Adam_Alexnet_Replication

April 29, 2024

0.0.1 Replication of Alexnet.

Due to limited computational resources the replciation is in limited scale. In particular, only the network were trained to only recognize first 20 classes of images. But this can be easily relaxed given sufficient computational power.

[1]:
```
import numpy as np
from functools import partial
from typing import Any, Optional

import os
import cv2
import time

import pandas as pd
import torch.nn.init as init

import torch
import torch.nn as nn
import torch.optim as optim
from torch.utils.data import Dataset, DataLoader
from PIL import Image
from torchvision import transforms

import matplotlib.pyplot as plt

# Device configuration
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
```

[2]:
```
# Neural Network Architecture.
class AlexNet(nn.Module):
    def __init__(self, num_classes: int = 1000, dropout: float = 0.5) -> None:
        super().__init__()
        self.features = nn.Sequential(
            nn.Conv2d(3, 96, kernel_size=11, stride=4, padding=2),
            nn.BatchNorm2d(96),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=3, stride=2),
            nn.Conv2d(96, 256, kernel_size=5, padding=2),
```

```
            nn.BatchNorm2d(256),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=3, stride=2),
            nn.Conv2d(256, 384, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.Conv2d(384, 384, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.Conv2d(384, 256, kernel_size=3, padding=1),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=3, stride=2),
        )
        self.avgpool = nn.AdaptiveAvgPool2d((6, 6))
        self.classifier = nn.Sequential(
            nn.Dropout(p=dropout),
            nn.Linear(256 * 6 * 6, 4096),
            nn.ReLU(inplace=True),
            nn.Dropout(p=dropout),
            nn.Linear(4096, 4096),
            nn.ReLU(inplace=True),
            nn.Linear(4096, num_classes),
        )

    def forward(self, x: torch.Tensor) -> torch.Tensor:
        x = self.features(x)
        x = self.avgpool(x)
        x = torch.flatten(x, 1)
        x = self.classifier(x)
        return x

    # Xavier Initialization
    def initialize_weights(self):
        for m in self.modules():
            if isinstance(m, nn.Conv2d) or isinstance(m, nn.Linear):
                # Initialize weights for convolutional and linear layers
                init.xavier_uniform_(m.weight)
                if m.bias is not None:
                    # Initialize biases if they exist
                    init.constant_(m.bias, 0)
```

[3]:
```
# Handle Exception for greyscale images
class ConvertToRGB(object):
    def __call__(self, img):
        if img.shape[0] == 1:  # Check if the image has only one channel
            img = torch.stack([img[0]] * 3, dim=0)  # Convert single channel to␣
 ↪RGB
        return img
```

```
# Preprocessing Images
preprocess = transforms.Compose([
    transforms.Resize(256),
    transforms.CenterCrop(224),
    transforms.ToTensor(),
    ConvertToRGB(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
    ])
```

[4]:
```
# the images were stored locally
image_path = "/Users/Limit/imagenet-object-localization-challenge_100"
filenames_image_path = []
label_train = []
root_image = []
counter = 0
current_label = 0

for root, _, filenames in os.walk(image_path):
    current_root = root
    for i in filenames:
        counter += 1
        # Print whenever one classes of images finished reading (each class has␣
 ↪1300 images)
        if ((counter) %1300 == 0):
            current_label += 1
            print(counter)
        # get labels
        label_train.append(current_label)
        temp = current_root + "\\" + i
        filenames_image_path.append(temp)
true_label = 0
correct_labels = 0
start_time = time.time()

print("image loading complete")

counter_1=0
x_train = []
# Due to limited computational resources, only train with first 20 classes of␣
 ↪images.
for i in range(26000):
    image_name = filenames_image_path[i]
    input_image = Image.open(image_name)
    input_tensor = preprocess(input_image)
    input_batch = input_tensor
    # move the input and model to GPU for speed if available
    if torch.cuda.is_available():
```

```
        input_batch = input_batch.to('cuda')
        model_alex_given.to('cuda')

    x_train.append(input_batch)
    counter_1 += 1
    # print the counter whenever finished processing the corresponding class
    if ((counter_1+1) %1300 == 0):
            counter_1 += 1
            print(counter_1)

print('image processing compelte')

y_train = label_train[:26000]
```

```
1300
2600
3900
5200
6500
7800
9100
10400
11700
13000
14300
15600
16900
18200
19500
20800
22100
23400
24700
26000
27300
28600
29900
31200
32500
33800
35100
36400
37700
39000
40300
41600
42900
44200
```

45500
46800
48100
49400
50700
52000
53300
54600
55900
57200
58500
59800
61100
62400
63700
65000
66300
67600
68900
70200
71500
72800
74100
75400
76700
78000
79300
80600
81900
83200
84500
85800
87100
88400
89700
91000
92300
93600
94900
96200
97500
98800
100100
101400
102700
104000
105300
106600

```
107900
109200
110500
111800
113100
114400
115700
117000
118300
119600
120900
122200
123500
124800
126100
127400
128700
image loading complete
1300
2600
3900
5200
6500
7800
9100
10400
11700
13000
14300
15600
16900
18200
19500
20800
22100
23400
24700
26000
image processing compelte
```

[5]:
```python
# Define dataset class
class CustomDataset(Dataset):
    def __init__(self, data, labels):
        self.data = data
        self.labels = labels

    def __len__(self):
```

```
        return len(self.data)

    def __getitem__(self, idx):
        return self.data[idx], torch.tensor(self.labels[idx])
```

[6]:
```
# Define the loss function
criterion = nn.CrossEntropyLoss()
# Initialize model
model = AlexNet().to(device)
model.initialize_weights()
# Define the optimizer
optimizer = optim.Adam(model.parameters(), lr=0.001, weight_decay=5e-4)

# Prepare the data
train_data = x_train  # List of input tensors
train_labels = y_train  # List of corresponding labels
dataset = CustomDataset(train_data, train_labels)
dataloader = DataLoader(dataset, batch_size=128, shuffle=True)

# Train the models
num_epochs = 10
for epoch in range(num_epochs):
    current_loss = 0.0
    for inputs, labels in dataloader:

        optimizer.zero_grad()
        outputs = model(inputs)
        # get the gradient before the update
        Before = list(model.parameters())[0].clone()

        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()

        current_loss += loss.item()
        # get the gradient after the update
        After = list(model.parameters())[0].clone()
    # Print the current learning detail
    for param_group in optimizer.param_groups:
        print("Learning rate:", param_group['lr'])
    print()
    print('another way to print learning rate:')
    for group in optimizer.param_groups:
        for p in group['params']:
            print(p.grad)  # Print gradients
    print('end of p.grad')
```

```
    print()
    # Verify whether gradient is computed successfully
    print(torch.equal(Before.data, After.data))
    print(f'Epoch {epoch+1} finished')
    epoch_loss = current_loss / len(dataset)
    print(f'Epoch [{epoch+1}/{num_epochs}], Loss: {loss.item():.4f}')
    print('************************************************************')
    print()
```

```
Learning rate: 0.001

another way to print learning rate:
tensor([[[[-2.2450e-04, -2.6235e-04,  1.3286e-05,  …, -4.0549e-04,
           -3.3386e-04, -5.6111e-04],
          [-1.3043e-04, -1.3913e-04, -2.6576e-05,  …, -1.9929e-04,
           -3.8673e-04, -5.9738e-04],
          [-9.8648e-05, -3.2555e-05,  1.3997e-04,  …, -1.4284e-04,
           -2.3201e-04, -4.2218e-04],
          …,
          [-2.4491e-04, -2.8296e-04, -9.6662e-05,  …, -1.7538e-04,
           -3.5029e-04, -4.0003e-04],
          [-4.0981e-04, -3.8546e-04, -1.6841e-04,  …, -2.8724e-04,
           -5.4481e-04, -4.2061e-04],
          [-4.3113e-04, -5.7506e-04, -2.9910e-04,  …, -6.1757e-04,
           -7.3993e-04, -5.0888e-04]],

         [[ 2.1652e-03,  2.1034e-03,  2.3343e-03,  …,  1.6621e-03,
            1.4816e-03,  1.2215e-03],
          [ 2.2524e-03,  2.3030e-03,  2.3419e-03,  …,  1.8686e-03,
            1.5167e-03,  1.2272e-03],
          [ 2.5604e-03,  2.6134e-03,  2.5874e-03,  …,  1.9380e-03,
            1.7817e-03,  1.5233e-03],
          …,
          [ 2.0643e-03,  2.0806e-03,  2.0535e-03,  …,  1.5873e-03,
            1.2570e-03,  1.2331e-03],
          [ 1.8915e-03,  1.9133e-03,  1.9438e-03,  …,  1.5486e-03,
            1.2238e-03,  1.3018e-03],
          [ 1.9480e-03,  1.8175e-03,  1.9074e-03,  …,  1.4511e-03,
            1.2744e-03,  1.3329e-03]],

         [[ 1.0050e-04,  1.7894e-04,  6.0103e-04,  …,  1.5318e-04,
           -4.7419e-05, -4.5425e-04],
          [ 1.0963e-04,  3.4200e-04,  5.1452e-04,  …,  4.3576e-04,
           -1.8559e-05, -3.6932e-04],
          [ 3.6491e-04,  6.4369e-04,  7.3898e-04,  …,  3.6338e-04,
            2.1941e-04, -9.0954e-05],
          …,
          [-9.2269e-05,  1.1274e-04,  2.0582e-04,  …, -6.6928e-05,
```

```
   -3.5984e-04, -4.3287e-04],
  [-3.4512e-04, -8.7569e-05,  2.0010e-05,  …, -1.4138e-04,
   -4.7387e-04, -3.5112e-04],
  [-4.7225e-04, -3.4872e-04, -1.5059e-04,  …, -4.0869e-04,
   -5.5646e-04, -3.5946e-04]]],


[[[ 8.7228e-04,  2.0809e-03,  3.1754e-03,  …,  4.2745e-03,
    3.5668e-03,  2.4240e-03],
  [ 2.5960e-03,  2.6544e-03,  3.8584e-03,  …,  3.8226e-03,
    2.1843e-03,  1.1908e-03],
  [ 4.1979e-03,  5.4852e-03,  4.7051e-03,  …,  2.4902e-03,
    1.5758e-03,  1.4826e-03],
  …,
  [ 6.2407e-03,  6.3033e-03,  6.0764e-03,  …,  3.9729e-03,
    2.5671e-03,  2.2804e-03],
  [ 6.1017e-03,  6.4274e-03,  6.0495e-03,  …,  5.4083e-03,
    4.4677e-03,  3.4453e-03],
  [ 6.2858e-03,  7.1168e-03,  6.0168e-03,  …,  6.8871e-03,
    5.4267e-03,  3.9873e-03]],

 [[-2.6602e-03, -1.9125e-03, -1.3427e-03,  …,  2.7827e-03,
    3.3923e-03,  3.8221e-03],
  [-1.1970e-03, -1.7534e-03, -1.5993e-03,  …,  3.5788e-03,
    3.3576e-03,  3.4934e-03],
  [-4.5049e-04, -4.0946e-04, -1.2256e-03,  …,  2.8361e-03,
    3.4070e-03,  4.2883e-03],
  …,
  [-4.5520e-04, -5.1864e-04, -1.7199e-04,  …,  1.7628e-03,
    7.1193e-04,  1.5707e-03],
  [-1.6157e-03, -8.5221e-04, -2.2690e-04,  …,  2.7229e-03,
    2.2141e-03,  2.2228e-03],
  [-1.8168e-03, -6.8410e-04, -4.0478e-04,  …,  3.8515e-03,
    2.4876e-03,  1.5894e-03]],

 [[-1.0227e-03, -1.5862e-03, -8.8575e-04,  …,  1.2441e-04,
    6.6647e-04,  1.8412e-03],
  [-3.5726e-04, -8.1069e-04, -4.4825e-04,  …,  1.5249e-03,
    1.7675e-03,  1.9034e-03],
  [ 2.2958e-04,  8.9246e-04, -3.0613e-04,  …,  2.1728e-03,
    3.2007e-03,  3.7246e-03],
  …,
  [-1.5564e-03, -2.1891e-03, -2.0420e-03,  …,  5.3427e-03,
    3.8548e-03,  4.0797e-03],
  [-2.8985e-03, -2.3295e-03, -1.5805e-03,  …,  6.6470e-03,
    5.7342e-03,  4.5484e-03],
  [-3.8722e-03, -2.3288e-03, -1.9014e-03,  …,  7.4626e-03,
    6.4747e-03,  5.4072e-03]]],
```

```
[[[-4.9443e-04, -5.5272e-04, -4.3362e-04,  …, -5.8670e-04,
   -1.0077e-03, -1.5394e-03],
  [-5.0052e-04, -6.0064e-04, -6.1253e-04,  …, -6.2991e-04,
   -6.5345e-04, -1.2926e-03],
  [-4.9849e-04, -3.9534e-04, -4.4179e-04,  …, -6.2165e-04,
   -7.4747e-04, -9.0831e-04],
  …,
  [-6.5374e-04, -5.8734e-04, -1.3524e-04,  …,  1.4915e-04,
   -2.2147e-05,  1.8527e-04],
  [-6.2436e-04, -3.3183e-04, -1.6235e-04,  …,  2.1338e-04,
    3.3285e-04,  4.3004e-04],
  [-5.4816e-04, -2.1557e-04, -3.8620e-04,  …,  3.2453e-04,
    4.9696e-04,  4.4259e-04]],

 [[ 2.9113e-04,  1.5038e-04,  2.1095e-04,  …, -1.3497e-04,
   -2.4305e-04, -5.0285e-04],
  [ 3.8801e-04,  1.9409e-04,  7.8242e-05,  …, -2.1607e-04,
   -5.6727e-06, -2.9055e-04],
  [ 4.4765e-04,  5.7057e-04,  4.9009e-04,  …, -1.3672e-04,
   -1.2094e-04,  1.0127e-04],
  …,
  [ 5.9474e-04,  7.0180e-04,  9.5088e-04,  …,  9.7629e-04,
    7.1613e-04,  8.9327e-04],
  [ 4.7314e-04,  7.3244e-04,  9.6231e-04,  …,  8.5401e-04,
    8.7221e-04,  1.0370e-03],
  [ 5.7751e-04,  9.7424e-04,  9.5804e-04,  …,  9.0472e-04,
    9.7441e-04,  9.4112e-04]],

 [[-5.8028e-04, -7.0661e-04, -5.7110e-04,  …, -1.5536e-03,
   -1.7731e-03, -1.6878e-03],
  [-3.7442e-04, -5.6967e-04, -5.1851e-04,  …, -1.3090e-03,
   -1.2430e-03, -1.4237e-03],
  [-3.6386e-04, -2.4535e-04, -1.8653e-04,  …, -1.1799e-03,
   -1.2660e-03, -1.1464e-03],
  …,
  [ 1.9567e-04,  2.8007e-04,  3.6589e-04,  …,  2.6968e-05,
   -2.3598e-04, -2.2487e-04],
  [ 4.8697e-06,  3.2485e-04,  5.8561e-04,  …,  2.9119e-05,
    3.6294e-06, -3.1966e-06],
  [-1.2032e-04,  3.1834e-04,  3.7644e-04,  …, -4.5154e-05,
    5.9100e-05, -6.0336e-05]]],


…,
```

```
[[[-2.6655e-03, -2.1954e-03, -1.9857e-03,  …, -1.8776e-03,
   -1.1333e-03, -1.3101e-03],
  [-2.2520e-03, -1.5418e-03, -1.4618e-03,  …, -9.3706e-04,
   -9.9944e-04, -6.9122e-04],
  [-2.0816e-03, -1.9547e-03, -1.1636e-03,  …, -5.0743e-04,
   -5.0281e-04, -2.8350e-04],
  …,
  [-1.6316e-03, -6.2708e-04, -3.6107e-04,  …, -6.4660e-04,
   -5.7673e-04, -1.2374e-03],
  [-1.3073e-03, -7.8761e-04, -7.4252e-04,  …, -4.8779e-04,
   -1.7469e-03, -1.9346e-03],
  [-8.1381e-04, -9.1683e-04, -7.1069e-04,  …, -1.3207e-03,
   -2.2830e-03, -2.3038e-03]],

 [[-2.2746e-03, -2.5346e-03, -2.8078e-03,  …, -3.0071e-03,
   -2.4150e-03, -2.4825e-03],
  [-2.5386e-03, -2.1397e-03, -2.3793e-03,  …, -2.4032e-03,
   -2.5532e-03, -2.1004e-03],
  [-2.8076e-03, -2.9655e-03, -2.4698e-03,  …, -2.1557e-03,
   -2.4227e-03, -1.9067e-03],
  …,
  [-2.8256e-03, -2.3383e-03, -2.5130e-03,  …, -2.2245e-03,
   -2.3573e-03, -2.2606e-03],
  [-2.3849e-03, -2.3736e-03, -2.5253e-03,  …, -1.6762e-03,
   -3.0286e-03, -2.7050e-03],
  [-1.8765e-03, -2.6272e-03, -2.3154e-03,  …, -2.2314e-03,
   -3.0785e-03, -2.7623e-03]],

 [[-3.7861e-03, -3.8497e-03, -4.3046e-03,  …, -4.6469e-03,
   -3.8601e-03, -3.7688e-03],
  [-4.1422e-03, -3.7939e-03, -4.3125e-03,  …, -4.2684e-03,
   -4.4499e-03, -3.7165e-03],
  [-4.5776e-03, -5.0314e-03, -4.4168e-03,  …, -3.7047e-03,
   -4.0233e-03, -3.1973e-03],
  …,
  [-4.6861e-03, -3.7673e-03, -3.8440e-03,  …, -3.4329e-03,
   -3.5886e-03, -3.1952e-03],
  [-3.9512e-03, -4.0480e-03, -4.2748e-03,  …, -3.0371e-03,
   -4.3148e-03, -3.7913e-03],
  [-3.1400e-03, -4.0268e-03, -3.9542e-03,  …, -3.1890e-03,
   -4.1864e-03, -3.8130e-03]]],


[[[-3.6215e-03, -2.7614e-03, -2.2490e-03,  …, -2.6739e-03,
   -1.7931e-03, -2.1586e-03],
  [-3.2102e-03, -2.1410e-03, -2.0325e-03,  …, -2.3342e-03,
   -1.5969e-03, -1.2916e-03],
  [-2.3461e-03, -2.4454e-03, -2.1601e-03,  …, -2.1008e-03,
```

```
  -1.8973e-03, -1.6098e-03],
 …,
 [-2.3051e-03, -1.3633e-03, -2.3185e-03,  …, -2.6538e-03,
  -2.3346e-03, -2.7877e-03],
 [-1.6522e-03, -1.6748e-03, -2.8022e-03,  …, -2.4592e-03,
  -3.1550e-03, -3.4136e-03],
 [-1.7396e-03, -2.4475e-03, -2.1567e-03,  …, -3.5344e-03,
  -4.1806e-03, -4.1003e-03]],

[[-2.1964e-03, -2.1983e-03, -2.2851e-03,  …, -4.2791e-03,
  -3.0871e-03, -3.3293e-03],
 [-2.5888e-03, -2.2504e-03, -2.3851e-03,  …, -4.2253e-03,
  -3.3291e-03, -2.9431e-03],
 [-2.5551e-03, -2.9439e-03, -2.6872e-03,  …, -4.0317e-03,
  -4.3252e-03, -3.9893e-03],
 …,
 [-3.6962e-03, -3.0089e-03, -4.0038e-03,  …, -4.1538e-03,
  -3.8580e-03, -3.8818e-03],
 [-3.0090e-03, -3.0863e-03, -4.3528e-03,  …, -3.7154e-03,
  -4.5580e-03, -4.5720e-03],
 [-3.0109e-03, -3.9552e-03, -3.8326e-03,  …, -4.6444e-03,
  -5.4849e-03, -4.9872e-03]],

[[-1.6217e-03, -1.3688e-03, -2.0235e-03,  …, -5.0882e-03,
  -3.2053e-03, -3.1847e-03],
 [-1.8820e-03, -2.0091e-03, -2.3547e-03,  …, -4.9815e-03,
  -4.0654e-03, -3.2035e-03],
 [-1.8360e-03, -2.8484e-03, -2.7284e-03,  …, -4.3713e-03,
  -4.6892e-03, -4.1446e-03],
 …,
 [-3.5491e-03, -2.4188e-03, -3.2593e-03,  …, -3.7752e-03,
  -3.2746e-03, -3.1026e-03],
 [-2.5314e-03, -2.6344e-03, -3.9226e-03,  …, -3.5209e-03,
  -4.4131e-03, -4.3478e-03],
 [-2.6983e-03, -3.8795e-03, -3.5043e-03,  …, -3.9787e-03,
  -5.1504e-03, -4.6708e-03]]],


[[[-4.1986e-03, -3.8586e-03, -3.9825e-03,  …, -3.3137e-03,
   -2.2789e-03, -2.8435e-03],
  [-3.7673e-03, -3.4877e-03, -3.3475e-03,  …, -3.0708e-03,
   -2.4150e-03, -2.3419e-03],
  [-3.2171e-03, -3.6350e-03, -3.1014e-03,  …, -3.0336e-03,
   -2.5601e-03, -2.1421e-03],
  …,
  [-3.0625e-03, -3.0683e-03, -3.2513e-03,  …, -4.2693e-03,
   -4.4701e-03, -3.9902e-03],
  [-2.8496e-03, -3.4464e-03, -3.7352e-03,  …, -4.3571e-03,
```

```
           -4.6453e-03, -3.8900e-03],
          [-2.8371e-03, -3.8581e-03, -3.5105e-03,  …, -4.8879e-03,
           -4.7479e-03, -3.7943e-03]],

         [[-4.2474e-03, -4.1948e-03, -4.3423e-03,  …, -4.8824e-03,
           -4.2503e-03, -4.7611e-03],
          [-4.8164e-03, -4.5795e-03, -4.2122e-03,  …, -5.1252e-03,
           -4.6690e-03, -4.7356e-03],
          [-5.0660e-03, -5.0325e-03, -4.1231e-03,  …, -5.4034e-03,
           -5.2949e-03, -5.0588e-03],
          …,
          [-5.6755e-03, -5.8352e-03, -5.9492e-03,  …, -7.0721e-03,
           -6.8430e-03, -6.4586e-03],
          [-5.6430e-03, -6.3440e-03, -6.3743e-03,  …, -6.9515e-03,
           -7.3662e-03, -6.5425e-03],
          [-5.7233e-03, -6.9041e-03, -6.0195e-03,  …, -7.1682e-03,
           -7.5161e-03, -6.7219e-03]],

         [[-5.6354e-03, -5.2485e-03, -5.6457e-03,  …, -5.8381e-03,
           -4.6919e-03, -5.3838e-03],
          [-6.2789e-03, -5.8804e-03, -5.8251e-03,  …, -6.0646e-03,
           -5.6051e-03, -5.6553e-03],
          [-6.3720e-03, -6.3353e-03, -5.2916e-03,  …, -6.5168e-03,
           -6.1605e-03, -5.6041e-03],
          …,
          [-6.5880e-03, -6.2767e-03, -6.0747e-03,  …, -7.8239e-03,
           -7.4370e-03, -6.4784e-03],
          [-6.1662e-03, -6.8824e-03, -6.7332e-03,  …, -7.9388e-03,
           -8.1456e-03, -7.0171e-03],
          [-5.8967e-03, -7.1783e-03, -6.1453e-03,  …, -7.9529e-03,
           -8.1542e-03, -7.2911e-03]]]])
tensor([ 6.0536e-09,  2.5146e-08,  7.0431e-09,  1.2806e-09, -9.7789e-09,
         1.0338e-07, -1.3970e-08,  2.0955e-08, -7.3342e-09,  8.8476e-09,
         8.2873e-09, -2.3283e-09, -1.1642e-09, -1.0477e-08,  3.2596e-09,
        -1.0477e-09,  1.5134e-08, -6.9849e-09, -3.7428e-08, -2.5611e-09,
         3.9145e-09,  1.1642e-08,  1.0987e-09, -6.9849e-10, -1.1874e-08,
        -7.9162e-09, -1.3533e-09,  9.7789e-09,  5.1223e-09, -2.0955e-08,
        -3.3178e-09, -1.5134e-09, -5.6461e-09, -7.7125e-10,  2.5379e-08,
         1.1059e-09,  9.7789e-09,  4.5402e-09,  0.0000e+00, -2.1188e-08,
        -3.0268e-09,  1.1059e-08, -5.8208e-10, -2.0955e-09,  1.6764e-08,
        -1.0565e-08, -2.2701e-09,  3.2596e-09,  1.6298e-08,  1.0899e-08,
        -5.2387e-10,  5.7044e-09, -3.4925e-10,  1.5367e-08, -1.8044e-09,
        -1.0186e-10,  1.2049e-08, -9.3132e-10, -2.0431e-08,  7.2177e-09,
         5.3551e-09, -3.4925e-10,  4.0745e-08, -1.6298e-09, -2.6776e-09,
        -1.4552e-09, -4.4005e-08, -6.5193e-09,  1.8626e-09,  6.1700e-09,
         6.4028e-10,  1.8335e-09,  1.7229e-08, -1.8976e-08,  2.5782e-08,
         9.3132e-10,  8.7311e-11, -1.3039e-08, -1.1176e-08, -1.8626e-08,
         5.5879e-09, -4.6566e-09, -3.0268e-09, -8.8476e-09, -3.1665e-08,
```

```
        -4.4529e-09, -9.0804e-09,  3.4634e-09, -1.5134e-09, -3.5856e-08,
         0.0000e+00, -1.8626e-09,  2.6193e-08, -2.7008e-08, -3.2829e-08,
         9.6770e-09])
tensor([-3.1339e-03,  4.4844e-03, -2.5021e-03, -2.1334e-03, -5.7859e-04,
         1.5974e-02,  9.0197e-04,  8.2594e-03,  2.1123e-03, -1.2218e-02,
         1.2826e-03,  1.5641e-03, -5.4733e-03,  1.0729e-03,  1.2404e-03,
        -1.3353e-03,  7.2340e-04, -4.3057e-03, -1.1468e-02,  1.2278e-03,
        -1.5542e-03,  3.7812e-03, -6.5647e-03, -4.7042e-04, -2.7936e-03,
         2.7164e-03, -2.1354e-03, -1.2023e-03,  1.0984e-02,  1.0446e-03,
         4.0112e-03, -3.1501e-03,  1.3591e-03, -1.7205e-03,  3.6888e-03,
        -2.1118e-03, -1.4983e-02, -4.8394e-03,  7.6642e-03,  7.9222e-04,
         8.9615e-03, -4.6549e-03, -3.1345e-03,  5.3394e-04, -3.6109e-03,
        -5.6301e-03, -3.6135e-03, -9.7899e-04,  4.5757e-04, -6.5295e-03,
        -3.0813e-03, -5.8450e-03,  1.1155e-03,  2.3415e-03, -2.7027e-03,
        -9.0805e-03, -1.3883e-03, -1.0023e-03, -6.4286e-06, -1.6716e-03,
         3.1024e-03, -2.0076e-03, -7.8981e-03, -2.2020e-03, -1.5661e-03,
         1.7878e-03,  3.3988e-03, -1.5375e-02, -7.8825e-04, -7.9101e-04,
        -3.3849e-03, -3.6194e-03,  4.6541e-04,  4.4619e-03,  3.3047e-03,
        -8.9137e-03, -2.2858e-03, -5.8426e-04,  2.7426e-03, -1.5572e-03,
         4.0223e-03,  3.3732e-02,  6.0934e-04,  2.0543e-03,  1.3523e-02,
        -1.3022e-02,  2.8139e-03, -1.7000e-03, -5.5451e-04,  3.2046e-03,
         3.1511e-03,  3.0859e-02, -6.4629e-03,  1.9544e-03,  1.4194e-03,
         1.7819e-03])
tensor([-2.8560e-03,  3.0074e-03,  2.5643e-03,  9.0516e-04,  1.9549e-03,
        -1.6031e-03, -6.7629e-03, -8.9809e-04, -5.5439e-03, -1.4078e-03,
         2.4820e-03,  1.7452e-03, -2.7878e-03,  2.3261e-04, -6.2247e-03,
         5.1916e-04,  7.3838e-04, -3.2245e-03,  6.7732e-03,  1.2178e-05,
         1.5647e-03, -7.7282e-04,  3.2598e-03, -8.8877e-04,  3.7427e-03,
        -3.6805e-03,  3.3076e-04, -3.1009e-04,  3.3078e-05, -7.7173e-04,
         4.2995e-03, -5.9331e-04, -5.3916e-04,  2.1728e-04, -2.3278e-03,
         5.8171e-04,  5.0776e-03, -1.1998e-03,  6.5763e-03,  9.6618e-04,
         6.9845e-03,  1.0769e-03,  1.2188e-03,  5.0085e-04, -5.7834e-04,
         5.6098e-03,  6.1288e-04, -1.2213e-03, -6.2244e-03,  5.4881e-03,
        -1.1396e-03, -2.6278e-03, -1.7259e-03, -5.0510e-03,  3.5231e-04,
         3.7806e-03, -1.4515e-03,  4.0540e-04,  1.7873e-03, -1.4796e-03,
         3.9151e-03,  4.6942e-04, -9.1711e-04, -4.3700e-04,  3.8396e-04,
        -1.5615e-03, -7.2524e-03,  3.3934e-04,  8.4716e-04, -3.7634e-03,
        -4.4309e-04,  2.5165e-03, -1.1027e-03,  1.8441e-03,  8.6240e-04,
        -1.9348e-04, -5.0092e-04, -1.4487e-03,  5.7852e-04, -5.8208e-03,
        -9.8558e-04,  4.3238e-03,  1.9879e-05, -7.2059e-03,  6.4160e-03,
         3.4306e-03, -5.8799e-03,  2.9400e-05,  1.1555e-03, -3.4086e-03,
         3.8617e-03,  5.0825e-03,  4.8999e-03, -6.3737e-03, -2.5977e-03,
         1.3853e-03])
tensor([[[[-5.8228e-04, -5.1785e-04, -3.6837e-04, -2.5006e-04, -4.5957e-05],
          [-4.8898e-04, -4.2059e-04, -2.8077e-04, -2.5921e-04, -9.6911e-05],
          [-2.5149e-04, -2.0462e-04, -1.6272e-04, -1.2628e-04, -5.8761e-05],
          [ 7.8214e-05,  4.4160e-05, -3.6201e-05, -3.9168e-06,  3.5908e-05],
          [ 3.1757e-04,  2.4566e-04,  1.3342e-04,  6.9916e-05,  4.4159e-05]],
```

```
 [[ 5.4918e-04,  9.9066e-04,  1.0349e-03,  4.9165e-04,  1.7957e-04],
  [ 5.7236e-04,  8.3786e-04,  9.0766e-04,  5.4220e-04,  2.1655e-04],
  [ 6.6798e-04,  6.2258e-04,  7.8586e-04,  5.7085e-04,  2.9597e-04],
  [ 4.9322e-04,  3.4305e-04,  7.4478e-04,  5.1909e-04,  4.5249e-04],
  [ 4.2391e-05,  2.3899e-04,  5.9483e-04,  6.0908e-04,  6.6370e-04]],

 [[ 5.9688e-04,  6.4951e-04,  6.6061e-04,  4.6489e-04,  1.0869e-04],
  [ 6.4017e-04,  7.7991e-04,  8.3771e-04,  5.9252e-04,  3.3800e-04],
  [ 5.8423e-04,  6.0769e-04,  6.5756e-04,  5.1784e-04,  2.5445e-04],
  [ 2.5155e-04,  2.4513e-05,  1.1822e-04, -5.6953e-05, -3.1647e-04],
  [-6.7390e-06, -2.4320e-04, -1.3175e-04, -2.9394e-04, -4.8842e-04]],

 …,

 [[-1.3841e-04, -2.0966e-04, -7.4729e-05, -2.5260e-05, -8.8818e-05],
  [-2.4894e-04, -1.4831e-04,  3.2742e-05,  4.8350e-05,  1.2314e-04],
  [-2.7550e-04, -2.8121e-04, -2.1557e-04, -1.4221e-04, -1.1242e-04],
  [-4.3504e-04, -6.9615e-04, -6.4167e-04, -6.2207e-04, -7.2250e-04],
  [-5.0997e-04, -7.4204e-04, -6.5956e-04, -7.6550e-04, -9.1394e-04]],

 [[-3.4734e-04, -4.2928e-04, -3.5175e-04, -2.6629e-04, -1.8373e-04],
  [-4.0423e-04, -3.5856e-04, -3.0277e-04, -2.0583e-04, -3.6400e-05],
  [-4.4556e-04, -4.1156e-04, -4.0884e-04, -2.7214e-04, -1.7533e-04],
  [-4.1955e-04, -6.1051e-04, -5.3570e-04, -5.2110e-04, -5.3227e-04],
  [-3.3781e-04, -4.7938e-04, -4.5576e-04, -4.9829e-04, -6.3015e-04]],

 [[-4.5959e-04, -5.9939e-04, -4.5402e-04, -3.3915e-04, -2.0584e-04],
  [-5.2952e-04, -5.3875e-04, -3.9606e-04, -3.0480e-04, -6.7786e-05],
  [-4.7736e-04, -4.8335e-04, -4.1803e-04, -3.4362e-04, -2.3052e-04],
  [-4.1005e-04, -6.1990e-04, -6.1441e-04, -5.1695e-04, -5.3112e-04],
  [-2.4082e-04, -4.1805e-04, -4.4346e-04, -4.7530e-04, -6.1471e-04]]],


[[[-2.6347e-04, -1.7698e-04,  4.8281e-06,  1.0341e-04,  2.0285e-04],
  [-1.3786e-04, -2.8949e-05,  1.4734e-04,  1.7694e-04,  2.1236e-04],
  [-6.5756e-05,  1.1454e-04,  2.9635e-04,  2.9334e-04,  2.5728e-04],
  [ 4.5018e-05,  1.9793e-04,  3.6994e-04,  3.9011e-04,  2.5771e-04],
  [ 4.8875e-05,  1.9492e-04,  3.3596e-04,  2.8480e-04,  2.1092e-04]],

 [[ 1.4559e-04,  5.9219e-05,  1.0883e-04, -9.2140e-05, -2.3212e-04],
  [ 1.4769e-04, -2.0938e-05, -4.8113e-05, -2.9804e-04, -1.3562e-04],
  [ 1.6144e-04, -1.3654e-04, -2.7593e-04, -2.9802e-04,  5.2528e-05],
  [ 2.1681e-04, -1.7320e-04, -2.5594e-04, -2.4963e-04,  1.7826e-04],
  [ 1.0438e-04, -2.1412e-04, -2.6014e-04, -1.6492e-04,  2.2001e-04]],

 [[ 2.9523e-04,  2.8932e-04,  1.9833e-04,  1.3077e-04, -9.3110e-06],
  [ 2.0406e-04,  1.9934e-04,  8.8172e-05,  4.8993e-05, -5.7990e-05],
```

```
  [ 8.9186e-05,  1.4157e-04, -3.8592e-05, -1.3748e-04, -7.9940e-05],
  [-5.1888e-05, -5.7436e-06,  6.5422e-05,  2.4121e-05,  1.5655e-04],
  [-5.2886e-05,  5.5108e-05,  3.3726e-05,  1.1193e-05,  2.5034e-04]],

 …,

 [[ 2.9023e-04,  3.5375e-04,  2.2543e-04,  1.5189e-04,  2.2731e-04],
  [ 1.5846e-04,  1.3924e-04,  6.3114e-05,  1.8086e-05,  3.9376e-05],
  [-2.0432e-05, -9.4619e-06, -3.9635e-05, -7.8356e-05,  6.1730e-05],
  [-3.6670e-05, -9.4389e-05, -2.8746e-05, -6.2733e-06,  1.1060e-04],
  [-2.2978e-05, -7.4280e-05, -4.4109e-05, -2.7942e-05,  5.2975e-05]],

 [[ 1.5403e-04,  1.7643e-04,  1.4826e-04,  1.6905e-04,  2.3088e-04],
  [ 6.7713e-05,  8.0683e-05,  4.3023e-05,  5.6608e-05,  6.3455e-05],
  [-2.2183e-05, -9.5662e-07, -1.5081e-05, -2.4888e-05,  5.0999e-05],
  [-2.3284e-05, -3.5783e-05, -2.0966e-05, -2.5409e-05,  1.3159e-05],
  [-2.7970e-05, -4.3785e-05, -3.5908e-05, -6.7062e-05, -6.3533e-05]],

 [[ 1.2863e-04,  1.9365e-04,  1.3885e-04,  6.7617e-05,  2.0934e-04],
  [ 3.4755e-05,  7.6155e-05,  1.3123e-05, -9.3471e-06,  3.3415e-05],
  [-5.5867e-05, -3.1390e-05, -4.2171e-05, -6.6817e-05,  8.5166e-06],
  [-2.6613e-05, -2.5717e-05, -4.5342e-05, -3.0525e-05, -4.6719e-05],
  [-1.6673e-05, -3.8822e-05, -5.1022e-05, -3.9648e-05, -1.3445e-04]]],


[[[ 4.7319e-04,  5.3432e-04,  4.2098e-04,  4.7704e-04,  2.7690e-04],
  [ 5.9709e-04,  5.1308e-04,  4.0899e-04,  4.1511e-04,  3.0956e-04],
  [ 5.1016e-04,  4.0757e-04,  3.4222e-04,  3.1444e-04,  2.1752e-04],
  [ 5.0985e-04,  4.1014e-04,  3.8095e-04,  3.3667e-04,  2.3017e-04],
  [ 4.0722e-04,  3.4512e-04,  3.1315e-04,  2.9604e-04,  1.9059e-04]],

 [[ 1.1222e-03,  1.8307e-04, -2.9869e-04, -3.9081e-04, -1.1962e-04],
  [ 4.6495e-04,  6.4975e-05, -2.9589e-04, -3.6040e-04, -2.3648e-04],
  [ 2.0088e-04, -9.7037e-05, -2.7776e-04, -2.4708e-04, -2.6044e-04],
  [ 3.2239e-05, -1.8565e-04, -2.9774e-04, -3.0665e-04, -1.6499e-04],
  [ 5.6633e-05, -1.0319e-04, -3.7711e-04, -2.7290e-04, -1.0954e-04]],

 [[-2.8358e-04, -1.9828e-04, -3.2749e-04, -5.1532e-04, -5.5312e-04],
  [-3.3063e-04, -2.7804e-04, -4.3811e-04, -5.5539e-04, -5.5907e-04],
  [-2.7302e-04, -1.4363e-04, -3.3553e-04, -3.2127e-04, -3.5099e-04],
  [-3.2438e-04, -1.5873e-04, -2.7268e-04, -2.2155e-04, -2.4713e-04],
  [-2.6091e-04, -1.0280e-05, -2.0800e-05, -6.4102e-05, -9.0747e-05]],

 …,

 [[-2.1837e-04,  1.8365e-04,  2.6184e-04,  2.4403e-04, -7.9076e-07],
  [ 2.4369e-05,  2.7599e-04,  3.2171e-04,  1.9894e-04,  7.4221e-05],
  [ 2.1211e-04,  4.5783e-04,  5.0534e-04,  5.0561e-04,  4.0675e-04],
```

```
  [ 2.0505e-04,  4.4060e-04,  5.8252e-04,  6.4248e-04,  4.9714e-04],
  [ 2.5364e-04,  5.5191e-04,  7.3317e-04,  7.1879e-04,  5.5288e-04]],

 [[ 2.3418e-04,  5.2200e-04,  5.9811e-04,  5.8694e-04,  4.0252e-04],
  [ 4.5321e-04,  6.3314e-04,  6.7610e-04,  5.6392e-04,  4.8515e-04],
  [ 5.5295e-04,  7.2146e-04,  7.8312e-04,  7.7423e-04,  6.8696e-04],
  [ 5.1272e-04,  6.7671e-04,  8.2931e-04,  8.7170e-04,  7.3638e-04],
  [ 5.4084e-04,  7.3526e-04,  8.9826e-04,  8.8175e-04,  7.0441e-04]],

 [[ 2.0421e-04,  5.8884e-04,  6.8616e-04,  6.1847e-04,  3.8450e-04],
  [ 4.5293e-04,  6.3427e-04,  7.5981e-04,  6.1405e-04,  4.8669e-04],
  [ 5.2710e-04,  7.0988e-04,  8.0233e-04,  7.8159e-04,  6.7548e-04],
  [ 5.0497e-04,  6.6715e-04,  8.4304e-04,  8.5367e-04,  7.3505e-04],
  [ 4.6591e-04,  6.9630e-04,  8.4708e-04,  8.7935e-04,  6.8853e-04]]],


…,


[[[ 1.9989e-05,  1.9370e-04,  2.9327e-04,  2.6131e-04,  2.9233e-04],
  [ 2.4566e-05,  1.6382e-04,  1.4420e-04,  1.5875e-04,  2.5979e-04],
  [-1.0903e-04,  4.5408e-05,  1.2060e-04,  7.8014e-05,  5.0880e-05],
  [-3.0414e-04, -1.6296e-04,  1.2186e-05,  5.2203e-06, -4.0733e-05],
  [-4.1555e-04, -3.3365e-04, -1.3643e-04, -1.5044e-04, -2.1076e-04]],

 [[-5.8952e-04, -1.8840e-04,  3.3998e-04,  1.6091e-04, -3.1639e-04],
  [-4.7886e-04, -3.0410e-05,  4.9076e-04,  5.6184e-04,  1.6711e-04],
  [-6.6482e-04, -1.4088e-04,  4.4150e-04,  5.7607e-04,  3.8640e-04],
  [-7.2434e-04, -1.4348e-04,  2.2216e-04,  7.1282e-04,  6.0137e-04],
  [-6.3307e-04, -3.6466e-04,  5.4030e-05,  7.6842e-04,  8.0900e-04]],

 [[ 9.9638e-05,  5.8219e-05,  8.1208e-05,  1.6910e-04, -3.0798e-06],
  [ 3.9932e-04,  2.1331e-04,  3.5354e-04,  3.9381e-04,  2.2985e-04],
  [ 4.5242e-04,  4.0979e-04,  3.8937e-04,  5.6203e-04,  5.2563e-04],
  [ 7.1383e-04,  4.2246e-04,  3.0912e-04,  4.3761e-04,  5.0695e-04],
  [ 6.9921e-04,  4.1578e-04,  3.3467e-04,  4.4678e-04,  6.4419e-04]],

 …,

 [[ 1.3449e-04, -1.0401e-04, -1.0731e-04, -8.8786e-05, -1.9398e-05],
  [ 1.6566e-04, -4.0730e-05, -7.6103e-05, -9.3004e-05, -1.4040e-04],
  [ 5.3730e-04,  1.9423e-04, -5.0492e-05, -9.5424e-05, -1.1537e-04],
  [ 5.8530e-04,  2.7515e-04, -1.0471e-04, -1.3193e-04,  4.0115e-05],
  [ 5.9426e-04,  8.9841e-05, -1.6840e-04, -1.4751e-04,  1.3135e-04]],

 [[ 1.4713e-04,  2.7099e-05,  2.6933e-05,  3.0629e-05,  2.3088e-04],
  [ 1.3190e-04,  8.2576e-05, -6.0312e-06, -1.1365e-04, -1.1209e-05],
  [ 3.9668e-04,  1.9120e-04,  2.8837e-05, -1.2802e-04, -1.0111e-04],
```

```
  [ 3.2170e-04,  1.9789e-04,  2.1270e-06, -1.3034e-04, -2.3872e-06],
  [ 4.0042e-04,  5.1028e-05, -1.0419e-04, -1.3311e-04,  6.2170e-05]],

 [[ 6.1343e-05, -8.3623e-05, -5.4147e-05, -9.7213e-05,  1.5212e-05],
  [ 4.5630e-05, -2.1902e-06, -1.2044e-04, -1.5979e-04, -1.4070e-04],
  [ 3.4174e-04,  1.6353e-05, -9.2221e-05, -2.3385e-04, -2.5626e-04],
  [ 2.6159e-04,  8.8044e-05, -1.4205e-04, -2.4092e-04, -9.2680e-05],
  [ 3.7733e-04, -5.3515e-05, -2.7889e-04, -3.2132e-04, -5.6095e-05]]],


[[[-1.7881e-04, -5.8228e-05,  5.1547e-05,  1.2142e-04,  1.4826e-04],
  [-1.4216e-04,  2.5687e-05,  2.2274e-04,  1.9293e-04,  1.3977e-04],
  [-1.8001e-04,  4.3859e-05,  2.5844e-04,  2.9511e-04,  1.8821e-04],
  [-3.5577e-04, -2.2771e-04,  9.2215e-05,  1.8646e-04,  1.1077e-04],
  [-5.2086e-04, -4.0223e-04, -1.3669e-04,  4.7785e-06, -1.3343e-05]],

 [[ 3.1971e-06, -1.3657e-05, -1.1985e-04,  4.5295e-05,  1.1204e-04],
  [ 9.6514e-05,  3.8805e-05,  7.2365e-05,  1.4674e-04,  1.1636e-04],
  [ 1.2846e-04,  6.6452e-05,  1.6647e-04,  3.0382e-04,  1.5265e-04],
  [ 1.7896e-04,  1.1623e-04,  2.5827e-04,  3.3484e-04,  2.1009e-04],
  [ 2.0946e-04,  2.0528e-04,  3.0884e-04,  3.4878e-04,  2.1221e-04]],

 [[ 1.5101e-04,  1.3127e-04,  9.1708e-05, -4.8437e-07,  9.7174e-06],
  [ 2.1876e-04,  1.3262e-04,  1.1076e-04,  7.4815e-05,  5.7800e-05],
  [ 3.2054e-04,  2.2330e-04,  1.0766e-04,  6.7757e-05,  9.0025e-05],
  [ 3.9770e-04,  2.7559e-04,  3.3617e-05, -1.7172e-05,  8.6371e-06],
  [ 3.2756e-04,  2.8000e-04,  1.3848e-04, -4.2411e-05, -3.8005e-05]],

 …,

 [[ 1.3452e-04,  7.7250e-05,  6.1974e-05, -4.4475e-05, -6.8834e-05],
  [ 2.5203e-04,  1.1631e-04,  6.8742e-05,  8.8824e-07, -1.7065e-05],
  [ 3.8450e-04,  2.1357e-04,  5.0101e-05,  2.1878e-05,  4.2402e-05],
  [ 4.1668e-04,  1.8132e-04, -2.1882e-05, -6.4975e-05, -5.0740e-05],
  [ 3.4382e-04,  1.7486e-04,  6.7651e-06, -1.3207e-04, -1.5914e-04]],

 [[-6.1171e-05, -6.1433e-05, -6.7133e-05, -1.2811e-04, -1.4227e-04],
  [ 2.7644e-05, -2.9583e-05, -2.4741e-05, -8.4066e-05, -8.3117e-05],
  [ 1.1662e-04,  2.7068e-05, -2.1483e-05, -4.5343e-05, -1.9267e-05],
  [ 1.6013e-04, -1.4113e-05, -7.1297e-05, -1.0344e-04, -7.5444e-05],
  [ 1.2892e-04,  4.1891e-06, -9.4963e-05, -1.5993e-04, -1.4731e-04]],

 [[-4.8182e-05, -5.8541e-05, -3.9667e-05, -1.2983e-04, -1.5773e-04],
  [ 3.2108e-05, -2.5558e-05, -4.2475e-05, -1.0379e-04, -8.6528e-05],
  [ 1.5525e-04,  2.0429e-05, -3.0547e-05, -7.1051e-05, -4.4240e-05],
  [ 1.5398e-04, -3.8944e-05, -1.1335e-04, -1.3316e-04, -1.3195e-04],
  [ 1.4417e-04, -4.2796e-05, -1.1918e-04, -2.0157e-04, -2.1144e-04]]],
```

```
        [[[ 1.2240e-04,  1.2144e-04,  2.1265e-04,  3.1731e-04,  2.9496e-04],
          [ 5.2139e-05,  2.0082e-05,  1.2365e-04,  1.1998e-04,  9.7836e-05],
          [ 1.2869e-04,  1.0183e-04,  1.1336e-04,  1.1550e-04,  1.1162e-04],
          [-1.2138e-04, -1.3412e-04, -8.1210e-05,  7.7633e-05,  1.9712e-04],
          [-1.2039e-04, -8.5155e-05,  4.9659e-05,  2.0871e-04,  2.7973e-04]],

         [[ 8.5405e-05, -4.5195e-05,  1.6283e-04, -1.5737e-04, -5.7022e-04],
          [-3.9705e-05, -1.1276e-04, -6.5279e-05, -2.1818e-04, -4.7590e-04],
          [-1.3574e-04, -3.5116e-04, -2.9897e-04, -3.8501e-04, -3.7009e-04],
          [-2.2734e-04, -3.9933e-04, -6.5731e-04, -7.0041e-04, -6.5979e-04],
          [-3.9559e-04, -8.5839e-04, -8.4888e-04, -8.0289e-04, -8.9346e-04]],

         [[-5.1153e-04, -4.8006e-04, -3.6159e-04, -2.9405e-04, -3.4677e-04],
          [-3.8135e-04, -3.9410e-04, -2.9927e-04, -1.1276e-04, -9.1053e-05],
          [-4.2010e-04, -3.7397e-04, -2.2372e-04, -3.8051e-05,  9.1124e-05],
          [-2.0540e-04, -6.1916e-05, -1.1039e-05,  5.4395e-05,  3.8999e-05],
          [-1.6560e-04, -3.0045e-05,  5.0225e-05,  1.0669e-04,  6.8221e-05]],

         …,

         [[-3.1946e-04, -2.1983e-04, -2.1590e-04,  5.9937e-05,  1.1959e-04],
          [-1.6346e-04, -2.0734e-04, -2.5633e-05,  2.5879e-04,  3.9676e-04],
          [-1.5874e-04, -1.4659e-04,  1.2140e-04,  2.7852e-04,  4.9640e-04],
          [-1.9628e-04, -1.1964e-05,  2.8757e-04,  3.3028e-04,  4.0256e-04],
          [-2.2946e-04, -3.6470e-05,  2.5026e-04,  3.5758e-04,  3.4126e-04]],

         [[-8.6239e-05,  6.1082e-06,  3.5249e-05,  2.4362e-04,  2.6460e-04],
          [-2.4150e-05,  3.0618e-05,  2.1534e-04,  3.3393e-04,  4.0139e-04],
          [ 1.7180e-05,  8.9938e-05,  2.3735e-04,  3.2359e-04,  4.3190e-04],
          [-1.2400e-04,  9.1822e-05,  2.1067e-04,  2.4951e-04,  3.1893e-04],
          [-1.7105e-04,  1.3064e-05,  2.0681e-04,  2.6584e-04,  2.7440e-04]],

         [[-4.5972e-05,  2.9723e-05,  6.7356e-05,  2.8248e-04,  3.9594e-04],
          [ 1.3272e-05,  1.8617e-05,  2.3512e-04,  3.7262e-04,  4.8094e-04],
          [ 3.3934e-05,  5.5310e-05,  2.5720e-04,  3.3955e-04,  4.4855e-04],
          [-1.2583e-04,  1.0359e-05,  2.1097e-04,  2.5779e-04,  3.2423e-04],
          [-1.7291e-04, -2.8172e-05,  1.9799e-04,  2.7389e-04,  2.4303e-04]]]])
tensor([-6.1646e-10, -2.7296e-10,  1.3574e-10,  3.6243e-10, -2.8891e-11,
         2.1126e-09, -1.6641e-09,  1.9035e-09,  1.0517e-10,  7.0077e-10,
        -3.0412e-09, -3.8641e-09,  1.0886e-09, -5.0818e-10, -3.0423e-10,
        -7.7989e-11,  3.6789e-10,  3.8017e-10,  2.5058e-09, -3.1905e-09,
         2.3954e-10, -2.5849e-10,  3.0059e-10,  2.4079e-10, -8.9130e-11,
        -7.7966e-10,  2.7608e-10,  2.9642e-09, -1.2241e-09,  3.3333e-10,
        -7.4829e-10,  4.9042e-09,  9.7634e-10,  1.2783e-09,  3.1628e-10,
         3.1256e-10, -2.4949e-10, -1.8900e-09,  1.3704e-09,  5.6696e-10,
        -5.0368e-10,  1.2841e-09,  5.2432e-10,  2.1316e-10,  5.3024e-10,
         9.3302e-10,  1.9231e-09, -1.4336e-09, -2.0974e-09, -3.2587e-09,
```

```
        -9.9953e-10,  9.0859e-10, -1.3042e-09, -5.1355e-10,  2.4428e-10,
         4.3556e-10, -5.9674e-10,  8.7834e-10,  9.9085e-11,  8.8312e-10,
         1.5107e-09, -5.9470e-10,  1.1310e-09, -1.2460e-10, -2.3482e-10,
        -1.3379e-09, -1.5419e-09, -3.8705e-10,  3.4498e-09, -2.3579e-10,
        -1.0334e-09, -8.6129e-10,  2.1674e-10,  5.2000e-10, -4.6630e-10,
        -1.7543e-09,  1.1134e-09,  1.2226e-09, -6.5734e-10, -8.3242e-10,
        -3.1973e-09,  4.8634e-09,  1.4831e-09,  2.4461e-09,  1.0264e-09,
         1.4098e-09, -6.6325e-09, -1.5768e-10, -2.8739e-10,  3.8040e-10,
        -4.1569e-10,  1.2595e-09,  1.2781e-10,  2.4075e-09, -4.9710e-09,
        -4.1734e-09, -7.6852e-11,  7.0986e-10, -4.1314e-10, -4.2990e-10,
        -2.1913e-11, -5.7776e-10, -5.1932e-10,  3.7306e-10, -4.1211e-09,
        -1.9901e-10, -3.6562e-10,  9.7543e-10,  4.8714e-10,  2.1398e-09,
         9.1131e-10,  4.3610e-10,  1.1005e-10, -7.6943e-10,  9.3678e-10,
        -2.6284e-10,  4.9317e-10, -7.9888e-10, -1.2703e-09, -1.0507e-09,
        -3.4062e-10, -1.3197e-09, -1.3838e-09,  1.5988e-09,  1.2733e-10,
        -2.4515e-09, -2.9582e-09, -2.4613e-09, -5.4723e-10, -2.8394e-09,
        -3.6152e-10, -9.7771e-10,  2.8353e-10,  1.8716e-10,  2.3774e-09,
        -6.2494e-10,  1.1937e-11, -3.6096e-09, -4.3929e-10,  2.8203e-09,
         1.3588e-09,  5.7144e-10, -2.1316e-09,  1.5944e-09,  6.0861e-10,
         6.2801e-10, -2.4031e-09,  6.0101e-10, -6.8093e-10,  1.7001e-09,
        -8.7812e-10, -1.4394e-10,  8.2061e-10, -4.7228e-09,  1.3933e-09,
         3.1309e-10, -1.6929e-09,  1.8243e-09,  3.8640e-09, -3.3810e-10,
         1.0835e-09, -1.2960e-11, -1.9411e-09,  1.5147e-09,  1.0318e-09,
         8.3166e-10, -2.6009e-09, -3.7658e-09,  1.2803e-09,  1.1146e-09,
        -4.2974e-10, -1.6625e-10,  1.7735e-10,  1.3247e-09, -1.7868e-09,
         4.9693e-10, -2.2158e-10,  1.3545e-09,  1.0164e-10, -1.2669e-09,
         1.2378e-09, -6.6237e-11, -9.6293e-10,  1.8752e-09,  1.2287e-09,
        -4.3747e-10,  6.4799e-10,  6.6620e-10,  6.0582e-10, -1.8194e-09,
         8.0683e-10,  5.5394e-10,  1.8770e-10,  5.6720e-10, -2.2438e-09,
        -4.5611e-10,  7.0372e-10,  6.1073e-10,  5.4925e-12,  5.7457e-10,
        -4.5961e-10,  1.5975e-09,  1.8268e-09, -8.3946e-10, -9.1086e-10,
         1.6705e-09,  1.5575e-09,  7.2447e-10,  1.8297e-09, -8.4896e-10,
        -1.3908e-09,  8.2252e-10,  4.1690e-10, -5.7499e-10, -1.0981e-10,
         5.2614e-10, -6.2059e-09, -1.0143e-10, -3.2194e-09,  1.7776e-10,
        -3.5679e-09, -4.3307e-09,  3.5621e-10, -2.2292e-09, -1.0206e-09,
         1.5679e-09, -6.9548e-10,  1.7693e-10,  7.0577e-10,  1.7961e-09,
        -1.6263e-09, -5.9266e-11,  3.4483e-09, -3.2495e-10,  1.3466e-10,
         3.9679e-10, -6.9710e-10,  2.3327e-09, -3.7173e-10, -1.0925e-09,
        -5.9845e-10, -2.0767e-09, -1.5343e-10,  1.6740e-11,  8.7219e-10,
        -5.4841e-10,  8.1821e-10,  2.2692e-09, -1.0326e-09, -2.7670e-09,
        -5.3396e-09, -6.5029e-10,  2.1066e-10,  1.1583e-10,  3.2060e-11,
        -5.7275e-10])
tensor([-2.5776e-03,  4.6746e-04, -4.8659e-03, -1.6277e-03,  1.6341e-03,
        -3.5718e-03, -8.6248e-03, -9.3682e-03, -9.4200e-04,  3.8827e-03,
         2.6970e-02,  8.6785e-03,  7.8650e-03,  6.6036e-03, -5.1930e-03,
         1.6656e-03, -2.5625e-03,  5.8394e-03,  1.2211e-02,  7.7886e-03,
        -2.5591e-03, -4.1722e-04,  4.4011e-04,  4.8210e-04,  2.5869e-03,
         3.2069e-03,  2.7563e-03,  9.2521e-03,  4.2460e-03,  4.0042e-03,
```

```
        -1.4979e-03,  1.0313e-02,  2.2411e-03,  1.3552e-02, -1.4844e-03,
        -4.6890e-04, -1.4181e-03,  1.2537e-03,  7.0504e-03,  1.2948e-04,
         3.7792e-03, -1.0532e-03,  8.4365e-04, -4.2080e-03, -2.6012e-03,
        -3.5559e-03, -3.4213e-03, -1.2518e-02,  4.4784e-03, -1.2906e-03,
         4.6797e-03, -6.6078e-03, -8.8801e-03, -8.3176e-03, -4.6825e-03,
         2.4309e-03, -4.1264e-03, -5.0668e-03,  1.0698e-03,  4.0210e-03,
         4.4714e-03, -7.6317e-04, -5.7261e-03, -2.2266e-03, -2.1704e-03,
        -3.3872e-03, -3.7313e-03,  1.3507e-03,  2.0307e-04, -4.0421e-03,
        -2.1555e-03, -7.3432e-03, -8.4126e-03, -7.1216e-03,  1.8771e-02,
        -9.3668e-03, -2.3965e-04, -2.5204e-03,  9.8738e-04,  1.9630e-03,
        -4.4360e-03, -9.7398e-03,  4.8471e-03,  2.7241e-03,  7.9489e-04,
        -6.5342e-03,  4.1628e-03,  1.7535e-04, -3.3791e-03,  4.2502e-03,
         3.3086e-03,  5.0574e-03,  3.8526e-03,  1.0617e-02,  7.0960e-03,
         1.2174e-02,  3.7288e-04, -1.2609e-03,  3.2090e-03,  2.3770e-03,
         4.8316e-03, -4.8960e-03,  6.5491e-03, -1.1992e-03,  4.9876e-03,
        -6.7114e-03, -1.0684e-03, -1.7774e-03,  2.3057e-03, -4.3009e-03,
        -4.1487e-03, -7.0132e-04,  2.9431e-03, -3.4864e-04, -2.3758e-03,
        -6.2823e-04, -1.0152e-03, -1.8373e-03,  1.1992e-03, -4.6009e-03,
         4.5067e-03, -5.2896e-03, -5.6163e-03, -4.8853e-03, -4.1586e-04,
        -5.2068e-04,  8.3416e-03,  2.2189e-03,  3.7729e-04, -1.2100e-02,
        -2.3830e-03,  1.9880e-03, -2.7578e-03, -2.4743e-03,  9.0414e-03,
        -7.6587e-04,  1.7162e-02, -5.5383e-03, -1.4307e-03, -9.4376e-03,
        -1.3345e-03, -4.7613e-03, -6.4023e-04, -8.0825e-03,  2.2091e-03,
         3.8881e-03,  1.5757e-02,  1.9317e-02,  1.2601e-02, -1.9611e-03,
        -4.7756e-03, -2.6574e-03,  1.7043e-03, -3.2060e-03,  1.3652e-02,
        -1.6844e-03,  1.3900e-03, -2.0210e-03, -7.0505e-03, -2.6900e-03,
         8.5993e-03,  5.4778e-04, -1.5985e-03, -6.0169e-03, -5.1887e-03,
        -1.8017e-03,  5.3277e-03, -6.8540e-03, -1.7559e-04,  3.1238e-03,
        -9.6166e-03, -2.8639e-03,  4.3359e-03,  1.9885e-02, -2.4030e-03,
         2.8496e-03, -1.7820e-03, -1.3773e-03,  3.7193e-03, -3.1241e-03,
         2.4734e-03,  1.1147e-03,  7.5769e-04,  5.1845e-03,  2.2253e-03,
        -1.3338e-03,  1.5268e-02,  1.7381e-03, -1.6138e-03,  8.5208e-03,
        -7.9827e-05,  8.3849e-03,  6.2192e-03,  6.6985e-03, -6.4368e-03,
        -9.7851e-04,  1.7410e-03,  2.1408e-04,  2.7712e-03,  1.8421e-03,
         2.6148e-03,  3.1654e-03, -4.1460e-03,  8.8338e-04, -1.4236e-03,
        -6.0960e-04,  2.6976e-03, -1.5414e-03, -7.7171e-03, -1.4097e-03,
        -2.5457e-03,  2.3551e-03, -6.1120e-03,  4.2371e-03, -1.5177e-03,
        -5.9404e-03, -1.5205e-03,  3.3471e-03,  6.1341e-03, -2.8803e-03,
         6.3540e-03, -6.2744e-03,  1.6646e-02,  1.2156e-02, -1.2879e-03,
         5.8893e-04, -3.0067e-03,  4.4580e-03, -4.6424e-03,  1.8443e-02,
         1.8947e-03, -1.7761e-03,  2.6803e-03,  1.9712e-03, -4.8216e-03,
         8.0682e-03,  1.5599e-03,  4.8990e-03,  5.2738e-04,  1.8892e-03,
        -7.2168e-03, -6.4261e-03,  9.6659e-03, -2.1031e-03,  3.1065e-03,
         9.9808e-04,  4.1706e-03, -1.8452e-04, -5.7180e-03,  3.8495e-06,
         3.1345e-03, -2.8175e-03, -2.4004e-04, -3.2073e-03, -6.1741e-03,
        -1.0883e-03])
tensor([-1.8697e-03,  9.1653e-04, -4.9514e-03, -1.7538e-03,  2.9593e-03,
        -6.3389e-03, -9.2908e-03, -1.5690e-02, -2.3762e-03,  6.9359e-03,
```

```
 1.2614e-02,  1.0115e-02,  6.3901e-03,  5.8022e-03, -4.7087e-03,
-2.7379e-04, -1.5009e-03,  7.4059e-03,  1.1128e-02,  8.5964e-03,
-2.6091e-03, -1.2101e-03,  1.7850e-03,  4.6626e-04,  3.0909e-03,
 4.0230e-03,  2.3593e-03,  5.8232e-03,  5.5753e-03,  4.6485e-03,
-1.9271e-03,  1.0006e-02,  3.2550e-03,  1.5555e-02, -1.9121e-03,
 1.1548e-03, -1.6447e-03,  2.9804e-03,  3.5028e-03, -1.6475e-03,
 4.5898e-03, -1.1048e-03,  1.1947e-04, -2.7415e-03, -3.2504e-03,
-3.6534e-03, -3.9604e-03, -9.2403e-03,  5.6006e-03,  1.2745e-03,
 9.1320e-03, -5.1435e-03, -1.0667e-02, -1.0941e-02, -1.0745e-02,
 3.4092e-03, -4.1902e-03, -6.1678e-03,  9.3266e-05,  4.9149e-03,
 4.7787e-03, -2.5416e-03, -6.0952e-03, -2.7120e-03, -2.1403e-03,
-2.4448e-03, -3.1213e-03,  2.3026e-03,  3.7925e-03, -5.0538e-03,
-4.1244e-03, -8.4860e-03, -9.8925e-03, -8.0247e-03,  7.7722e-03,
-3.9699e-03,  1.5710e-03, -2.8190e-03,  2.9864e-03,  2.6429e-03,
-8.1444e-03, -1.4159e-02,  5.1083e-03,  4.6816e-03,  1.4071e-04,
-6.0326e-03,  4.8899e-03,  7.3242e-04, -5.0923e-03,  4.3665e-03,
 2.8161e-03,  3.8910e-03,  3.8739e-03,  4.5391e-04,  1.3298e-02,
 3.2847e-03,  2.0972e-04, -2.2093e-04,  1.4395e-03,  1.7095e-03,
 3.7368e-03, -4.9194e-03,  8.8172e-03,  5.2288e-04,  2.6414e-03,
-9.8347e-03, -7.2975e-04, -2.1503e-03, -1.5885e-03, -8.2744e-03,
-1.6107e-03, -9.4696e-04,  3.1557e-03,  2.6291e-03, -2.7793e-03,
-1.2012e-03, -2.2297e-04, -1.8577e-03,  1.7268e-03, -4.1096e-03,
 4.2546e-03, -5.7045e-03, -5.4519e-03, -7.1204e-03,  4.5367e-04,
 7.7435e-04,  3.7984e-03,  3.0191e-03, -3.2862e-04, -1.8538e-02,
-2.3336e-03,  4.6354e-03, -5.3777e-04, -2.0615e-03,  8.0812e-03,
-2.5828e-03,  2.5562e-03, -8.9200e-03, -2.3512e-03, -9.2051e-03,
-1.4798e-03, -4.5156e-03,  3.3719e-03, -6.8362e-03,  2.4081e-03,
 4.8098e-03,  8.1897e-04,  1.6629e-02,  7.4155e-03,  2.6429e-04,
-4.5221e-03, -1.1180e-03,  1.0321e-03, -2.3328e-03,  7.6357e-03,
-3.6069e-03,  3.4701e-03, -5.5847e-04, -1.0753e-02, -2.0139e-03,
 4.1968e-03,  5.3291e-04,  1.7256e-03, -6.3610e-03, -5.0903e-03,
-5.1965e-03,  6.8328e-03, -6.2659e-03,  6.3527e-04,  7.8126e-04,
-1.0721e-02, -3.0606e-03,  6.6814e-03,  1.4898e-02, -2.2244e-03,
 5.9664e-03, -2.1441e-03, -1.4797e-03,  4.3303e-03, -2.9500e-03,
 3.9283e-03, -6.6456e-04, -6.1184e-04,  5.8023e-03,  1.8906e-03,
-5.0010e-04,  5.8345e-03,  2.1843e-03,  7.3056e-04,  1.2932e-02,
-2.4571e-04,  6.1439e-03,  6.0884e-03,  3.6550e-03, -5.5293e-03,
-1.0944e-03,  2.2762e-03, -2.1927e-05,  2.7498e-03,  3.3659e-03,
 5.2644e-03,  4.9052e-03, -6.7363e-03,  1.6773e-03, -5.3695e-04,
 6.5254e-04,  5.3083e-03, -4.0399e-03, -5.7703e-03, -1.0737e-03,
-2.7293e-03,  2.3431e-03, -3.8975e-03,  5.7293e-03, -2.1399e-03,
-4.9314e-03,  1.2234e-03, -1.2435e-03,  6.1943e-03, -3.6075e-04,
 6.6703e-03, -7.7635e-03,  1.1024e-02,  1.0377e-02, -2.3910e-03,
 1.9991e-03, -3.4080e-03,  5.9004e-03, -3.9644e-03,  1.0013e-02,
-1.5116e-03, -1.3249e-03,  2.4762e-03,  9.4518e-05, -4.2805e-03,
 1.4816e-03, -1.8465e-03,  4.6232e-03, -3.0750e-03,  1.7619e-03,
-7.8841e-03, -6.3075e-03,  6.4033e-03, -3.1482e-03,  2.6566e-03,
-8.9319e-04,  6.5180e-03, -1.3512e-04, -5.4124e-03,  2.4210e-03,
```

```
         6.4726e-03, -2.8879e-03,  1.5507e-03, -3.2230e-03, -6.8332e-03,
        -1.9218e-04])
tensor([[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  …,

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,


[[[ 1.3009e-02,  8.7709e-03,  3.7459e-03],
   [ 8.7660e-03,  4.8083e-03,  1.4789e-03],
   [ 8.6171e-03,  6.0993e-03,  3.0718e-03]],

  [[-4.4172e-04, -2.7067e-03, -2.2347e-03],
   [ 9.7517e-04, -2.2630e-03, -2.5792e-03],
   [-1.6521e-04, -4.1454e-03, -3.3992e-03]],

  [[ 4.3811e-04,  2.4118e-03,  1.1492e-03],
   [-9.0869e-04,  1.6457e-03,  2.8890e-03],
```

```
  [-3.2353e-03,  6.5730e-04,  2.4718e-03]],

 …,

 [[ 2.7132e-02,  2.0954e-02,  1.2671e-02],
  [ 2.5305e-02,  1.9699e-02,  1.1858e-02],
  [ 2.5141e-02,  2.3043e-02,  1.4550e-02]],

 [[-1.3251e-03, -8.4047e-03, -8.2254e-03],
  [ 9.5996e-04, -7.3452e-03, -8.2259e-03],
  [ 9.9289e-06, -6.2842e-03, -6.1410e-03]],

 [[-2.3117e-03,  3.3361e-03,  4.7953e-03],
  [-5.2920e-04,  6.8673e-03,  9.6408e-03],
  [-2.6406e-04,  7.5044e-03,  8.9470e-03]]],


[[[ 1.1792e-02,  1.4667e-02,  1.7012e-02],
  [ 7.8975e-03,  7.7979e-03,  7.3741e-03],
  [ 8.2298e-03,  6.8513e-03,  4.7462e-03]],

 [[-9.3193e-03, -9.8423e-03, -1.1545e-02],
  [-1.2839e-02, -1.0646e-02, -9.0065e-03],
  [-1.2050e-02, -8.8736e-03, -4.9475e-03]],

 [[-1.1715e-02, -1.1504e-02, -8.5520e-03],
  [-1.8479e-02, -1.6512e-02, -1.0928e-02],
  [-6.0588e-03, -2.7089e-03,  8.4614e-04]],

 …,

 [[ 1.7445e-02,  1.9700e-02,  2.3363e-02],
  [ 1.5280e-02,  1.4282e-02,  1.1565e-02],
  [ 1.0792e-02,  6.0934e-03,  1.8347e-03]],

 [[-7.9685e-03, -1.0020e-02, -1.1688e-02],
  [-1.2161e-02, -1.2137e-02, -1.0886e-02],
  [-1.2042e-02, -1.0903e-02, -8.4785e-03]],

 [[-1.3426e-02, -1.0219e-02, -7.4783e-03],
  [-1.8671e-02, -1.4039e-02, -9.4112e-03],
  [-9.2838e-03, -9.9562e-04,  2.2636e-03]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]])
tensor([ 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -3.7811e-04,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  9.7928e-05,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.4137e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  1.6178e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  6.2008e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  5.4456e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -5.1845e-03,  0.0000e+00, -1.8318e-03, -9.5388e-06,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -3.7224e-04,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  1.3967e-02,
```

```
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  3.1613e-02,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00, -5.5211e-04,  0.0000e+00,
 0.0000e+00, -6.6245e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 9.7755e-05,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
-1.8236e-05,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
-3.8972e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
-4.1388e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00, -3.7803e-03,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  1.0324e-03,  0.0000e+00,
 0.0000e+00,  0.0000e+00, -1.9906e-03,  0.0000e+00, -2.8636e-04,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  1.2246e-03,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00, -4.2784e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  1.3655e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -4.0090e-03,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

```
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  1.6349e-02, -7.1604e-03,  0.0000e+00])
tensor([[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 4.6718e-03, -3.6317e-03, -4.8242e-03],
          [ 1.0410e-02, -3.0947e-03, -6.3942e-03],
          [ 9.0960e-03, -5.7201e-03, -1.1506e-02]],

         [[ 1.8919e-02,  7.3809e-03,  8.5891e-04],
          [-3.2180e-03, -1.2487e-02, -9.2375e-03],
          [-8.8650e-03, -2.0362e-02, -1.1763e-02]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 1.2490e-02,  5.5379e-02,  7.9576e-02],
          [ 1.9741e-02,  6.3659e-02,  9.3632e-02],
          [ 1.7835e-02,  5.5790e-02,  8.4135e-02]],

         [[ 3.2309e-02,  2.6407e-02,  1.8573e-02],
```

```
   [ 2.3921e-02,  1.7285e-02,  1.2862e-02],
   [ 2.0194e-02,  1.5549e-02,  1.5830e-02]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  …,

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  1.9623e-04,  1.9446e-04],
  [ 0.0000e+00,  1.6231e-04,  1.3572e-04],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 1.2521e-05,  5.8998e-06, -6.3330e-05],
           [ 1.0524e-05,  2.2274e-05, -4.6598e-05]],

          [[ 0.0000e+00, -7.4834e-05, -2.0131e-04],
           [ 1.0287e-04,  4.0657e-05, -5.4797e-06],
           [ 1.3807e-05, -3.9222e-05,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([-3.5784e-02,  1.7517e-02,  0.0000e+00, -3.4298e-03, -2.0663e-02,
        -1.5892e-02, -1.4944e-03, -1.6923e-03, -1.2437e-02,  3.4697e-05,
         0.0000e+00,  0.0000e+00,  1.1576e-02,  0.0000e+00,  1.4404e-04,
         1.2713e-04,  1.3597e-05,  0.0000e+00,  0.0000e+00,  1.1906e-04,
        -4.1819e-04,  5.4936e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -6.6001e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,  1.3882e-04,
        -2.3011e-02, -2.5209e-03, -3.1362e-04,  0.0000e+00, -4.8653e-04,
        -6.5833e-04, -1.1942e-03,  1.9913e-02,  7.0845e-04,  1.2237e-02,
        -1.9795e-03,  3.9350e-03, -4.8556e-06, -6.0075e-04,  0.0000e+00,
        -2.4406e-03, -5.7482e-04, -5.6751e-05,  7.8307e-03, -3.1745e-03,
         0.0000e+00,  2.3840e-02, -2.6269e-03,  0.0000e+00,  8.1421e-03,
        -7.8694e-05,  1.5778e-03,  0.0000e+00, -2.2593e-04,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -2.0171e-04, -8.2862e-04, -1.3522e-02,
        -9.7561e-03,  1.7797e-02, -3.8412e-03,  1.2025e-02,  4.9746e-03,
         4.7867e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,  2.2528e-02,
         0.0000e+00,  0.0000e+00,  2.6874e-05,  7.0544e-05, -3.4418e-04,
         3.1609e-05,  0.0000e+00,  1.1193e-03,  0.0000e+00,  9.5286e-05,
         0.0000e+00,  5.7087e-03, -4.5232e-03, -5.7958e-03,  0.0000e+00,
        -1.0849e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00, -6.1041e-04,
         0.0000e+00,  3.0890e-04, -1.4908e-02,  0.0000e+00, -2.1755e-03,
        -4.3281e-02, -2.0645e-03, -3.6489e-03,  1.1766e-03,  0.0000e+00,
        -5.2713e-03, -3.4733e-02, -5.8674e-05,  1.8358e-04,  7.1904e-03,
        -1.0375e-05,  1.4016e-03,  0.0000e+00, -9.0290e-04,  2.2983e-02,
         3.4355e-03, -6.0318e-04, -2.4704e-04,  0.0000e+00,  5.0000e-03,
         2.7816e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00, -6.8123e-04,
         0.0000e+00,  1.3170e-04,  0.0000e+00, -4.4236e-04,  2.1835e-03,
         5.1412e-05, -2.1524e-03, -2.1513e-03,  1.3618e-02,  3.9940e-05,
```

```
-2.0239e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  1.2515e-02,
 5.5054e-05, -5.0232e-02, -2.3025e-02,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00, -2.1800e-02, -2.2891e-06, -6.9848e-03,
 0.0000e+00, -3.2751e-05, -7.6977e-04, -1.1497e-03,  0.0000e+00,
 7.6472e-02,  0.0000e+00,  1.7898e-04,  9.9001e-04,  4.2088e-03,
 0.0000e+00, -2.4569e-02,  0.0000e+00,  7.9241e-04,  2.3914e-03,
-1.6399e-04,  1.4149e-04,  2.2778e-03, -1.6071e-04,  4.0609e-02,
-2.9733e-03,  2.4834e-03,  0.0000e+00,  1.4606e-04,  2.5920e-02,
 9.7647e-03,  0.0000e+00, -2.7158e-02, -3.1484e-03,  3.0777e-02,
 8.3060e-04, -1.9662e-02, -3.8072e-04,  1.5393e-05,  4.3374e-03,
 0.0000e+00, -1.2341e-04, -3.9672e-04, -3.7225e-04,  4.6993e-03,
-2.4704e-04,  0.0000e+00, -1.6112e-03,  8.0897e-04, -5.2616e-04,
 0.0000e+00, -1.5460e-03,  2.8533e-03, -2.5950e-04,  0.0000e+00,
-1.9808e-02,  6.8652e-02,  3.7246e-03, -6.1132e-06,  0.0000e+00,
-5.7335e-05,  3.0279e-04, -1.0080e-03, -1.9315e-03, -4.1191e-04,
 5.1347e-03, -7.4003e-04,  2.6855e-03, -8.4881e-05,  2.9022e-03,
-3.6736e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00, -1.2320e-02,  8.7137e-04,
-3.5936e-04, -7.8706e-03,  1.7649e-02, -9.5470e-05,  0.0000e+00,
-2.2319e-04, -6.2181e-05,  0.0000e+00,  2.4394e-03, -5.6872e-04,
 3.9628e-03, -9.4498e-04,  0.0000e+00,  8.6902e-04,  6.1183e-03,
 0.0000e+00, -9.3880e-03,  0.0000e+00,  1.8859e-03,  1.9006e-02,
 4.9100e-04, -3.1882e-03,  0.0000e+00,  0.0000e+00, -3.6555e-05,
 9.9351e-04,  2.2183e-02, -1.5043e-03,  0.0000e+00,  0.0000e+00,
-1.1357e-04,  0.0000e+00, -5.7484e-04,  1.8561e-02,  2.5606e-04,
 0.0000e+00,  2.5472e-03, -1.6057e-02, -6.3867e-04,  0.0000e+00,
 0.0000e+00,  1.0496e-01, -2.1955e-02,  0.0000e+00, -2.9501e-03,
 0.0000e+00,  0.0000e+00,  1.1495e-04,  0.0000e+00, -9.5050e-02,
 0.0000e+00, -2.9931e-03,  1.1603e-04,  0.0000e+00, -1.7020e-03,
 1.4873e-03,  5.8404e-04,  0.0000e+00,  4.4173e-02, -8.8478e-04,
 0.0000e+00, -1.0363e-03,  0.0000e+00,  0.0000e+00,  6.3054e-04,
 1.3219e-03,  0.0000e+00,  1.5785e-04, -2.6750e-03, -3.6060e-04,
 9.0409e-04, -2.3594e-03,  0.0000e+00,  1.8639e-06,  6.8333e-03,
-1.2422e-02,  2.9073e-03, -1.5932e-03, -1.4780e-02,  0.0000e+00,
-3.6643e-02,  1.2038e-03,  0.0000e+00,  1.1341e-02,  1.5636e-04,
 1.8198e-03, -1.8422e-04, -2.9245e-04, -3.6361e-03, -8.0989e-05,
 0.0000e+00,  2.2345e-04,  0.0000e+00,  0.0000e+00,  2.7421e-04,
 0.0000e+00,  0.0000e+00,  1.6509e-04, -1.3429e-02,  9.9302e-05,
-1.6642e-04,  0.0000e+00,  0.0000e+00, -1.0148e-03, -5.4653e-05,
-1.7512e-04,  4.0876e-02,  9.1328e-03,  0.0000e+00, -3.8030e-02,
-1.4710e-05,  0.0000e+00,  0.0000e+00,  4.0111e-02, -8.1469e-03,
 6.4778e-03, -2.9701e-02,  0.0000e+00, -1.4340e-02,  1.0564e-02,
 2.7181e-02, -4.3290e-03,  0.0000e+00,  0.0000e+00, -4.8877e-04,
 0.0000e+00, -1.8068e-04,  0.0000e+00,  0.0000e+00, -1.2856e-04,
-2.5240e-03, -5.0924e-06,  0.0000e+00, -2.4700e-04,  0.0000e+00,
 7.5964e-04,  2.2536e-03,  3.7948e-02,  0.0000e+00,  0.0000e+00,
-6.5066e-03, -1.1365e-03,  2.4302e-04, -2.2545e-03,  1.8496e-05,
-1.5240e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00, -3.4832e-03,
```

```
         1.2532e-04, -3.7169e-03, -2.2399e-03,  0.0000e+00, -3.4282e-05,
         2.5132e-04,  1.8578e-04,  0.0000e+00, -2.0174e-04])
tensor([[[[0., 0., 0.],
          [0., 0., 0.],
          [0., 0., 0.]],

         [[0., 0., 0.],
          [0., 0., 0.],
          [0., 0., 0.]],

         [[0., 0., 0.],
          [0., 0., 0.],
          [0., 0., 0.]],

         …,

         [[0., 0., 0.],
          [0., 0., 0.],
          [0., 0., 0.]],

         [[0., 0., 0.],
          [0., 0., 0.],
          [0., 0., 0.]],

         [[0., 0., 0.],
          [0., 0., 0.],
          [0., 0., 0.]]],


        [[[0., 0., 0.],
          [0., 0., 0.],
          [0., 0., 0.]],

         [[0., 0., 0.],
          [0., 0., 0.],
          [0., 0., 0.]],

         [[0., 0., 0.],
          [0., 0., 0.],
          [0., 0., 0.]],

         …,

         [[0., 0., 0.],
          [0., 0., 0.],
          [0., 0., 0.]],

         [[0., 0., 0.],
```

```
   [0., 0., 0.],
   [0., 0., 0.]],

  [[0., 0., 0.],
   [0., 0., 0.],
   [0., 0., 0.]]],


[[[0., 0., 0.],
   [0., 0., 0.],
   [0., 0., 0.]],

  [[0., 0., 0.],
   [0., 0., 0.],
   [0., 0., 0.]],

  [[0., 0., 0.],
   [0., 0., 0.],
   [0., 0., 0.]],

 …,

  [[0., 0., 0.],
   [0., 0., 0.],
   [0., 0., 0.]],

  [[0., 0., 0.],
   [0., 0., 0.],
   [0., 0., 0.]],

  [[0., 0., 0.],
   [0., 0., 0.],
   [0., 0., 0.]]],


…,


[[[0., 0., 0.],
   [0., 0., 0.],
   [0., 0., 0.]],

  [[0., 0., 0.],
   [0., 0., 0.],
   [0., 0., 0.]],

  [[0., 0., 0.],
   [0., 0., 0.],
```

```
  [0., 0., 0.]],

 …,

 [[0., 0., 0.],
  [0., 0., 0.],
  [0., 0., 0.]],

 [[0., 0., 0.],
  [0., 0., 0.],
  [0., 0., 0.]],

 [[0., 0., 0.],
  [0., 0., 0.],
  [0., 0., 0.]]],


[[[0., 0., 0.],
  [0., 0., 0.],
  [0., 0., 0.]],

 [[0., 0., 0.],
  [0., 0., 0.],
  [0., 0., 0.]],

 [[0., 0., 0.],
  [0., 0., 0.],
  [0., 0., 0.]],

 …,

 [[0., 0., 0.],
  [0., 0., 0.],
  [0., 0., 0.]],

 [[0., 0., 0.],
  [0., 0., 0.],
  [0., 0., 0.]],

 [[0., 0., 0.],
  [0., 0., 0.],
  [0., 0., 0.]]],


[[[0., 0., 0.],
  [0., 0., 0.],
  [0., 0., 0.]],
```

```
          [[0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.]],

          [[0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.]],

          …,

          [[0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.]],

          [[0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.]],

          [[0., 0., 0.],
           [0., 0., 0.],
           [0., 0., 0.]]])
tensor([ 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  2.0324e-01, -4.2485e-02,
        -4.5929e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00, -2.0137e-03,
         0.0000e+00,  2.0787e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         1.9747e-01, -7.0126e-04, -3.3509e-03,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  1.1618e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -5.3895e-01,  0.0000e+00,
         4.4229e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  1.0331e-04,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  2.5296e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  5.0179e-06,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -3.0610e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  1.1091e-02,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  7.6002e-03,
         0.0000e+00,  0.0000e+00, -5.5575e-04,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.9964e-01,
         0.0000e+00,  0.0000e+00, -5.2109e-03,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

```
         0.0000e+00,  0.0000e+00, -1.1339e-03,  1.3507e-02, -4.3648e-03,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  1.5222e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -2.0736e-03,  0.0000e+00,
         0.0000e+00,  7.1113e-02,  0.0000e+00,  0.0000e+00, -4.2841e-05,
         0.0000e+00,  0.0000e+00, -2.8472e-03,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  1.9773e-01,  0.0000e+00,  1.8934e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.2713e-03,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  2.1544e-02,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -3.5943e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -2.4208e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         1.3630e-01,  0.0000e+00,  0.0000e+00,  1.7387e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  1.2898e-01,  0.0000e+00,
         0.0000e+00,  1.0803e-02,  5.6647e-04,  0.0000e+00,  0.0000e+00,
         2.2851e-03,  0.0000e+00, -1.1284e-01,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  3.5612e-02,  0.0000e+00,
         3.6480e-05,  0.0000e+00,  5.7610e-04,  0.0000e+00,  0.0000e+00,
         0.0000e+00])
tensor([[0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        …,
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.]])
tensor([-0.0032,  0.0000,  0.0004,  …,  0.0000,  0.0000, -0.0046])
tensor([[ 0.0000e+00,  0.0000e+00,  3.3484e-05,  …,  0.0000e+00,
          0.0000e+00, -7.9144e-06],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        …,
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00]])
tensor([ 0.0030,  0.0000,  0.0000,  …,  0.0000,  0.0000, -0.0017])
tensor([[ 2.9035e-04,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
```

```
          0.0000e+00, -1.3560e-02],
        [ 1.9053e-06,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  3.2321e-03],
        [ 9.4820e-07,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  1.0271e-05],
        …,
        [ 3.9385e-11,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  2.7845e-09],
        [ 4.0027e-11,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  2.4489e-09],
        [ 4.3306e-11,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  3.2091e-09]])
tensor([-3.9245e-02, -8.2047e-03,  5.2651e-02,  4.3162e-02, -1.6058e-02,
        -1.7880e-02, -1.3595e-01, -2.7035e-02,  5.5422e-02,  5.6681e-03,
         4.4107e-02, -3.5732e-02,  1.5202e-02, -1.9814e-02,  8.9718e-03,
         1.7127e-02,  1.8945e-02, -2.1571e-02,  6.8749e-02, -1.0291e-02,
         8.7174e-06,  1.7208e-06,  1.7550e-06,  1.8161e-06,  1.7243e-06,
         1.8107e-06,  1.8412e-06,  1.7333e-06,  1.8202e-06,  1.8668e-06,
         1.8273e-06,  1.7604e-06,  1.8238e-06,  1.7866e-06,  1.7701e-06,
         1.7575e-06,  1.8036e-06,  1.8710e-06,  1.7723e-06,  1.8320e-06,
         1.8133e-06,  1.7833e-06,  1.8204e-06,  1.7943e-06,  1.7948e-06,
         1.8300e-06,  1.8160e-06,  1.8018e-06,  1.8187e-06,  1.7410e-06,
         1.8504e-06,  1.8243e-06,  1.8782e-06,  1.7411e-06,  1.8323e-06,
         1.8396e-06,  1.8151e-06,  1.8933e-06,  1.7666e-06,  1.8263e-06,
         1.8265e-06,  1.7960e-06,  1.9182e-06,  1.8419e-06,  1.9168e-06,
         1.8316e-06,  1.8384e-06,  1.8562e-06,  1.7239e-06,  1.9109e-06,
         1.8061e-06,  1.8404e-06,  1.7514e-06,  1.8021e-06,  1.8428e-06,
         1.7197e-06,  1.8712e-06,  1.8014e-06,  1.8149e-06,  1.6769e-06,
         1.8058e-06,  1.7987e-06,  1.8010e-06,  1.8439e-06,  1.7988e-06,
         1.7874e-06,  1.8274e-06,  1.8431e-06,  1.8820e-06,  1.7494e-06,
         1.7910e-06,  1.8113e-06,  1.8709e-06,  1.7549e-06,  1.7722e-06,
         1.8031e-06,  1.8250e-06,  1.8134e-06,  1.6569e-06,  1.8467e-06,
         1.8631e-06,  1.7561e-06,  1.8338e-06,  1.7923e-06,  1.8258e-06,
         1.8460e-06,  1.8121e-06,  1.7917e-06,  1.7705e-06,  1.9668e-06,
         1.8210e-06,  1.8506e-06,  1.8306e-06,  1.7244e-06,  1.8279e-06,
         1.7711e-06,  1.8494e-06,  1.7932e-06,  1.8016e-06,  1.7566e-06,
         1.7177e-06,  1.7906e-06,  1.7203e-06,  1.8591e-06,  1.8013e-06,
         1.7725e-06,  1.8158e-06,  1.8129e-06,  1.8606e-06,  1.8107e-06,
         1.7615e-06,  1.7517e-06,  1.7778e-06,  1.8562e-06,  1.8235e-06,
         1.7448e-06,  1.8175e-06,  1.7892e-06,  1.7248e-06,  1.7630e-06,
         1.8493e-06,  1.8361e-06,  1.8442e-06,  1.7687e-06,  1.7376e-06,
         1.8436e-06,  1.7481e-06,  1.8526e-06,  1.7952e-06,  1.8670e-06,
         1.8357e-06,  1.8471e-06,  1.8309e-06,  1.7671e-06,  1.7526e-06,
         1.7671e-06,  1.7491e-06,  1.8518e-06,  1.8400e-06,  1.7980e-06,
         1.8570e-06,  1.7253e-06,  1.7865e-06,  1.8130e-06,  1.8427e-06,
         1.7234e-06,  1.8186e-06,  1.7776e-06,  1.8183e-06,  1.8804e-06,
         1.8428e-06,  1.8681e-06,  1.8725e-06,  1.7413e-06,  1.7991e-06,
         1.8017e-06,  1.8419e-06,  1.7829e-06,  1.8381e-06,  1.8504e-06,
```

```
1.8031e-06,  1.7445e-06,  1.7773e-06,  1.8483e-06,  1.7358e-06,
1.8714e-06,  1.7726e-06,  1.7046e-06,  1.8582e-06,  1.8289e-06,
1.7757e-06,  1.8253e-06,  1.8374e-06,  1.8639e-06,  1.7720e-06,
1.7452e-06,  1.7644e-06,  1.8394e-06,  1.7978e-06,  1.7778e-06,
1.7698e-06,  1.8677e-06,  1.7378e-06,  1.8834e-06,  1.8193e-06,
1.8742e-06,  1.8054e-06,  1.8201e-06,  1.7725e-06,  1.8235e-06,
1.7609e-06,  1.8710e-06,  1.7355e-06,  1.8172e-06,  1.8288e-06,
1.7910e-06,  1.8636e-06,  1.8408e-06,  1.8185e-06,  1.8039e-06,
1.8287e-06,  1.7838e-06,  1.7833e-06,  1.7679e-06,  1.7180e-06,
1.8608e-06,  1.7661e-06,  1.8029e-06,  1.7273e-06,  1.7671e-06,
1.8130e-06,  1.8105e-06,  1.8118e-06,  1.8363e-06,  1.8428e-06,
1.8289e-06,  1.8010e-06,  1.8156e-06,  1.7882e-06,  1.8207e-06,
1.7742e-06,  1.7685e-06,  1.8261e-06,  1.8318e-06,  1.8235e-06,
1.8340e-06,  1.7812e-06,  1.8036e-06,  1.8070e-06,  1.8813e-06,
1.8348e-06,  1.8176e-06,  1.7221e-06,  1.7301e-06,  1.7546e-06,
1.8469e-06,  1.7918e-06,  1.8039e-06,  1.8212e-06,  1.8007e-06,
1.7485e-06,  1.8078e-06,  1.7507e-06,  1.8311e-06,  1.8350e-06,
1.8718e-06,  1.8345e-06,  1.8255e-06,  1.7844e-06,  1.8261e-06,
1.8053e-06,  1.7581e-06,  1.8499e-06,  1.7966e-06,  1.7409e-06,
1.8473e-06,  1.7889e-06,  1.7279e-06,  1.7739e-06,  1.8365e-06,
1.7917e-06,  1.8289e-06,  1.8031e-06,  1.8320e-06,  1.8438e-06,
1.7110e-06,  1.7688e-06,  1.7252e-06,  1.7661e-06,  1.8403e-06,
1.7086e-06,  1.8224e-06,  1.7829e-06,  1.7938e-06,  1.8298e-06,
1.8004e-06,  1.8592e-06,  1.7968e-06,  1.8162e-06,  1.8566e-06,
1.7953e-06,  1.8480e-06,  1.7557e-06,  1.8775e-06,  1.7377e-06,
1.8079e-06,  1.8530e-06,  1.7567e-06,  1.8529e-06,  1.7568e-06,
1.7680e-06,  1.8454e-06,  1.9075e-06,  1.7798e-06,  1.8973e-06,
1.8408e-06,  1.7869e-06,  1.7598e-06,  1.7971e-06,  1.8130e-06,
1.8168e-06,  1.8233e-06,  1.7278e-06,  1.7647e-06,  1.8111e-06,
1.8373e-06,  1.8505e-06,  1.7390e-06,  1.8152e-06,  1.8413e-06,
1.8433e-06,  1.8982e-06,  1.8699e-06,  1.8394e-06,  1.7381e-06,
1.8295e-06,  1.8654e-06,  1.8019e-06,  1.7981e-06,  1.7916e-06,
1.7951e-06,  1.7318e-06,  1.8235e-06,  1.9079e-06,  1.7075e-06,
1.7557e-06,  1.8609e-06,  1.7754e-06,  1.8339e-06,  1.7828e-06,
1.8355e-06,  1.8489e-06,  1.8231e-06,  1.8257e-06,  1.8233e-06,
1.8316e-06,  1.8036e-06,  1.8120e-06,  1.7622e-06,  1.8097e-06,
1.8430e-06,  1.8265e-06,  1.7204e-06,  1.7556e-06,  1.8129e-06,
1.8340e-06,  1.7226e-06,  1.8052e-06,  1.7865e-06,  1.7813e-06,
1.9104e-06,  1.8265e-06,  1.8552e-06,  1.8154e-06,  1.8130e-06,
1.8752e-06,  1.8339e-06,  1.7127e-06,  1.8309e-06,  1.7464e-06,
1.8260e-06,  1.7642e-06,  1.8377e-06,  1.8771e-06,  1.7815e-06,
1.8744e-06,  1.7938e-06,  1.8455e-06,  1.8261e-06,  1.7421e-06,
1.7807e-06,  1.8492e-06,  1.8287e-06,  1.8146e-06,  1.8368e-06,
1.7796e-06,  1.8203e-06,  1.7747e-06,  1.8106e-06,  1.7919e-06,
1.7662e-06,  1.8787e-06,  1.8361e-06,  1.7436e-06,  1.7487e-06,
1.8495e-06,  1.8057e-06,  1.7775e-06,  1.7901e-06,  1.7599e-06,
1.8531e-06,  1.8765e-06,  1.8610e-06,  1.8855e-06,  1.8145e-06,
1.8273e-06,  1.7750e-06,  1.7964e-06,  1.8351e-06,  1.8379e-06,
```

```
1.8464e-06,  1.7990e-06,  1.7749e-06,  1.7862e-06,  1.8071e-06,
1.7686e-06,  1.7592e-06,  1.7152e-06,  1.8055e-06,  1.8030e-06,
1.7632e-06,  1.7834e-06,  1.7441e-06,  1.7614e-06,  1.7728e-06,
1.7686e-06,  1.8309e-06,  1.8473e-06,  1.8205e-06,  1.7740e-06,
1.8621e-06,  1.7705e-06,  1.8019e-06,  1.8397e-06,  1.7920e-06,
1.8305e-06,  1.7994e-06,  1.7238e-06,  1.8403e-06,  1.7961e-06,
1.8187e-06,  1.8348e-06,  1.7269e-06,  1.7977e-06,  1.7901e-06,
1.7873e-06,  1.7983e-06,  1.7998e-06,  1.8334e-06,  1.8143e-06,
1.7397e-06,  1.7884e-06,  1.8067e-06,  1.7233e-06,  1.8574e-06,
1.8295e-06,  1.7281e-06,  1.9022e-06,  1.7759e-06,  1.8182e-06,
1.8380e-06,  1.8124e-06,  1.7535e-06,  1.8729e-06,  1.8021e-06,
1.7900e-06,  1.8377e-06,  1.8103e-06,  1.7963e-06,  1.8002e-06,
1.7930e-06,  1.8032e-06,  1.8058e-06,  1.8043e-06,  1.7339e-06,
1.7574e-06,  1.8207e-06,  1.7992e-06,  1.7438e-06,  1.8478e-06,
1.7181e-06,  1.7574e-06,  1.7424e-06,  1.8002e-06,  1.8065e-06,
1.7340e-06,  1.8088e-06,  1.8339e-06,  1.8311e-06,  1.8134e-06,
1.8347e-06,  1.8714e-06,  1.7745e-06,  1.8097e-06,  1.7883e-06,
1.7231e-06,  1.8426e-06,  1.8164e-06,  1.8077e-06,  1.8049e-06,
1.8087e-06,  1.7921e-06,  1.7226e-06,  1.8368e-06,  1.7593e-06,
1.7558e-06,  1.8061e-06,  1.7968e-06,  1.8050e-06,  1.8505e-06,
1.8345e-06,  1.8381e-06,  1.7229e-06,  1.7478e-06,  1.8763e-06,
1.7800e-06,  1.7882e-06,  1.7179e-06,  1.7814e-06,  1.8415e-06,
1.8135e-06,  1.7921e-06,  1.7259e-06,  1.8314e-06,  1.8983e-06,
1.7746e-06,  1.7515e-06,  1.8004e-06,  1.8190e-06,  1.7459e-06,
1.8497e-06,  1.7216e-06,  1.9108e-06,  1.7721e-06,  1.8280e-06,
1.8668e-06,  1.7777e-06,  1.8204e-06,  1.7745e-06,  1.7842e-06,
1.7971e-06,  1.8204e-06,  1.7589e-06,  1.8170e-06,  1.7154e-06,
1.8311e-06,  1.7528e-06,  1.8087e-06,  1.7966e-06,  1.7832e-06,
1.9151e-06,  1.8679e-06,  1.7583e-06,  1.7552e-06,  1.7905e-06,
1.7210e-06,  1.8041e-06,  1.7666e-06,  1.8286e-06,  1.7898e-06,
1.7796e-06,  1.7912e-06,  1.7628e-06,  1.7638e-06,  1.8270e-06,
1.8616e-06,  1.7407e-06,  1.8095e-06,  1.7193e-06,  1.8070e-06,
1.7488e-06,  1.8785e-06,  1.8001e-06,  1.8166e-06,  1.8010e-06,
1.7402e-06,  1.7142e-06,  1.7929e-06,  1.8262e-06,  1.7529e-06,
1.8153e-06,  1.7414e-06,  1.8439e-06,  1.8505e-06,  1.7844e-06,
1.7809e-06,  1.8495e-06,  1.8162e-06,  1.7461e-06,  1.8088e-06,
1.8322e-06,  1.7887e-06,  1.7772e-06,  1.7873e-06,  1.7457e-06,
1.7970e-06,  1.7697e-06,  1.8670e-06,  1.8153e-06,  1.8612e-06,
1.8033e-06,  1.7360e-06,  1.7538e-06,  1.8207e-06,  1.8542e-06,
1.8095e-06,  1.7797e-06,  1.8691e-06,  1.8745e-06,  1.8155e-06,
1.8175e-06,  1.7972e-06,  1.8045e-06,  1.8345e-06,  1.8331e-06,
1.7371e-06,  1.8028e-06,  1.8081e-06,  1.7908e-06,  1.7704e-06,
1.7881e-06,  1.8092e-06,  1.8130e-06,  1.7233e-06,  1.8120e-06,
1.8163e-06,  1.8105e-06,  1.7438e-06,  1.8044e-06,  1.7702e-06,
1.8504e-06,  1.7887e-06,  1.7653e-06,  1.7705e-06,  1.8097e-06,
1.8053e-06,  1.7907e-06,  1.8466e-06,  1.7212e-06,  1.8444e-06,
1.7654e-06,  1.8565e-06,  1.7841e-06,  1.8245e-06,  1.7945e-06,
1.9339e-06,  1.8113e-06,  1.8064e-06,  1.7700e-06,  1.8171e-06,
```

```
1.8480e-06,  1.7927e-06,  1.7663e-06,  1.7769e-06,  1.7935e-06,
1.8373e-06,  1.8252e-06,  1.8146e-06,  1.8338e-06,  1.8233e-06,
1.8110e-06,  1.8786e-06,  1.8171e-06,  1.7594e-06,  1.7659e-06,
1.8706e-06,  1.7568e-06,  1.9096e-06,  1.8519e-06,  1.7788e-06,
1.7941e-06,  1.8407e-06,  1.7865e-06,  1.8151e-06,  1.7977e-06,
1.7789e-06,  1.8066e-06,  1.8293e-06,  1.7782e-06,  1.8256e-06,
1.8072e-06,  1.7502e-06,  1.8365e-06,  1.7419e-06,  1.8143e-06,
1.7486e-06,  1.8473e-06,  1.7821e-06,  1.8124e-06,  1.8235e-06,
1.7305e-06,  1.8301e-06,  1.7319e-06,  1.7558e-06,  1.7518e-06,
1.8602e-06,  1.8156e-06,  1.8752e-06,  1.7627e-06,  1.8339e-06,
1.8542e-06,  1.8548e-06,  1.8252e-06,  1.8447e-06,  1.7201e-06,
1.7773e-06,  1.7913e-06,  1.8622e-06,  1.8205e-06,  1.8156e-06,
1.8781e-06,  1.7546e-06,  1.7537e-06,  1.7571e-06,  1.8478e-06,
1.7562e-06,  1.8141e-06,  1.7928e-06,  1.8003e-06,  1.8116e-06,
1.7712e-06,  1.8232e-06,  1.7812e-06,  1.7129e-06,  1.7411e-06,
1.8323e-06,  1.7853e-06,  1.8115e-06,  1.8165e-06,  1.8575e-06,
1.9000e-06,  1.7892e-06,  1.8010e-06,  1.8131e-06,  1.8346e-06,
1.8799e-06,  1.7954e-06,  1.7514e-06,  1.7697e-06,  1.7528e-06,
1.8626e-06,  1.8177e-06,  1.8212e-06,  1.7285e-06,  1.8468e-06,
1.7274e-06,  1.8112e-06,  1.7727e-06,  1.8584e-06,  1.8138e-06,
1.8286e-06,  1.8040e-06,  1.8397e-06,  1.8160e-06,  1.8068e-06,
1.8117e-06,  1.7309e-06,  1.8191e-06,  1.8351e-06,  1.8084e-06,
1.8077e-06,  1.8363e-06,  1.7747e-06,  1.7374e-06,  1.7219e-06,
1.8465e-06,  1.7981e-06,  1.7735e-06,  1.6318e-06,  1.8679e-06,
1.8290e-06,  1.7598e-06,  1.8147e-06,  1.8042e-06,  1.8136e-06,
1.7027e-06,  1.7788e-06,  1.7973e-06,  1.8526e-06,  1.7297e-06,
1.7244e-06,  1.7927e-06,  1.7880e-06,  1.7250e-06,  1.8385e-06,
1.7322e-06,  1.7392e-06,  1.8039e-06,  1.8084e-06,  1.7705e-06,
1.7280e-06,  1.8175e-06,  1.8346e-06,  1.7721e-06,  1.8105e-06,
1.7798e-06,  1.8084e-06,  1.7529e-06,  1.7851e-06,  1.7815e-06,
1.8760e-06,  1.8091e-06,  1.7944e-06,  1.8596e-06,  1.8531e-06,
1.7587e-06,  1.7890e-06,  1.7276e-06,  1.8115e-06,  1.7775e-06,
1.8479e-06,  1.7789e-06,  1.7513e-06,  1.7464e-06,  1.8023e-06,
1.8176e-06,  1.8323e-06,  1.7973e-06,  1.8594e-06,  1.7810e-06,
1.7808e-06,  1.7270e-06,  1.8626e-06,  1.7892e-06,  1.8014e-06,
1.8039e-06,  1.8267e-06,  1.7793e-06,  1.8227e-06,  1.8750e-06,
1.7530e-06,  1.7376e-06,  1.7999e-06,  1.8195e-06,  1.8065e-06,
1.9278e-06,  1.8301e-06,  1.8351e-06,  1.8678e-06,  1.7954e-06,
1.8938e-06,  1.7906e-06,  1.7803e-06,  1.6961e-06,  1.7134e-06,
1.7818e-06,  1.8253e-06,  1.7840e-06,  1.8595e-06,  1.8094e-06,
1.7619e-06,  1.8589e-06,  1.7407e-06,  1.7830e-06,  1.7977e-06,
1.8306e-06,  1.7110e-06,  1.8555e-06,  1.8341e-06,  1.7258e-06,
1.7437e-06,  1.8359e-06,  1.8156e-06,  1.7495e-06,  1.8199e-06,
1.8181e-06,  1.7762e-06,  1.8192e-06,  1.8052e-06,  1.8273e-06,
1.8328e-06,  1.8036e-06,  1.8140e-06,  1.8534e-06,  1.7319e-06,
1.7964e-06,  1.8094e-06,  1.8663e-06,  1.7964e-06,  1.8961e-06,
1.9486e-06,  1.8151e-06,  1.7829e-06,  1.7758e-06,  1.7416e-06,
1.7795e-06,  1.8084e-06,  1.8558e-06,  1.8447e-06,  1.8305e-06,
```

```
          1.8076e-06,  1.8438e-06,  1.7528e-06,  1.8556e-06,  1.7509e-06,
          1.8375e-06,  1.7948e-06,  1.8257e-06,  1.8030e-06,  1.8339e-06,
          1.8557e-06,  1.8309e-06,  1.7712e-06,  1.7471e-06,  1.7590e-06,
          1.8269e-06,  1.8620e-06,  1.8238e-06,  1.8712e-06,  1.7980e-06,
          1.8562e-06,  1.9058e-06,  1.7921e-06,  1.7691e-06,  1.7989e-06,
          1.7362e-06,  1.8726e-06,  1.7292e-06,  1.8342e-06,  1.8245e-06,
          1.8715e-06,  1.8626e-06,  1.8641e-06,  1.8643e-06,  1.8198e-06,
          1.8674e-06,  1.7922e-06,  1.7188e-06,  1.7756e-06,  1.8131e-06,
          1.8410e-06,  1.8954e-06,  1.8418e-06,  1.8474e-06,  1.8032e-06,
          1.8015e-06,  1.7701e-06,  1.7690e-06,  1.7096e-06,  1.7240e-06,
          1.9166e-06,  1.8272e-06,  1.8535e-06,  1.9072e-06,  1.8896e-06,
          1.8351e-06,  1.8085e-06,  1.8051e-06,  1.8486e-06,  1.8073e-06,
          1.8244e-06,  1.8111e-06,  1.8028e-06,  1.8335e-06,  1.8451e-06,
          1.8237e-06,  1.8286e-06,  1.7706e-06,  1.7572e-06,  1.7820e-06,
          1.7533e-06,  1.8182e-06,  1.8079e-06,  1.8240e-06,  1.8081e-06,
          1.8251e-06,  1.8082e-06,  1.8839e-06,  1.8642e-06,  1.7994e-06,
          1.7683e-06,  1.8044e-06,  1.7868e-06,  1.7499e-06,  1.8889e-06,
          1.7534e-06,  1.7415e-06,  1.8141e-06,  1.7695e-06,  1.8136e-06,
          1.8294e-06,  1.7767e-06,  1.8268e-06,  1.8193e-06,  1.7272e-06,
          1.7477e-06,  1.8417e-06,  1.8241e-06,  1.8328e-06,  1.9603e-06])
end of p.grad

False
Epoch 1 finished
Epoch [1/10], Loss: 2.0627
************************************************************

Learning rate: 0.001

another way to print learning rate:
tensor([[[[-7.3280e-04, -4.7365e-04,  1.7247e-04,  …,  1.9832e-03,
            2.1671e-03,  2.6696e-03],
          [-1.4190e-04, -1.6004e-04, -1.6317e-04,  …,  2.8018e-03,
            3.0133e-03,  3.2748e-03],
          [-2.0058e-04,  6.4376e-05, -1.6218e-04,  …,  2.3786e-03,
            2.5567e-03,  2.4045e-03],
          …,
          [-2.8301e-03, -3.1046e-03, -3.8356e-03,  …, -5.2173e-03,
           -4.9356e-03, -4.7123e-03],
          [-4.1248e-03, -4.4643e-03, -4.4993e-03,  …, -4.3475e-03,
           -4.4187e-03, -4.0343e-03],
          [-4.9349e-03, -5.0598e-03, -4.6522e-03,  …, -3.4053e-03,
           -3.2711e-03, -2.7619e-03]],

         [[ 6.1123e-04,  5.7573e-04,  1.0527e-03,  …,  2.7499e-03,
            2.7148e-03,  3.1698e-03],
          [ 1.1549e-03,  8.7638e-04,  7.2167e-04,  …,  3.5016e-03,
            3.3883e-03,  3.6937e-03],
```

```
  [ 1.0265e-03,  1.1233e-03,  9.2014e-04,  …,  3.1403e-03,
    3.2383e-03,  2.9192e-03],
  …,
  [-1.4697e-03, -2.0072e-03, -2.9429e-03,  …, -4.2602e-03,
   -4.0033e-03, -3.7028e-03],
  [-3.1343e-03, -3.6732e-03, -3.7538e-03,  …, -3.6297e-03,
   -3.6243e-03, -3.1054e-03],
  [-3.8883e-03, -3.9660e-03, -3.7996e-03,  …, -2.8366e-03,
   -2.6488e-03, -2.2402e-03]],

 [[ 3.0518e-03,  2.8636e-03,  3.3344e-03,  …,  4.9303e-03,
    4.8026e-03,  5.3230e-03],
  [ 3.5540e-03,  3.1196e-03,  2.9459e-03,  …,  5.4432e-03,
    5.2099e-03,  5.5928e-03],
  [ 3.1969e-03,  3.2502e-03,  2.9896e-03,  …,  4.8816e-03,
    4.9641e-03,  4.6700e-03],
  …,
  [ 4.6506e-04, -2.3888e-04, -1.0538e-03,  …, -2.0415e-03,
   -1.8478e-03, -1.5498e-03],
  [-1.3364e-03, -1.9336e-03, -1.9729e-03,  …, -1.5487e-03,
   -1.4658e-03, -8.4554e-04],
  [-1.9344e-03, -1.9508e-03, -1.9892e-03,  …, -8.7323e-04,
   -5.1843e-04, -6.6770e-05]]],


[[[ 5.4656e-03,  3.2323e-03,  2.2967e-03,  …,  2.4061e-03,
    1.7844e-03,  2.8731e-03],
  [ 1.8269e-03,  2.5526e-04,  4.1510e-04,  …,  3.1103e-03,
    1.4901e-03,  2.2307e-03],
  [ 2.0539e-03,  6.9264e-04,  3.7736e-04,  …,  3.8121e-03,
    3.3020e-03,  2.5619e-03],
  …,
  [ 6.2552e-03,  7.0346e-03,  5.3427e-03,  …,  5.9541e-03,
    5.5936e-03,  4.0157e-03],
  [ 8.6536e-03,  7.9684e-03,  5.9094e-03,  …,  7.5302e-03,
    7.2028e-03,  5.3385e-03],
  [ 6.7627e-03,  6.3930e-03,  5.4239e-03,  …,  8.6946e-03,
    7.4663e-03,  5.2804e-03]],

 [[ 1.0110e-02,  8.0152e-03,  7.4898e-03,  …,  8.4238e-03,
    7.2716e-03,  7.5679e-03],
  [ 6.3251e-03,  5.5265e-03,  5.8421e-03,  …,  8.6422e-03,
    7.0378e-03,  7.1332e-03],
  [ 6.5470e-03,  6.0572e-03,  6.1281e-03,  …,  9.4795e-03,
    8.7159e-03,  7.2813e-03],
  …,
  [ 1.0135e-02,  1.1144e-02,  9.9445e-03,  …,  1.0038e-02,
    9.4345e-03,  7.9402e-03],
```

```
  [ 1.2524e-02,  1.2019e-02,  9.9797e-03,  …,  1.1277e-02,
    1.0875e-02,  8.6958e-03],
  [ 1.1286e-02,  1.1067e-02,  9.3951e-03,  …,  1.2263e-02,
    1.1153e-02,  8.5550e-03]],

 [[ 7.0736e-03,  5.6012e-03,  4.4254e-03,  …,  6.3421e-03,
    5.2583e-03,  4.9242e-03],
  [ 3.1451e-03,  2.5179e-03,  2.8072e-03,  …,  6.4097e-03,
    5.0255e-03,  4.8746e-03],
  [ 3.0386e-03,  2.9556e-03,  3.2815e-03,  …,  6.9223e-03,
    6.3941e-03,  5.2040e-03],
 …,
  [ 5.3964e-03,  6.6886e-03,  5.7050e-03,  …,  8.5708e-03,
    7.5997e-03,  5.6518e-03],
  [ 7.2541e-03,  7.6226e-03,  5.9224e-03,  …,  8.8844e-03,
    8.7070e-03,  6.2572e-03],
  [ 5.7755e-03,  6.2108e-03,  5.4073e-03,  …,  9.4318e-03,
    8.5586e-03,  5.4776e-03]]],


[[[ 1.2690e-04, -1.0129e-04, -5.0310e-04,  …, -1.5131e-03,
   -1.9889e-03, -2.5538e-03],
  [-5.4496e-04,  2.8688e-06, -4.3001e-04,  …, -1.7682e-03,
   -2.3307e-03, -3.0597e-03],
  [-6.7989e-04, -7.2914e-04, -4.6916e-04,  …, -1.7574e-03,
   -1.9743e-03, -2.0671e-03],
 …,
  [-5.9795e-04, -4.7456e-04, -6.5868e-04,  …, -3.1067e-04,
   -3.6258e-04, -8.6337e-04],
  [-1.2469e-03, -5.0093e-04, -4.3952e-04,  …,  4.3409e-04,
    5.0271e-04, -1.1004e-04],
  [-1.9401e-03, -1.4208e-03, -1.1502e-03,  …,  2.2491e-04,
    1.0043e-04, -2.9693e-04]],

 [[ 8.6731e-04,  8.3034e-04,  6.9967e-04,  …, -1.8661e-04,
   -7.7725e-04, -1.4346e-03],
  [ 1.1217e-04,  9.4892e-04,  6.1126e-04,  …, -4.8701e-04,
   -1.1757e-03, -2.0898e-03],
  [-3.5610e-04,  2.3811e-04,  3.6536e-04,  …, -4.6489e-04,
   -7.3028e-04, -1.1057e-03],
 …,
  [ 4.9473e-04,  5.5464e-04,  2.9358e-04,  …,  8.3184e-04,
    6.4504e-04, -2.5443e-04],
  [-5.1705e-04,  1.3838e-04,  2.2293e-04,  …,  1.3894e-03,
    1.0261e-03,  2.9902e-04],
  [-1.2645e-03, -6.6419e-04, -5.3138e-04,  …,  7.0095e-04,
    5.4185e-04,  1.4010e-04]],
```

```
 [[ 9.4341e-04,  1.2460e-03,  1.1740e-03,  …,  8.4341e-04,
    6.2570e-05, -7.4819e-04],
  [ 8.6147e-05,  1.1987e-03,  9.6690e-04,  …,  5.2481e-04,
   -3.3407e-04, -1.5314e-03],
  [-6.6187e-04,  3.3903e-04,  6.5938e-04,  …,  3.5040e-04,
   -1.8201e-04, -7.1875e-04],
  …,
  [ 8.8028e-04,  7.0515e-04,  5.3344e-04,  …,  9.6869e-04,
    4.9861e-04, -4.5908e-04],
  [-1.8478e-04,  3.3452e-04,  5.7197e-04,  …,  1.4093e-03,
    6.9541e-04,  2.0140e-05],
  [-8.6814e-04, -2.8131e-04, -9.0827e-05,  …,  4.6757e-04,
    8.1686e-05, -3.7539e-04]]],


…,


[[[-1.0179e-03, -8.3416e-04, -1.7611e-04,  …,  3.2747e-04,
    2.7054e-04,  2.9650e-04],
  [-1.3779e-03, -1.0950e-03, -8.9429e-04,  …,  1.2445e-03,
    1.6697e-03,  1.6156e-03],
  [-1.4248e-03, -1.7339e-03, -1.5246e-03,  …,  1.0185e-03,
    1.7668e-03,  2.1246e-03],
  …,
  [ 1.3650e-04,  1.6590e-04,  3.0113e-04,  …,  1.8284e-03,
    1.0992e-03,  1.0198e-03],
  [ 5.6520e-04,  5.8812e-04,  1.1844e-03,  …,  2.9965e-03,
    2.2021e-03,  1.7898e-03],
  [ 1.2259e-03,  1.4922e-03,  2.3525e-03,  …,  3.4828e-03,
    3.5969e-03,  2.9052e-03]],

 [[ 3.8934e-04,  9.2802e-04,  1.6678e-03,  …,  1.3283e-03,
    1.4476e-03,  1.5838e-03],
  [ 5.1593e-04,  1.5882e-03,  2.0097e-03,  …,  2.8385e-03,
    3.1304e-03,  3.3384e-03],
  [ 1.1562e-03,  8.5129e-04,  1.1100e-03,  …,  3.1983e-03,
    3.8369e-03,  4.2202e-03],
  …,
  [ 2.7962e-03,  3.0195e-03,  3.3430e-03,  …,  4.8745e-03,
    4.5593e-03,  3.9832e-03],
  [ 3.3391e-03,  3.7088e-03,  4.0594e-03,  …,  6.2453e-03,
    5.4461e-03,  4.9419e-03],
  [ 4.0901e-03,  3.7734e-03,  4.8241e-03,  …,  5.9556e-03,
    5.6445e-03,  5.3925e-03]],

 [[ 1.4850e-04,  4.4057e-04,  1.3404e-03,  …,  1.5658e-03,
    1.8987e-03,  2.2817e-03],
```

```
  [ 2.5225e-04,  1.0231e-03,  1.6625e-03,  …,  2.2480e-03,
    2.9325e-03,  3.5391e-03],
  [ 1.1067e-03,  6.8657e-04,  1.1139e-03,  …,  2.7532e-03,
    3.2299e-03,  4.0992e-03],
 …,
  [ 3.5384e-03,  3.6657e-03,  3.8564e-03,  …,  5.1125e-03,
    4.7466e-03,  4.3914e-03],
  [ 4.3832e-03,  4.7333e-03,  4.8496e-03,  …,  6.7218e-03,
    5.9402e-03,  5.5996e-03],
  [ 4.7307e-03,  4.0345e-03,  5.1630e-03,  …,  6.0887e-03,
    5.8013e-03,  5.5697e-03]]],


[[[ 5.3791e-04,  5.2703e-04,  5.4622e-04,  …,  6.1236e-04,
    3.7820e-04,  7.1253e-04],
  [ 1.3867e-03,  1.5183e-03,  1.5822e-03,  …,  8.4121e-04,
    6.7100e-04,  1.7838e-03],
  [ 3.2006e-03,  3.1060e-03,  1.6113e-03,  …,  1.5706e-03,
    2.2492e-03,  2.9385e-03],
 …,
  [ 5.4285e-03,  5.4167e-03,  5.1416e-03,  …,  4.1087e-03,
    4.2179e-03,  5.1137e-03],
  [ 5.2728e-03,  5.1430e-03,  5.6709e-03,  …,  6.2833e-03,
    5.7743e-03,  6.4123e-03],
  [ 7.4207e-03,  7.7366e-03,  7.6349e-03,  …,  7.2655e-03,
    8.3011e-03,  8.0856e-03]],

 [[ 3.0436e-03,  3.2335e-03,  2.6867e-03,  …,  2.4162e-03,
    2.4141e-03,  2.8201e-03],
  [ 3.6220e-03,  4.3550e-03,  4.3593e-03,  …,  2.9029e-03,
    2.6853e-03,  3.9755e-03],
  [ 5.5282e-03,  5.0176e-03,  3.8522e-03,  …,  4.3511e-03,
    3.9973e-03,  4.6859e-03],
 …,
  [ 6.5770e-03,  6.7910e-03,  6.7739e-03,  …,  6.7028e-03,
    6.9006e-03,  7.1364e-03],
  [ 6.2243e-03,  7.2956e-03,  7.5483e-03,  …,  8.4755e-03,
    7.6758e-03,  8.3317e-03],
  [ 8.1998e-03,  8.1287e-03,  8.6169e-03,  …,  9.3145e-03,
    9.7509e-03,  9.7664e-03]],

 [[ 1.9173e-03,  1.1545e-03,  7.2662e-04,  …,  9.6089e-04,
    1.1528e-03,  1.8651e-03],
  [ 2.1583e-03,  2.1565e-03,  1.9394e-03,  …,  5.7949e-04,
    3.4838e-04,  1.9617e-03],
  [ 3.3899e-03,  2.2682e-03,  1.0476e-03,  …,  1.9379e-03,
    7.1386e-04,  1.6618e-03],
 …,
```

```
  [ 4.0405e-03,  4.2378e-03,  4.1304e-03,  …,  3.8872e-03,
    3.8350e-03,  3.8638e-03],
  [ 3.6386e-03,  4.5802e-03,  4.9453e-03,  …,  5.7282e-03,
    4.8035e-03,  5.5098e-03],
  [ 5.1131e-03,  4.9128e-03,  5.7481e-03,  …,  6.4843e-03,
    6.5580e-03,  6.6601e-03]]],


[[[-2.2845e-03, -4.2417e-03, -5.3481e-03,  …, -5.4370e-03,
   -3.9351e-03, -2.6988e-03],
  [-4.6954e-03, -5.8764e-03, -5.9517e-03,  …, -1.8966e-03,
   -2.1275e-03, -5.4964e-04],
  [-4.3011e-03, -5.5029e-03, -6.6617e-03,  …,  1.1556e-03,
    2.0634e-03,  2.1247e-03],
  …,
  [ 3.9797e-03,  5.0186e-03,  5.5318e-03,  …,  8.1416e-03,
    8.4572e-03,  8.8839e-03],
  [ 4.4325e-03,  4.4855e-03,  4.7325e-03,  …,  9.1816e-03,
    9.2609e-03,  9.3687e-03],
  [ 5.0783e-03,  5.2601e-03,  5.5205e-03,  …,  9.8232e-03,
    1.0315e-02,  9.8193e-03]],

 [[-5.4211e-03, -7.0584e-03, -8.1914e-03,  …, -9.3471e-03,
   -8.2354e-03, -7.5317e-03],
  [-8.0890e-03, -8.7690e-03, -9.0494e-03,  …, -6.3757e-03,
   -6.3796e-03, -5.2543e-03],
  [-8.0619e-03, -9.0520e-03, -9.4784e-03,  …, -3.2935e-03,
   -2.7875e-03, -2.6203e-03],
  …,
  [ 4.0429e-04,  2.1030e-03,  3.0151e-03,  …,  3.8866e-03,
    3.9776e-03,  4.2407e-03],
  [ 1.0178e-03,  1.7860e-03,  2.0443e-03,  …,  4.9547e-03,
    4.5831e-03,  4.2921e-03],
  [ 1.6182e-03,  1.3121e-03,  1.7547e-03,  …,  4.7943e-03,
    4.9369e-03,  5.0283e-03]],

 [[-6.9517e-04, -2.6379e-03, -3.2196e-03,  …, -2.9383e-03,
   -1.6419e-03, -5.7436e-04],
  [-3.4837e-03, -4.8177e-03, -4.8517e-03,  …, -1.7556e-03,
   -1.0859e-03,  3.5309e-04],
  [-3.9018e-03, -5.4208e-03, -5.6708e-03,  …,  3.1697e-04,
    7.4637e-04,  1.1099e-03],
  …,
  [ 4.1058e-03,  5.7935e-03,  6.6558e-03,  …,  6.4542e-03,
    6.2795e-03,  6.7736e-03],
  [ 4.5064e-03,  5.1901e-03,  5.6750e-03,  …,  7.8601e-03,
    7.2145e-03,  6.8083e-03],
  [ 4.8942e-03,  4.1654e-03,  4.6688e-03,  …,  7.3718e-03,
```

```
           7.4956e-03,  7.2123e-03]]]])
tensor([-8.1491e-10,  6.7055e-08,  7.2177e-09, -1.9500e-09, -1.6997e-08,
         3.0268e-08, -1.8044e-09,  1.2666e-07,  6.4028e-09, -4.7730e-08,
        -1.7819e-08,  9.5461e-09, -2.4680e-08, -1.3970e-09, -2.1188e-08,
         1.3970e-09,  1.3504e-08, -7.3574e-08, -1.0575e-06, -5.1223e-09,
         2.3865e-09, -1.9791e-09,  3.5623e-08, -1.9674e-08, -4.4180e-08,
         1.3912e-08, -2.5029e-09, -5.3551e-09, -3.7253e-09, -1.6298e-08,
         5.8208e-10, -2.3283e-10, -1.3970e-09, -6.5484e-11,  8.7894e-09,
         1.1970e-09, -4.8662e-08, -5.1223e-09, -9.4878e-09, -4.5402e-09,
         3.4925e-10, -8.8476e-09,  1.2806e-08,  2.6776e-09, -6.9849e-09,
        -4.1677e-08,  1.6764e-08,  1.3388e-08,  1.3970e-08,  8.6729e-09,
         2.7649e-09,  2.9569e-08,  1.0041e-08,  7.4506e-08,  8.3819e-09,
        -1.6298e-09, -3.2363e-08,  1.1642e-09,  2.7721e-09,  4.1910e-09,
         1.2456e-08, -3.4925e-10,  3.8883e-08, -2.5684e-09, -5.4715e-09,
         3.1214e-09,  1.8161e-08,  9.7323e-08, -1.1642e-09,  4.7730e-09,
        -1.4770e-09, -1.9325e-08, -1.8859e-08, -1.9325e-08, -2.9569e-08,
        -1.6531e-08, -5.8790e-09,  9.0804e-09, -6.6939e-09,  2.3283e-10,
        -1.0361e-08,  3.8184e-08, -1.1409e-08, -7.4506e-09, -2.0023e-08,
        -1.7229e-08,  4.1444e-08,  2.4447e-09,  9.0222e-09,  8.1491e-09,
         1.4785e-08, -4.9360e-08, -9.3132e-10, -2.4680e-08,  4.9826e-08,
        -9.8953e-09])
tensor([-1.2044e-03,  2.9163e-03,  5.6998e-03, -1.9723e-03, -1.8406e-03,
         2.9451e-02,  5.2111e-03,  2.0084e-02,  5.0292e-03,  1.0041e-02,
         2.3714e-03, -8.5765e-03, -1.3060e-02,  4.0091e-03,  4.4985e-03,
        -1.2550e-03,  6.0773e-03,  2.2229e-02,  4.6876e-05,  5.1813e-03,
        -1.2990e-03, -1.4168e-02,  3.1562e-03,  7.3165e-03,  6.5506e-03,
        -3.1953e-03, -1.6370e-03,  6.7020e-03,  1.1474e-02,  2.6564e-02,
        -3.3017e-03, -3.9177e-03,  3.8136e-03, -2.5796e-03, -6.3204e-03,
        -4.2579e-03,  6.7533e-03, -5.5583e-03, -2.8618e-02, -4.2299e-03,
        -3.3534e-02,  5.7118e-03,  4.8832e-03, -2.8205e-03, -8.3021e-03,
         6.3378e-03,  6.9915e-03,  8.4038e-04,  1.9533e-02,  9.2759e-03,
        -3.0318e-04, -1.1968e-02, -5.1389e-04,  2.8042e-02,  2.8907e-03,
         6.3373e-03, -9.1967e-02, -2.6388e-03,  2.6137e-03,  5.9596e-03,
        -7.7023e-04, -3.4057e-03,  1.2849e-02, -4.1127e-03,  4.8861e-04,
         3.0993e-03,  6.4507e-03,  4.6647e-03, -1.0971e-03,  3.6708e-03,
        -3.3031e-03,  1.0847e-02,  1.5790e-02, -1.1753e-02, -2.0143e-02,
         9.2922e-03, -2.0506e-03,  7.7045e-03, -1.0840e-02,  7.3664e-03,
        -4.7661e-02, -4.1330e-03, -8.9536e-03,  3.2863e-03, -4.5378e-03,
         5.5386e-03,  3.8654e-03, -9.2607e-03, -5.5722e-04, -1.3167e-02,
         1.9086e-02, -2.0466e-03, -1.4400e-02,  6.0659e-03,  2.3311e-02,
        -1.6509e-02])
tensor([-2.1525e-03, -1.5952e-03,  6.4777e-03,  7.7527e-04,  6.1992e-04,
         3.5992e-02,  8.3305e-03,  2.0132e-02,  9.6484e-03,  8.3334e-03,
         2.1503e-03,  1.0538e-03, -7.4111e-03, -1.3693e-04,  1.2442e-02,
         5.1319e-06,  6.3897e-03,  1.3343e-02,  1.0003e-03, -1.7957e-03,
         2.5258e-03, -1.4376e-02,  5.3806e-03,  4.2294e-03,  1.0030e-02,
        -5.5598e-04,  2.7603e-04,  4.8576e-03,  1.6570e-02,  2.7082e-02,
         1.2400e-03, -8.2838e-03,  3.9837e-03,  7.1031e-04, -5.2226e-03,
```

```
        -1.2553e-04,  1.2512e-02, -5.5865e-04, -1.3364e-02, -1.1075e-02,
        -1.4878e-02,  3.2232e-03,  5.4816e-03, -7.4622e-04, -2.0432e-03,
         3.4371e-03,  4.8037e-03,  2.7013e-03,  9.6907e-03,  8.6345e-03,
         1.8067e-03, -1.6254e-02,  6.5098e-03,  1.1833e-02,  6.1219e-03,
         8.4553e-03, -2.6246e-02,  3.7250e-03,  3.2670e-03,  4.5889e-03,
        -1.8185e-04,  9.9901e-04,  1.2668e-02,  1.6772e-03,  1.7498e-03,
         6.3725e-03,  1.7625e-02,  1.0133e-02, -1.5251e-03,  4.7781e-03,
        -4.0556e-04,  5.6375e-03,  5.6878e-03, -1.8074e-02, -2.3972e-02,
         6.8184e-03,  1.1455e-03, -3.3337e-04, -1.1769e-02,  8.8521e-03,
        -1.0049e-02, -1.0610e-02,  8.3863e-04,  1.4343e-02, -4.7877e-03,
         1.1960e-02,  1.7571e-02, -9.2102e-03,  4.2312e-04, -1.9673e-02,
         3.1967e-03,  1.6934e-03, -8.2236e-03,  8.0266e-03,  1.0468e-02,
        -1.8146e-02])
tensor([[[[-1.2884e-03, -1.8511e-03, -1.1879e-03, -1.1313e-03, -7.8369e-04],
          [-1.5458e-03, -1.7355e-03, -1.1589e-03, -1.0756e-03, -8.8608e-04],
          [-1.1012e-03, -1.2007e-03, -1.0378e-03, -6.1942e-04, -2.7662e-04],
          [-1.0892e-03, -1.1534e-03, -5.4892e-04, -5.4588e-05,  3.2653e-04],
          [-5.3947e-05, -1.5471e-04,  3.4166e-04,  8.1859e-04,  1.0736e-03]],

         [[ 6.0441e-04,  4.1563e-04,  2.3396e-04,  2.7374e-04,  6.5763e-04],
          [ 3.3979e-04,  3.7453e-04,  5.1416e-05,  4.4315e-04,  8.6124e-04],
          [ 4.2511e-04,  4.7152e-04,  1.0064e-04,  1.1877e-04,  2.3778e-04],
          [-2.6984e-04, -6.7975e-05, -6.9506e-04, -1.0439e-03, -1.1600e-03],
          [-2.3568e-04, -2.1204e-04, -8.7508e-04, -8.7112e-04, -1.1208e-03]],

         [[-6.5431e-04, -9.0269e-04, -9.5029e-04, -1.2045e-03, -5.5441e-04],
          [-4.8627e-04, -3.5705e-04, -4.0617e-04, -3.8667e-04,  7.6341e-04],
          [-1.3991e-04,  2.9610e-05, -1.7401e-04,  3.6296e-04,  1.1445e-03],
          [ 4.4579e-04,  4.9327e-04,  7.7754e-04,  1.0091e-03,  1.3423e-03],
          [ 7.1388e-04,  8.3041e-04,  1.2061e-03,  1.2097e-03,  1.3893e-03]],

         …,

         [[-1.5221e-03, -2.1502e-03, -1.6861e-03, -1.7347e-03, -9.8189e-04],
          [-1.3598e-03, -1.8811e-03, -1.1452e-03, -8.6166e-04, -9.3689e-05],
          [-1.1655e-03, -1.2459e-03, -4.0057e-04,  2.5166e-04,  6.4609e-04],
          [-7.7339e-04, -2.5838e-04,  3.2125e-04,  9.2934e-04,  1.5338e-03],
          [-5.2828e-05,  3.6113e-04,  1.1683e-03,  1.4167e-03,  1.5086e-03]],

         [[-1.6200e-03, -2.1815e-03, -1.4630e-03, -1.3639e-03, -8.8517e-04],
          [-1.7759e-03, -2.1998e-03, -1.2328e-03, -9.8726e-04, -6.3734e-04],
          [-1.6091e-03, -1.4188e-03, -8.2939e-04, -2.8479e-04, -5.1698e-05],
          [-1.5934e-03, -1.2380e-03, -2.7201e-04,  4.3061e-04,  7.2818e-04],
          [-4.9059e-04, -1.4816e-04,  6.8652e-04,  8.1197e-04,  7.9544e-04]],

         [[-1.1359e-03, -1.6809e-03, -1.1831e-03, -8.9472e-04, -3.9964e-04],
          [-1.2714e-03, -1.5356e-03, -7.9196e-04, -2.3576e-04,  1.7919e-04],
          [-1.0562e-03, -7.2210e-04,  1.0959e-04,  3.9125e-04,  4.4835e-04],
```

```
  [-4.9372e-04, -4.2865e-04,  2.8421e-04,  7.4344e-04,  8.8168e-04],
  [-1.9995e-04,  1.1093e-04,  4.4695e-04,  5.0506e-04,  4.6035e-04]]],


[[[ 1.9597e-04,  1.0426e-04, -1.0807e-05, -1.1404e-04, -2.8208e-04],
  [ 2.0057e-04,  1.1257e-04,  6.5956e-05,  2.2281e-05, -4.6221e-05],
  [ 2.2619e-04,  2.4936e-04,  1.8969e-04,  1.3614e-04,  1.7666e-04],
  [ 8.3479e-05,  1.4374e-04, -3.1491e-05,  1.6969e-05, -7.4229e-05],
  [ 8.9676e-05,  1.2309e-04, -1.8572e-04, -1.8999e-04, -3.6520e-04]],

 [[ 3.2737e-04,  4.9857e-04,  1.1650e-04,  7.6426e-04,  1.0567e-03],
  [ 8.8905e-05, -9.7767e-05, -1.8067e-04,  4.5641e-04,  7.7534e-04],
  [-7.8649e-05, -2.7563e-04, -7.4079e-05,  4.0023e-04,  6.8430e-04],
  [ 3.5449e-04,  2.4499e-04,  3.4775e-04,  6.2514e-04,  8.0669e-04],
  [ 1.0424e-03,  7.5670e-04,  7.4283e-04,  7.5270e-04,  1.1394e-03]],

 [[ 3.6725e-04,  6.1072e-04,  5.0936e-04,  7.8482e-04,  9.4958e-04],
  [-9.0839e-05,  1.5368e-04,  1.3616e-04,  4.5065e-04,  8.8063e-04],
  [-5.1285e-04, -4.1190e-04, -2.8680e-04,  7.6798e-05,  2.2083e-04],
  [-4.5786e-04, -4.9413e-04, -3.0518e-04,  2.3416e-05, -9.0854e-05],
  [-3.5483e-04, -3.6205e-04, -4.2732e-04, -2.5067e-04,  1.4317e-04]],

 …,

 [[ 1.0525e-04,  2.3341e-04,  2.6052e-05,  1.5962e-04,  7.2179e-05],
  [-1.3430e-06,  1.5028e-04,  4.9357e-05,  5.8408e-05,  5.3948e-05],
  [-1.0667e-04, -6.9257e-05, -6.4436e-05, -3.8553e-05, -6.6436e-05],
  [-6.7361e-05, -1.5257e-04, -2.0124e-04, -1.5236e-04, -2.7584e-04],
  [-2.0141e-04, -1.7735e-04, -5.5614e-04, -4.5297e-04, -3.8387e-04]],

 [[-5.3163e-05,  2.1391e-05, -1.9893e-04, -1.4157e-04, -3.5162e-04],
  [-1.6017e-05,  3.6147e-05, -7.0075e-05, -4.4426e-05, -1.1954e-04],
  [-1.2452e-04, -5.2203e-05, -1.0022e-04,  3.5931e-05,  3.3250e-05],
  [-2.5860e-05, -1.0039e-04, -1.9179e-04,  1.3157e-06, -9.0938e-05],
  [-1.8614e-05, -7.0987e-05, -4.0003e-04, -1.9885e-04, -2.2573e-04]],

 [[ 1.1115e-04,  1.1806e-04, -3.2484e-05, -9.5148e-05, -1.5660e-04],
  [ 1.7704e-04,  8.4835e-05, -1.4390e-05, -1.4969e-06, -2.0162e-04],
  [ 3.4604e-05, -9.6298e-06,  7.1394e-06, -2.4773e-05, -1.7766e-04],
  [-2.0299e-05, -1.2382e-05, -1.7274e-04, -2.1577e-04, -2.4858e-04],
  [-2.9845e-05, -2.0025e-04, -6.0419e-04, -4.2280e-04, -2.6314e-04]]],


[[[ 1.4391e-03,  7.0979e-04,  4.5287e-04,  1.2490e-04,  4.3487e-04],
  [ 1.2173e-03,  8.6746e-04,  6.3943e-04,  3.6469e-04,  8.3227e-05],
  [ 9.3536e-04,  1.1378e-03,  9.9672e-04,  6.1097e-04,  3.8487e-04],
  [ 7.7179e-04,  7.4136e-04,  7.0948e-04,  4.9849e-04,  5.0149e-04],
  [ 9.2417e-04,  8.8750e-04,  5.3814e-04,  2.4816e-04,  1.9223e-04]],
```

```
 [[ 1.0388e-03,  1.3081e-03,  8.3470e-04,  4.8436e-04,  6.4257e-04],
  [ 8.3305e-04,  9.6664e-04,  3.7576e-04,  2.8695e-04,  5.2842e-04],
  [ 5.5418e-04,  6.3499e-04, -2.6134e-04,  6.5023e-05,  5.4181e-04],
  [-7.0473e-05, -1.5776e-04, -8.2539e-04, -3.6625e-04,  2.7004e-04],
  [-2.1138e-04, -2.7384e-04, -8.2166e-04, -2.5138e-04,  1.8135e-04]],

 [[-5.9503e-04, -5.4037e-04, -8.2084e-04, -6.6187e-04, -5.0804e-04],
  [-4.5163e-04, -5.1879e-04, -1.1327e-03, -9.7669e-04, -6.7170e-04],
  [-8.5246e-05, -2.1668e-04, -1.1532e-03, -1.3981e-03, -8.3499e-04],
  [ 3.0656e-04, -1.2571e-04, -7.7346e-04, -1.0842e-03, -6.1649e-04],
  [ 6.7605e-04,  5.0347e-04, -9.5407e-06, -1.8676e-04, -3.9528e-05]],

 …,

 [[ 1.2319e-03,  6.4799e-04,  1.8489e-04, -8.5958e-05,  6.5311e-06],
  [ 1.2986e-03,  7.5737e-04,  5.1374e-05, -7.6222e-05, -5.3957e-05],
  [ 1.0320e-03,  8.9892e-04,  6.1504e-04,  1.8119e-04,  2.2628e-04],
  [ 1.3369e-03,  1.2765e-03,  8.2307e-04,  3.2388e-04,  6.4983e-04],
  [ 1.5886e-03,  1.5861e-03,  1.3506e-03,  6.3119e-04,  7.7373e-04]],

 [[ 1.8147e-03,  1.0418e-03,  8.4681e-04,  3.2694e-04,  7.5801e-04],
  [ 1.7526e-03,  1.2341e-03,  8.8121e-04,  6.4897e-04,  4.3739e-04],
  [ 1.3993e-03,  1.4959e-03,  1.3529e-03,  8.6894e-04,  6.7388e-04],
  [ 1.3586e-03,  1.3981e-03,  1.1986e-03,  8.9521e-04,  1.1829e-03],
  [ 1.5472e-03,  1.4824e-03,  1.2907e-03,  6.4863e-04,  7.6207e-04]],

 [[ 1.2551e-03,  6.5099e-04,  2.8229e-04, -1.1033e-04, -1.5653e-04],
  [ 1.1899e-03,  7.5997e-04,  1.6742e-04, -8.7481e-05, -1.2827e-04],
  [ 7.0737e-04,  6.5302e-04,  7.4005e-04,  4.6817e-04,  1.6728e-04],
  [ 6.6976e-04,  7.2720e-04,  6.5417e-04,  4.7147e-04,  6.1995e-04],
  [ 9.0667e-04,  9.1788e-04,  8.7014e-04,  6.6575e-04,  6.6554e-04]]],


…,


[[[-3.3481e-04, -4.7716e-04, -4.4572e-04, -4.8802e-04, -7.6148e-04],
  [-5.3168e-04, -7.1165e-04, -7.7579e-04, -8.2802e-04, -8.0049e-04],
  [-5.2855e-04, -6.4008e-04, -7.6347e-04, -7.4377e-04, -8.8087e-04],
  [-3.5617e-04, -4.4489e-04, -4.9790e-04, -4.2328e-04, -5.3708e-04],
  [-3.1719e-04, -3.7324e-04, -2.7459e-04, -2.8202e-04, -4.7575e-04]],

 [[-8.6578e-04, -5.2165e-04, -1.6094e-04,  2.8207e-04,  4.2266e-04],
  [-3.6006e-04,  6.8821e-05,  3.1732e-04,  5.7476e-04,  7.0974e-04],
  [-3.4141e-04,  6.5325e-05,  3.8616e-04,  5.9181e-04,  5.7766e-04],
  [-9.6418e-05, -2.8522e-05,  3.7700e-04,  5.7253e-04,  6.1074e-04],
  [-2.9477e-04, -2.2776e-04,  1.1260e-04,  3.3045e-04,  5.2719e-04]],
```

```
 [[-4.8534e-05,  1.6304e-04,  3.1822e-04,  4.2243e-04,  9.4800e-04],
  [-1.1901e-04, -1.2758e-05,  1.4663e-04,  6.1446e-04,  1.0881e-03],
  [-2.2773e-04,  6.2087e-05,  4.2371e-04,  6.8024e-04,  1.1903e-03],
  [-1.9442e-04,  1.6656e-04,  4.4874e-04,  9.3464e-04,  1.3717e-03],
  [-1.6840e-04,  1.7568e-04,  4.7342e-04,  9.0854e-04,  9.6755e-04]],

 …,

 [[-4.2288e-04, -3.6852e-04, -3.6153e-04, -4.1105e-04, -6.2605e-05],
  [-5.9148e-04, -6.1359e-04, -3.6080e-04, -8.6176e-05,  1.3858e-04],
  [-5.6438e-04, -5.2798e-04, -3.0375e-04, -9.1932e-05,  1.8223e-04],
  [-3.5087e-04, -2.4244e-04, -3.6965e-05,  9.1601e-05,  2.0466e-04],
  [-2.2764e-04, -2.3992e-04,  5.0105e-05,  1.0134e-04, -3.8828e-05]],

 [[-6.8558e-04, -6.5673e-04, -6.4003e-04, -6.4131e-04, -7.3253e-04],
  [-8.3884e-04, -7.8689e-04, -6.9329e-04, -6.5455e-04, -5.6599e-04],
  [-8.0117e-04, -8.5334e-04, -7.1373e-04, -5.6254e-04, -6.2909e-04],
  [-5.4259e-04, -5.7464e-04, -4.2194e-04, -4.0060e-04, -5.3012e-04],
  [-3.7649e-04, -4.9510e-04, -3.9553e-04, -3.6542e-04, -5.3072e-04]],

 [[-2.1315e-04, -4.3189e-04, -3.9425e-04, -3.8503e-04, -2.9138e-04],
  [-4.7202e-04, -5.3824e-04, -2.2812e-04, -1.6252e-04, -8.4095e-05],
  [-3.5248e-04, -3.5939e-04, -2.8733e-04, -1.9059e-04, -2.2689e-04],
  [-1.5099e-04, -7.3164e-05, -9.4582e-05, -1.1240e-04, -3.8397e-04],
  [-2.3613e-05, -4.0836e-05, -5.2353e-05, -7.6000e-05, -3.7320e-04]]],


[[[ 1.5988e-04,  3.7264e-04,  4.1411e-04,  9.1334e-04,  7.0816e-04],
  [ 2.1620e-04,  3.8503e-04,  2.9009e-04,  7.6949e-04,  1.0349e-03],
  [ 2.2651e-04,  1.9714e-04,  3.3761e-04,  8.5875e-04,  1.1751e-03],
  [ 8.4217e-05,  3.1123e-04,  2.5640e-04,  3.8451e-04,  7.6042e-04],
  [ 1.0380e-04,  2.8358e-04,  3.2144e-04,  1.3727e-04,  5.0206e-04]],

 [[-7.0341e-04, -1.3767e-03, -1.7044e-03, -1.8731e-03, -1.6603e-03],
  [-6.1906e-04, -9.8691e-04, -1.6291e-03, -1.7979e-03, -1.4332e-03],
  [-4.2249e-04, -7.3283e-04, -9.7856e-04, -1.3588e-03, -1.3368e-03],
  [ 1.3070e-04, -5.2197e-04, -1.1632e-03, -1.3849e-03, -1.3320e-03],
  [ 1.0341e-04, -5.0951e-04, -1.0079e-03, -1.3840e-03, -1.4014e-03]],

 [[-5.5006e-04, -7.5223e-04, -1.0275e-03, -1.0623e-03, -1.1265e-03],
  [-2.7649e-04, -3.0257e-04, -3.7312e-04, -7.6672e-04, -7.9882e-04],
  [-1.3165e-04,  2.6457e-04,  2.3698e-04, -1.7141e-04, -4.1508e-04],
  [-1.6267e-05,  8.7666e-05,  9.6709e-06, -2.9101e-04, -4.3934e-04],
  [ 3.6227e-04,  1.0062e-04, -2.0602e-04, -2.4566e-04, -4.1369e-04]],

 …,
```

```
 [[ 7.4011e-04,  7.9510e-04,  4.5600e-04,  7.0931e-04,  8.4564e-04],
  [ 6.0291e-04,  8.6495e-04,  6.7022e-04,  6.5446e-04,  8.5241e-04],
  [ 8.2092e-04,  9.8769e-04,  9.0945e-04,  6.9932e-04,  9.7126e-04],
  [ 4.9257e-04,  7.1755e-04,  5.9022e-04,  2.7215e-04,  3.7896e-04],
  [ 6.7924e-04,  5.5824e-04,  5.6364e-04,  4.4715e-04,  4.7625e-04]],

 [[ 1.1425e-03,  1.1531e-03,  8.4597e-04,  1.3453e-03,  1.1607e-03],
  [ 9.7119e-04,  1.1312e-03,  7.9795e-04,  1.2116e-03,  1.3862e-03],
  [ 1.0829e-03,  1.0042e-03,  9.4964e-04,  1.0686e-03,  1.3390e-03],
  [ 6.9711e-04,  8.9158e-04,  6.8991e-04,  5.7537e-04,  7.0731e-04],
  [ 6.4321e-04,  7.0173e-04,  6.3433e-04,  5.7007e-04,  6.8185e-04]],

 [[ 5.0217e-04,  6.8529e-04,  5.4692e-04,  7.8182e-04,  8.9542e-04],
  [ 1.6236e-04,  3.6404e-04,  6.3416e-04,  7.1748e-04,  9.2620e-04],
  [ 2.6650e-04,  3.8511e-04,  3.4969e-04,  5.1671e-04,  7.3657e-04],
  [ 1.4111e-04,  2.1517e-04,  1.2672e-04,  1.8554e-04,  4.3281e-04],
  [ 2.7428e-04,  2.9363e-04,  3.4510e-04,  2.7784e-04,  4.7259e-04]]],


[[[-3.0359e-04, -3.1950e-04, -4.5444e-04, -3.1410e-04, -6.0282e-04],
  [-2.9784e-04, -2.7503e-04, -2.4128e-04, -1.9880e-04, -3.3545e-04],
  [-2.4461e-04, -1.5032e-04, -1.3840e-04, -2.5018e-04, -3.7827e-04],
  [ 2.4863e-05,  6.8289e-05,  8.6052e-05, -8.9667e-05, -1.1398e-04],
  [ 1.4455e-04,  1.1052e-04,  1.9416e-04,  4.2587e-05,  1.2252e-04]],

 [[-1.4697e-04, -9.0722e-06, -2.0495e-04,  4.9011e-05, -1.2603e-04],
  [-2.7953e-04, -2.2005e-04, -2.2176e-04, -1.2926e-04, -1.2895e-04],
  [-3.7980e-04, -4.1404e-04, -3.4115e-04, -2.2033e-04, -1.3219e-04],
  [-4.1836e-04, -4.5676e-04, -1.7226e-04, -1.0418e-04, -2.1312e-04],
  [-5.6169e-04, -7.4662e-04, -4.4080e-04, -4.3815e-04, -4.8282e-04]],

 [[ 3.7070e-04,  3.2524e-04,  5.5918e-04,  8.8648e-04,  1.0606e-03],
  [ 6.0394e-04,  4.6666e-04,  5.3469e-04,  7.5276e-04,  8.7198e-04],
  [ 8.4612e-04,  7.6439e-04,  7.0453e-04,  7.0328e-04,  7.1527e-04],
  [ 6.9632e-04,  9.5859e-04,  9.6987e-04,  8.3115e-04,  5.4158e-04],
  [ 5.7176e-04,  6.3302e-04,  7.6161e-04,  5.5638e-04,  9.5147e-05]],

 …,

 [[-2.1446e-04, -2.2111e-04, -1.6224e-04,  2.0354e-05, -4.9607e-05],
  [-8.6342e-05, -1.2053e-04, -7.0655e-05,  3.1437e-05,  1.4128e-04],
  [ 4.1086e-05,  6.0647e-05,  1.2836e-05, -2.8960e-05,  2.1509e-05],
  [ 2.4888e-04,  3.3140e-04,  2.3260e-04,  9.1844e-05, -2.1596e-05],
  [ 8.8127e-05,  2.6586e-04,  1.9853e-04, -3.0165e-05, -2.9169e-04]],

 [[-3.8843e-04, -3.5990e-04, -4.5934e-04, -3.4871e-04, -7.4357e-04],
  [-3.9889e-04, -3.2195e-04, -2.8256e-04, -2.4593e-04, -2.5100e-04],
  [-4.3735e-04, -3.4868e-04, -3.2274e-04, -3.8572e-04, -4.0985e-04],
```

```
          [-5.1201e-05, -2.0308e-05, -4.6101e-05, -1.2341e-04, -2.1703e-04],
          [-5.3318e-05,  5.3832e-06,  5.3882e-05, -3.9969e-05, -2.1113e-04]],

         [[-4.5451e-04, -4.3874e-04, -4.1833e-04, -3.5433e-04, -5.5299e-04],
          [-4.4021e-04, -4.3456e-04, -3.6599e-04, -3.4337e-04, -3.3357e-04],
          [-4.0795e-04, -3.2295e-04, -3.3444e-04, -2.6289e-04, -3.0010e-04],
          [-2.7951e-05,  9.8525e-05,  4.6469e-06, -1.5497e-04, -3.4414e-04],
          [-1.9812e-05, -3.0523e-05, -1.0255e-04, -2.3589e-04, -2.9436e-04]]])
tensor([ 1.1823e-09, -6.4347e-11, -2.3797e-09, -1.3370e-09,  5.9461e-09,
        -4.9152e-10,  9.2064e-10,  1.2099e-08, -1.7503e-09,  1.7178e-09,
         1.3716e-08, -2.1960e-09,  7.4567e-10,  5.1523e-09, -1.1346e-09,
        -1.9036e-09,  1.0323e-10,  1.2105e-09, -1.5036e-10,  1.2067e-08,
        -7.7671e-10, -5.9411e-10, -2.1737e-10,  6.6757e-10,  8.6351e-10,
        -2.7864e-09,  3.7153e-10, -2.4765e-09, -9.5331e-09, -1.6811e-08,
        -4.4020e-10, -4.7085e-09, -1.8413e-09, -3.3775e-09, -4.4474e-10,
         5.3842e-10,  2.2010e-09, -1.5771e-09,  4.1339e-10, -1.3587e-09,
         9.4803e-10, -5.3763e-09, -1.3024e-09,  2.2737e-09,  1.2551e-10,
         1.1188e-08, -1.1878e-09,  9.4442e-09,  1.3347e-08,  2.2537e-10,
        -1.8006e-09, -3.3415e-09,  1.7635e-08, -7.1041e-09,  1.0289e-09,
        -5.6564e-09, -1.7452e-09,  5.7310e-10,  1.6568e-09,  1.6241e-09,
         3.0968e-09,  9.0643e-10,  1.6507e-09,  9.7157e-10, -7.8603e-10,
        -1.8827e-10, -4.1609e-10, -3.9441e-09, -5.3678e-09,  5.4723e-10,
         4.8146e-11,  2.4432e-09, -1.4165e-09,  2.3438e-09,  4.4845e-09,
         1.3201e-08,  5.3318e-10, -3.4083e-10,  2.2209e-10, -1.2478e-09,
         2.5747e-09,  2.6619e-09, -1.5568e-09, -1.3165e-10,  3.1764e-09,
         3.9171e-09, -2.0385e-09, -6.4855e-09, -1.2134e-09, -2.0196e-09,
        -2.5128e-09,  1.3392e-10,  4.9131e-09,  8.7795e-10, -6.4662e-10,
         5.8578e-09,  6.8530e-10,  3.2742e-11, -4.8589e-10, -1.5916e-12,
        -3.2220e-09, -2.5993e-09, -2.3835e-09,  2.9028e-09, -6.2816e-09,
        -7.3603e-09,  1.8742e-09,  1.3674e-09, -1.0323e-10, -4.8551e-09,
         1.1396e-09,  8.0615e-10, -5.6218e-10,  1.1196e-09,  5.3802e-11,
        -8.3936e-10,  1.3665e-09,  7.9785e-10,  2.4356e-09, -9.0752e-10,
        -1.5469e-09,  1.7064e-10, -2.6012e-10,  2.4465e-09,  3.6205e-09,
         1.1660e-09, -1.0978e-08, -3.1455e-09, -2.6368e-09,  1.1210e-10,
         3.5607e-10, -1.1014e-09,  5.4752e-10, -2.5317e-09, -9.5224e-10,
        -1.0591e-09, -3.9930e-09,  1.9704e-08, -3.5061e-09,  1.9511e-08,
        -6.4100e-09,  5.7982e-10,  1.1672e-09,  1.7337e-09,  9.2314e-10,
         1.9759e-09,  3.9929e-09,  2.0799e-09, -4.8871e-09,  3.5372e-10,
         1.2660e-09,  3.9199e-10,  4.8426e-09, -1.1990e-09, -2.1908e-08,
         1.4846e-09,  3.0104e-10,  7.5306e-10,  4.9945e-09,  1.3531e-09,
         3.9801e-09,  3.0648e-09, -1.2340e-09,  1.1182e-09, -3.7950e-09,
         2.5966e-10,  5.7528e-09, -2.4152e-09, -4.3542e-09,  1.4144e-09,
         5.2982e-09,  1.7984e-09,  3.2924e-10, -4.4127e-09, -5.2198e-09,
        -1.2064e-09,  2.4592e-09,  1.0714e-09, -2.2455e-09,  2.4102e-11,
         1.6371e-11,  1.1453e-09,  1.2442e-09,  6.0142e-10, -1.1419e-09,
        -1.0381e-09, -2.7671e-09,  1.0686e-09, -2.3611e-09,  7.7807e-10,
         1.5357e-09, -2.4916e-09,  2.8972e-09, -1.7755e-10,  1.2274e-09,
         2.6284e-10, -1.8094e-09, -2.1239e-09,  1.6599e-09,  1.2881e-09,
```

```
         4.2904e-09,  9.5542e-10, -2.0797e-08, -2.2878e-09, -1.5065e-09,
        -1.7830e-10,  1.6498e-09, -4.1500e-09, -1.2933e-09, -6.2118e-10,
         1.6163e-09, -6.9249e-10, -1.4160e-09,  6.4390e-10, -1.8231e-09,
        -3.5491e-09, -1.9154e-09, -3.5900e-09,  1.3270e-09, -7.0941e-11,
         1.8122e-10,  7.3458e-09, -4.8922e-09, -2.9093e-09, -1.9558e-09,
        -1.1651e-09, -1.4893e-10,  7.6071e-09,  5.5479e-11, -5.2807e-09,
         1.9298e-09,  1.8447e-09,  5.2899e-10,  4.7395e-10, -2.8194e-10,
         2.9967e-09,  1.7874e-10,  2.9537e-09,  5.8025e-10, -7.8053e-09,
         2.5636e-09, -8.2591e-09, -6.4266e-09,  1.0982e-09,  2.6575e-09,
         8.6575e-10, -7.6989e-10, -3.9454e-09,  3.9381e-10, -2.5350e-09,
         1.8481e-09, -4.4594e-10, -2.3356e-09, -8.9489e-10,  5.9390e-10,
         2.0873e-10])
tensor([ 5.3096e-03, -1.9001e-03,  6.2270e-03, -7.7724e-03, -2.1428e-04,
        -2.7535e-03,  4.0561e-03,  8.0327e-03,  1.6805e-02, -2.8370e-03,
        -2.1867e-02,  4.4438e-04,  9.7561e-03, -9.2851e-03, -1.1236e-03,
        -6.6711e-03,  2.7338e-03,  2.8068e-03, -8.4909e-03, -3.6101e-03,
        -3.8669e-04, -2.6543e-02, -4.4224e-03,  5.6594e-04, -8.9277e-04,
         1.9539e-03,  2.1124e-02, -4.5516e-02,  2.1405e-02, -7.3315e-03,
         2.7244e-04, -4.5213e-03, -6.7913e-03, -5.5799e-03,  5.6325e-03,
         1.0676e-03,  1.6719e-03,  4.5602e-03, -9.6154e-03, -1.1171e-03,
        -2.8695e-03, -2.8426e-03,  4.0807e-03,  1.0269e-02,  1.9893e-03,
        -6.7958e-02, -6.1221e-03,  2.6116e-02, -1.9886e-02,  4.4014e-03,
        -4.0310e-03,  9.3755e-03,  1.6385e-02,  1.7003e-02,  7.5198e-03,
        -1.2267e-02, -8.3750e-03,  2.0169e-02,  1.8635e-02, -1.1353e-03,
         5.1904e-03,  3.0673e-03, -6.8708e-03,  2.2349e-03,  2.0905e-03,
         2.3605e-04,  7.4219e-04,  2.3043e-03, -5.2209e-02,  2.4963e-03,
         1.3522e-02,  2.4416e-02,  5.0387e-03, -6.7971e-03,  2.9534e-02,
        -2.8664e-02, -5.1806e-03, -7.2233e-04,  4.4601e-03,  5.7965e-03,
         1.3871e-02,  2.4184e-02,  5.0813e-03, -1.8117e-03,  1.5335e-03,
         1.7555e-02,  3.9605e-03, -1.8033e-02,  4.0116e-04,  2.2542e-03,
         1.1259e-02, -1.1295e-03,  2.1854e-02,  7.6512e-03, -2.0737e-02,
         4.4165e-02,  8.9358e-04, -4.7995e-04, -7.2200e-05, -4.3733e-03,
         2.2638e-03,  4.8564e-04, -1.2530e-03, -1.8616e-03,  1.9264e-03,
         1.7157e-02, -3.9007e-03, -7.1345e-03,  3.7150e-03,  8.5317e-03,
        -1.5668e-03, -6.5744e-04,  2.7477e-03,  5.0679e-03, -4.8104e-03,
         2.9736e-03,  2.7957e-03, -1.4952e-03, -5.4387e-03,  1.1504e-02,
        -9.4069e-03,  3.0096e-03,  1.7185e-04,  6.8848e-03, -3.0157e-03,
        -5.4756e-03,  2.4049e-02, -5.2046e-03,  7.2439e-03,  2.5020e-02,
         2.4084e-03,  3.9390e-03,  4.4648e-03,  3.2982e-03, -1.4041e-03,
         1.9200e-02,  6.9131e-03, -1.1788e-02, -3.0583e-03,  9.0949e-03,
         2.6671e-03,  7.4246e-03,  1.6537e-02,  3.2244e-04, -4.5540e-03,
         3.8236e-03,  2.8542e-02,  2.4248e-03,  3.2542e-03,  5.9971e-04,
         3.1721e-03, -2.3897e-03, -1.0911e-02, -2.3896e-02, -6.0194e-03,
         9.1734e-04, -6.3632e-04, -1.9099e-03,  1.8173e-02, -9.4180e-03,
         1.5019e-02, -7.1647e-03, -9.8245e-03, -2.8501e-03,  6.8185e-03,
         5.8309e-03, -2.7906e-03, -4.9672e-03,  8.7823e-03,  2.2465e-03,
         8.2674e-03, -6.5391e-03, -2.8354e-03,  3.8530e-03, -3.0674e-02,
        -8.4318e-04,  2.6261e-03, -2.7095e-03,  2.6472e-03, -1.7207e-03,
```

```
        -3.4739e-03,  1.5540e-03, -5.1644e-04, -7.3538e-03,  4.4896e-03,
         3.6943e-03, -8.7276e-03, -5.0909e-03, -3.0100e-03, -4.4701e-03,
         6.7981e-03, -2.3196e-04, -4.2522e-03, -1.8715e-02, -6.0570e-04,
         1.9900e-03, -5.7686e-03,  3.1997e-03,  1.6876e-03,  7.8129e-04,
         8.1596e-03, -4.8910e-03, -9.8782e-03,  4.3998e-03,  4.0972e-03,
        -1.2494e-03, -7.1779e-03, -1.2203e-04, -9.4726e-04,  4.4668e-03,
         1.6927e-03, -4.7749e-03,  8.2466e-03,  5.0639e-03, -6.6762e-03,
         1.1957e-03, -7.6155e-03,  8.1163e-03, -1.8218e-03,  2.3453e-03,
         2.0861e-03,  1.5632e-02, -3.6543e-02, -7.0090e-03, -5.9502e-02,
         4.3847e-03, -3.8393e-03,  1.6826e-02,  5.1819e-04, -4.3613e-02,
         1.2453e-02,  7.2529e-04, -1.3024e-02, -1.5627e-02,  9.8761e-03,
         2.4418e-02,  2.1381e-02, -1.4891e-02,  1.4100e-03, -2.2607e-03,
        -8.7227e-04,  9.0689e-03, -1.1606e-03,  2.4432e-03, -1.2351e-02,
         4.7925e-03, -2.1896e-02,  5.4568e-03,  6.7762e-03, -3.3232e-02,
        -6.3474e-03,  7.6166e-03, -4.4930e-03, -6.5571e-03,  7.8927e-04,
        -4.0634e-03])
tensor([ 1.2298e-02, -4.6716e-03,  1.1779e-02, -2.5705e-03, -2.2122e-04,
         6.5277e-04,  2.8086e-03, -1.4230e-02,  2.2734e-02, -6.1334e-03,
        -1.0207e-02,  4.9505e-03,  1.0694e-02, -5.4062e-03, -1.9214e-04,
        -5.4182e-03,  3.8715e-03, -6.4460e-04, -1.9435e-03,  3.4604e-03,
        -3.6722e-03, -2.1970e-02, -9.7907e-03, -4.5132e-04, -4.6340e-04,
         1.8862e-03,  1.7586e-02, -3.2666e-02,  2.1887e-02, -1.0426e-02,
        -6.1118e-04, -7.4320e-03, -1.0746e-02, -6.9226e-03,  1.0382e-02,
         5.8186e-03,  2.1603e-03,  4.3280e-03, -6.6991e-03,  1.1821e-03,
        -8.9790e-04, -1.2872e-02,  5.4572e-03,  1.3333e-02,  2.6204e-04,
        -4.5493e-02, -7.5877e-03,  5.0031e-03, -1.5532e-02,  2.4024e-03,
        -7.0093e-03,  7.4639e-03,  8.3033e-03,  3.2985e-03, -3.2572e-03,
        -5.6456e-03, -7.8494e-03,  1.4461e-02,  1.5113e-02, -5.7955e-03,
         8.7851e-03,  2.9833e-03, -7.1363e-03,  5.5262e-03,  9.4958e-04,
        -4.5736e-03,  4.3617e-03,  6.1037e-03, -4.6233e-02, -1.2462e-03,
         1.4612e-02,  1.9393e-02,  3.8420e-03, -4.8085e-03,  2.3945e-02,
        -2.5587e-02, -1.4840e-03, -2.3638e-03,  5.5587e-03,  6.8897e-03,
         6.9223e-03,  1.0860e-02,  6.9699e-03, -1.4050e-03,  6.6133e-03,
         2.5614e-02,  3.1947e-03, -2.1084e-02,  1.7615e-03,  4.3470e-03,
         6.8270e-03,  3.0079e-04,  1.9924e-02,  1.3601e-02, -1.6145e-02,
         4.1771e-02,  3.2735e-03, -2.2991e-03, -7.2674e-04, -8.3694e-03,
         2.0055e-03, -1.5701e-04, -7.0601e-03, -3.5568e-03,  2.8401e-03,
         7.1293e-03, -5.9152e-03, -6.5698e-03,  8.1607e-03, -1.1846e-03,
        -3.0188e-03,  1.2792e-03,  6.9828e-03,  7.6276e-03, -7.5214e-03,
         4.5006e-03, -2.8224e-04,  7.5309e-04, -9.3066e-03,  6.4291e-03,
         2.6251e-03,  5.1261e-03, -8.6928e-04,  4.3604e-03, -3.6715e-04,
        -3.7478e-03,  2.3356e-02, -1.5085e-02,  9.4275e-03,  1.5794e-02,
         5.8568e-03,  8.8721e-03,  5.7916e-03,  6.6222e-04,  2.9122e-03,
         1.6290e-02,  1.6437e-02, -2.6452e-02, -3.8780e-03, -9.6936e-03,
         3.2038e-03,  5.0789e-03,  1.4644e-02, -7.1704e-03, -1.0381e-02,
         5.0198e-03,  2.3067e-02, -5.3173e-03,  1.6548e-02, -6.1946e-04,
         4.4576e-03, -4.9907e-04, -4.8302e-03, -2.1049e-02, -1.6212e-03,
         3.8228e-03,  1.0831e-03, -4.9894e-03,  3.1216e-03, -1.4626e-02,
```

```
         1.2551e-02, -1.6325e-02, -1.0790e-02, -3.4511e-03,  1.0057e-02,
         8.9286e-03, -5.7236e-03, -8.4687e-03,  8.2854e-03,  4.7779e-03,
        -7.3674e-03, -6.8493e-03, -4.0052e-03,  1.2472e-02, -1.6084e-02,
        -1.0850e-03,  3.0311e-03, -6.0045e-03, -4.8669e-04, -2.8592e-03,
        -6.7094e-03,  4.4179e-03, -3.4016e-03, -3.8443e-03,  7.9503e-03,
         1.8063e-03,  5.3089e-03, -9.1160e-03, -8.1203e-03, -9.7443e-03,
         8.1446e-03, -7.6889e-03, -1.1110e-02, -2.5448e-02, -1.5793e-03,
         4.5671e-03, -1.1667e-02,  9.3894e-03,  2.2022e-03, -7.8819e-03,
         1.3598e-02, -5.7460e-03, -1.0976e-02,  1.2423e-02,  3.7253e-03,
         4.2257e-03, -6.3391e-03, -3.2630e-03, -3.8124e-03,  7.8098e-03,
         3.0087e-03, -4.4573e-03,  2.5115e-03,  2.7541e-03, -9.3501e-03,
         6.6008e-03, -1.6540e-02,  6.1930e-03, -5.4260e-03, -1.2075e-04,
         2.5479e-04,  7.0224e-03, -2.8353e-02, -5.8673e-03, -3.4817e-02,
         4.7556e-03, -6.7870e-03,  1.3870e-02,  1.1929e-03, -2.7976e-02,
         5.4744e-03,  1.7780e-03, -1.0888e-02, -1.5561e-02,  7.5545e-03,
         1.2502e-02,  1.7245e-02, -1.4715e-02, -1.0213e-03, -2.2815e-03,
        -4.7375e-03, -1.6267e-03, -4.1841e-05,  4.5099e-03, -9.6629e-03,
         5.5752e-03, -1.1712e-02,  1.5741e-03,  9.6286e-03, -1.9071e-02,
        -9.3298e-03,  1.4463e-02, -9.7888e-03, -3.6331e-04, -2.6705e-03,
        -6.6928e-03])
tensor([[[[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         …,

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]]],


        [[[ 0.0000,  0.0000,  0.0000],
```

```
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],
```

```
 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


…,


[[[-0.0333, -0.0321, -0.0322],
  [-0.0326, -0.0254, -0.0223],
  [-0.0116, -0.0043, -0.0083]],

 [[ 0.0009,  0.0137,  0.0213],
  [ 0.0017,  0.0137,  0.0194],
  [ 0.0041,  0.0109,  0.0141]],

 [[-0.0112, -0.0041, -0.0033],
  [-0.0168, -0.0086, -0.0069],
  [-0.0070,  0.0006,  0.0010]],

 …,

 [[-0.0092,  0.0039,  0.0166],
  [-0.0095,  0.0038,  0.0156],
  [-0.0045,  0.0075,  0.0152]],

 [[-0.0040,  0.0131,  0.0267],
  [-0.0042,  0.0085,  0.0179],
  [ 0.0134,  0.0180,  0.0188]],

 [[-0.0082,  0.0009,  0.0026],
  [-0.0072,  0.0005, -0.0014],
  [-0.0013,  0.0074,  0.0045]]],


[[[-0.0334, -0.0312, -0.0300],
  [-0.0419, -0.0383, -0.0335],
  [-0.0340, -0.0308, -0.0282]],

 [[-0.0475, -0.0331, -0.0374],
  [-0.0331, -0.0188, -0.0254],
  [-0.0358, -0.0237, -0.0246]],

 [[-0.0353, -0.0222, -0.0258],
  [-0.0304, -0.0207, -0.0247],
  [-0.0352, -0.0220, -0.0246]],

 …,
```

```
          [[-0.0645, -0.0454, -0.0365],
           [-0.0453, -0.0281, -0.0278],
           [-0.0358, -0.0286, -0.0339]],

          [[-0.0563, -0.0360, -0.0367],
           [-0.0463, -0.0299, -0.0290],
           [-0.0487, -0.0341, -0.0292]],

          [[-0.0322, -0.0207, -0.0286],
           [-0.0284, -0.0218, -0.0315],
           [-0.0326, -0.0271, -0.0341]]],


         [[[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          …,

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]]]])
tensor([ 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  1.7447e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -1.8931e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  2.4631e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -4.1003e-04,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

```
0.0000e+00, -1.3339e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  1.2262e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  1.2438e-03,  0.0000e+00,  2.3036e-03,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -6.1234e-06,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -2.8452e-02,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  4.0461e-02,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00, -1.4987e-02,  0.0000e+00,
0.0000e+00, -3.1133e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00, -6.1635e-05,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
4.9196e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

```
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          1.6471e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -4.6884e-07,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  6.7202e-02,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  6.8064e-04,  0.0000e+00,
          0.0000e+00,  0.0000e+00, -2.7943e-03,  0.0000e+00,  7.2165e-05,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00, -6.0151e-03,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -1.6620e-03, -5.2466e-04,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  5.2111e-02, -1.6275e-05,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  5.8198e-05,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -3.1834e-02, -9.8098e-02,  0.0000e+00])
tensor([[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[-3.1704e-02, -1.8289e-02, -1.8449e-02],
           [-2.2088e-02, -1.2385e-02, -1.8489e-02],
           [-9.1146e-03, -2.7238e-02, -2.5195e-02]],

          [[-9.3675e-02, -9.6923e-02, -4.7851e-02],
           [-1.3656e-01, -9.6401e-02, -1.3635e-02],
           [-2.1402e-01, -1.3630e-01, -1.6479e-02]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


         [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[-2.0084e-03, -5.2984e-03, -2.3676e-04],
  [-3.2935e-04, -1.3650e-03,  4.2119e-03],
  [ 2.3693e-04,  1.9450e-04,  2.8512e-03]],

 [[ 1.2512e-04, -5.7362e-05, -1.3499e-03],
  [-8.0715e-04, -1.0871e-03, -5.0946e-04],
  [ 1.1052e-03, -2.2319e-03,  1.9262e-03]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[-9.7222e-05, -1.1651e-04, -1.4049e-04],
  [-1.0653e-04, -1.5960e-04, -2.0583e-04],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[-2.5481e-02, -4.8785e-02, -2.6264e-02],
  [-3.4360e-02, -4.1081e-02, -1.2406e-02],
  [-2.9812e-02, -2.3308e-02, -3.7828e-03]],

 [[-1.9605e-02, -3.7091e-02, -4.1731e-02],
  [-3.4812e-02, -6.1652e-02, -4.9825e-02],
  [-5.7235e-02, -7.7974e-02, -5.3466e-02]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,
```

```
          [[ 9.0898e-06,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  7.9379e-05],
           [-4.7860e-05,  1.0564e-04,  1.2872e-04],
           [ 1.7575e-06,  1.0201e-04,  5.0385e-05]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


         [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([-3.6688e-02,  1.7949e-03, -9.4679e-05, -1.6276e-02, -1.0441e-01,
         2.4901e-02,  5.7236e-03, -1.2466e-02, -2.0287e-02, -7.7511e-04,
         1.1194e-03,  7.4595e-03,  3.6302e-02,  1.8802e-04,  1.0210e-03,
         1.2453e-02,  0.0000e+00, -1.2190e-04,  0.0000e+00, -1.3177e-03,
         8.9530e-03,  5.9656e-03,  2.2730e-04,  4.8155e-03, -1.6002e-04,
         1.7900e-03,  0.0000e+00,  2.9993e-02,  0.0000e+00,  4.3829e-03,
         8.3705e-02,  8.5011e-03,  8.3700e-03,  0.0000e+00, -7.2058e-04,
        -6.4231e-03,  3.5875e-02, -4.1734e-02,  3.5006e-02,  6.5519e-03,
         8.1789e-03,  7.3645e-03,  5.2046e-03, -8.7958e-03, -1.5092e-04,
```

```
 2.8620e-02, -3.1427e-03, -5.2394e-03,  4.9776e-02, -6.7485e-04,
 0.0000e+00, -6.0785e-03,  3.3350e-03, -6.8140e-04, -9.2122e-02,
-1.9854e-04,  1.5898e-02,  8.1682e-04, -8.3332e-03,  5.9681e-05,
-1.2594e-04,  0.0000e+00, -1.1950e-03, -2.4500e-04, -2.2537e-02,
 1.3456e-02,  2.9011e-02,  2.7824e-02,  7.2991e-02,  1.1666e-01,
 8.8851e-04,  0.0000e+00,  1.4881e-03,  7.5394e-04, -7.0545e-05,
-1.9551e-03,  3.0105e-03,  2.5790e-03, -2.4938e-03, -9.9216e-03,
-3.6047e-04,  0.0000e+00,  2.0182e-03, -3.5090e-05,  6.4071e-04,
 1.8768e-02, -3.9323e-02,  1.1066e-02,  9.3855e-03,  0.0000e+00,
-9.1942e-03, -1.5717e-03,  0.0000e+00,  0.0000e+00,  4.7292e-04,
 0.0000e+00,  2.0729e-02, -1.9815e-02,  0.0000e+00,  1.4093e-03,
 2.9864e-02,  1.9301e-03, -2.8237e-02, -1.9080e-02, -9.9993e-04,
-1.0194e-02, -4.6790e-03,  4.2323e-04,  7.2203e-04,  1.2902e-01,
 5.2093e-05,  2.5722e-03, -2.0182e-02, -5.0711e-03,  6.9529e-02,
 1.2286e-02,  3.8628e-02, -3.6633e-04, -8.7094e-05, -1.6841e-04,
 3.8112e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.7254e-03,
 0.0000e+00, -8.9385e-03,  1.2173e-03,  3.1355e-04, -3.1175e-02,
 0.0000e+00,  0.0000e+00, -5.8113e-05,  1.9003e-02,  5.3742e-03,
 5.3778e-02,  6.3448e-05, -3.9507e-04,  3.5264e-03,  1.9077e-04,
-1.6022e-03,  1.4195e-01,  7.6219e-02,  9.5724e-03,  0.0000e+00,
 0.0000e+00,  0.0000e+00, -8.2885e-03, -1.9078e-02,  2.4347e-04,
 0.0000e+00, -8.8182e-03,  5.0396e-03,  7.6603e-05, -3.9055e-04,
-5.9882e-02,  1.3434e-05, -1.6449e-04, -9.2413e-04, -1.3047e-01,
-5.8904e-05,  1.4744e-02,  0.0000e+00,  0.0000e+00,  3.3231e-02,
-2.8644e-05, -8.8521e-05,  7.4493e-03,  1.1403e-03,  1.3838e-02,
-1.2623e-02, -8.8776e-05,  4.4203e-03, -2.3810e-04, -3.2775e-02,
 5.2379e-02, -8.5620e-05, -5.7020e-03, -6.3701e-03,  4.4627e-03,
 4.3701e-04, -2.0382e-02, -2.5573e-03,  4.3980e-03, -1.2566e-02,
 0.0000e+00, -3.0091e-03,  4.5744e-03, -2.6598e-04,  8.3940e-03,
 8.9488e-04,  0.0000e+00,  0.0000e+00, -2.5657e-02,  3.5472e-03,
-1.5685e-04, -5.2959e-02, -3.1305e-03,  2.3277e-04, -2.5670e-03,
 6.9298e-02,  2.9660e-02, -2.4754e-05,  1.1560e-03, -2.7598e-03,
-2.2093e-04,  1.4064e-02, -7.6413e-03,  3.9433e-03, -1.2216e-03,
-1.5667e-03, -5.2604e-03,  4.1201e-03,  5.7967e-04, -3.0129e-02,
-5.6824e-03, -2.5456e-04,  0.0000e+00,  3.6760e-04,  0.0000e+00,
 0.0000e+00,  7.9789e-05, -9.7257e-04, -9.5030e-02,  3.3768e-04,
-1.8544e-03, -3.6523e-02, -7.2934e-02,  4.0945e-03,  0.0000e+00,
 4.9555e-05, -1.7378e-03,  0.0000e+00,  8.4583e-03,  8.0902e-03,
-5.3693e-03,  2.3692e-04,  3.2043e-04,  7.9284e-03,  4.1744e-02,
 6.3465e-04, -9.2906e-03,  0.0000e+00, -5.0780e-03,  6.2474e-02,
 0.0000e+00,  1.4215e-03, -7.7114e-05,  5.2675e-04, -4.7182e-02,
-2.4799e-04,  9.1349e-02, -2.7841e-02,  0.0000e+00,  0.0000e+00,
-1.1429e-03,  0.0000e+00,  7.2656e-03,  1.1332e-02,  2.3827e-05,
 1.2736e-03, -2.7722e-02,  7.5002e-03, -3.1872e-02, -2.8911e-05,
 5.1429e-03, -6.5550e-02, -1.1608e-02, -3.3981e-04,  7.2716e-02,
 1.1304e-03,  0.0000e+00, -1.3498e-04,  0.0000e+00,  3.7031e-03,
 1.5550e-02, -5.2709e-02,  6.0137e-03,  0.0000e+00, -4.9374e-04,
 5.7017e-03,  3.7391e-03,  7.7051e-05, -1.3585e-02,  1.4197e-02,
```

```
        -1.3778e-03,  1.9260e-03,  7.3948e-05, -3.8151e-03, -1.2004e-02,
        -3.3429e-03, -2.4811e-03, -1.2313e-03, -4.9987e-03,  0.0000e+00,
        -5.0344e-03, -1.1624e-02,  1.1053e-04,  0.0000e+00,  5.9217e-05,
        -5.3263e-02, -2.0007e-02,  4.7769e-03, -2.6769e-02,  0.0000e+00,
         5.4983e-02,  1.0025e-02,  1.7233e-05, -1.8682e-02,  2.5798e-04,
         7.4100e-04, -9.9626e-03,  1.1921e-02,  3.6981e-03,  1.1578e-05,
         1.1009e-03, -1.6596e-03,  0.0000e+00, -9.7963e-03,  3.5173e-03,
        -2.0022e-05,  2.1767e-04, -2.7261e-03,  1.5587e-01, -2.6056e-05,
        -1.5325e-03,  4.0836e-03, -1.6416e-03, -5.2665e-02,  0.0000e+00,
        -7.0171e-04,  2.2164e-02,  6.7884e-03,  0.0000e+00,  6.8571e-03,
        -4.2379e-04, -2.2591e-05,  1.1320e-03, -2.0329e-02, -3.1282e-02,
        -1.7914e-03,  1.6486e-02,  4.8752e-03, -1.5933e-03, -8.3915e-02,
        -1.5584e-02,  2.2126e-03,  0.0000e+00, -7.9807e-04, -2.0745e-04,
         2.0107e-03, -2.3169e-03,  2.6880e-02,  1.6485e-02, -3.6252e-04,
        -1.6164e-04, -1.8336e-02,  1.4660e-02, -1.2373e-03, -1.1438e-03,
        -1.1068e-03,  1.5384e-04, -1.4640e-02, -4.2800e-03,  0.0000e+00,
         7.7305e-02, -2.1117e-02,  3.8170e-04,  5.5854e-04, -3.4539e-02,
        -2.3367e-05, -2.7088e-04,  0.0000e+00,  2.1025e-02,  7.0168e-03,
         1.5233e-03,  3.1799e-02, -2.7526e-03, -1.3818e-04,  9.1171e-03,
        -3.9343e-03, -1.8216e-02,  1.1102e-04,  0.0000e+00])
tensor([[[[-2.4337e-03, -8.9142e-03,  5.2915e-04],
          [-4.5725e-03, -7.1946e-03, -2.8895e-03],
          [ 6.2991e-03, -1.3715e-03,  6.4167e-04]],

         [[-2.0965e-04,  1.6192e-04,  0.0000e+00],
          [-2.0727e-03, -2.1130e-03, -4.1039e-04],
          [-3.1831e-05, -5.5566e-05, -3.2599e-05]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[-1.0682e-04, -4.8258e-05, -1.0485e-03],
          [-6.6306e-04,  5.1047e-05,  8.3990e-05],
          [ 1.8526e-04,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,
```

```
          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


         [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([-6.3481e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -1.0702e-05,  2.0134e-03,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -7.2273e-02, -8.1953e-01,
        -1.3451e-02,  1.6602e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -1.2292e-02, -2.4479e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -4.2887e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -3.6598e-02,  3.2357e-02,  1.1818e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  3.5770e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -9.1780e-04, -4.9840e-02,
```

```
          0.0000e+00,  0.0000e+00,  3.1887e-03,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  8.3653e-01,  0.0000e+00,
         -3.8742e-01,  0.0000e+00,  0.0000e+00,  0.0000e+00,  2.7007e-01,
          0.0000e+00,  4.6942e-04,  0.0000e+00, -1.3946e-04,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  7.2986e-02, -9.9644e-04,  0.0000e+00,
          0.0000e+00,  1.0931e-02,  5.2078e-02,  0.0000e+00, -7.1202e-04,
         -9.4627e-04,  9.2873e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  1.0637e-01,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         -4.8057e-03,  0.0000e+00,  0.0000e+00, -3.9866e-02,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  4.1266e-04,
          0.0000e+00,  0.0000e+00,  6.0754e-02,  0.0000e+00,  0.0000e+00,
         -1.9490e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  4.0276e-03,
          0.0000e+00,  0.0000e+00,  0.0000e+00, -3.0473e-04, -5.9068e-02,
          0.0000e+00,  0.0000e+00,  1.1591e-02,  1.2568e-03,  0.0000e+00,
          0.0000e+00,  7.2871e-03, -6.3459e-05,  0.0000e+00,  0.0000e+00,
          5.4246e-04,  0.0000e+00,  5.1441e-03,  0.0000e+00,  0.0000e+00,
         -9.6361e-05,  0.0000e+00, -1.9933e-02,  1.5514e-01, -2.1199e-02,
          1.0904e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.0977e-02,
          7.6936e-05,  0.0000e+00,  1.2775e-03,  0.0000e+00,  0.0000e+00,
          1.3441e-02,  0.0000e+00, -3.6458e-04,  6.6490e-03,  0.0000e+00,
          0.0000e+00, -2.8325e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  9.5260e-02,  0.0000e+00,  0.0000e+00,
          1.0409e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  2.9744e-01,  0.0000e+00, -7.1178e-02,  0.0000e+00,
         -1.8593e-02,  8.3816e-05,  0.0000e+00,  1.5862e-02,  1.1396e-02,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  2.4850e-03,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  5.1273e-02,  0.0000e+00,
          0.0000e+00,  0.0000e+00, -4.5261e-02,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  7.4818e-02,  6.3576e-02,  0.0000e+00,  8.5091e-03,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          7.8157e-01, -1.9571e-03,  0.0000e+00, -1.3711e-02,  0.0000e+00,
          0.0000e+00,  0.0000e+00, -3.3593e-03,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00, -6.8216e-01,  2.5736e-03,
         -1.3889e-02,  1.3667e-02,  5.2170e-04, -1.1742e-02,  0.0000e+00,
          1.7990e-01,  0.0000e+00, -2.0826e-01,  0.0000e+00,  0.0000e+00,
          3.4487e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.7193e-03,
          5.3907e-04,  0.0000e+00,  0.0000e+00,  4.1502e-01,  0.0000e+00,
          0.0000e+00,  8.7031e-03, -1.3975e-02,  0.0000e+00,  0.0000e+00,
          0.0000e+00])
tensor([[-0.0003, -0.0002,  0.0000,  …,  0.0000,  0.0000,  0.0000],
        [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
        [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
        …,
        [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
```

```
        [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
        [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]])
tensor([-0.0044,  0.0000,  0.0016,  …, -0.0001,  0.0005,  0.0005])
tensor([[0.0000, 0.0000, 0.0004,  …, 0.0000, 0.0000, 0.0004],
        [0.0000, 0.0000, 0.0000,  …, 0.0000, 0.0000, 0.0000],
        [0.0000, 0.0000, 0.0000,  …, 0.0000, 0.0000, 0.0000],
        …,
        [0.0000, 0.0000, 0.0000,  …, 0.0000, 0.0000, 0.0000],
        [0.0000, 0.0000, 0.0000,  …, 0.0000, 0.0000, 0.0000],
        [0.0000, 0.0000, 0.0000,  …, 0.0000, 0.0000, 0.0000]])
tensor([ 1.5359e-03,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
        -3.8190e-04, -2.5495e-05])
tensor([[1.1468e-04, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 8.7469e-06,
         1.3502e-05],
        [1.8215e-05, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 1.8593e-05,
         3.6040e-06],
        [1.9837e-05, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 5.1129e-04,
         1.1331e-05],
        …,
        [8.2761e-08, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 2.9171e-10,
         1.8469e-09],
        [8.3117e-08, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 2.9292e-10,
         1.8522e-09],
        [7.9660e-08, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 3.0698e-10,
         1.7416e-09]])
tensor([ 1.6567e-02, -8.0167e-03,  7.6510e-02,  2.3928e-02, -9.2317e-02,
        -2.0567e-02,  6.5565e-02, -1.1329e-02,  5.0863e-03,  1.0379e-01,
        -2.7121e-02,  5.5883e-03, -1.8585e-02,  4.7190e-02,  2.7031e-02,
         3.2907e-02, -4.1251e-02, -1.1459e-01, -3.2949e-02, -4.0302e-02,
         7.8598e-06,  3.2790e-06,  3.0802e-06,  2.8421e-06,  3.3204e-06,
         2.8835e-06,  2.8148e-06,  3.2084e-06,  2.8040e-06,  2.7916e-06,
         2.8006e-06,  3.0380e-06,  2.8014e-06,  2.8319e-06,  3.0084e-06,
         3.1073e-06,  2.8511e-06,  2.8007e-06,  2.9802e-06,  2.7892e-06,
         2.8516e-06,  2.9583e-06,  2.8805e-06,  2.8984e-06,  2.8168e-06,
         2.7962e-06,  2.8377e-06,  2.8776e-06,  2.8109e-06,  3.1561e-06,
         2.7777e-06,  2.8603e-06,  2.8108e-06,  3.1470e-06,  2.8030e-06,
         2.7988e-06,  2.8392e-06,  2.7965e-06,  3.0046e-06,  2.8254e-06,
         2.7847e-06,  2.8013e-06,  2.8462e-06,  2.8754e-06,  2.8185e-06,
         2.8004e-06,  2.8439e-06,  2.8067e-06,  3.2893e-06,  2.8360e-06,
         2.8362e-06,  2.8045e-06,  3.1041e-06,  2.8713e-06,  2.8121e-06,
         3.3066e-06,  2.8066e-06,  2.8894e-06,  2.8772e-06,  2.7942e-06,
         2.8911e-06,  2.8786e-06,  2.8730e-06,  2.7982e-06,  2.8502e-06,
         2.8028e-06,  2.8145e-06,  2.7968e-06,  2.8202e-06,  3.1349e-06,
         2.9312e-06,  2.8700e-06,  2.8035e-06,  3.0897e-06,  2.9668e-06,
         2.8003e-06,  2.7951e-06,  2.8850e-06,  2.7664e-06,  2.7989e-06,
         2.8099e-06,  3.0536e-06,  2.7911e-06,  2.8654e-06,  2.8494e-06,
         2.8055e-06,  2.8375e-06,  2.8736e-06,  3.0225e-06,  2.8685e-06,
         2.8182e-06,  2.8102e-06,  2.8227e-06,  3.3222e-06,  2.8181e-06,
```

```
2.9854e-06,  2.8101e-06,  2.8983e-06,  2.7916e-06,  3.0709e-06,
3.2942e-06,  2.9285e-06,  3.3133e-06,  2.8353e-06,  2.8674e-06,
2.9868e-06,  2.8054e-06,  2.8676e-06,  2.8043e-06,  2.8520e-06,
3.0460e-06,  2.8102e-06,  3.0273e-06,  2.7943e-06,  2.8313e-06,
3.1150e-06,  2.8258e-06,  2.9449e-06,  3.2715e-06,  2.9744e-06,
2.8080e-06,  2.7880e-06,  2.8037e-06,  3.0380e-06,  3.1784e-06,
2.7916e-06,  3.1102e-06,  2.7902e-06,  2.9098e-06,  2.7784e-06,
2.8039e-06,  2.7943e-06,  2.7993e-06,  2.7582e-06,  3.1120e-06,
3.0337e-06,  3.0919e-06,  2.8225e-06,  2.8281e-06,  2.8576e-06,
2.7921e-06,  3.2402e-06,  2.9173e-06,  2.8288e-06,  2.8409e-06,
3.2880e-06,  2.8400e-06,  2.9866e-06,  2.8000e-06,  2.7939e-06,
2.8037e-06,  2.7955e-06,  2.8176e-06,  2.8356e-06,  2.8253e-06,
2.8453e-06,  2.8234e-06,  2.8906e-06,  2.7969e-06,  2.8121e-06,
2.7870e-06,  3.0937e-06,  2.9633e-06,  2.8189e-06,  3.2246e-06,
2.8015e-06,  2.9856e-06,  2.7615e-06,  2.8015e-06,  2.7811e-06,
3.0377e-06,  2.8365e-06,  2.7896e-06,  2.8191e-06,  2.9701e-06,
3.0986e-06,  3.0411e-06,  2.8025e-06,  2.8906e-06,  2.9952e-06,
3.0395e-06,  2.8187e-06,  3.1294e-06,  2.8088e-06,  2.8078e-06,
2.7962e-06,  2.8231e-06,  2.7906e-06,  2.9762e-06,  2.7853e-06,
3.0574e-06,  2.7813e-06,  3.1972e-06,  2.8698e-06,  2.7908e-06,
2.9178e-06,  2.8235e-06,  2.8034e-06,  2.8213e-06,  2.8178e-06,
2.8033e-06,  2.9563e-06,  2.9586e-06,  3.0291e-06,  3.2962e-06,
2.8144e-06,  3.0541e-06,  2.8574e-06,  3.2243e-06,  2.9918e-06,
2.8171e-06,  2.7746e-06,  2.8431e-06,  2.8029e-06,  2.8267e-06,
2.7908e-06,  2.8583e-06,  2.7718e-06,  2.9339e-06,  2.8234e-06,
2.9991e-06,  3.0033e-06,  2.8326e-06,  2.8174e-06,  2.8351e-06,
2.8112e-06,  2.9303e-06,  2.8659e-06,  2.8867e-06,  2.8044e-06,
2.7899e-06,  2.7961e-06,  3.2496e-06,  3.2353e-06,  3.0894e-06,
2.8003e-06,  2.9250e-06,  2.8589e-06,  2.8500e-06,  2.8933e-06,
3.1158e-06,  2.8562e-06,  3.1122e-06,  2.8162e-06,  2.8223e-06,
2.8118e-06,  2.8110e-06,  2.8181e-06,  2.9330e-06,  2.7918e-06,
2.8935e-06,  3.0648e-06,  2.8111e-06,  2.9370e-06,  3.1443e-06,
2.8005e-06,  2.8401e-06,  3.2594e-06,  2.9710e-06,  2.8077e-06,
2.8896e-06,  2.8101e-06,  2.8813e-06,  2.8187e-06,  2.8046e-06,
3.3586e-06,  2.9835e-06,  3.2159e-06,  2.9882e-06,  2.8059e-06,
3.3589e-06,  2.8094e-06,  2.9460e-06,  2.9123e-06,  2.7933e-06,
2.8997e-06,  2.8130e-06,  2.9359e-06,  2.8468e-06,  2.8087e-06,
2.8317e-06,  2.7959e-06,  2.8323e-06,  2.7968e-06,  2.8280e-06,
2.7648e-06,  2.8136e-06,  3.0827e-06,  2.8008e-06,  3.0743e-06,
3.0087e-06,  2.8053e-06,  2.8518e-06,  3.0051e-06,  2.8077e-06,
2.8402e-06,  2.8222e-06,  3.0685e-06,  2.9222e-06,  2.8529e-06,
2.8461e-06,  2.8141e-06,  3.2629e-06,  3.0086e-06,  2.8089e-06,
2.8108e-06,  2.7843e-06,  3.1550e-06,  2.8162e-06,  2.7784e-06,
2.8428e-06,  2.8011e-06,  2.8697e-06,  2.8040e-06,  3.1830e-06,
2.8028e-06,  2.8270e-06,  2.9218e-06,  2.8328e-06,  2.7796e-06,
2.9401e-06,  3.2157e-06,  2.7997e-06,  2.7941e-06,  3.3616e-06,
3.0924e-06,  2.8147e-06,  2.9623e-06,  2.8036e-06,  2.9322e-06,
2.8260e-06,  2.8249e-06,  2.8236e-06,  2.8040e-06,  2.7938e-06,
```

```
2.8000e-06,  2.8008e-06,  2.7872e-06,  3.0890e-06,  2.9086e-06,
2.8282e-06,  2.8747e-06,  3.3357e-06,  3.0956e-06,  2.8247e-06,
2.8246e-06,  3.2716e-06,  2.8332e-06,  2.9097e-06,  2.9902e-06,
2.8101e-06,  2.8521e-06,  2.8117e-06,  2.8382e-06,  2.8150e-06,
2.8561e-06,  2.8041e-06,  3.3612e-06,  2.8360e-06,  3.1077e-06,
2.8003e-06,  3.0449e-06,  2.8239e-06,  2.8269e-06,  2.9846e-06,
2.8185e-06,  2.8818e-06,  2.7957e-06,  2.7960e-06,  3.1732e-06,
2.9428e-06,  2.8057e-06,  2.8097e-06,  2.8061e-06,  2.8322e-06,
2.9322e-06,  2.8630e-06,  2.9785e-06,  2.8862e-06,  2.8956e-06,
3.0490e-06,  2.8116e-06,  2.8135e-06,  3.1115e-06,  3.1085e-06,
2.7989e-06,  2.8693e-06,  2.8013e-06,  2.8151e-06,  3.0332e-06,
2.7904e-06,  2.7897e-06,  2.8005e-06,  2.7935e-06,  2.8794e-06,
2.8097e-06,  2.9991e-06,  2.8624e-06,  2.8005e-06,  2.8037e-06,
2.8124e-06,  2.8363e-06,  2.7993e-06,  2.9464e-06,  2.8347e-06,
2.9997e-06,  3.0328e-06,  3.2941e-06,  2.8932e-06,  2.8278e-06,
3.0324e-06,  2.9644e-06,  3.1103e-06,  3.0357e-06,  2.9922e-06,
2.9109e-06,  2.8054e-06,  2.8217e-06,  2.8365e-06,  2.9529e-06,
2.8035e-06,  2.7762e-06,  2.8891e-06,  2.7924e-06,  2.8966e-06,
2.8277e-06,  2.8700e-06,  3.2290e-06,  2.8047e-06,  2.8092e-06,
2.8337e-06,  2.8291e-06,  3.2201e-06,  2.8583e-06,  2.9360e-06,
2.8140e-06,  2.8832e-06,  2.8681e-06,  2.8703e-06,  2.7965e-06,
3.1378e-06,  2.9402e-06,  2.8642e-06,  3.2461e-06,  2.8109e-06,
2.8054e-06,  3.2287e-06,  2.8024e-06,  2.9447e-06,  2.8116e-06,
2.8027e-06,  2.8218e-06,  3.0860e-06,  2.8139e-06,  2.7830e-06,
2.9070e-06,  2.8037e-06,  2.8260e-06,  2.9382e-06,  2.9301e-06,
2.8650e-06,  2.8228e-06,  2.8255e-06,  2.8388e-06,  3.2192e-06,
3.0922e-06,  2.7989e-06,  2.9055e-06,  3.1193e-06,  2.7895e-06,
3.3157e-06,  3.0644e-06,  3.1591e-06,  2.9205e-06,  2.8500e-06,
3.1385e-06,  2.7868e-06,  2.7971e-06,  2.8072e-06,  2.8713e-06,
2.8268e-06,  2.8045e-06,  2.9208e-06,  2.8961e-06,  2.9021e-06,
3.2527e-06,  2.8036e-06,  2.8239e-06,  2.8733e-06,  2.8670e-06,
2.8586e-06,  2.9023e-06,  3.2625e-06,  2.7961e-06,  3.0739e-06,
3.0662e-06,  2.8622e-06,  2.8573e-06,  2.8779e-06,  2.8009e-06,
2.7946e-06,  2.8180e-06,  3.1890e-06,  3.1124e-06,  2.7990e-06,
2.7853e-06,  2.9127e-06,  3.3397e-06,  3.0083e-06,  2.8115e-06,
2.8376e-06,  2.9591e-06,  3.2852e-06,  2.7975e-06,  2.8157e-06,
3.0057e-06,  3.0977e-06,  2.9341e-06,  2.8462e-06,  3.1575e-06,
2.7959e-06,  3.2725e-06,  2.8215e-06,  2.9381e-06,  2.8294e-06,
2.8098e-06,  2.9842e-06,  2.8325e-06,  3.0018e-06,  2.8264e-06,
2.8473e-06,  2.8452e-06,  3.0424e-06,  2.7963e-06,  3.3172e-06,
2.7971e-06,  3.0875e-06,  2.8550e-06,  2.7915e-06,  2.7966e-06,
2.8229e-06,  2.7944e-06,  3.0373e-06,  3.0971e-06,  2.8702e-06,
3.2849e-06,  2.8025e-06,  3.0187e-06,  2.8357e-06,  2.9220e-06,
2.9257e-06,  2.9190e-06,  3.0535e-06,  2.9823e-06,  2.8048e-06,
2.8070e-06,  2.8482e-06,  2.8975e-06,  3.2731e-06,  2.8781e-06,
3.1170e-06,  2.8020e-06,  2.8148e-06,  2.8354e-06,  2.9146e-06,
3.1787e-06,  3.2938e-06,  2.9107e-06,  2.8233e-06,  3.0625e-06,
2.8510e-06,  3.1559e-06,  2.7899e-06,  2.7805e-06,  2.8833e-06,
```

2.9450e-06, 2.8049e-06, 2.8707e-06, 3.0913e-06, 2.9031e-06,
2.7956e-06, 2.8995e-06, 2.9238e-06, 2.7507e-06, 3.1390e-06,
2.8327e-06, 3.0079e-06, 2.8051e-06, 2.8632e-06, 2.8086e-06,
2.8178e-06, 3.2016e-06, 3.0840e-06, 2.8222e-06, 2.8137e-06,
2.8745e-06, 2.9754e-06, 2.7993e-06, 2.8144e-06, 2.8416e-06,
2.8210e-06, 2.8823e-06, 2.8279e-06, 2.8357e-06, 2.8230e-06,
3.1414e-06, 2.8595e-06, 2.8670e-06, 2.9056e-06, 3.0442e-06,
2.9378e-06, 2.7919e-06, 2.8046e-06, 3.2517e-06, 2.7965e-06,
2.8334e-06, 2.8036e-06, 3.1169e-06, 2.8579e-06, 2.9557e-06,
2.8125e-06, 2.9520e-06, 3.0660e-06, 2.9714e-06, 2.7980e-06,
2.8888e-06, 2.9395e-06, 2.8032e-06, 3.3005e-06, 2.8045e-06,
2.6515e-06, 2.9011e-06, 2.9604e-06, 2.7900e-06, 2.9254e-06,
2.8300e-06, 2.8476e-06, 2.8713e-06, 2.9790e-06, 2.7783e-06,
2.8015e-06, 2.8828e-06, 3.0562e-06, 2.9427e-06, 2.9055e-06,
2.7864e-06, 2.8370e-06, 2.7906e-06, 2.7503e-06, 2.8286e-06,
2.8599e-06, 2.8371e-06, 2.8310e-06, 3.0550e-06, 3.0449e-06,
2.8096e-06, 3.0657e-06, 2.8089e-06, 2.7900e-06, 3.0178e-06,
2.9177e-06, 2.8271e-06, 2.8174e-06, 2.8426e-06, 2.8811e-06,
2.9936e-06, 2.8468e-06, 2.8211e-06, 2.9416e-06, 2.8206e-06,
2.7860e-06, 3.0980e-06, 2.7709e-06, 3.1265e-06, 2.9256e-06,
3.1159e-06, 2.8064e-06, 2.9109e-06, 2.8197e-06, 2.8339e-06,
3.1899e-06, 2.8003e-06, 3.1783e-06, 3.1197e-06, 3.1083e-06,
2.8217e-06, 2.8166e-06, 2.7853e-06, 2.9842e-06, 2.8239e-06,
2.8298e-06, 2.8003e-06, 2.8003e-06, 2.8101e-06, 3.3014e-06,
2.9423e-06, 2.9560e-06, 2.8175e-06, 2.8492e-06, 2.8135e-06,
2.7936e-06, 3.0430e-06, 3.0355e-06, 3.0737e-06, 2.8059e-06,
3.0268e-06, 2.8479e-06, 2.8913e-06, 2.8742e-06, 2.8666e-06,
3.0062e-06, 2.7869e-06, 2.9398e-06, 3.3640e-06, 3.1396e-06,
2.8194e-06, 2.8232e-06, 2.7918e-06, 2.8177e-06, 2.8233e-06,
2.8395e-06, 2.9324e-06, 2.8571e-06, 2.8019e-06, 2.8150e-06,
2.8078e-06, 2.7922e-06, 3.1067e-06, 3.0143e-06, 3.0946e-06,
2.8099e-06, 2.8486e-06, 2.8093e-06, 3.2494e-06, 2.7971e-06,
3.2368e-06, 2.8917e-06, 2.7981e-06, 2.8102e-06, 2.8363e-06,
2.7985e-06, 2.8380e-06, 2.7896e-06, 2.8679e-06, 2.7912e-06,
2.8169e-06, 3.2072e-06, 2.8160e-06, 2.7955e-06, 2.8034e-06,
2.8673e-06, 2.7800e-06, 2.7885e-06, 3.2036e-06, 3.3369e-06,
2.8078e-06, 2.8435e-06, 2.9934e-06, 2.7841e-06, 2.8134e-06,
2.8182e-06, 3.0270e-06, 2.8772e-06, 2.7886e-06, 2.8717e-06,
2.7215e-06, 2.8465e-06, 2.8977e-06, 2.8139e-06, 3.2113e-06,
3.2391e-06, 2.7824e-06, 2.9323e-06, 3.2922e-06, 2.8158e-06,
3.2100e-06, 3.1291e-06, 2.8081e-06, 2.8769e-06, 2.9822e-06,
3.2348e-06, 2.8136e-06, 2.8173e-06, 2.9745e-06, 2.8314e-06,
2.9400e-06, 2.8559e-06, 3.1102e-06, 2.8761e-06, 2.9723e-06,
2.8347e-06, 2.9046e-06, 2.8720e-06, 2.8572e-06, 2.8071e-06,
3.0729e-06, 2.9154e-06, 3.3257e-06, 2.8576e-06, 2.9649e-06,
2.7967e-06, 2.9833e-06, 3.0845e-06, 3.1221e-06, 2.8478e-06,
2.8246e-06, 2.8096e-06, 2.9829e-06, 2.8569e-06, 2.9433e-06,
2.9773e-06, 3.2537e-06, 2.8189e-06, 2.8013e-06, 2.8099e-06,

```
        2.8836e-06,  2.7906e-06,  2.9736e-06,  2.8398e-06,  2.7921e-06,
        3.0504e-06,  3.2535e-06,  2.9184e-06,  2.8382e-06,  2.7893e-06,
        2.8155e-06,  2.7947e-06,  2.8306e-06,  2.7980e-06,  2.9007e-06,
        2.8053e-06,  2.8797e-06,  2.9913e-06,  3.3719e-06,  3.3459e-06,
        2.9820e-06,  2.8046e-06,  2.9145e-06,  2.8486e-06,  2.8133e-06,
        2.7930e-06,  2.8119e-06,  3.1381e-06,  2.7965e-06,  2.8922e-06,
        2.8397e-06,  3.2992e-06,  2.8149e-06,  2.8174e-06,  3.2840e-06,
        3.1448e-06,  2.7829e-06,  2.8373e-06,  3.0923e-06,  2.7926e-06,
        2.8388e-06,  2.8570e-06,  2.8384e-06,  2.8864e-06,  2.8070e-06,
        2.8302e-06,  2.9088e-06,  2.8377e-06,  2.8094e-06,  3.1819e-06,
        2.9362e-06,  2.8069e-06,  2.7849e-06,  2.8507e-06,  2.8259e-06,
        2.8376e-06,  2.8661e-06,  2.9503e-06,  2.9919e-06,  3.1215e-06,
        2.9175e-06,  2.8694e-06,  2.7851e-06,  2.8046e-06,  2.8050e-06,
        2.8073e-06,  2.7978e-06,  3.1019e-06,  2.8012e-06,  2.8848e-06,
        2.8068e-06,  2.8708e-06,  2.8112e-06,  2.7892e-06,  2.7867e-06,
        2.8057e-06,  2.8052e-06,  2.7942e-06,  3.1162e-06,  3.0940e-06,
        2.8410e-06,  2.8127e-06,  2.7963e-06,  2.8114e-06,  2.9279e-06,
        2.8123e-06,  2.9242e-06,  2.8409e-06,  3.0059e-06,  2.8649e-06,
        3.2050e-06,  2.7989e-06,  3.2528e-06,  2.8251e-06,  2.7875e-06,
        2.8142e-06,  2.8009e-06,  2.8030e-06,  2.7839e-06,  2.7945e-06,
        2.7885e-06,  2.8767e-06,  3.3404e-06,  2.9176e-06,  2.8499e-06,
        2.7940e-06,  2.8950e-06,  2.8315e-06,  2.8360e-06,  2.9036e-06,
        2.8581e-06,  3.0073e-06,  3.0248e-06,  3.3533e-06,  3.2065e-06,
        2.8804e-06,  2.7966e-06,  2.8018e-06,  2.8056e-06,  2.8782e-06,
        2.8104e-06,  2.8979e-06,  2.8291e-06,  2.7956e-06,  2.8222e-06,
        2.8348e-06,  2.7971e-06,  2.9046e-06,  2.8078e-06,  2.8203e-06,
        2.7944e-06,  2.7612e-06,  3.0070e-06,  3.0525e-06,  2.8726e-06,
        3.0938e-06,  2.8721e-06,  2.8611e-06,  2.8289e-06,  2.8630e-06,
        2.7837e-06,  2.7888e-06,  2.7925e-06,  2.8325e-06,  2.8977e-06,
        2.9584e-06,  2.8471e-06,  2.9745e-06,  3.1126e-06,  2.8612e-06,
        3.0801e-06,  3.1432e-06,  2.8166e-06,  2.9660e-06,  2.8250e-06,
        2.7978e-06,  2.9566e-06,  2.8612e-06,  2.8304e-06,  3.2299e-06,
        3.1447e-06,  2.7859e-06,  2.8071e-06,  2.8282e-06,  2.9037e-06])
end of p.grad

False
Epoch 2 finished
Epoch [2/10], Loss: 2.0083
************************************************************

Learning rate: 0.001

another way to print learning rate:
tensor([[[[-1.7939e-03, -1.9108e-03, -2.0208e-03,  …, -1.4831e-03,
           -2.4398e-03, -2.0092e-03],
          [-1.5743e-03, -7.2293e-04, -2.5273e-04,  …, -6.3787e-04,
           -1.7466e-03, -1.2078e-03],
          [-1.7477e-03, -9.4947e-04, -5.0870e-04,  …, -2.0927e-03,
```

```
  -2.3541e-03, -1.1393e-03],
 …,
 [-4.8791e-03, -6.5154e-03, -7.7866e-03,  …, -8.5469e-03,
  -8.2720e-03, -7.9532e-03],
 [-4.4313e-03, -6.2039e-03, -7.5090e-03,  …, -7.7664e-03,
  -7.3724e-03, -6.9527e-03],
 [-5.6238e-03, -6.8574e-03, -7.1235e-03,  …, -7.5506e-03,
  -7.0994e-03, -6.2871e-03]],

[[ 4.9017e-03,  4.7470e-03,  5.0086e-03,  …,  6.3914e-03,
   5.3827e-03,  5.5061e-03],
 [ 5.5592e-03,  6.3211e-03,  7.0041e-03,  …,  7.5756e-03,
   6.3396e-03,  6.5400e-03],
 [ 5.7571e-03,  6.6871e-03,  7.2947e-03,  …,  6.0155e-03,
   5.7383e-03,  6.9191e-03],
 …,
 [ 1.6948e-03, -1.2678e-04, -1.2233e-03,  …, -2.5234e-03,
  -2.5830e-03, -2.4715e-03],
 [ 2.1493e-03,  7.1804e-05, -1.0023e-03,  …, -1.7614e-03,
  -1.6766e-03, -1.4582e-03],
 [ 1.2909e-03, -2.4710e-04, -2.5985e-04,  …, -1.2959e-03,
  -1.2895e-03, -8.1915e-04]],

[[ 6.5818e-03,  6.1459e-03,  6.6571e-03,  …,  7.9186e-03,
   7.1899e-03,  7.3158e-03],
 [ 7.0076e-03,  7.5724e-03,  8.2652e-03,  …,  9.1173e-03,
   8.3574e-03,  8.7378e-03],
 [ 6.9180e-03,  7.6981e-03,  8.1310e-03,  …,  7.7612e-03,
   7.7840e-03,  9.0411e-03],
 …,
 [ 3.0425e-03,  8.0117e-04, -1.3361e-05,  …, -3.0891e-03,
  -2.8641e-03, -2.2439e-03],
 [ 3.5138e-03,  9.7915e-04, -4.7732e-05,  …, -1.4875e-03,
  -1.3593e-03, -9.5699e-04],
 [ 3.4388e-03,  9.8117e-04,  6.9265e-04,  …, -3.4822e-04,
  -2.2789e-04,  4.3535e-04]]],


[[[ 7.4118e-02,  7.3882e-02,  7.3790e-02,  …,  6.6892e-02,
    6.5120e-02,  6.2468e-02],
 [ 7.6011e-02,  7.4824e-02,  7.4197e-02,  …,  6.6922e-02,
   6.5690e-02,  6.3372e-02],
 [ 7.5070e-02,  7.2932e-02,  7.2483e-02,  …,  6.4673e-02,
   6.3427e-02,  6.2845e-02],
 …,
 [ 7.1743e-02,  6.9325e-02,  7.0018e-02,  …,  6.2214e-02,
   6.2740e-02,  6.2928e-02],
 [ 7.3332e-02,  7.0809e-02,  6.9501e-02,  …,  6.3698e-02,
```

```
    6.3621e-02,  6.3419e-02],
  [ 7.0484e-02,  6.7565e-02,  6.6246e-02,  …,  6.3868e-02,
    6.4129e-02,  6.3588e-02]],

 [[ 3.2077e-02,  3.1182e-02,  2.9189e-02,  …,  1.9468e-02,
    1.7399e-02,  1.4309e-02],
  [ 3.4247e-02,  3.2715e-02,  2.9767e-02,  …,  1.9347e-02,
    1.8050e-02,  1.5065e-02],
  [ 3.2853e-02,  3.0534e-02,  2.7789e-02,  …,  1.6831e-02,
    1.5523e-02,  1.4276e-02],
  …,
  [ 2.8331e-02,  2.5514e-02,  2.4687e-02,  …,  1.4983e-02,
    1.5023e-02,  1.4801e-02],
  [ 2.9947e-02,  2.6871e-02,  2.4093e-02,  …,  1.6577e-02,
    1.6014e-02,  1.5184e-02],
  [ 2.6362e-02,  2.3357e-02,  2.1180e-02,  …,  1.6184e-02,
    1.6366e-02,  1.5328e-02]],

 [[ 3.5658e-02,  3.5066e-02,  3.4440e-02,  …,  2.9769e-02,
    2.8238e-02,  2.4812e-02],
  [ 3.8009e-02,  3.7323e-02,  3.6151e-02,  …,  3.0856e-02,
    2.9562e-02,  2.5973e-02],
  [ 3.7397e-02,  3.6626e-02,  3.5084e-02,  …,  2.8597e-02,
    2.6787e-02,  2.4399e-02],
  …,
  [ 3.0876e-02,  2.8620e-02,  2.8895e-02,  …,  2.2544e-02,
    2.2225e-02,  2.1713e-02],
  [ 3.1303e-02,  2.8900e-02,  2.7554e-02,  …,  2.3412e-02,
    2.2573e-02,  2.1431e-02],
  [ 2.6817e-02,  2.5158e-02,  2.4580e-02,  …,  2.1681e-02,
    2.2111e-02,  2.0833e-02]]],


[[[ 1.7898e-03,  1.8665e-03,  5.0902e-04,  …,  2.0934e-03,
    2.7893e-03,  3.7027e-03],
  [ 5.0465e-03,  3.9750e-03,  2.7405e-03,  …,  1.3181e-03,
    3.0521e-03,  4.0334e-03],
  [ 5.7184e-03,  4.7166e-03,  4.8683e-03,  …,  1.1920e-03,
    1.3767e-03,  3.0286e-03],
  …,
  [ 6.4503e-03,  6.5355e-03,  5.8515e-03,  …,  5.0286e-03,
    3.5404e-03,  2.9971e-03],
  [ 5.0925e-03,  4.9920e-03,  3.8871e-03,  …,  2.1378e-03,
    3.1765e-03,  2.3933e-03],
  [ 3.5676e-03,  2.4984e-03,  1.1917e-03,  …,  1.5746e-03,
    1.4354e-03, -4.6751e-04]],

 [[-4.1025e-03, -4.1745e-03, -5.6895e-03,  …, -6.4175e-03,
```

```
   -6.1856e-03, -5.7966e-03],
  [-5.1941e-04, -1.5736e-03, -2.8628e-03,  …, -6.5745e-03,
   -5.1251e-03, -4.5679e-03],
  [ 4.5236e-04, -5.0868e-04, -4.3374e-04,  …, -5.9025e-03,
   -6.2259e-03, -5.0499e-03],
  …,
  [ 1.3548e-03,  1.7775e-03,  9.9618e-04,  …, -7.8000e-05,
   -2.0025e-03, -2.8758e-03],
  [-2.2656e-04, -2.8643e-04, -1.3179e-03,  …, -3.4453e-03,
   -2.6606e-03, -3.8120e-03],
  [-1.9823e-03, -3.1222e-03, -4.2748e-03,  …, -4.3326e-03,
   -4.6143e-03, -6.8442e-03]],

 [[-6.1265e-04, -1.2695e-03, -2.9276e-03,  …, -2.7754e-03,
   -2.4802e-03, -2.2709e-03],
  [ 3.0649e-03,  1.2955e-03, -2.9300e-04,  …, -2.9877e-03,
   -1.5250e-03, -1.0770e-03],
  [ 3.6727e-03,  2.2884e-03,  2.0182e-03,  …, -2.9204e-03,
   -3.0252e-03, -2.0618e-03],
  …,
  [ 2.8876e-03,  2.7926e-03,  1.6817e-03,  …,  2.1904e-03,
    3.8663e-04, -9.8373e-06],
  [ 3.8676e-04,  8.2637e-05, -9.4566e-04,  …, -1.0822e-03,
   -3.8681e-05, -1.1675e-03],
  [-1.5643e-03, -2.6180e-03, -3.7503e-03,  …, -1.7600e-03,
   -1.9056e-03, -4.0988e-03]]],


…,


[[[-1.0964e-02, -1.0058e-02, -1.0956e-02,  …, -6.8314e-03,
   -6.1759e-03, -7.6024e-03],
  [-9.8255e-03, -8.7224e-03, -8.6839e-03,  …, -5.9037e-03,
   -5.1038e-03, -5.6349e-03],
  [-8.4017e-03, -8.6422e-03, -7.4157e-03,  …, -5.3283e-03,
   -5.6584e-03, -5.4252e-03],
  …,
  [-5.5369e-03, -4.8369e-03, -5.8155e-03,  …, -5.9851e-03,
   -6.1550e-03, -6.4420e-03],
  [-5.7725e-03, -5.8494e-03, -5.8875e-03,  …, -8.0853e-03,
   -7.6398e-03, -7.0261e-03],
  [-7.0517e-03, -8.3037e-03, -8.0302e-03,  …, -8.0589e-03,
   -7.9531e-03, -8.6683e-03]],

 [[-7.8734e-03, -6.7772e-03, -7.5096e-03,  …, -4.2776e-03,
   -3.6783e-03, -4.7096e-03],
  [-6.3770e-03, -5.1159e-03, -5.0511e-03,  …, -3.4173e-03,
```

```
   -2.6579e-03, -2.6888e-03],
  [-4.3438e-03, -4.7853e-03, -3.5821e-03,  …, -2.0527e-03,
   -2.3197e-03, -1.8111e-03],
  …,
  [-2.4820e-03, -1.2979e-03, -2.0975e-03,  …, -1.3882e-03,
   -1.5554e-03, -2.1124e-03],
  [-2.4740e-03, -2.0317e-03, -1.9206e-03,  …, -3.6872e-03,
   -3.2534e-03, -2.9242e-03],
  [-3.2786e-03, -4.1354e-03, -3.6838e-03,  …, -3.6366e-03,
   -3.7816e-03, -4.7943e-03]],

 [[-1.7516e-02, -1.5923e-02, -1.6300e-02,  …, -1.2961e-02,
   -1.2653e-02, -1.3581e-02],
  [-1.5865e-02, -1.4209e-02, -1.4106e-02,  …, -1.2139e-02,
   -1.1346e-02, -1.1166e-02],
  [-1.4288e-02, -1.4754e-02, -1.3501e-02,  …, -1.1013e-02,
   -1.1316e-02, -1.0380e-02],
  …,
  [-1.3362e-02, -1.2407e-02, -1.2968e-02,  …, -1.0770e-02,
   -1.0485e-02, -1.0391e-02],
  [-1.3361e-02, -1.2903e-02, -1.2675e-02,  …, -1.3057e-02,
   -1.2271e-02, -1.1553e-02],
  [-1.4168e-02, -1.4946e-02, -1.4151e-02,  …, -1.2987e-02,
   -1.3273e-02, -1.3668e-02]]],


[[[-2.1939e-02, -2.0934e-02, -2.0073e-02,  …, -1.5065e-02,
   -1.4469e-02, -1.5905e-02],
  [-2.0811e-02, -1.9520e-02, -1.8373e-02,  …, -1.4275e-02,
   -1.3783e-02, -1.4325e-02],
  [-1.7861e-02, -1.7372e-02, -1.6726e-02,  …, -1.3556e-02,
   -1.3949e-02, -1.3944e-02],
  …,
  [-2.0740e-02, -2.0210e-02, -2.0787e-02,  …, -1.9119e-02,
   -1.8180e-02, -1.7497e-02],
  [-2.1472e-02, -2.2399e-02, -2.2943e-02,  …, -2.0114e-02,
   -1.9243e-02, -1.7762e-02],
  [-2.3708e-02, -2.4839e-02, -2.4120e-02,  …, -2.0122e-02,
   -1.9902e-02, -1.9377e-02]],

 [[-1.3008e-02, -1.2247e-02, -1.1675e-02,  …, -5.5858e-03,
   -4.7731e-03, -5.5900e-03],
  [-1.1827e-02, -1.0822e-02, -9.7411e-03,  …, -4.8099e-03,
   -4.2629e-03, -4.4856e-03],
  [-9.0327e-03, -8.6050e-03, -7.7483e-03,  …, -3.7989e-03,
   -4.4883e-03, -4.1467e-03],
  …,
  [-1.1909e-02, -1.0937e-02, -1.1336e-02,  …, -8.4065e-03,
```

```
   -7.3454e-03, -7.1573e-03],
  [-1.2250e-02, -1.2550e-02, -1.2970e-02,  …, -9.6791e-03,
   -8.9808e-03, -7.7943e-03],
  [-1.3881e-02, -1.4478e-02, -1.3473e-02,  …, -9.7227e-03,
   -9.7967e-03, -9.7345e-03]],

 [[-2.9952e-02, -2.8826e-02, -2.8092e-02,  …, -2.1582e-02,
   -2.0797e-02, -2.1379e-02],
  [-2.8319e-02, -2.6548e-02, -2.5420e-02,  …, -2.0373e-02,
   -1.9903e-02, -2.0072e-02],
  [-2.4578e-02, -2.4239e-02, -2.3320e-02,  …, -1.9045e-02,
   -1.9920e-02, -1.9252e-02],
  …,
  [-2.6365e-02, -2.5285e-02, -2.5571e-02,  …, -2.1811e-02,
   -2.1296e-02, -2.0967e-02],
  [-2.6732e-02, -2.6865e-02, -2.7123e-02,  …, -2.3693e-02,
   -2.3308e-02, -2.2001e-02],
  [-2.8104e-02, -2.8678e-02, -2.7345e-02,  …, -2.4561e-02,
   -2.4891e-02, -2.4415e-02]]],


[[[-9.4379e-03, -8.9847e-03, -9.0479e-03,  …, -2.6365e-03,
   -1.0305e-04, -1.0051e-03],
  [-1.1507e-02, -1.1352e-02, -8.8342e-03,  …, -4.0458e-03,
   -3.1050e-03, -2.8805e-03],
  [-1.4170e-02, -1.5366e-02, -1.2893e-02,  …, -3.4884e-03,
   -3.3104e-03, -2.3928e-03],
  …,
  [-3.2635e-03,  1.5661e-03,  8.0331e-04,  …,  6.5837e-03,
    5.9970e-03,  4.1033e-03],
  [-1.0608e-03,  5.5245e-04,  2.7833e-03,  …,  7.4995e-03,
    6.4277e-03,  4.5660e-03],
  [-9.9251e-05, -9.4703e-04,  9.5485e-04,  …,  6.0273e-03,
    4.4261e-03,  3.5218e-03]],

 [[-1.1926e-02, -1.1428e-02, -1.1124e-02,  …, -3.5927e-03,
   -1.5867e-03, -2.0948e-03],
  [-1.3117e-02, -1.3069e-02, -1.0193e-02,  …, -5.5187e-03,
   -4.6119e-03, -3.9907e-03],
  [-1.3965e-02, -1.5657e-02, -1.3183e-02,  …, -4.6548e-03,
   -4.5496e-03, -3.3008e-03],
  …,
  [-2.5673e-03,  1.9573e-03,  8.0559e-04,  …,  6.2531e-03,
    5.8000e-03,  4.1691e-03],
  [-1.6273e-04,  8.3399e-04,  3.0262e-03,  …,  6.8873e-03,
    5.6990e-03,  4.0945e-03],
  [ 8.7536e-04, -1.1980e-04,  1.4593e-03,  …,  5.0953e-03,
    3.6090e-03,  2.5689e-03]],
```

```
          [[-8.5943e-03, -8.7528e-03, -8.4513e-03,  …,  6.4752e-04,
             2.6441e-03,  2.4012e-03],
           [-9.1635e-03, -9.7431e-03, -6.8901e-03,  …, -1.6815e-03,
            -4.9790e-04,  6.2424e-04],
           [-9.4243e-03, -1.1727e-02, -9.2303e-03,  …,  5.2449e-05,
             4.3030e-04,  1.6921e-03],
          …,
           [ 2.1493e-03,  6.0163e-03,  4.2283e-03,  …,  9.2810e-03,
             9.2704e-03,  8.1234e-03],
           [ 4.7446e-03,  5.1188e-03,  6.5054e-03,  …,  1.0076e-02,
             9.1797e-03,  7.8920e-03],
           [ 6.0397e-03,  4.5941e-03,  5.7878e-03,  …,  8.4758e-03,
             6.8636e-03,  5.9553e-03]]]])
tensor([ 3.6904e-08,  3.9116e-08, -8.3819e-09, -4.5402e-09, -7.4506e-09,
        -4.8894e-08,  3.5812e-08,  4.2841e-08, -3.4837e-08, -1.5367e-08,
        -7.4506e-09,  1.3970e-09,  2.6543e-08,  9.3132e-09, -1.3388e-08,
        -1.3333e-09, -2.0955e-09,  0.0000e+00,  1.2806e-09, -1.0710e-08,
         5.8208e-09,  1.8626e-08,  3.0268e-08,  3.4226e-08, -5.8208e-11,
         7.7300e-08,  6.4938e-09,  3.4925e-08,  3.6322e-08,  3.1432e-08,
         1.0012e-08, -3.9581e-09, -1.9558e-08, -4.4238e-09,  1.6764e-08,
        -6.4756e-09,  1.7975e-07,  2.6589e-07,  2.9686e-09, -8.1491e-10,
        -7.9162e-09,  9.7789e-08, -2.3283e-09,  7.7998e-09,  2.7940e-09,
         3.7253e-09, -1.3970e-09, -6.0536e-09, -2.3283e-08, -6.0536e-09,
         4.1327e-09,  5.1223e-08, -2.7649e-08, -3.2596e-08,  5.5879e-09,
         2.2643e-08, -1.3271e-08, -1.3039e-08, -4.2783e-09,  5.9372e-09,
         1.2107e-08, -1.0710e-08,  2.6193e-09, -3.9581e-09, -1.6182e-08,
        -2.0955e-09, -4.4471e-08, -4.5635e-08,  3.2596e-09,  3.3528e-08,
        -3.7544e-09,  1.9092e-08, -7.6368e-08, -1.5832e-08, -3.3528e-08,
        -1.8626e-08,  1.2442e-09, -2.1653e-08, -1.3039e-08,  1.3504e-08,
         1.8626e-08,  2.7940e-08,  1.1642e-10,  3.1083e-08, -1.3039e-07,
         2.2817e-08,  5.5879e-08,  2.6077e-08,  6.6357e-09,  2.1420e-08,
         8.3819e-09,  2.2352e-08,  3.3760e-08, -1.4435e-08, -5.9605e-08,
         1.8161e-08])
tensor([-5.7217e-03, -2.7909e-02, -4.1102e-03, -3.4914e-03,  6.6228e-03,
        -5.1941e-03,  5.9156e-02, -3.2235e-02,  4.4233e-02,  2.2045e-02,
         1.8770e-02, -1.4696e-02, -2.1543e-02,  1.5326e-02,  5.4687e-02,
        -6.3750e-03,  2.8806e-02,  4.8004e-02, -8.4326e-03,  1.2127e-02,
        -6.1887e-03, -2.9345e-02, -7.1726e-03,  3.3587e-02,  2.3505e-02,
         1.6964e-02,  3.2004e-04,  1.9060e-02,  2.5235e-02, -8.7206e-04,
        -1.9991e-02, -7.8160e-03,  2.4299e-02, -4.8413e-03,  1.0346e-02,
        -7.3227e-03,  6.7952e-05,  3.1040e-04, -3.3755e-02, -8.4160e-03,
        -3.1061e-02, -7.6541e-04, -3.9850e-03, -9.6386e-03, -1.1399e-02,
        -8.0326e-03,  4.2418e-03,  2.0412e-02,  4.5867e-02,  2.6253e-04,
         2.3811e-03, -3.2785e-02,  2.7116e-02,  4.5285e-02,  6.1761e-03,
         2.2907e-02, -2.2460e-01, -4.9576e-04, -7.8119e-03,  2.5608e-02,
         1.7004e-03,  1.0872e-03, -3.6609e-03, -7.9645e-03,  5.7664e-03,
         4.3203e-02,  5.7447e-02, -4.1063e-02, -1.1590e-02,  4.5172e-02,
```

```
        -6.4982e-03,  9.0825e-03,  3.0464e-02, -1.9614e-02, -2.4912e-02,
         1.8821e-02,  2.8101e-03,  2.4437e-02, -8.0286e-03,  2.0037e-02,
        -9.2111e-02, -1.9984e-02, -1.1612e-02,  3.3416e-02, -6.2435e-02,
         6.1152e-03,  5.5062e-02, -1.0733e-02, -1.4500e-03, -2.5839e-02,
         2.3508e-02, -4.6654e-02, -4.1957e-02,  2.8771e-02,  2.0961e-02,
        -2.1457e-02])
tensor([-7.0615e-03, -2.7747e-02, -1.5574e-03, -4.9050e-03,  7.8682e-03,
         1.8642e-02,  2.4030e-02, -1.9142e-03,  2.2724e-02, -7.1191e-03,
        -2.2384e-03, -1.2068e-02, -8.6970e-03,  1.3097e-02,  2.4525e-02,
        -4.5571e-03,  1.6565e-03,  1.1117e-02,  6.5242e-03,  1.1421e-02,
        -7.5783e-03, -1.0962e-02, -2.4310e-02,  1.6254e-02,  2.0626e-03,
        -2.5374e-03, -1.5410e-03,  1.1998e-02,  2.4668e-03,  1.8678e-02,
        -1.5154e-02, -7.9269e-03,  9.9301e-03, -5.8020e-03, -7.2938e-03,
        -7.7214e-03, -2.6391e-03, -4.9864e-03, -3.2562e-02,  2.8927e-03,
        -4.4179e-02,  4.3840e-04,  1.1521e-03, -8.6663e-03, -4.7998e-03,
        -3.6217e-03,  3.8699e-03,  1.1161e-02,  2.0327e-02,  4.8728e-03,
        -6.7515e-03,  7.2257e-03,  1.5947e-02,  1.7581e-02,  8.0121e-03,
        -3.9350e-04, -2.0906e-02, -9.2829e-03, -9.0919e-03,  1.5779e-02,
        -1.2744e-02, -7.5476e-03, -3.2886e-03, -7.0028e-03, -1.1983e-02,
         1.9225e-02,  2.6981e-02, -2.8957e-02, -6.3897e-03,  1.9558e-02,
        -5.7069e-03,  3.8608e-03,  1.8149e-02, -9.4961e-03, -1.1441e-02,
        -3.1236e-05,  5.9081e-04,  1.3647e-02, -1.0328e-02,  8.0563e-03,
        -3.3079e-03,  4.6604e-03, -8.5165e-03,  1.6161e-02, -3.2954e-02,
        -2.0649e-03,  2.0905e-02, -2.5670e-03, -2.3459e-03, -8.4911e-03,
        -2.7834e-03, -1.8105e-02, -5.7025e-03,  1.2369e-02,  7.1385e-03,
        -1.5990e-02])
tensor([[[[ 3.8637e-04,  7.7956e-04,  9.7670e-04,  8.5147e-04,  3.3180e-04],
          [ 4.7164e-04,  9.3628e-04,  1.0185e-03,  1.1781e-03,  8.8011e-04],
          [ 5.3961e-04,  4.5859e-04,  1.1403e-03,  1.2162e-03,  1.1072e-03],
          [-1.0037e-03, -5.6959e-04,  5.5889e-04,  1.1770e-03,  8.9725e-04],
          [-1.0538e-03, -7.4676e-04,  3.3979e-04,  8.5516e-04,  5.1991e-04]],

         [[ 6.8840e-04,  9.1278e-04,  7.4592e-04,  9.4844e-04,  4.1164e-04],
          [ 7.2220e-04,  1.0343e-04,  5.4313e-05,  3.5743e-04,  5.1320e-04],
          [ 1.0240e-03,  9.5360e-05,  6.3706e-04,  1.1145e-03,  1.2275e-03],
          [ 7.5201e-04,  2.6115e-04,  5.0647e-04,  8.1044e-04,  6.8652e-04],
          [ 9.9594e-04,  3.1648e-04,  7.5003e-04,  8.7090e-04,  8.8659e-04]],

         [[ 6.6783e-04,  3.8346e-04,  1.6119e-04,  7.3959e-04,  7.0099e-04],
          [ 1.4631e-03,  9.1020e-04,  6.5740e-04,  8.9279e-04,  8.8615e-04],
          [ 2.6312e-03,  1.9149e-03,  1.3383e-03,  1.5119e-03,  1.2677e-03],
          [ 2.4252e-03,  1.7525e-03,  1.3783e-03,  1.2063e-03,  6.8868e-04],
          [ 2.7618e-03,  1.9689e-03,  1.4639e-03,  1.4649e-03,  7.8337e-04]],

         …,

         [[ 9.0010e-04,  1.0032e-03,  6.6916e-04,  7.6256e-04,  8.8754e-04],
          [ 1.0301e-03,  8.3585e-04,  4.7139e-04,  8.2405e-04,  1.0815e-03],
```

```
  [ 1.8346e-03,  1.2928e-03,  1.3022e-03,  1.3388e-03,  1.5289e-03],
  [ 9.2723e-04,  6.6649e-04,  1.2218e-03,  1.1826e-03,  9.9613e-04],
  [ 9.6475e-04,  8.0605e-04,  1.0820e-03,  1.2565e-03,  1.0088e-03]],

 [[ 8.7097e-04,  1.1536e-03,  1.3512e-03,  1.2515e-03,  1.0254e-03],
  [ 9.8739e-04,  1.0922e-03,  1.2665e-03,  1.3964e-03,  1.1784e-03],
  [ 1.6066e-03,  1.1544e-03,  1.7658e-03,  1.7659e-03,  1.6294e-03],
  [ 2.7095e-04,  5.1504e-04,  1.4212e-03,  1.4747e-03,  1.1689e-03],
  [ 1.8031e-04,  4.8787e-04,  1.1954e-03,  1.1854e-03,  9.8903e-04]],

 [[ 7.2073e-04,  9.6506e-04,  1.1489e-03,  9.9814e-04,  7.4071e-04],
  [ 1.3390e-03,  1.4767e-03,  1.6093e-03,  1.6553e-03,  1.0717e-03],
  [ 1.4876e-03,  1.9687e-03,  2.1848e-03,  1.9277e-03,  1.4188e-03],
  [ 1.3833e-03,  1.7354e-03,  2.1452e-03,  1.9850e-03,  1.4003e-03],
  [ 1.1292e-03,  1.6548e-03,  2.0919e-03,  2.0286e-03,  1.4358e-03]]],


[[[-3.0728e-04, -2.7363e-04, -4.5253e-04, -3.9584e-04, -2.9254e-04],
  [-3.6854e-04, -4.3814e-04, -5.2059e-04, -3.2331e-04, -2.2173e-04],
  [-6.3009e-04, -6.1899e-04, -7.1040e-04, -5.1393e-04, -3.5909e-04],
  [-3.2971e-04, -4.3746e-04, -7.1693e-04, -7.6665e-04, -6.1971e-04],
  [-2.6108e-04, -7.3906e-04, -1.0387e-03, -1.0836e-03, -1.1322e-03]],

 [[ 7.9684e-04,  4.8983e-04,  6.1662e-04,  6.4151e-04,  6.1803e-04],
  [ 6.3264e-04,  5.7547e-04,  5.9773e-04,  4.1407e-04,  5.6075e-04],
  [-2.7464e-04, -5.5352e-04, -7.5737e-04, -6.0247e-04, -2.3016e-04],
  [-1.8856e-04, -3.6183e-04, -4.0349e-04,  1.6830e-04,  2.0261e-04],
  [-6.7528e-05,  1.7035e-04,  1.8866e-04,  5.5865e-04,  5.6589e-04]],

 [[ 1.2062e-03,  1.0386e-03,  4.9327e-04,  4.6268e-04,  2.9995e-04],
  [ 9.4789e-04,  8.5140e-04,  4.8393e-04,  6.5559e-04,  4.2558e-04],
  [ 8.0799e-05,  8.3042e-06, -1.8381e-04, -1.3393e-04, -4.4442e-05],
  [ 2.8908e-04,  1.0282e-04,  5.1731e-05,  2.8914e-04,  3.2804e-04],
  [ 6.7988e-04,  6.2474e-04,  5.2439e-04,  6.6593e-04,  7.9746e-04]],

 …,

 [[ 5.0961e-05, -9.6438e-05, -5.4939e-04, -4.6520e-04, -2.9092e-04],
  [-4.6792e-05, -2.6963e-04, -5.8255e-04, -4.0491e-04, -2.9741e-04],
  [-8.6835e-04, -1.0589e-03, -1.1928e-03, -1.0971e-03, -9.1948e-04],
  [-6.9135e-04, -8.9628e-04, -1.0097e-03, -9.6046e-04, -8.7732e-04],
  [-7.1848e-04, -8.7928e-04, -8.5531e-04, -8.8398e-04, -7.5519e-04]],

 [[-2.0595e-04, -2.4259e-04, -3.8315e-04, -2.6516e-04, -2.5437e-04],
  [-2.0267e-04, -2.3519e-04, -3.0654e-04, -2.4206e-04, -2.2146e-04],
  [-9.0326e-04, -9.9632e-04, -1.0954e-03, -1.0676e-03, -1.0294e-03],
  [-8.8697e-04, -9.7607e-04, -1.0778e-03, -1.0856e-03, -1.0423e-03],
  [-9.3892e-04, -1.0095e-03, -1.0338e-03, -1.0773e-03, -1.0299e-03]],
```

```
 [[-1.8785e-04, -2.2644e-04, -4.6018e-04, -4.5414e-04, -4.2938e-04],
  [-1.2001e-04, -9.5737e-05, -2.7811e-04, -2.0576e-04, -3.1902e-04],
  [ 2.1529e-04,  1.9680e-04,  1.3581e-05,  3.6675e-05,  8.3566e-05],
  [ 4.0802e-04,  2.5946e-04,  1.4556e-04,  1.4976e-04,  1.5609e-04],
  [ 1.8913e-04,  1.3083e-04,  5.8568e-05,  7.7419e-05,  1.2431e-04]]],


[[[ 9.4506e-04,  8.4179e-04,  9.0234e-04,  1.0702e-03,  1.1672e-03],
  [ 1.3018e-03,  1.0973e-03,  9.3291e-04,  1.5070e-03,  1.5048e-03],
  [ 1.4576e-03,  1.3431e-03,  1.1995e-03,  1.0610e-03,  8.5884e-04],
  [ 1.1092e-03,  9.9760e-04,  5.8468e-04,  8.5187e-04,  1.0891e-03],
  [ 6.0017e-04,  4.5512e-04,  4.8282e-04,  8.4869e-04,  9.6210e-04]],

 [[-1.4664e-04, -5.6352e-04, -7.9359e-04, -2.9689e-04,  3.1322e-05],
  [-3.3179e-04, -5.2583e-04, -8.2644e-04, -6.3646e-04, -4.6301e-04],
  [-4.4618e-04, -6.9536e-04, -1.0272e-03, -6.6024e-04, -4.0556e-04],
  [-7.2565e-04, -9.6299e-04, -1.1640e-03, -5.4022e-04, -6.1372e-04],
  [-6.2989e-04, -7.1691e-04, -1.3460e-03, -1.1793e-03, -1.7170e-03]],

 [[ 5.1022e-04,  1.5480e-04, -4.4380e-04, -3.1658e-04, -6.6877e-05],
  [ 1.8811e-04, -4.1294e-05, -2.8077e-04, -2.6964e-04, -4.3628e-04],
  [-3.8102e-04, -3.7430e-04, -9.3261e-04, -4.7547e-04, -4.4799e-04],
  [-8.3014e-04, -6.7124e-04, -1.0609e-03, -7.2480e-04, -4.3350e-04],
  [-5.9145e-04, -5.6329e-04, -7.2004e-04, -7.9382e-04, -1.0408e-03]],

 …,

 [[ 6.4446e-04,  6.3870e-04,  2.9920e-04,  2.8696e-04,  6.4662e-04],
  [ 4.3391e-04,  5.4886e-04,  9.8180e-05,  1.3480e-04,  2.4242e-04],
  [ 3.9051e-04,  3.0086e-04,  3.4480e-05,  1.5851e-04,  1.0609e-06],
  [ 2.3624e-04,  3.3368e-04,  9.6780e-05,  5.0129e-06, -7.7459e-05],
  [ 2.9614e-04,  5.3380e-04,  1.3566e-04, -2.7851e-04, -7.0163e-04]],

 [[ 5.3902e-04,  5.7855e-04,  2.7548e-04,  2.8720e-04,  4.5818e-04],
  [ 3.3194e-04,  3.6862e-04,  1.0823e-04,  1.9409e-04,  2.7489e-04],
  [ 6.3904e-04,  6.2365e-04,  2.4379e-04,  2.5645e-04,  2.6808e-05],
  [ 4.7094e-04,  7.0639e-04,  4.1701e-04,  1.1224e-04, -1.2666e-04],
  [ 4.1344e-04,  6.1550e-04,  2.3595e-04, -1.8696e-04, -2.6429e-04]],

 [[ 6.1828e-04,  7.9909e-04,  7.5296e-04,  9.6375e-04,  1.0456e-03],
  [ 9.7860e-04,  9.9554e-04,  5.9866e-04,  7.1868e-04,  6.5496e-04],
  [ 8.4078e-04,  8.6672e-04,  4.6484e-04,  6.2245e-04,  8.8510e-04],
  [ 4.3998e-04,  4.4608e-04,  1.8294e-04,  1.2861e-04,  2.2086e-04],
  [ 2.9623e-04,  2.1552e-04,  9.5068e-05, -7.3247e-07, -1.9524e-04]]],


…,
```

```
[[[ 6.1865e-04,  8.1067e-04,  9.6372e-04,  5.8709e-04,  2.9052e-04],
  [ 3.1631e-04,  3.8679e-04,  5.8264e-04,  3.6679e-04,  2.2564e-04],
  [ 1.2112e-04,  7.2374e-05,  1.1030e-04,  5.3278e-05,  1.2998e-04],
  [ 2.1168e-04,  1.5918e-04,  3.7525e-04,  5.0096e-04,  5.3555e-04],
  [ 2.4739e-04,  2.9380e-04,  6.4303e-04,  5.5479e-04,  5.5814e-04]],

 [[-1.0787e-03, -1.3729e-03, -8.1684e-04,  1.1081e-04,  3.9203e-04],
  [-1.3370e-03, -1.6335e-03, -1.1075e-03,  3.2464e-05,  5.8066e-04],
  [-1.8336e-03, -1.8509e-03, -1.0892e-03, -2.0564e-04,  2.5245e-04],
  [-1.6108e-03, -1.2878e-03, -8.8660e-04, -6.3265e-04, -2.2840e-04],
  [-1.4597e-03, -1.5411e-03, -1.2314e-03, -8.9197e-04, -5.8374e-04]],

 [[-5.3607e-04, -7.1017e-04, -8.8251e-04, -5.2173e-04, -5.3430e-05],
  [-8.7142e-04, -1.1067e-03, -1.2729e-03, -9.4134e-04, -4.4041e-04],
  [-1.2635e-03, -1.4411e-03, -1.4201e-03, -1.0221e-03, -5.3046e-04],
  [-1.2698e-03, -1.3370e-03, -1.0488e-03, -8.4358e-04, -5.2159e-04],
  [-1.3260e-03, -1.2525e-03, -1.1799e-03, -9.0435e-04, -6.3608e-04]],

 …,

 [[ 4.5622e-05, -1.5420e-04, -3.4528e-04, -3.7699e-04, -2.2901e-04],
  [ 2.3040e-05, -1.4908e-04, -1.8710e-04, -3.7139e-04, -1.6909e-04],
  [-6.1575e-05, -1.4889e-04, -8.7383e-05, -1.4695e-04, -2.5497e-05],
  [-1.1456e-04, -1.7386e-04,  7.8517e-05,  4.2534e-05,  3.3703e-05],
  [-2.6117e-04, -2.6497e-04, -5.8126e-05, -6.8574e-05, -5.9581e-05]],

 [[-1.6065e-04, -2.5995e-04, -3.7993e-04, -5.2887e-04, -5.7456e-04],
  [-2.0291e-04, -2.8805e-04, -3.1206e-04, -4.7939e-04, -5.2148e-04],
  [-1.6180e-04, -2.0069e-04, -1.1664e-04, -3.5363e-04, -3.9242e-04],
  [-1.8542e-04, -1.9338e-04, -3.3730e-05, -2.1522e-04, -3.6369e-04],
  [-1.9408e-04, -2.3306e-04, -4.6630e-05, -1.4793e-04, -2.9697e-04]],

 [[-3.5275e-05, -2.0220e-04, -3.2637e-04, -4.0847e-04, -3.3505e-04],
  [-7.9665e-05, -1.2348e-04, -9.4941e-05, -2.0108e-04,  8.2343e-05],
  [-3.0464e-04, -3.4107e-04, -1.4496e-05, -4.5970e-05,  3.2361e-05],
  [-3.2501e-04, -2.5627e-04,  6.0692e-05, -4.6756e-05, -1.6062e-04],
  [-3.4408e-04, -2.4958e-04,  9.4203e-05, -3.3874e-06, -3.1686e-05]]],


[[[-7.2079e-05,  4.5370e-04,  3.6943e-04,  5.3700e-04,  5.0708e-04],
  [-4.3592e-04, -5.8036e-06,  4.1406e-04,  9.2736e-04,  6.4620e-04],
  [-1.0425e-03, -7.9293e-04, -1.2209e-04,  4.3289e-04,  1.5427e-04],
  [-8.0333e-04, -4.1231e-04,  1.0786e-05,  3.0544e-04,  4.0846e-04],
  [-5.1412e-05, -1.7147e-04, -3.1456e-04,  3.2113e-04,  5.6183e-04]],

 [[ 3.1626e-04,  1.2040e-03,  1.6201e-03,  1.1615e-03,  8.8479e-04],
```

```
  [ 3.5395e-04,  1.1042e-03,  1.2368e-03,  6.9215e-04,  4.7174e-04],
  [-1.1388e-04,  1.1952e-03,  8.3815e-04, -2.2125e-04,  1.8482e-04],
  [ 1.8824e-04,  9.8757e-04,  2.1290e-04, -4.0249e-04,  1.7794e-04],
  [ 3.0708e-04,  9.0425e-04,  5.9784e-04,  2.5355e-04,  4.0225e-04]],

 [[-6.3989e-04, -7.9914e-04, -2.2234e-04, -2.6218e-04, -4.5434e-04],
  [-5.9476e-04, -7.7993e-04, -4.8982e-04, -5.5859e-04, -5.3382e-04],
  [-5.5612e-04, -5.5794e-04, -1.9120e-04, -4.8344e-04, -8.6112e-04],
  [ 9.0846e-05,  1.6586e-04,  5.3364e-04,  7.5375e-06, -4.0348e-05],
  [-2.2622e-04,  1.7919e-04,  4.8748e-04,  2.8684e-04,  8.9800e-05]],

 …,

 [[-4.9830e-04, -3.6864e-04, -1.2861e-04,  4.2320e-04,  5.7723e-04],
  [-2.0407e-05, -7.5171e-05,  3.9952e-05,  3.6603e-04,  4.0863e-04],
  [-2.3550e-04, -5.4703e-04, -2.7906e-04,  8.1401e-05, -1.4094e-04],
  [-1.8061e-04, -5.3414e-04, -4.2841e-04, -2.2276e-04, -2.2331e-04],
  [-3.5780e-04, -5.4951e-04, -4.1463e-04, -3.8877e-04, -4.1608e-04]],

 [[-2.0969e-05,  2.2510e-05,  1.5931e-04,  7.5799e-04,  7.5316e-04],
  [ 1.5795e-04,  6.2669e-05,  4.6380e-05,  5.0503e-04,  7.2832e-04],
  [-2.4086e-04, -4.7274e-04, -4.3559e-04, -2.1375e-05,  9.8870e-05],
  [-3.0836e-04, -7.7319e-04, -7.5453e-04, -3.0553e-04, -4.9588e-05],
  [-3.6049e-04, -7.0183e-04, -7.7471e-04, -3.5477e-04, -7.5760e-05]],

 [[-8.0780e-04, -7.6394e-04, -7.0270e-04,  1.3732e-04,  1.1271e-04],
  [-1.1883e-03, -8.9799e-04, -3.7849e-04,  1.6290e-05, -2.1154e-04],
  [-1.2709e-03, -1.0950e-03, -3.0598e-04, -5.8431e-05, -2.1267e-04],
  [-7.1363e-04, -5.3985e-04, -3.2819e-04,  2.6384e-04,  1.3231e-04],
  [-2.3775e-04, -4.2777e-04, -4.7342e-04,  1.9853e-05, -3.0802e-05]]],


[[[ 3.2769e-04,  2.8081e-04,  5.3704e-05, -1.7180e-04, -4.5080e-04],
  [ 2.3295e-04,  3.4849e-04,  1.1918e-04, -1.1303e-04, -2.6953e-04],
  [ 2.5588e-04,  2.2805e-04,  1.3362e-04,  3.4531e-05, -1.2851e-04],
  [ 2.3048e-04,  1.0510e-05, -1.7061e-04, -1.8576e-04, -3.4265e-04],
  [ 1.7382e-04,  1.7333e-04,  1.0063e-04,  9.5654e-05, -1.4374e-04]],

 [[-1.2453e-03, -8.4066e-04, -1.1784e-03, -1.3482e-03, -9.0128e-04],
  [-7.6762e-04, -5.3624e-04, -8.9110e-04, -8.3411e-04, -8.2628e-05],
  [-7.6721e-04, -6.9294e-04, -9.5769e-04, -6.4044e-04,  8.7097e-05],
  [-7.1151e-04, -4.8130e-04, -4.8816e-04, -4.9714e-05,  4.0007e-04],
  [-6.1911e-04, -6.3120e-04, -2.9406e-04,  1.4476e-04,  3.7791e-04]],

 [[-1.3110e-03, -1.0637e-03, -5.5024e-04, -7.9921e-04, -7.4872e-04],
  [-1.3517e-03, -9.8757e-04, -4.7330e-04, -5.0782e-04, -5.3884e-04],
  [-1.5586e-03, -1.1827e-03, -7.4794e-04, -7.0434e-04, -3.6233e-04],
  [-1.4837e-03, -1.0621e-03, -4.6163e-04, -5.9737e-04, -1.4302e-04],
```

```
          [-1.5990e-03, -1.1412e-03, -9.1989e-04, -3.9884e-04, -1.9258e-04]],

         …,

         [[-1.2081e-03, -1.1540e-03, -1.0676e-03, -1.6990e-03, -1.9170e-03],
          [-1.2203e-03, -1.2911e-03, -1.1968e-03, -1.6122e-03, -1.8387e-03],
          [-1.3633e-03, -1.5866e-03, -1.6172e-03, -1.8370e-03, -1.6141e-03],
          [-1.2935e-03, -1.5444e-03, -1.3577e-03, -1.5120e-03, -8.3451e-04],
          [-1.4020e-03, -1.5209e-03, -1.4072e-03, -8.8387e-04, -7.9852e-04]],

         [[-8.3282e-04, -9.2077e-04, -1.2113e-03, -1.6437e-03, -1.6309e-03],
          [-8.1938e-04, -9.8618e-04, -1.2766e-03, -1.5982e-03, -1.5603e-03],
          [-9.5742e-04, -1.2193e-03, -1.4902e-03, -1.7409e-03, -1.4385e-03],
          [-1.0374e-03, -1.3082e-03, -1.3690e-03, -1.5061e-03, -9.0696e-04],
          [-1.1551e-03, -1.2975e-03, -1.3071e-03, -9.6705e-04, -7.8990e-04]],

         [[ 5.3664e-04,  6.1730e-04,  6.5436e-04,  5.6494e-04,  4.9171e-04],
          [ 4.0397e-04,  4.0047e-04,  5.6009e-04,  5.7136e-04,  5.6455e-04],
          [ 5.4236e-04,  5.9230e-04,  6.5455e-04,  6.5135e-04,  4.2736e-04],
          [ 6.4880e-04,  5.7645e-04,  6.4049e-04,  5.8652e-04,  4.7941e-04],
          [ 7.1104e-04,  6.8102e-04,  6.3243e-04,  6.5654e-04,  7.4309e-04]]]])
tensor([-1.8563e-09, -1.0377e-09,  7.7762e-10, -5.0932e-10, -1.0541e-09,
         9.2496e-10, -5.2563e-10,  1.0351e-08, -1.5293e-09, -8.5365e-09,
        -1.2153e-09, -1.7413e-09,  4.0407e-09,  1.5205e-08,  2.6375e-10,
         4.0072e-09,  6.6967e-10, -6.3268e-09, -6.6491e-09, -1.8982e-09,
         1.8626e-09, -6.0000e-09, -2.4195e-10,  1.3562e-09,  2.3397e-09,
         1.4511e-08,  7.5131e-09,  1.8020e-09,  8.7766e-11, -2.0149e-09,
         3.3542e-09,  1.1072e-08,  3.8171e-09, -8.6216e-09, -1.1125e-08,
         1.8827e-09,  1.0145e-09,  9.0495e-09,  9.3659e-10,  1.3051e-09,
        -6.2391e-10,  1.2113e-10, -1.2769e-09, -3.4581e-10, -1.3474e-09,
         5.2696e-09, -8.7471e-10, -9.2547e-09,  1.0562e-08,  1.8277e-10,
        -1.9113e-09, -4.8096e-09, -4.3522e-09, -3.8594e-09, -2.2875e-09,
         2.3489e-09,  8.4656e-09,  9.7771e-10, -5.0190e-09, -1.9219e-08,
         2.1114e-09, -5.2498e-10, -9.4678e-09, -1.3822e-09,  1.1878e-09,
        -1.1314e-09,  4.2837e-09, -5.0386e-10, -1.4468e-08,  1.7847e-09,
        -3.6899e-09, -6.1755e-10,  2.1021e-10,  6.4338e-09, -3.3386e-09,
         9.5196e-09,  5.3827e-10,  2.9631e-10,  2.1937e-09, -1.4494e-09,
         1.1588e-08, -7.0497e-10,  1.7579e-09,  2.5611e-10, -6.7030e-10,
         6.3264e-09, -1.0243e-08,  8.8694e-09,  2.8303e-09, -1.1405e-09,
        -2.9853e-09, -3.9273e-10,  3.7582e-09, -1.0453e-09, -3.8091e-09,
         4.6899e-09, -2.0818e-09, -3.3288e-10,  7.3987e-10,  1.7538e-10,
        -3.6424e-09, -1.6328e-09,  2.2073e-09,  1.1078e-09, -3.2665e-09,
         9.2707e-09,  2.2446e-09,  8.5211e-09, -5.4480e-09,  7.3947e-09,
        -1.3770e-09,  2.8012e-09,  8.7020e-09,  2.3670e-09, -1.0219e-09,
        -2.9229e-09,  8.2173e-10,  7.6307e-10, -3.5526e-10,  4.6435e-09,
         7.4919e-09,  1.9468e-09, -4.2201e-10, -2.7303e-09,  2.8278e-09,
         1.1792e-10, -8.2105e-10,  4.1609e-11, -5.2842e-10, -8.0285e-09,
         6.7313e-10, -7.0449e-09, -1.2998e-09,  4.6217e-10, -2.1177e-10,
```

```
        -1.0251e-08,  1.0108e-08, -7.4187e-09,  2.4520e-09, -2.8425e-09,
         2.3167e-09, -2.7985e-09,  2.3417e-09,  2.4913e-09,  5.9672e-09,
        -4.3074e-09, -8.7130e-10,  1.1739e-09,  2.8840e-09, -2.3233e-09,
        -2.2023e-09, -2.5598e-09, -2.6048e-09,  1.7848e-09, -1.7007e-09,
         2.7745e-09, -3.2491e-09,  9.1040e-10, -2.8657e-10,  6.5825e-10,
        -1.3909e-09,  1.5852e-09, -1.6939e-09, -1.7296e-09,  2.2698e-09,
         8.2886e-09,  1.3293e-08,  3.1204e-09, -2.6954e-10,  2.0764e-09,
         3.0164e-08,  3.3340e-09,  1.5815e-08, -4.1668e-09,  9.5950e-09,
         2.7676e-09, -4.2562e-10,  4.5786e-09, -2.5975e-09, -1.4807e-09,
         3.9427e-09, -1.6965e-09,  3.5616e-09,  6.5666e-10,  1.5370e-10,
        -6.0446e-11, -4.4668e-10, -2.8556e-09,  2.1056e-09,  1.0106e-09,
        -6.3390e-09,  1.7333e-09, -1.2558e-09, -4.1039e-09, -1.5351e-09,
         1.6753e-09,  2.4988e-09,  8.8039e-10, -3.0482e-10,  3.6617e-09,
         3.7815e-11,  5.2722e-10,  6.9300e-09,  3.4252e-09,  1.4643e-10,
         3.2378e-09, -4.4084e-09,  4.7532e-10,  3.2269e-09, -4.1958e-09,
        -9.8134e-10, -3.8999e-09, -2.4461e-09, -3.3985e-09,  1.8736e-09,
        -9.0495e-11,  2.5859e-10,  3.5930e-09, -1.0584e-09, -1.9458e-09,
        -1.2833e-09,  4.6890e-09, -1.9575e-09,  2.7674e-09,  5.1315e-09,
         3.8926e-10, -5.9572e-10,  5.3021e-10,  1.4789e-09,  8.7782e-09,
        -1.6419e-09, -6.2755e-10, -2.1708e-09,  5.7410e-10,  2.1422e-09,
         1.0817e-08, -9.3110e-11, -2.3915e-09, -1.2847e-09,  5.4206e-09,
        -3.9149e-09, -9.7977e-09,  5.1917e-09, -2.3219e-10,  4.4274e-09,
        -1.6588e-09,  4.4976e-09, -4.9118e-09,  3.0735e-10,  1.1366e-09,
         3.9695e-09,  2.3902e-09,  4.2955e-09,  9.4951e-10,  3.1955e-09,
        -7.7350e-09])
tensor([-0.0003, -0.0016, -0.0042, -0.0017,  0.0008,  0.0004, -0.0051, -0.0014,
         0.0205,  0.0026, -0.0040, -0.0024, -0.0105, -0.0045, -0.0055,  0.0025,
        -0.0011, -0.0060, -0.0317, -0.0403,  0.0022, -0.0337, -0.0007,  0.0033,
         0.0019, -0.0106,  0.0070, -0.0307,  0.0059, -0.0056, -0.0010, -0.0215,
        -0.0162, -0.0215,  0.0306,  0.0036,  0.0022,  0.0051, -0.0236, -0.0030,
         0.0026, -0.0127, -0.0009,  0.0060, -0.0036, -0.1054, -0.0087,  0.0157,
        -0.0488,  0.0084,  0.0025,  0.0092,  0.0221,  0.0019,  0.0041, -0.0200,
         0.0032,  0.0338, -0.0083, -0.0278,  0.0005,  0.0039,  0.0091, -0.0066,
        -0.0016,  0.0013,  0.0077, -0.0027, -0.0836,  0.0007,  0.0283,  0.0219,
        -0.0053,  0.0026,  0.0099, -0.0218, -0.0049,  0.0018,  0.0021,  0.0076,
         0.0217,  0.0324,  0.0052,  0.0105,  0.0064,  0.0179, -0.0002,  0.0014,
         0.0084,  0.0047, -0.0086, -0.0035,  0.0263,  0.0203, -0.0485,  0.0048,
         0.0023, -0.0025, -0.0006,  0.0029,  0.0200, -0.0008,  0.0010, -0.0009,
        -0.0002,  0.0268,  0.0019,  0.0110,  0.0073, -0.0014,  0.0057, -0.0046,
         0.0036,  0.0062,  0.0007,  0.0007,  0.0052,  0.0014,  0.0036,  0.0183,
        -0.0271,  0.0026,  0.0026,  0.0019,  0.0016, -0.0006,  0.0046, -0.0038,
         0.0010,  0.0422,  0.0032, -0.0080,  0.0014,  0.0002, -0.0039,  0.0595,
        -0.0016, -0.0147, -0.0022,  0.0109, -0.0047, -0.0020,  0.0055, -0.0025,
         0.0011,  0.0026, -0.0118,  0.0037,  0.0189,  0.0010,  0.0052, -0.0060,
        -0.0102, -0.0214, -0.0037,  0.0057,  0.0022, -0.0003, -0.0056, -0.0056,
        -0.0083, -0.0001, -0.0065,  0.0022,  0.0166,  0.0224, -0.0056, -0.0044,
         0.0049,  0.0270,  0.0074,  0.0035,  0.0048, -0.0082, -0.0021,  0.0055,
         0.0029, -0.0056,  0.0054, -0.0077,  0.0008,  0.0027,  0.0022,  0.0058,
```

```
        -0.0003,  0.0005, -0.0079, -0.0015, -0.0038,  0.0032,  0.0156,  0.0094,
         0.0018, -0.0164, -0.0054, -0.0012,  0.0034,  0.0053,  0.0030,  0.0137,
        -0.0015,  0.0050,  0.0113,  0.0066,  0.0107,  0.0061, -0.0004,  0.0044,
        -0.0066,  0.0240, -0.0062,  0.0055, -0.0065,  0.0059,  0.0032, -0.0081,
         0.0011,  0.0194,  0.0029,  0.0022,  0.0019, -0.0024, -0.0053, -0.0180,
        -0.1155,  0.0018,  0.0007,  0.0238, -0.0076, -0.0195,  0.0090, -0.0015,
        -0.0227, -0.0200,  0.0006,  0.0178,  0.0615, -0.0245,  0.0139,  0.0037,
         0.0098,  0.0153,  0.0064, -0.0053, -0.0224,  0.0032, -0.0189, -0.0140,
         0.0066, -0.0800,  0.0011,  0.0063,  0.0007, -0.0010,  0.0034,  0.0001])
tensor([ 1.9366e-03,  5.1566e-04, -2.6468e-03, -1.5283e-03, -2.5608e-03,
         1.7326e-03, -5.1725e-03, -4.6380e-03,  1.6995e-02, -2.5514e-03,
        -3.3574e-03, -6.4664e-03,  4.3460e-03, -6.6523e-03, -7.9323e-03,
         5.6531e-03, -1.6911e-03, -1.0941e-02, -2.8736e-02, -3.6579e-02,
         6.3668e-04, -1.5140e-02, -3.8424e-03,  2.4790e-03,  1.2856e-03,
        -6.2770e-03,  5.4797e-03, -2.6346e-02,  8.1825e-03, -4.4324e-03,
        -3.9236e-04, -9.8403e-03, -1.0763e-02, -2.1944e-02,  2.0217e-02,
         5.5660e-03,  3.0805e-03,  4.7349e-03, -8.6056e-03, -2.3190e-03,
         4.3468e-03, -5.2599e-03, -2.6182e-04,  8.4027e-03, -5.0478e-03,
        -6.6027e-02, -6.0671e-03,  9.5576e-03, -2.5698e-02,  5.8220e-03,
         5.1093e-03,  1.3009e-02,  1.5588e-02,  5.7186e-03,  1.1435e-02,
        -1.9143e-02,  1.1178e-02,  1.5744e-02, -1.1852e-03, -1.8928e-02,
         3.3886e-04,  7.7846e-03,  6.7258e-03, -1.1283e-02, -3.7519e-03,
         2.0589e-03,  1.2473e-02, -1.4759e-04, -5.1668e-02, -1.8072e-03,
         2.1378e-02,  1.3438e-02, -6.3084e-03,  6.0882e-03, -3.9967e-03,
        -2.4629e-03, -1.8682e-03, -4.4196e-03,  2.3543e-03,  8.4057e-03,
         1.6587e-02,  1.8357e-02, -1.0176e-03,  3.2612e-03,  8.5847e-03,
         1.9772e-02, -7.5796e-03, -1.0189e-02,  1.1463e-02,  6.7401e-03,
        -1.2414e-02, -1.6254e-03,  1.6740e-02,  1.5302e-02, -3.2875e-02,
        -4.4550e-03,  9.9880e-04, -2.1467e-03,  2.9056e-03,  2.4141e-03,
         4.0681e-03, -4.4246e-03, -2.7600e-03, -5.7465e-03, -3.5893e-03,
         1.5278e-02, -9.6161e-05,  1.5110e-02,  7.3328e-03, -1.0129e-03,
         9.2266e-03, -8.3652e-03,  6.2202e-03,  1.1513e-02,  5.4297e-03,
        -4.2316e-03,  6.7995e-03,  1.8240e-03,  1.4730e-03,  1.7545e-02,
        -1.2238e-02,  1.6518e-03,  7.1229e-03,  4.7394e-03,  6.4847e-03,
         3.9609e-03, -7.3843e-05, -7.3157e-03,  3.9752e-03,  3.4098e-02,
         2.0138e-03,  4.6446e-03,  4.1891e-03,  4.8870e-03, -2.0051e-03,
         4.0980e-02, -2.1752e-03, -2.1078e-02, -5.8004e-03,  4.6593e-03,
        -6.1796e-03, -2.1551e-03,  1.2001e-02,  3.8589e-03,  3.5287e-04,
        -2.6261e-03, -4.1083e-04, -2.5965e-03,  6.0451e-03, -2.0835e-03,
         7.3103e-03, -6.5901e-03, -1.0533e-03, -6.6579e-03, -5.5946e-03,
         4.8145e-03,  3.1925e-03, -1.1541e-03, -1.9553e-03, -1.2876e-02,
        -6.9805e-03, -2.9077e-03, -1.5142e-02, -1.0155e-03,  1.5920e-02,
         2.1518e-02, -3.9504e-03,  1.6605e-03,  3.0604e-03,  2.0502e-02,
         1.3597e-03,  4.0194e-03,  4.3376e-03, -1.7740e-02, -1.4999e-02,
         5.1702e-03,  5.5060e-03, -5.1875e-03,  2.3069e-03, -8.1768e-03,
        -2.7814e-06,  5.2517e-03,  4.3620e-03,  1.1388e-02, -5.6789e-03,
         6.7176e-04, -1.6499e-02,  1.9512e-03, -7.2679e-03,  7.2826e-04,
         1.3852e-02,  7.7262e-03, -1.1960e-03, -1.3396e-02, -7.6867e-03,
```

```
        -3.8599e-03,  1.4077e-03,  7.6771e-03,  2.2477e-03,  1.9293e-03,
        -6.1542e-03,  7.1801e-03,  2.2647e-03,  1.2579e-02,  4.1909e-03,
         1.3484e-02, -1.0254e-02,  9.2410e-03, -1.9661e-03,  2.0826e-02,
        -5.1451e-03,  7.4481e-03, -2.3434e-03, -1.0843e-02,  4.0369e-03,
        -9.2673e-03, -5.3097e-04,  1.8745e-02,  2.3177e-03,  5.2148e-03,
         2.8058e-03, -3.0556e-03, -2.4069e-02, -2.7552e-02, -3.8462e-02,
         3.7844e-03,  6.8925e-04,  1.5739e-02, -1.0637e-02, -3.2672e-02,
         6.2016e-03, -2.7366e-03, -1.5740e-02, -6.0982e-03,  2.2654e-03,
         1.8932e-02,  3.5494e-02, -1.4191e-02,  8.5355e-03,  6.7914e-03,
         1.9180e-02,  1.2764e-02, -3.9835e-03, -3.2552e-03, -1.2416e-02,
         7.9200e-03, -1.0047e-02, -1.2116e-02,  9.1253e-03, -2.5669e-02,
         4.1254e-03,  4.9254e-03,  2.0437e-03,  2.3571e-03,  3.0519e-03,
        -5.1951e-03])
tensor([[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,
```

```
[[[-1.2956e-02, -1.7065e-02, -2.0176e-02],
  [-8.0149e-03, -1.1918e-02, -1.2030e-02],
  [-1.3348e-02, -1.3110e-02, -1.1230e-02]],

 [[-1.8031e-02, -1.8961e-02, -1.8043e-02],
  [-1.1405e-02, -1.2373e-02, -1.1839e-02],
  [-6.8767e-03, -8.8937e-03, -1.0845e-02]],

 [[-1.2113e-02, -1.1698e-02, -1.5050e-02],
  [-7.8521e-03, -9.3338e-03, -1.5616e-02],
  [-1.2659e-02, -1.4295e-02, -2.0830e-02]],

 …,

 [[-9.4261e-03, -8.4070e-03, -6.5371e-03],
  [-9.4465e-03, -5.5043e-03, -4.0859e-03],
  [-1.5173e-02, -9.8592e-03, -7.9381e-03]],

 [[-8.1519e-03, -7.4388e-03, -3.8587e-03],
  [-1.3221e-02, -1.3113e-02, -7.5724e-03],
  [-1.3772e-02, -1.7528e-02, -1.3044e-02]],

 [[-2.5134e-02, -2.7114e-02, -2.7046e-02],
  [-1.9644e-02, -2.3876e-02, -2.4089e-02],
  [-1.5767e-02, -1.9883e-02, -2.2708e-02]]],


[[[-1.0961e-02, -3.0155e-03, -6.5369e-03],
  [-5.1222e-03,  5.5650e-03,  1.1228e-02],
  [-1.1339e-02, -1.1906e-03,  3.6435e-03]],

 [[-1.5176e-02, -6.5623e-03, -8.5670e-03],
  [-7.1852e-03, -4.0069e-04, -4.6008e-03],
  [-9.3983e-03, -5.5473e-03, -1.0115e-02]],

 [[ 5.6118e-04,  1.3539e-02,  8.8109e-03],
  [ 9.1159e-03,  2.1371e-02,  1.2288e-02],
  [-2.5987e-03,  3.0621e-03, -8.9214e-03]],

 …,

 [[-1.7287e-02, -8.5057e-04,  2.1471e-03],
  [-1.6314e-02, -4.5429e-03, -9.3090e-03],
  [-1.9539e-02, -1.6384e-02, -2.2811e-02]],

 [[-2.0159e-02, -1.4342e-02, -1.3875e-02],
  [-1.8396e-02, -1.6202e-02, -2.0615e-02],
```

```
          [-1.9152e-02, -2.2153e-02, -2.7644e-02]],

         [[-1.0511e-02, -1.3927e-03, -4.9028e-03],
          [-1.9665e-05,  9.6772e-03,  9.0540e-03],
          [-2.7170e-04,  6.5131e-03,  3.1528e-03]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([ 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  1.0925e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  3.5698e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  1.8287e-02,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -2.3394e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

```
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  6.4311e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  9.6593e-03,  0.0000e+00, -1.7904e-03,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  9.9855e-03,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00, -5.6214e-03,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  3.9740e-08,  0.0000e+00, -3.0749e-02,  0.0000e+00,
 0.0000e+00,  9.2976e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 2.5516e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 3.7258e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
-2.2254e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  6.0025e-03,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

```
          0.0000e+00,  0.0000e+00,  0.0000e+00,  4.4949e-03,  0.0000e+00,
          0.0000e+00,  0.0000e+00, -5.2802e-03,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  7.7815e-03,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -3.0740e-03,  5.2616e-04,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -2.4908e-02, -4.0229e-04,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  2.2974e-02,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -6.2241e-02, -5.5094e-02,  0.0000e+00])
tensor([[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[ 3.2357e-02,  4.8416e-02,  3.9762e-02],
           [ 2.3730e-02,  5.3695e-02,  4.3650e-02],
           [ 9.9465e-03,  4.3553e-02,  4.2189e-02]],

          [[-8.9638e-02, -4.3736e-02, -1.6998e-03],
           [-1.3665e-01, -7.7892e-02, -2.4601e-02],
           [-6.9118e-02, -2.5014e-02,  1.1328e-02]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


         [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 3.2287e-03,  1.2042e-02,  2.1554e-02],
  [ 1.0971e-02,  1.1684e-02,  9.8201e-03],
  [ 1.0813e-02,  9.2013e-03,  6.5607e-03]],

 [[ 8.6134e-03,  6.7625e-03,  6.9715e-03],
  [ 7.5374e-03,  9.5330e-03,  1.2523e-02],
  [ 1.9451e-02,  2.6832e-02,  3.1341e-02]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,
```

```
[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[-1.9200e-02, -7.0061e-03, -1.6084e-02],
  [-7.6557e-04,  1.2128e-02, -6.8226e-03],
  [ 1.2534e-02,  2.2025e-02,  3.4020e-03]],

 [[-8.5774e-02, -4.8697e-02, -1.8925e-02],
  [-7.0294e-02, -3.7566e-02, -1.2596e-02],
  [-5.2350e-02, -4.1750e-02, -2.5530e-02]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  3.4021e-04,  0.0000e+00],
  [-1.0301e-05,  0.0000e+00,  0.0000e+00],
```

```
          [ 3.1031e-04,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([ 3.5082e-02,  2.1811e-02,  0.0000e+00, -8.9701e-03,  7.6773e-02,
         4.5932e-02,  2.1831e-03,  1.9891e-02,  6.5554e-02,  0.0000e+00,
        -6.1201e-04,  2.1743e-02, -1.5292e-02,  6.7022e-05,  0.0000e+00,
        -7.0127e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,  4.5038e-04,
         2.6277e-03,  1.7951e-04,  3.6359e-03,  4.1282e-03, -1.5799e-04,
        -1.4033e-02,  0.0000e+00,  6.5799e-04,  0.0000e+00, -9.8319e-03,
         2.5444e-03, -4.5291e-02, -5.5132e-06,  0.0000e+00, -2.7070e-03,
         6.4546e-04,  5.6834e-04,  5.4673e-02, -1.8847e-03,  2.9245e-02,
         0.0000e+00, -1.3066e-02,  2.4284e-03, -1.1847e-02,  0.0000e+00,
         3.4177e-02,  6.1181e-04, -2.3698e-04,  2.4349e-02,  1.2712e-03,
         0.0000e+00,  9.9387e-03,  2.4668e-03,  2.3489e-03,  8.6816e-03,
         1.6183e-03,  2.6095e-02,  1.6927e-03, -3.4995e-02,  0.0000e+00,
         0.0000e+00,  8.1014e-05, -4.2122e-03,  1.3316e-04, -3.6609e-02,
         2.6790e-02, -1.9144e-02,  1.3544e-02, -2.7460e-02,  1.1967e-02,
         5.8415e-04,  0.0000e+00,  0.0000e+00, -3.2254e-04, -4.1699e-05,
        -4.7609e-05,  0.0000e+00, -2.0028e-03,  4.8942e-03, -3.7693e-03,
```

```
 2.7945e-03,  1.4131e-03,  1.8454e-03,  0.0000e+00, -5.1939e-03,
 0.0000e+00,  2.6588e-02,  1.2685e-02, -8.4497e-03,  3.4957e-05,
-5.2465e-03, -3.6560e-04, -1.2222e-03,  0.0000e+00,  0.0000e+00,
 0.0000e+00, -1.2775e-02, -3.6422e-02,  0.0000e+00, -8.9804e-03,
 9.5124e-03, -3.2075e-03,  6.3473e-02,  3.9615e-05, -1.2750e-04,
 8.2057e-05, -9.5161e-03, -9.7950e-04, -7.3774e-03, -6.3700e-02,
-1.6848e-03,  2.9083e-03,  1.2509e-03,  1.3263e-03, -6.9061e-02,
 3.9975e-02,  2.7012e-02,  1.1443e-03,  0.0000e+00, -2.3783e-02,
-7.2761e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  2.1009e-03,
 0.0000e+00,  1.3964e-03, -6.8591e-03,  5.1245e-04, -1.4008e-02,
 0.0000e+00,  0.0000e+00,  3.0760e-04,  7.4685e-02, -4.2793e-04,
 7.1299e-02,  0.0000e+00,  6.0127e-04, -4.8474e-04,  2.9147e-05,
-1.3810e-03,  4.4741e-02,  2.1455e-05, -2.1351e-03,  0.0000e+00,
 0.0000e+00,  0.0000e+00, -6.3224e-02, -1.2340e-02,  1.1673e-03,
 0.0000e+00,  2.0427e-03, -3.6074e-03, -2.8234e-04,  0.0000e+00,
-6.5069e-02,  0.0000e+00,  7.3654e-04,  4.2296e-04, -6.2442e-02,
-1.4582e-02, -3.4326e-02,  0.0000e+00, -2.1760e-04,  2.9476e-02,
-1.7321e-04,  2.0902e-03, -1.7823e-03, -3.1715e-03,  1.1982e-02,
-1.7799e-02,  2.4636e-04,  3.3853e-03,  9.1821e-05,  1.9093e-03,
 2.1377e-02,  7.6169e-04,  3.4608e-03, -6.6676e-03,  5.5087e-03,
-8.4151e-05,  5.4814e-02,  4.3196e-03,  4.1430e-03,  8.6893e-04,
 0.0000e+00,  2.6235e-02, -3.3794e-02,  1.2565e-02, -8.9251e-03,
 2.2834e-04,  0.0000e+00, -7.0053e-04, -6.7856e-02, -1.8842e-02,
 1.0810e-03,  1.5238e-03,  6.9410e-03, -8.7124e-03,  2.6592e-04,
-7.0913e-03,  4.3263e-02,  4.9309e-03, -7.1808e-03, -8.4245e-04,
 8.0143e-03,  1.6084e-03, -8.7925e-03, -2.2777e-04,  1.3079e-02,
 2.6168e-03, -9.5991e-04, -1.2902e-03,  2.3724e-03, -2.7143e-03,
-8.9278e-04,  0.0000e+00,  0.0000e+00,  1.5659e-03,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  2.0257e-03,  4.2662e-02,  8.6058e-04,
 1.9325e-05, -4.5026e-03, -1.6707e-02,  1.9682e-03,  2.2519e-05,
-1.5028e-02,  2.3560e-03,  0.0000e+00, -8.0269e-03,  9.7398e-03,
 2.2403e-03,  1.2498e-03,  0.0000e+00,  7.6057e-04, -2.9144e-02,
 1.5369e-03,  1.7497e-02,  0.0000e+00, -1.5465e-04, -2.3893e-02,
 0.0000e+00, -1.4731e-02,  0.0000e+00,  5.7057e-04,  5.4367e-02,
 1.3736e-04, -3.8114e-03,  2.3607e-02,  0.0000e+00,  0.0000e+00,
 3.0564e-05,  0.0000e+00, -7.6254e-04,  8.2174e-02,  0.0000e+00,
 2.3862e-02,  4.6632e-03,  4.6696e-02,  8.2458e-03,  0.0000e+00,
-5.8534e-05, -1.9132e-01, -1.5440e-02, -4.4703e-03, -8.4005e-03,
 0.0000e+00,  0.0000e+00,  3.6258e-05,  0.0000e+00, -4.1363e-02,
-2.6255e-03,  1.3215e-01, -8.5047e-05,  0.0000e+00, -2.1301e-03,
 3.2293e-03,  6.4825e-03, -1.7155e-03,  4.5345e-02, -3.8099e-03,
-3.8755e-04,  1.9241e-05,  4.0835e-03,  1.9774e-04, -3.7123e-03,
-8.3280e-03,  4.6037e-04,  2.8777e-04,  3.0918e-03, -1.0814e-03,
-1.4586e-02,  2.2129e-03,  7.5888e-03,  0.0000e+00,  1.2634e-03,
-1.9117e-02, -4.3224e-03,  4.2813e-05,  6.7422e-02,  0.0000e+00,
 7.1946e-03,  1.6497e-03,  0.0000e+00,  2.3901e-02,  3.7336e-04,
 2.2456e-05,  4.9567e-03, -8.8338e-03, -6.0107e-03, -1.0840e-02,
 0.0000e+00, -2.3066e-03, -3.3586e-04,  5.0598e-03,  1.8342e-03,
```

```
         0.0000e+00, -4.8373e-03, -2.0212e-02,  9.0695e-02, -2.4226e-03,
         1.1114e-02,  0.0000e+00, -1.3714e-03, -8.6242e-02,  0.0000e+00,
         1.0323e-03, -5.9953e-02, -2.1456e-02,  0.0000e+00, -3.2254e-02,
         1.5589e-05, -6.8417e-04, -1.2461e-05,  4.4545e-03,  3.6853e-02,
        -1.8602e-02,  3.1441e-02, -1.0911e-02, -1.2317e-04, -5.3372e-02,
        -4.4270e-02,  2.9260e-03,  4.8888e-05,  0.0000e+00,  0.0000e+00,
         1.9729e-03, -1.9243e-03, -3.9120e-03, -9.6057e-03, -3.7281e-03,
         3.4409e-03, -3.3016e-03,  0.0000e+00, -3.6322e-04, -1.8405e-03,
        -4.8361e-03, -3.2317e-03,  2.9447e-02,  0.0000e+00, -3.5937e-04,
         8.1219e-03,  1.1509e-02, -9.6038e-04,  2.3640e-03,  6.4492e-02,
         2.6543e-05,  2.2144e-03, -2.3569e-03, -4.4211e-03, -9.0395e-03,
         2.5053e-04, -1.6769e-03, -6.0096e-03,  5.4482e-04,  1.0596e-02,
         0.0000e+00, -1.1423e-02,  1.3695e-03,  0.0000e+00])
tensor([[[[-0.0020, -0.0056, -0.0039],
          [-0.0004, -0.0035, -0.0019],
          [-0.0013, -0.0038, -0.0021]],

         [[ 0.0000,  0.0000, -0.0001],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         …,

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]]],


        [[[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],
```

```
 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


…,
```

```
[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
```

```
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]]],


        [[[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         …,

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]]]])
tensor([-9.8882e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  1.7016e-03,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  1.2084e-01, -1.6412e-01,
        -9.4159e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         4.3692e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -9.2273e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -1.0282e-01,  0.0000e+00, -2.8182e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -4.8083e-03,  8.6613e-03,
         0.0000e+00,  0.0000e+00, -1.2320e-03,  0.0000e+00,  0.0000e+00,
        -3.8549e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  3.6514e-01,  0.0000e+00,
        -2.3562e-01, -3.5854e-05, -4.5113e-04,  0.0000e+00,  1.2692e-01,
         0.0000e+00, -3.9008e-03,  0.0000e+00, -1.3698e-02,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -2.4995e-02,  1.3481e-01,  0.0000e+00,
         0.0000e+00, -5.1732e-02, -5.0253e-03,  0.0000e+00,  0.0000e+00,
```

```
        -9.2759e-04,  1.4970e-02,  0.0000e+00,  0.0000e+00, -7.7273e-04,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -2.9169e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -1.5849e-04,  0.0000e+00,  2.2091e-01, -7.0830e-03,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  6.3360e-03,
        -1.8069e-02,  0.0000e+00,  3.3362e-03,  0.0000e+00,  0.0000e+00,
         4.6132e-02,  0.0000e+00, -3.9115e-03,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  1.3011e-02,  2.7315e-01,
         0.0000e+00,  0.0000e+00,  5.4310e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -6.9222e-03,  0.0000e+00,  5.2525e-03,  0.0000e+00,  0.0000e+00,
         2.4800e-03, -1.0328e-04,  1.3874e-02,  4.2648e-01, -6.0499e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  6.6889e-04, -2.4045e-03,  2.2270e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -1.4994e-01,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  3.7254e-02,  0.0000e+00,  0.0000e+00,
        -4.3567e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.8501e-03,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -3.2834e-01,  0.0000e+00, -1.6714e-01,  5.5202e-03,
        -6.3038e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00, -3.7903e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  1.9814e-02,  0.0000e+00,
         3.9675e-06,  0.0000e+00,  2.4098e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -1.3051e-02, -1.1736e-02,  0.0000e+00,  1.0840e-02,
        -1.6080e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -5.0153e-02,  0.0000e+00,  0.0000e+00, -6.0132e-02,  0.0000e+00,
        -3.2078e-04,  0.0000e+00, -4.0123e-04,  0.0000e+00,  0.0000e+00,
        -2.1932e-04,  0.0000e+00,  0.0000e+00, -1.9443e-01,  0.0000e+00,
         0.0000e+00, -3.8776e-02,  9.8007e-03,  1.5938e-03,  0.0000e+00,
         7.4797e-02,  0.0000e+00, -3.2647e-01,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.9615e-03,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -4.1523e-05,  1.2679e-04,
        -5.8467e-03,  0.0000e+00,  0.0000e+00, -1.2231e-01,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  2.0670e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00])
tensor([[ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
        [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
        [-0.0007, -0.0001, -0.0011,  …,  0.0000,  0.0000,  0.0000],
        …,
        [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
        [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
        [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]])
tensor([-1.0580e-03,  0.0000e+00, -9.1295e-03,  …,  0.0000e+00,
        -8.9318e-06,  0.0000e+00])
tensor([[ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
```

```
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        …,
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [-6.5682e-05,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00]])
tensor([ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000, -0.0004])
tensor([[0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         1.7945e-05],
        [0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         3.9333e-04],
        [0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         2.5328e-06],
        …,
        [0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         1.0361e-09],
        [0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         1.0422e-09],
        [0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         9.9648e-10]])
tensor([ 9.5329e-03,  3.8800e-02, -1.0517e-02,  1.8539e-03, -6.1069e-02,
         3.4389e-02,  5.7388e-02,  5.4286e-02, -1.0778e-01,  4.8075e-02,
        -8.3648e-02,  3.9192e-03,  2.6845e-02,  7.8843e-02,  2.3714e-03,
         1.0774e-02, -3.4420e-02, -4.0322e-02, -2.8913e-02, -4.1021e-03,
         8.1626e-06,  3.8486e-06,  3.8474e-06,  3.7647e-06,  3.8175e-06,
         3.7910e-06,  3.7278e-06,  3.8477e-06,  3.6959e-06,  3.7031e-06,
         3.7210e-06,  3.8411e-06,  3.7198e-06,  3.7532e-06,  3.8361e-06,
         3.8471e-06,  3.7726e-06,  3.7062e-06,  3.8302e-06,  3.7005e-06,
         3.7687e-06,  3.8146e-06,  3.7067e-06,  3.7972e-06,  3.6663e-06,
         3.7154e-06,  3.7491e-06,  3.7844e-06,  3.7413e-06,  3.8573e-06,
         3.7035e-06,  3.7778e-06,  3.7108e-06,  3.8576e-06,  3.7212e-06,
         3.7155e-06,  3.7555e-06,  3.7000e-06,  3.8396e-06,  3.7444e-06,
         3.7046e-06,  3.7284e-06,  3.6908e-06,  3.7059e-06,  3.6957e-06,
         3.7203e-06,  3.7078e-06,  3.7189e-06,  3.8334e-06,  3.7008e-06,
         3.7578e-06,  3.7010e-06,  3.8515e-06,  3.7843e-06,  3.7301e-06,
         3.8381e-06,  3.7013e-06,  3.7966e-06,  3.7866e-06,  3.6678e-06,
         3.7956e-06,  3.7863e-06,  3.6748e-06,  3.7114e-06,  3.7645e-06,
         3.6719e-06,  3.7289e-06,  3.6992e-06,  3.7064e-06,  3.8510e-06,
         3.8087e-06,  3.7735e-06,  3.7041e-06,  3.8501e-06,  3.8284e-06,
         3.7053e-06,  3.7081e-06,  3.7831e-06,  3.6435e-06,  3.7198e-06,
         3.7251e-06,  3.8472e-06,  3.7032e-06,  3.7770e-06,  3.7682e-06,
         3.6935e-06,  3.7541e-06,  3.7884e-06,  3.8384e-06,  3.7015e-06,
         3.7331e-06,  3.6963e-06,  3.7344e-06,  3.8183e-06,  3.7376e-06,
         3.8298e-06,  3.7336e-06,  3.8003e-06,  3.6805e-06,  3.8501e-06,
```

```
3.8509e-06,  3.8094e-06,  3.8318e-06,  3.6847e-06,  3.7789e-06,
3.8349e-06,  3.7150e-06,  3.7728e-06,  3.7154e-06,  3.7662e-06,
3.8391e-06,  3.6956e-06,  3.8377e-06,  3.7099e-06,  3.7504e-06,
3.8542e-06,  3.7509e-06,  3.8112e-06,  3.8404e-06,  3.8218e-06,
3.6957e-06,  3.6857e-06,  3.7257e-06,  3.8329e-06,  3.8565e-06,
3.7181e-06,  3.8528e-06,  3.6944e-06,  3.7968e-06,  3.6973e-06,
3.7060e-06,  3.7008e-06,  3.7238e-06,  3.6499e-06,  3.8488e-06,
3.8361e-06,  3.8419e-06,  3.7417e-06,  3.6889e-06,  3.7770e-06,
3.7052e-06,  3.8577e-06,  3.8057e-06,  3.7645e-06,  3.6975e-06,
3.8522e-06,  3.7636e-06,  3.8289e-06,  3.6788e-06,  3.7069e-06,
3.7012e-06,  3.7086e-06,  3.6915e-06,  3.6746e-06,  3.7506e-06,
3.7684e-06,  3.7524e-06,  3.7934e-06,  3.6955e-06,  3.6972e-06,
3.7033e-06,  3.8546e-06,  3.8248e-06,  3.7309e-06,  3.8525e-06,
3.6975e-06,  3.8304e-06,  3.6664e-06,  3.7104e-06,  3.6913e-06,
3.8416e-06,  3.7504e-06,  3.7100e-06,  3.7362e-06,  3.8268e-06,
3.8535e-06,  3.8392e-06,  3.7249e-06,  3.7988e-06,  3.8383e-06,
3.8481e-06,  3.6958e-06,  3.8490e-06,  3.7064e-06,  3.6921e-06,
3.6974e-06,  3.7452e-06,  3.7040e-06,  3.8281e-06,  3.7123e-06,
3.8417e-06,  3.7096e-06,  3.8588e-06,  3.7861e-06,  3.6737e-06,
3.8064e-06,  3.7042e-06,  3.6943e-06,  3.7251e-06,  3.6889e-06,
3.6968e-06,  3.8230e-06,  3.8147e-06,  3.8405e-06,  3.8531e-06,
3.6990e-06,  3.8442e-06,  3.7769e-06,  3.8556e-06,  3.8297e-06,
3.7447e-06,  3.6865e-06,  3.7697e-06,  3.7093e-06,  3.7457e-06,
3.7076e-06,  3.7762e-06,  3.7005e-06,  3.8148e-06,  3.7402e-06,
3.8306e-06,  3.8361e-06,  3.7524e-06,  3.7513e-06,  3.7546e-06,
3.7368e-06,  3.8082e-06,  3.7806e-06,  3.7926e-06,  3.7155e-06,
3.7005e-06,  3.7100e-06,  3.8564e-06,  3.8456e-06,  3.8545e-06,
3.7133e-06,  3.8088e-06,  3.7806e-06,  3.7703e-06,  3.7891e-06,
3.8556e-06,  3.7727e-06,  3.8560e-06,  3.7441e-06,  3.7527e-06,
3.6895e-06,  3.7350e-06,  3.7443e-06,  3.8179e-06,  3.7160e-06,
3.7869e-06,  3.8510e-06,  3.7339e-06,  3.8071e-06,  3.8597e-06,
3.6956e-06,  3.7604e-06,  3.8500e-06,  3.8252e-06,  3.7279e-06,
3.7978e-06,  3.7217e-06,  3.7982e-06,  3.6914e-06,  3.7085e-06,
3.7630e-06,  3.8271e-06,  3.8599e-06,  3.8307e-06,  3.7082e-06,
3.7970e-06,  3.6927e-06,  3.8213e-06,  3.8081e-06,  3.7010e-06,
3.7987e-06,  3.7070e-06,  3.8105e-06,  3.7653e-06,  3.6889e-06,
3.7493e-06,  3.7021e-06,  3.6761e-06,  3.7046e-06,  3.6937e-06,
3.6883e-06,  3.7183e-06,  3.8496e-06,  3.7176e-06,  3.8496e-06,
3.8340e-06,  3.6909e-06,  3.7041e-06,  3.8341e-06,  3.6990e-06,
3.7631e-06,  3.7482e-06,  3.8487e-06,  3.8012e-06,  3.7644e-06,
3.7635e-06,  3.7354e-06,  3.8428e-06,  3.8409e-06,  3.7373e-06,
3.7355e-06,  3.7005e-06,  3.8620e-06,  3.7405e-06,  3.6924e-06,
3.7005e-06,  3.7005e-06,  3.6828e-06,  3.7312e-06,  3.8504e-06,
3.7275e-06,  3.7063e-06,  3.8052e-06,  3.7530e-06,  3.6836e-06,
3.8164e-06,  3.8499e-06,  3.6975e-06,  3.6920e-06,  3.8085e-06,
3.8473e-06,  3.7003e-06,  3.8248e-06,  3.7150e-06,  3.8171e-06,
3.7051e-06,  3.7384e-06,  3.6988e-06,  3.7306e-06,  3.7235e-06,
3.6715e-06,  3.6984e-06,  3.6964e-06,  3.8528e-06,  3.7999e-06,
```

```
3.7466e-06,  3.7851e-06,  3.8119e-06,  3.8477e-06,  3.7456e-06,
3.7483e-06,  3.8322e-06,  3.7184e-06,  3.8018e-06,  3.8276e-06,
3.7045e-06,  3.7674e-06,  3.7023e-06,  3.7065e-06,  3.6957e-06,
3.6764e-06,  3.7241e-06,  3.7619e-06,  3.7589e-06,  3.8562e-06,
3.7216e-06,  3.8458e-06,  3.7028e-06,  3.7082e-06,  3.8290e-06,
3.7430e-06,  3.7882e-06,  3.7073e-06,  3.7080e-06,  3.8549e-06,
3.8158e-06,  3.7084e-06,  3.7361e-06,  3.7311e-06,  3.7555e-06,
3.8123e-06,  3.7697e-06,  3.8289e-06,  3.7923e-06,  3.7970e-06,
3.8446e-06,  3.7108e-06,  3.7052e-06,  3.8508e-06,  3.8473e-06,
3.7214e-06,  3.7734e-06,  3.6580e-06,  3.6896e-06,  3.8440e-06,
3.7085e-06,  3.6908e-06,  3.7332e-06,  3.7032e-06,  3.7834e-06,
3.7343e-06,  3.8367e-06,  3.7813e-06,  3.7180e-06,  3.7094e-06,
3.6911e-06,  3.7639e-06,  3.7151e-06,  3.8177e-06,  3.7647e-06,
3.8336e-06,  3.8464e-06,  3.8573e-06,  3.7866e-06,  3.7525e-06,
3.8406e-06,  3.8214e-06,  3.8577e-06,  3.8418e-06,  3.8303e-06,
3.7956e-06,  3.6972e-06,  3.6934e-06,  3.7635e-06,  3.8097e-06,
3.6948e-06,  3.6875e-06,  3.7967e-06,  3.7009e-06,  3.7963e-06,
3.7489e-06,  3.7797e-06,  3.8630e-06,  3.6989e-06,  3.7197e-06,
3.7588e-06,  3.7593e-06,  3.8601e-06,  3.7750e-06,  3.8160e-06,
3.7371e-06,  3.7869e-06,  3.7840e-06,  3.6760e-06,  3.6916e-06,
3.8594e-06,  3.8120e-06,  3.7730e-06,  3.8639e-06,  3.6690e-06,
3.7362e-06,  3.8581e-06,  3.7018e-06,  3.8172e-06,  3.7333e-06,
3.7017e-06,  3.7547e-06,  3.8549e-06,  3.7010e-06,  3.6959e-06,
3.8009e-06,  3.7182e-06,  3.7497e-06,  3.8079e-06,  3.8093e-06,
3.7792e-06,  3.7454e-06,  3.7515e-06,  3.7615e-06,  3.8548e-06,
3.8513e-06,  3.6901e-06,  3.8043e-06,  3.8589e-06,  3.7096e-06,
3.6966e-06,  3.8522e-06,  3.8573e-06,  3.8098e-06,  3.7604e-06,
3.8638e-06,  3.6891e-06,  3.7161e-06,  3.6978e-06,  3.7841e-06,
3.7450e-06,  3.7129e-06,  3.8058e-06,  3.7905e-06,  3.7957e-06,
3.8571e-06,  3.7239e-06,  3.7479e-06,  3.7830e-06,  3.7808e-06,
3.7759e-06,  3.7990e-06,  3.8465e-06,  3.7066e-06,  3.8523e-06,
3.8343e-06,  3.7750e-06,  3.7742e-06,  3.7867e-06,  3.7092e-06,
3.7027e-06,  3.6944e-06,  3.8674e-06,  3.8569e-06,  3.7095e-06,
3.7101e-06,  3.8051e-06,  3.7905e-06,  3.8342e-06,  3.7351e-06,
3.7570e-06,  3.8183e-06,  3.8439e-06,  3.7256e-06,  3.6690e-06,
3.8338e-06,  3.8419e-06,  3.8146e-06,  3.7640e-06,  3.8524e-06,
3.7232e-06,  3.8450e-06,  3.7015e-06,  3.8109e-06,  3.7345e-06,
3.7265e-06,  3.8269e-06,  3.7500e-06,  3.8290e-06,  3.7469e-06,
3.7733e-06,  3.7603e-06,  3.8509e-06,  3.7016e-06,  3.8371e-06,
3.7125e-06,  3.8420e-06,  3.7691e-06,  3.7043e-06,  3.7245e-06,
3.7005e-06,  3.7015e-06,  3.8383e-06,  3.8430e-06,  3.7847e-06,
3.8468e-06,  3.7251e-06,  3.8355e-06,  3.7571e-06,  3.8134e-06,
3.8059e-06,  3.8040e-06,  3.8454e-06,  3.8249e-06,  3.7269e-06,
3.6937e-06,  3.6742e-06,  3.8000e-06,  3.8554e-06,  3.7837e-06,
3.8544e-06,  3.7104e-06,  3.6640e-06,  3.7464e-06,  3.7984e-06,
3.8567e-06,  3.8512e-06,  3.8055e-06,  3.7510e-06,  3.8509e-06,
3.7667e-06,  3.8535e-06,  3.6909e-06,  3.6999e-06,  3.7865e-06,
3.8136e-06,  3.7059e-06,  3.7854e-06,  3.8508e-06,  3.7929e-06,
```

```
3.7024e-06,  3.7956e-06,  3.8071e-06,  3.6736e-06,  3.8601e-06,
3.6774e-06,  3.8352e-06,  3.7093e-06,  3.7781e-06,  3.6991e-06,
3.7413e-06,  3.8600e-06,  3.8499e-06,  3.7435e-06,  3.7380e-06,
3.7856e-06,  3.8273e-06,  3.7096e-06,  3.7035e-06,  3.7665e-06,
3.7381e-06,  3.7851e-06,  3.7554e-06,  3.7595e-06,  3.7498e-06,
3.8651e-06,  3.7765e-06,  3.7858e-06,  3.8043e-06,  3.8410e-06,
3.8110e-06,  3.7065e-06,  3.7415e-06,  3.8583e-06,  3.7280e-06,
3.7494e-06,  3.7285e-06,  3.8557e-06,  3.7763e-06,  3.8226e-06,
3.6956e-06,  3.8208e-06,  3.8439e-06,  3.8296e-06,  3.7216e-06,
3.7937e-06,  3.8133e-06,  3.7013e-06,  3.8428e-06,  3.7146e-06,
3.6126e-06,  3.7002e-06,  3.8258e-06,  3.6986e-06,  3.8030e-06,
3.6800e-06,  3.7704e-06,  3.7777e-06,  3.8274e-06,  3.6952e-06,
3.6980e-06,  3.7881e-06,  3.8425e-06,  3.8203e-06,  3.7974e-06,
3.7021e-06,  3.7563e-06,  3.7160e-06,  3.6264e-06,  3.7561e-06,
3.7742e-06,  3.7124e-06,  3.7571e-06,  3.8480e-06,  3.8388e-06,
3.7064e-06,  3.8457e-06,  3.6921e-06,  3.7011e-06,  3.8375e-06,
3.8065e-06,  3.7546e-06,  3.7478e-06,  3.7584e-06,  3.7832e-06,
3.8313e-06,  3.7629e-06,  3.7392e-06,  3.8136e-06,  3.7415e-06,
3.6921e-06,  3.8570e-06,  3.6929e-06,  3.8429e-06,  3.8042e-06,
3.8569e-06,  3.7099e-06,  3.8017e-06,  3.6636e-06,  3.7579e-06,
3.8656e-06,  3.7248e-06,  3.8641e-06,  3.8539e-06,  3.8543e-06,
3.6718e-06,  3.7396e-06,  3.6997e-06,  3.8267e-06,  3.7496e-06,
3.6964e-06,  3.7023e-06,  3.7140e-06,  3.7265e-06,  3.8520e-06,
3.8135e-06,  3.8200e-06,  3.6948e-06,  3.7642e-06,  3.7039e-06,
3.7100e-06,  3.8492e-06,  3.8402e-06,  3.8494e-06,  3.7368e-06,
3.8305e-06,  3.7674e-06,  3.7929e-06,  3.7802e-06,  3.7825e-06,
3.8383e-06,  3.7056e-06,  3.8167e-06,  3.8003e-06,  3.8576e-06,
3.7429e-06,  3.7484e-06,  3.6878e-06,  3.7442e-06,  3.6855e-06,
3.7017e-06,  3.8121e-06,  3.7741e-06,  3.7256e-06,  3.7462e-06,
3.7006e-06,  3.7034e-06,  3.8496e-06,  3.8373e-06,  3.8581e-06,
3.7102e-06,  3.6499e-06,  3.7293e-06,  3.8500e-06,  3.6953e-06,
3.8604e-06,  3.7951e-06,  3.6920e-06,  3.7189e-06,  3.6622e-06,
3.7071e-06,  3.7495e-06,  3.7077e-06,  3.7871e-06,  3.7040e-06,
3.7487e-06,  3.8530e-06,  3.7366e-06,  3.6949e-06,  3.7320e-06,
3.7792e-06,  3.6984e-06,  3.6693e-06,  3.8456e-06,  3.7646e-06,
3.7365e-06,  3.7630e-06,  3.8331e-06,  3.6551e-06,  3.7055e-06,
3.7444e-06,  3.8335e-06,  3.7826e-06,  3.7021e-06,  3.7821e-06,
3.5748e-06,  3.6895e-06,  3.7944e-06,  3.7323e-06,  3.8584e-06,
3.8631e-06,  3.6815e-06,  3.8077e-06,  3.8351e-06,  3.7316e-06,
3.8559e-06,  3.8525e-06,  3.7363e-06,  3.7898e-06,  3.8338e-06,
3.8561e-06,  3.7363e-06,  3.6854e-06,  3.8226e-06,  3.7531e-06,
3.8187e-06,  3.7727e-06,  3.8471e-06,  3.7880e-06,  3.8260e-06,
3.6949e-06,  3.8029e-06,  3.7872e-06,  3.7100e-06,  3.6897e-06,
3.8478e-06,  3.8057e-06,  3.7748e-06,  3.7760e-06,  3.8182e-06,
3.6930e-06,  3.8300e-06,  3.8510e-06,  3.8549e-06,  3.7680e-06,
3.7512e-06,  3.7300e-06,  3.8290e-06,  3.7086e-06,  3.8139e-06,
3.8283e-06,  3.8525e-06,  3.6939e-06,  3.6820e-06,  3.7309e-06,
3.7914e-06,  3.7086e-06,  3.8262e-06,  3.7669e-06,  3.6906e-06,
```

```
          3.8505e-06,  3.8454e-06,  3.8076e-06,  3.7518e-06,  3.6897e-06,
          3.7077e-06,  3.6866e-06,  3.7563e-06,  3.7187e-06,  3.7924e-06,
          3.6977e-06,  3.7895e-06,  3.8326e-06,  3.8300e-06,  3.8282e-06,
          3.8255e-06,  3.6775e-06,  3.8018e-06,  3.6753e-06,  3.7383e-06,
          3.6397e-06,  3.6879e-06,  3.8533e-06,  3.5994e-06,  3.7918e-06,
          3.7591e-06,  3.8567e-06,  3.6998e-06,  3.6931e-06,  3.8442e-06,
          3.8475e-06,  3.7017e-06,  3.7531e-06,  3.8530e-06,  3.7004e-06,
          3.7567e-06,  3.7686e-06,  3.7567e-06,  3.7936e-06,  3.6924e-06,
          3.7528e-06,  3.8013e-06,  3.7569e-06,  3.7288e-06,  3.8614e-06,
          3.8119e-06,  3.7257e-06,  3.6961e-06,  3.7756e-06,  3.6906e-06,
          3.6630e-06,  3.7747e-06,  3.8212e-06,  3.8361e-06,  3.8587e-06,
          3.8055e-06,  3.7826e-06,  3.6960e-06,  3.7124e-06,  3.7207e-06,
          3.7387e-06,  3.7089e-06,  3.8474e-06,  3.7058e-06,  3.6288e-06,
          3.7233e-06,  3.7816e-06,  3.7369e-06,  3.6976e-06,  3.6943e-06,
          3.7053e-06,  3.7266e-06,  3.6671e-06,  3.8551e-06,  3.8529e-06,
          3.6788e-06,  3.7012e-06,  3.6887e-06,  3.7049e-06,  3.8086e-06,
          3.7242e-06,  3.6603e-06,  3.7531e-06,  3.8326e-06,  3.7742e-06,
          3.8580e-06,  3.7017e-06,  3.8497e-06,  3.6788e-06,  3.6871e-06,
          3.6935e-06,  3.7041e-06,  3.7021e-06,  3.7118e-06,  3.6894e-06,
          3.6896e-06,  3.7814e-06,  3.8131e-06,  3.8019e-06,  3.7618e-06,
          3.7055e-06,  3.6556e-06,  3.7480e-06,  3.7573e-06,  3.8005e-06,
          3.7712e-06,  3.8465e-06,  3.8365e-06,  3.8299e-06,  3.8668e-06,
          3.6913e-06,  3.7095e-06,  3.7088e-06,  3.6991e-06,  3.6922e-06,
          3.7242e-06,  3.7936e-06,  3.7534e-06,  3.7029e-06,  3.7476e-06,
          3.7559e-06,  3.6986e-06,  3.8007e-06,  3.6921e-06,  3.6991e-06,
          3.7025e-06,  3.6744e-06,  3.8349e-06,  3.8436e-06,  3.7807e-06,
          3.8544e-06,  3.7835e-06,  3.7766e-06,  3.7541e-06,  3.7799e-06,
          3.7132e-06,  3.7218e-06,  3.6953e-06,  3.6883e-06,  3.7958e-06,
          3.8264e-06,  3.7620e-06,  3.8254e-06,  3.8555e-06,  3.6900e-06,
          3.8476e-06,  3.8544e-06,  3.6957e-06,  3.8261e-06,  3.7410e-06,
          3.6939e-06,  3.8226e-06,  3.7786e-06,  3.7592e-06,  3.8646e-06,
          3.8519e-06,  3.6991e-06,  3.7047e-06,  3.7507e-06,  3.5290e-06])
end of p.grad

False
Epoch 3 finished
Epoch [3/10], Loss: 1.5295
************************************************************

Learning rate: 0.001

another way to print learning rate:
tensor([[[[-6.8952e-05, -1.0099e-05,  8.5358e-05,  …, -4.4818e-05,
           -6.6466e-04, -3.6443e-04],
          [ 2.5587e-05,  2.4360e-05,  2.5006e-04,  …,  4.9234e-04,
            5.1061e-04,  5.0107e-04],
          [-1.4571e-04, -3.1204e-04, -1.6085e-04,  …,  6.6886e-04,
            1.2300e-03,  1.3962e-03],
```

```
 …,
  [ 6.8418e-04,  5.1442e-04,  1.0747e-03,  …,  1.8590e-03,
    2.2210e-03,  2.6742e-03],
  [ 1.7481e-03,  1.1456e-03,  1.4030e-03,  …,  2.2086e-03,
    2.5143e-03,  2.7480e-03],
  [ 2.6835e-03,  2.4903e-03,  2.4505e-03,  …,  3.0639e-03,
    2.9838e-03,  2.6167e-03]],

 [[-6.5574e-04, -5.1026e-04, -3.4469e-04,  …, -2.9748e-04,
   -8.9985e-04, -6.1187e-04],
  [-5.3844e-04, -4.3647e-04, -1.9776e-04,  …,  1.4779e-04,
    3.3607e-04,  2.9697e-04],
  [-5.6702e-04, -6.4159e-04, -5.4784e-04,  …,  5.1406e-04,
    1.0046e-03,  1.0094e-03],
  …,
  [ 4.4069e-04,  9.3366e-05,  6.0325e-04,  …,  1.1717e-03,
    1.3109e-03,  1.6391e-03],
  [ 1.5269e-03,  8.2042e-04,  1.1060e-03,  …,  1.5034e-03,
    1.6898e-03,  1.9869e-03],
  [ 2.2155e-03,  1.9154e-03,  1.8951e-03,  …,  2.1012e-03,
    2.0368e-03,  1.8580e-03]],

 [[-5.4521e-04, -3.4013e-04, -1.5139e-04,  …,  4.6837e-04,
    3.0648e-05,  4.4929e-04],
  [-4.7569e-04, -3.3492e-04, -2.2087e-04,  …,  6.7221e-04,
    9.6198e-04,  1.1277e-03],
  [-9.4299e-04, -8.9372e-04, -8.5466e-04,  …,  8.3716e-04,
    1.4348e-03,  1.6079e-03],
  …,
  [ 6.6859e-05, -2.1792e-04,  6.3421e-05,  …, -3.1832e-04,
   -1.0170e-04,  3.6320e-04],
  [ 1.2538e-03,  5.9920e-04,  7.1005e-04,  …, -3.4907e-05,
    3.5198e-04,  7.7422e-04],
  [ 2.0752e-03,  1.6513e-03,  1.4858e-03,  …,  5.1628e-04,
    5.1459e-04,  5.0516e-04]]],


[[[ 8.8199e-03,  8.5400e-03,  8.4086e-03,  …,  6.1273e-03,
    5.7594e-03,  5.8849e-03],
  [ 8.3873e-03,  9.0286e-03,  9.5251e-03,  …,  4.4580e-03,
    3.9585e-03,  5.9147e-03],
  [ 9.9412e-03,  9.6087e-03,  8.8481e-03,  …,  4.6000e-03,
    4.1465e-03,  5.3060e-03],
  …,
  [ 7.9035e-03,  8.0988e-03,  6.8561e-03,  …,  4.3397e-03,
    5.2021e-03,  6.6997e-03],
  [ 5.8084e-03,  6.1470e-03,  5.5342e-03,  …,  4.4502e-03,
    6.1940e-03,  6.9746e-03],
```

```
  [ 5.9338e-03,  6.0781e-03,  5.9938e-03,  …,  4.1080e-03,
    6.0618e-03,  6.8412e-03]],

 [[ 2.8948e-03,  2.8312e-03,  2.4685e-03,  …,  7.3210e-04,
    5.5139e-04,  8.5126e-04],
  [ 2.9131e-03,  3.6991e-03,  3.9884e-03,  …, -1.1328e-03,
   -1.0309e-03,  8.3522e-04],
  [ 5.1631e-03,  4.8964e-03,  3.6859e-03,  …, -5.8764e-04,
   -8.3593e-04, -3.4165e-05],
  …,
  [ 3.7383e-03,  4.1527e-03,  2.9187e-03,  …, -1.2104e-03,
   -5.1106e-04,  5.6395e-04],
  [ 2.0039e-03,  2.3172e-03,  1.5426e-03,  …, -1.5529e-03,
    3.6858e-04,  8.4709e-04],
  [ 2.1993e-03,  2.2097e-03,  2.1432e-03,  …, -1.8420e-03,
    1.3600e-04,  1.0672e-03]],

 [[ 5.1073e-03,  4.5468e-03,  3.9957e-03,  …,  3.8693e-03,
    3.8856e-03,  4.2008e-03],
  [ 4.8886e-03,  5.4889e-03,  5.9070e-03,  …,  2.2406e-03,
    2.0802e-03,  3.8381e-03],
  [ 7.2527e-03,  6.9005e-03,  6.0698e-03,  …,  2.8776e-03,
    2.6153e-03,  3.4456e-03],
  …,
  [ 6.4898e-03,  6.9168e-03,  5.9660e-03,  …,  1.5850e-03,
    2.1280e-03,  3.3105e-03],
  [ 5.0426e-03,  5.4101e-03,  4.4862e-03,  …,  1.2721e-03,
    3.0690e-03,  3.5152e-03],
  [ 5.0853e-03,  5.4315e-03,  5.3199e-03,  …,  8.8064e-04,
    2.9932e-03,  3.7473e-03]]],


[[[-3.4763e-04, -9.7401e-04, -1.3305e-03,  …, -2.0253e-03,
   -2.2831e-03, -2.2987e-03],
  [-5.4350e-04, -1.3598e-03, -1.5494e-03,  …, -2.4660e-03,
   -2.4863e-03, -2.3647e-03],
  [-5.9636e-04, -1.4004e-03, -1.4740e-03,  …, -2.5770e-03,
   -2.8651e-03, -2.6134e-03],
  …,
  [-1.4014e-03, -1.4621e-03, -1.2752e-03,  …, -2.8424e-03,
   -2.7763e-03, -2.0692e-03],
  [-1.3448e-03, -1.5789e-03, -1.1376e-03,  …, -2.9680e-03,
   -2.7826e-03, -2.6139e-03],
  [-7.6453e-04, -1.3095e-03, -1.2092e-03,  …, -3.4529e-03,
   -2.9217e-03, -2.5985e-03]],

 [[ 8.3508e-04,  1.2879e-04, -2.4705e-04,  …, -5.1048e-04,
   -1.0295e-03, -1.1281e-03],
```

```
  [ 5.6894e-04, -2.4585e-04, -4.1512e-04,  …, -9.3582e-04,
   -1.0092e-03, -9.5111e-04],
  [ 5.2823e-04, -2.4069e-04, -3.0816e-04,  …, -1.0725e-03,
   -1.4084e-03, -1.0134e-03],
  …,
  [-4.4096e-04, -6.0600e-04, -3.9452e-04,  …, -1.5828e-03,
   -1.6364e-03, -1.3075e-03],
  [-4.1018e-04, -7.2616e-04, -2.0157e-04,  …, -1.7220e-03,
   -1.6469e-03, -1.7940e-03],
  [ 2.2608e-04, -2.3521e-04,  4.7357e-05,  …, -2.3346e-03,
   -1.8665e-03, -1.7224e-03]],

 [[ 1.3711e-03,  7.2813e-04,  4.8298e-04,  …, -3.3127e-05,
   -4.9202e-04, -6.0222e-04],
  [ 1.2117e-03,  5.8355e-04,  4.4786e-04,  …, -3.2191e-04,
   -2.2958e-04, -1.9267e-04],
  [ 1.2715e-03,  8.2570e-04,  6.6538e-04,  …, -4.9022e-04,
   -5.9469e-04, -1.3580e-04],
  …,
  [ 4.9489e-04,  3.4631e-04,  3.2990e-04,  …, -5.6211e-04,
   -7.0963e-04, -6.9953e-04],
  [ 4.5193e-04,  3.5332e-05,  3.8842e-04,  …, -9.9208e-04,
   -9.3685e-04, -1.2581e-03],
  [ 1.1184e-03,  6.3610e-04,  6.6483e-04,  …, -1.6274e-03,
   -1.1654e-03, -1.1167e-03]]],


…,


[[[-7.8009e-04, -1.8545e-03, -2.8059e-03,  …, -3.3992e-03,
   -2.6166e-03, -1.6698e-03],
  [-9.8853e-04, -2.1096e-03, -3.1849e-03,  …, -2.6121e-03,
   -2.4158e-03, -2.1844e-03],
  [-1.3953e-03, -2.7562e-03, -3.2507e-03,  …, -2.4972e-03,
   -2.4723e-03, -2.2032e-03],
  …,
  [-6.4566e-04, -1.1440e-03, -1.5466e-03,  …, -1.4399e-03,
   -7.5114e-04, -3.5465e-04],
  [-3.2880e-04, -7.5355e-04, -1.5480e-03,  …, -1.2950e-03,
   -1.6764e-04, -5.0099e-04],
  [-1.4207e-04, -7.0702e-04, -1.1913e-03,  …, -9.0377e-04,
    4.5113e-05,  2.5108e-04]],

 [[-2.1531e-03, -3.1135e-03, -3.9266e-03,  …, -4.5691e-03,
   -3.7551e-03, -2.6078e-03],
  [-2.3092e-03, -3.6030e-03, -4.6062e-03,  …, -3.5775e-03,
   -3.1911e-03, -2.7688e-03],
```

```
  [-2.6349e-03, -4.0941e-03, -4.5222e-03,  …, -3.3828e-03,
   -3.3245e-03, -2.9959e-03],
  …,
  [-1.4648e-03, -2.0850e-03, -2.4402e-03,  …, -1.5259e-03,
   -1.0270e-03, -8.5338e-04],
  [-1.2681e-03, -1.8786e-03, -2.5258e-03,  …, -1.2835e-03,
   -3.7838e-04, -9.8989e-04],
  [-8.7345e-04, -1.5417e-03, -2.0022e-03,  …, -5.2879e-04,
    1.1053e-04,  1.8409e-05]],

 [[-3.6566e-03, -4.7141e-03, -5.3255e-03,  …, -4.8297e-03,
   -4.3528e-03, -3.1062e-03],
  [-3.8712e-03, -5.2756e-03, -5.9190e-03,  …, -4.0912e-03,
   -3.9961e-03, -3.3829e-03],
  [-4.1305e-03, -5.7493e-03, -5.8500e-03,  …, -4.1991e-03,
   -4.2678e-03, -3.6567e-03],
  …,
  [-2.1519e-03, -2.7216e-03, -3.0064e-03,  …, -2.3226e-03,
   -1.7073e-03, -1.3115e-03],
  [-1.7358e-03, -2.4078e-03, -2.9869e-03,  …, -1.8867e-03,
   -9.8167e-04, -1.5306e-03],
  [-1.1545e-03, -1.9130e-03, -2.1689e-03,  …, -7.3544e-04,
   -1.7548e-04, -4.3710e-04]]],


[[[ 1.1171e-04, -5.2474e-04, -9.9409e-04,  …, -3.9492e-04,
   -1.3536e-04, -3.0152e-04],
  [-1.4631e-05, -4.7999e-04, -9.5057e-04,  …, -8.7977e-05,
   -5.3257e-05, -3.3695e-04],
  [ 4.5778e-04,  4.9235e-05, -3.2851e-04,  …,  2.2464e-04,
    1.7732e-04,  2.0553e-04],
  …,
  [ 1.1059e-04, -6.5021e-05,  3.6627e-04,  …,  6.5730e-04,
    5.8936e-04,  9.6311e-04],
  [ 1.4284e-04,  2.2916e-04,  1.9394e-04,  …,  5.4048e-04,
    3.3087e-04,  3.8528e-04],
  [-6.7229e-05, -3.5482e-04, -3.4789e-04,  …,  3.4454e-04,
   -3.1972e-06,  5.7599e-05]],

 [[ 8.2419e-04,  1.9459e-04, -2.9440e-04,  …,  4.9226e-04,
    8.0128e-04,  6.3734e-04],
  [ 7.2782e-04,  2.0245e-04, -1.2227e-04,  …,  8.7676e-04,
    9.0768e-04,  6.1069e-04],
  [ 1.3008e-03,  7.8792e-04,  5.5761e-04,  …,  1.2045e-03,
    1.0673e-03,  1.0236e-03],
  …,
  [ 1.7312e-03,  1.4551e-03,  1.8544e-03,  …,  2.0469e-03,
    1.8407e-03,  2.1298e-03],
```

```
  [ 1.7963e-03,  1.7110e-03,  1.8268e-03,  …,  2.0139e-03,
    1.7466e-03,  1.7208e-03],
  [ 2.0834e-03,  1.7918e-03,  1.7847e-03,  …,  2.1499e-03,
    1.8343e-03,  1.8164e-03]],

 [[-1.2969e-03, -1.9424e-03, -2.2018e-03,  …, -1.0646e-03,
   -6.9597e-04, -4.5165e-04],
  [-1.3197e-03, -1.8428e-03, -1.8496e-03,  …, -6.9643e-04,
   -6.4555e-04, -4.8169e-04],
  [-6.6286e-04, -1.1574e-03, -9.6422e-04,  …, -2.4498e-04,
   -3.5154e-04,  2.4419e-05],
  …,
  [ 1.1985e-03,  7.6027e-04,  1.0517e-03,  …,  8.7560e-04,
    7.7897e-04,  1.1401e-03],
  [ 1.3923e-03,  1.2101e-03,  1.2223e-03,  …,  1.0179e-03,
    6.9095e-04,  7.8485e-04],
  [ 1.9244e-03,  1.5950e-03,  1.7108e-03,  …,  1.5424e-03,
    1.0709e-03,  1.0994e-03]]],


[[[-3.5772e-03, -2.6050e-03, -2.5447e-03,  …, -3.9065e-03,
   -2.7909e-03, -1.5274e-03],
  [-3.3945e-03, -3.6992e-03, -3.5644e-03,  …, -2.4187e-03,
   -2.8181e-03, -2.0925e-03],
  [-3.9533e-03, -4.5031e-03, -4.5158e-03,  …, -1.8658e-03,
   -2.1305e-03, -2.3160e-03],
  …,
  [-1.1974e-03, -2.8027e-03, -4.3611e-03,  …, -4.4325e-03,
   -3.5619e-03, -3.9130e-03],
  [-1.3691e-03, -3.0645e-03, -4.8727e-03,  …, -2.9903e-03,
   -2.0892e-03, -3.0542e-03],
  [-2.1867e-03, -5.2382e-03, -6.1568e-03,  …, -2.0966e-03,
   -1.1505e-03, -6.5527e-04]],

 [[-4.3764e-03, -3.8684e-03, -3.5064e-03,  …, -5.5493e-03,
   -4.0452e-03, -2.1518e-03],
  [-3.8824e-03, -4.5246e-03, -4.4830e-03,  …, -4.2914e-03,
   -4.4058e-03, -2.8658e-03],
  [-4.2930e-03, -4.8262e-03, -4.7852e-03,  …, -3.6233e-03,
   -3.3921e-03, -2.7701e-03],
  …,
  [-4.7978e-04, -2.2908e-03, -3.6543e-03,  …, -4.2941e-03,
   -3.4815e-03, -4.2468e-03],
  [-7.1146e-04, -2.5923e-03, -4.4548e-03,  …, -2.5906e-03,
   -1.7297e-03, -3.3965e-03],
  [-1.1716e-03, -4.4857e-03, -5.4666e-03,  …, -1.2154e-03,
   -4.7243e-04, -5.4633e-04]],
```

```
          [[-6.0939e-03, -6.3684e-03, -6.6743e-03,  …, -7.5694e-03,
            -5.7143e-03, -3.0118e-03],
           [-5.4616e-03, -6.7594e-03, -7.2831e-03,  …, -6.7518e-03,
            -6.4335e-03, -4.3234e-03],
           [-6.0627e-03, -7.0977e-03, -7.5074e-03,  …, -5.8760e-03,
            -5.3832e-03, -4.4731e-03],
           …,
           [-1.0278e-03, -2.8471e-03, -4.3914e-03,  …, -5.6888e-03,
            -5.0485e-03, -6.2942e-03],
           [-1.2491e-03, -3.2124e-03, -5.4212e-03,  …, -4.6228e-03,
            -3.9268e-03, -5.7992e-03],
           [-1.7617e-03, -4.7753e-03, -5.8768e-03,  …, -3.2853e-03,
            -2.6229e-03, -2.8124e-03]]])
tensor([ 9.3132e-10,  6.1686e-08, -1.0128e-08,  1.6298e-09,  1.5716e-08,
        -5.4133e-08, -3.4226e-08, -3.1316e-08,  3.7253e-09, -1.1350e-08,
        -1.0943e-08,  8.1491e-10,  1.6298e-09,  3.4925e-10,  1.3737e-08,
        -2.5611e-09,  1.9209e-09,  9.0804e-09,  8.9640e-09,  3.4925e-09,
        -2.6776e-09, -1.2224e-08, -2.6776e-09,  9.7789e-09,  4.7730e-09,
        -4.6566e-10, -8.8476e-09, -1.1642e-10,  3.3760e-09, -2.9038e-08,
        -8.1491e-10,  2.4098e-08,  5.0059e-09,  3.2305e-09,  2.9686e-09,
        -2.5611e-09,  2.0955e-09,  3.2596e-09, -5.5879e-09,  1.8626e-09,
         4.7730e-09, -1.8917e-10,  4.0745e-09,  1.3970e-09,  6.4829e-09,
        -8.0327e-09,  7.5233e-09,  6.0536e-09,  2.3283e-10,  1.7462e-09,
         5.8208e-10,  3.8504e-08,  1.1642e-09, -1.5832e-08,  5.8571e-09,
         8.9640e-09,  4.4238e-08, -3.8417e-09, -2.8522e-09, -4.6566e-09,
        -3.9581e-09, -2.4156e-09, -1.4320e-08,  3.0850e-09,  3.4925e-09,
         1.4203e-08, -2.2817e-08,  2.5611e-09,  1.6298e-09,  3.4925e-09,
        -1.1642e-10, -1.3897e-09,  1.7346e-08, -7.6834e-09, -1.8626e-09,
        -4.2201e-09, -5.8208e-11,  7.4506e-09, -6.1700e-09, -2.7940e-09,
        -4.5635e-08,  2.1071e-08,  4.0745e-09,  3.0035e-08,  6.1700e-09,
        -1.2107e-08, -3.7253e-09,  1.3504e-08, -5.8208e-10,  1.8626e-09,
        -1.8335e-09, -1.7462e-09,  3.1403e-08, -1.3970e-09, -7.6834e-09,
        -4.0745e-09])
tensor([-0.0044, -0.0122,  0.0038,  0.0040,  0.0122, -0.0174,  0.0062, -0.0222,
         0.0064,  0.0106,  0.0084,  0.0036, -0.0087, -0.0038,  0.0026,  0.0036,
         0.0064, -0.0015, -0.0046, -0.0011,  0.0077,  0.0055,  0.0170,  0.0068,
         0.0109,  0.0027,  0.0014,  0.0057, -0.0009, -0.0146,  0.0091, -0.0051,
         0.0084,  0.0054,  0.0032,  0.0088,  0.0011, -0.0073,  0.0071, -0.0055,
         0.0074,  0.0062,  0.0045,  0.0075,  0.0086, -0.0020,  0.0042,  0.0039,
        -0.0036, -0.0012, -0.0043,  0.0066,  0.0057, -0.0016,  0.0009,  0.0204,
        -0.0428, -0.0002,  0.0062,  0.0043, -0.0029, -0.0028,  0.0016, -0.0007,
        -0.0028,  0.0054,  0.0022,  0.0077,  0.0056,  0.0066,  0.0029,  0.0042,
        -0.0025, -0.0029, -0.0031,  0.0057,  0.0006, -0.0039,  0.0003, -0.0016,
        -0.0464,  0.0038,  0.0017,  0.0014, -0.0218,  0.0039,  0.0036,  0.0124,
         0.0001, -0.0048, -0.0011, -0.0116, -0.0155, -0.0017,  0.0001, -0.0017])
tensor([-6.4814e-03, -6.8357e-04,  2.0268e-03,  5.0587e-03, -2.4970e-04,
         3.5653e-03,  6.9965e-03,  2.6482e-03,  5.5728e-03, -1.2225e-03,
        -3.7390e-03,  1.9706e-03, -5.5492e-03, -3.4345e-03,  6.4307e-03,
```

```
          1.9503e-03, -2.9832e-03, -4.9322e-03, -9.7090e-04, -1.1109e-03,
          5.9692e-03, -4.6670e-03, -7.0652e-03,  5.0788e-03, -4.0200e-03,
          8.2228e-04,  3.0694e-03,  3.1022e-03, -2.2962e-03,  5.2609e-03,
          3.8138e-04, -5.8232e-03,  5.2300e-03,  2.6189e-03,  3.1275e-04,
          4.1195e-03,  2.1479e-03, -7.1403e-03,  2.7633e-03, -6.5802e-03,
         -3.8033e-03,  3.0373e-03,  7.6041e-04,  3.0697e-03,  1.2356e-03,
          3.2385e-03,  1.4834e-03,  3.7474e-03, -2.8403e-03,  1.2215e-03,
         -6.5329e-03, -2.5285e-03,  4.0940e-03, -3.7663e-03,  8.9425e-04,
         -2.5453e-03, -1.0893e-02,  3.8840e-04,  4.1852e-03,  1.8381e-03,
         -8.6252e-03,  3.5750e-04, -1.9892e-03,  1.2988e-03, -7.2904e-03,
          7.2425e-03,  5.4845e-03, -1.6879e-03,  6.5876e-04,  5.2734e-03,
          4.0472e-03,  4.0497e-03, -5.0983e-03,  2.6191e-04, -4.3786e-03,
         -1.8286e-03,  4.1546e-03, -2.6905e-03, -3.9609e-03, -3.7105e-03,
         -7.3701e-03,  9.4198e-03,  3.4214e-03,  3.3393e-03, -4.7346e-03,
          1.0327e-03,  7.5551e-03,  3.7264e-03,  8.2794e-05, -6.6668e-03,
         -7.9999e-03,  5.8505e-03, -7.0317e-03,  1.9476e-03, -1.5116e-03,
         -4.2775e-03])
tensor([[[[ 5.4016e-04,  5.5252e-04,  1.1276e-04,  3.7081e-04,  6.9104e-04],
           [ 6.7952e-04,  3.6284e-04,  1.8487e-04,  1.7955e-04,  2.4341e-04],
           [ 8.1862e-04,  8.0066e-04,  2.8656e-04,  2.0879e-04,  1.8438e-04],
           [ 1.0720e-03,  1.1247e-03,  1.1140e-03,  7.6435e-04,  9.4510e-04],
           [ 8.4048e-04,  4.1466e-04,  3.2375e-04,  6.3639e-04,  9.3838e-04]],

          [[-9.3059e-04, -9.6092e-04, -8.4443e-04, -3.0273e-04, -1.4782e-03],
           [-9.6070e-04, -4.2754e-04, -1.4273e-03, -8.8535e-04, -1.2126e-03],
           [-1.1880e-03, -2.8185e-04, -1.1234e-03, -9.6440e-05, -8.5599e-05],
           [-5.8904e-04, -2.2346e-04, -1.2892e-04,  6.1568e-04,  3.8989e-04],
           [ 4.4353e-05, -8.5416e-05,  1.5700e-04,  5.5251e-04,  4.2116e-04]],

          [[-1.5911e-04, -3.5026e-04, -6.9331e-04, -6.0823e-04, -3.4492e-04],
           [-4.6874e-04, -4.4045e-04, -9.9652e-04, -1.2706e-03, -9.4216e-04],
           [-6.8789e-04, -9.2177e-04, -1.3091e-03, -1.2948e-03, -1.2239e-03],
           [-7.8200e-04, -8.6540e-04, -9.4422e-04, -9.0170e-04, -7.9498e-04],
           [-3.7816e-04, -2.4736e-04, -7.9109e-04, -8.8338e-04, -7.2642e-04]],

          …,

          [[ 9.1017e-04,  8.6810e-04,  6.2096e-04,  4.0015e-04,  4.9622e-04],
           [ 6.8432e-04,  7.5204e-04,  5.7959e-04,  3.3545e-04,  3.4929e-04],
           [ 5.0070e-04,  6.0460e-04,  5.6217e-04,  2.3579e-04,  3.2710e-04],
           [ 9.0371e-04,  8.8796e-04,  8.8537e-04,  6.0906e-04,  7.4076e-04],
           [ 9.9402e-04,  7.1847e-04,  5.6056e-04,  4.1744e-04,  4.9255e-04]],

          [[ 8.9364e-04,  8.5801e-04,  5.7848e-04,  5.6639e-04,  6.1689e-04],
           [ 8.5938e-04,  8.0958e-04,  6.1335e-04,  5.1791e-04,  6.0508e-04],
           [ 7.7549e-04,  8.4844e-04,  6.8666e-04,  4.1143e-04,  5.4115e-04],
           [ 1.0971e-03,  1.2385e-03,  1.1074e-03,  7.7587e-04,  7.9564e-04],
           [ 1.1738e-03,  1.0336e-03,  8.8418e-04,  7.3284e-04,  7.0997e-04]],
```

```
 [[ 9.1756e-04,  6.9408e-04,  1.7773e-04, -4.3088e-05,  2.0775e-04],
  [ 5.2110e-04,  6.1130e-04,  1.6506e-04, -3.7556e-04, -3.8805e-05],
  [ 4.0903e-04,  4.6951e-04,  2.4793e-04, -3.5054e-04, -1.0128e-04],
  [ 9.7067e-04,  8.5350e-04,  5.8206e-04,  1.9680e-04,  5.4079e-04],
  [ 1.1411e-03,  8.4731e-04,  1.2967e-04,  1.7105e-04,  6.7497e-05]]],


[[[-1.2876e-04,  8.8706e-05,  3.5063e-04, -3.7332e-05,  3.0804e-04],
  [ 2.3868e-05,  1.4214e-04,  1.0848e-04,  1.3640e-04,  1.9220e-04],
  [ 4.1379e-04,  2.2609e-04,  1.7946e-04,  1.4506e-04,  7.1598e-05],
  [ 9.3671e-05, -1.0679e-04, -1.3714e-04, -6.7869e-05, -1.7595e-04],
  [ 1.9594e-04, -9.8937e-05, -2.8737e-04, -5.9195e-04, -1.0723e-03]],

 [[ 9.9584e-04,  1.0407e-03, -3.3882e-04, -1.2441e-04,  5.5379e-04],
  [ 1.0216e-03,  7.1866e-04, -6.0237e-04, -5.3089e-04,  1.2587e-04],
  [ 4.3881e-04,  2.2658e-04, -8.1934e-04, -6.4012e-04, -6.2741e-06],
  [ 4.4539e-04,  2.3956e-04, -6.9305e-04, -3.9629e-04,  1.5582e-04],
  [ 3.3184e-04,  3.7942e-04, -3.1739e-04, -4.8969e-04,  6.4680e-05]],

 [[ 1.0634e-03,  9.7087e-04,  7.0802e-04,  4.6164e-04,  4.5124e-04],
  [ 7.7564e-04,  6.4995e-04,  5.4561e-04,  2.6086e-04, -8.5175e-05],
  [ 4.4091e-04,  4.4450e-04,  2.4849e-04,  1.1881e-04, -1.4249e-04],
  [ 2.6883e-04,  1.8940e-04,  7.6072e-05, -2.1375e-04, -1.4611e-04],
  [ 1.8053e-04,  5.1619e-04,  5.5449e-04,  2.5696e-04, -2.4745e-04]],

 …,

 [[-9.7989e-05, -2.7251e-05,  1.1555e-05,  1.5676e-05,  1.7756e-05],
  [-7.3616e-05, -7.6534e-05, -1.4175e-04, -1.1150e-04, -1.7191e-04],
  [ 6.1387e-06, -1.9208e-05, -3.1454e-05,  2.9299e-05, -1.0173e-04],
  [-3.4531e-05, -1.4204e-04, -2.5161e-04, -2.6643e-04, -3.1195e-04],
  [-5.7808e-07, -8.5211e-05, -1.7424e-04, -2.1693e-04, -3.8088e-04]],

 [[-4.6487e-05, -1.1727e-05, -2.2554e-05, -1.0792e-05,  2.3211e-05],
  [-8.9117e-05, -6.8219e-05, -1.1101e-04, -1.0275e-04, -3.5823e-05],
  [ 7.1034e-05,  1.1003e-04,  6.9439e-05,  4.3373e-05, -5.8599e-06],
  [ 4.8651e-05,  4.1804e-05, -6.0304e-05, -5.7562e-05, -1.0830e-04],
  [ 5.0491e-05,  1.6934e-05, -6.4686e-05, -2.8452e-05, -9.1970e-05]],

 [[ 2.6234e-04,  3.7995e-04,  1.9959e-05,  1.2091e-06,  1.4951e-04],
  [ 5.0345e-04,  3.6640e-04,  1.2399e-04, -9.1569e-05, -2.2887e-04],
  [-1.7916e-04,  5.1217e-06, -4.0320e-05,  1.2064e-04, -2.0194e-04],
  [-1.6036e-04, -3.1820e-04, -1.9663e-04, -4.5506e-04, -8.5887e-04],
  [ 8.2449e-05, -5.2748e-05,  2.3805e-05, -8.1189e-06, -2.5879e-04]]],


[[[-9.7912e-05, -6.9740e-05, -3.0444e-04, -3.4164e-04, -1.8655e-04],
```

```
  [-2.8043e-04, -2.5927e-04, -4.9702e-04, -5.3311e-04, -2.9895e-04],
  [-5.7521e-04, -6.3389e-04, -5.3422e-04, -8.1307e-04, -7.1890e-04],
  [-5.3282e-04, -6.7063e-04, -7.0383e-04, -5.2273e-04, -5.3745e-04],
  [-5.1190e-04, -5.9282e-04, -6.1560e-04, -4.7646e-04, -4.8107e-04]],

 [[ 2.5769e-04,  1.4606e-04,  9.6880e-05,  5.9498e-04,  6.2778e-04],
  [ 9.9842e-05,  1.0103e-04, -9.6276e-06,  2.2473e-04,  3.3046e-04],
  [ 3.7009e-05, -2.7433e-04, -3.9721e-04, -1.9406e-04, -1.6680e-04],
  [-2.1225e-04, -4.1699e-04, -8.5584e-04, -4.4613e-04, -3.7839e-04],
  [-1.1730e-04, -3.1633e-04, -7.7794e-04, -3.1971e-04, -1.2976e-04]],

 [[ 5.1939e-04,  1.5454e-04,  7.0995e-06,  5.4661e-05,  2.3819e-04],
  [ 4.4554e-04,  2.2964e-04, -1.1365e-05, -6.8698e-05,  7.0172e-05],
  [ 1.2003e-04,  4.2418e-05, -2.0037e-04, -2.3124e-04, -3.5353e-04],
  [-7.6515e-05, -9.5536e-05, -3.1494e-04, -3.4753e-04, -3.7863e-04],
  [-1.4025e-04, -1.2299e-04, -4.5867e-04, -2.4744e-04,  5.6642e-05]],

 …,

 [[-4.7615e-04, -5.5240e-04, -6.3893e-04, -5.6416e-04, -5.0550e-04],
  [-4.9574e-04, -4.8453e-04, -6.1033e-04, -5.7829e-04, -5.3549e-04],
  [-7.3822e-04, -7.3456e-04, -8.1301e-04, -7.7277e-04, -7.5788e-04],
  [-7.2278e-04, -7.1513e-04, -8.6816e-04, -7.4579e-04, -6.3968e-04],
  [-6.6300e-04, -6.9397e-04, -8.2280e-04, -7.4075e-04, -6.7045e-04]],

 [[-6.8803e-04, -7.2349e-04, -7.6559e-04, -6.3187e-04, -6.0291e-04],
  [-6.7528e-04, -7.0347e-04, -7.7662e-04, -6.6726e-04, -6.6193e-04],
  [-8.4222e-04, -8.5239e-04, -9.2408e-04, -8.5127e-04, -8.3065e-04],
  [-8.0464e-04, -8.4339e-04, -8.9146e-04, -7.5969e-04, -6.6934e-04],
  [-7.8197e-04, -8.3051e-04, -9.0742e-04, -8.6116e-04, -8.0282e-04]],

 [[ 2.3353e-04,  7.3659e-05, -3.0408e-04, -1.9508e-04, -1.4383e-04],
  [ 9.7414e-05,  5.0335e-05, -3.1280e-04, -3.2924e-04, -4.3599e-04],
  [-5.7840e-04, -6.9251e-04, -8.9229e-04, -6.4523e-04, -7.2575e-04],
  [-7.2563e-04, -8.7024e-04, -1.1616e-03, -8.3225e-04, -5.5455e-04],
  [-6.3531e-04, -6.4931e-04, -1.0778e-03, -8.6385e-04, -5.9169e-04]]],


…,


[[[ 4.3780e-04,  4.3476e-04,  9.7652e-05,  1.8186e-04,  2.6350e-04],
  [ 2.7460e-05,  2.6559e-04,  2.2291e-04,  5.8379e-04,  6.7937e-04],
  [-5.1219e-04, -3.5986e-04, -4.8431e-04, -1.9024e-05,  9.1344e-05],
  [-7.3616e-04, -3.6955e-04, -2.2252e-04, -6.4047e-04, -3.4393e-04],
  [-5.8473e-04, -5.5441e-05,  1.5600e-05,  2.9711e-05,  2.8706e-04]],

 [[-1.2437e-03, -7.3934e-04, -7.8212e-04, -1.9338e-04,  5.6150e-05],
```

```
  [-1.7181e-03, -1.2379e-03, -1.2845e-03, -8.4922e-04,  1.5055e-04],
  [-2.2298e-03, -1.8212e-03, -1.9871e-03, -1.5320e-03,  2.3954e-04],
  [-1.4740e-03, -8.9770e-04, -1.1484e-03, -6.8432e-04,  1.5319e-06],
  [-1.8263e-04, -6.7056e-04, -5.3654e-04,  6.0658e-05,  4.9635e-04]],

 [[-8.6954e-04, -1.0845e-03, -1.2265e-03, -1.0873e-03, -3.4619e-04],
  [-1.1833e-03, -1.5678e-03, -1.2209e-03, -1.0256e-03, -3.5638e-04],
  [-1.2782e-03, -1.6788e-03, -1.7216e-03, -1.6080e-03, -1.2119e-03],
  [-5.0349e-04, -1.0511e-03, -9.6694e-04, -6.1873e-04, -4.6243e-04],
  [-1.9261e-04, -7.6920e-04, -7.2575e-04, -1.1704e-04,  3.0290e-04]],

 …,

 [[ 3.0686e-04,  4.0979e-04,  4.8925e-05,  9.8988e-05, -3.1317e-05],
  [ 2.7499e-04,  3.0208e-04,  1.0903e-05, -3.1121e-05, -5.8139e-05],
  [ 1.4191e-04,  1.4528e-04, -9.8022e-06, -1.3126e-04,  5.4507e-06],
  [ 2.1566e-04,  2.8087e-04,  2.2075e-04,  4.4651e-04,  5.3483e-04],
  [ 7.7602e-05,  1.8002e-04,  2.0455e-04,  4.2481e-04,  7.5541e-04]],

 [[ 1.7219e-04,  2.3139e-04,  8.1686e-05,  8.6914e-05,  5.7525e-05],
  [ 2.9454e-04,  2.8112e-04,  1.1254e-04,  1.0567e-04, -2.8031e-05],
  [ 2.8825e-04,  3.5579e-04,  1.1500e-04,  7.3994e-05,  5.8062e-05],
  [ 2.0851e-04,  2.3618e-04,  1.9300e-04,  2.4284e-04,  2.5927e-04],
  [ 7.0782e-05,  1.8241e-04,  1.8202e-04,  3.4971e-04,  5.1421e-04]],

 [[ 2.2952e-04,  3.1015e-04, -1.1041e-04, -2.7124e-04, -3.9043e-04],
  [-7.6485e-05, -1.6388e-04, -4.9380e-04, -9.6374e-04, -6.6502e-04],
  [-2.7368e-04, -4.7661e-04, -4.3077e-04, -8.2248e-04,  3.7704e-05],
  [ 5.2446e-05,  4.4371e-05,  2.9096e-04,  6.9532e-04,  1.1392e-03],
  [-3.5379e-05,  1.5077e-06,  2.3888e-04,  7.2390e-04,  1.2491e-03]]],


[[[ 9.4797e-04,  1.0086e-03,  8.1892e-04,  1.5691e-04,  2.7360e-05],
  [ 9.5706e-04,  1.0460e-03,  4.3786e-04, -6.9386e-05, -2.8065e-04],
  [ 1.1172e-03,  1.1494e-03,  1.0047e-03,  2.2204e-04,  3.8760e-05],
  [ 6.3471e-04,  5.1060e-04,  5.8150e-04,  2.1358e-04, -2.2548e-05],
  [ 1.4226e-04,  6.1198e-05,  2.3219e-04,  4.6572e-05,  2.6896e-05]],

 [[-3.9761e-04, -1.0447e-03, -1.7261e-03, -2.3770e-03, -1.2281e-03],
  [-1.7038e-04, -6.5800e-04, -8.8540e-04, -2.0659e-03, -1.3062e-03],
  [-1.3319e-04, -3.8173e-04,  1.7570e-04, -7.7119e-04, -1.0630e-03],
  [-4.9598e-04, -2.7707e-04,  1.5237e-04, -9.4438e-05, -3.7814e-04],
  [-6.8320e-04, -4.2614e-04,  1.9211e-04,  1.3586e-04, -5.9413e-05]],

 [[ 1.1657e-04, -2.0380e-04, -5.1317e-04, -1.3513e-03, -1.5601e-03],
  [ 8.0483e-05, -6.6007e-05,  2.1982e-05, -7.8746e-04, -1.2340e-03],
  [ 7.2381e-05,  2.1786e-04,  4.0009e-04, -7.8858e-05, -5.7241e-04],
  [-2.0906e-04,  1.1670e-05,  2.8919e-04,  4.7024e-05,  6.4729e-06],
```

```
  [-4.0910e-04, -1.4112e-04,  8.0121e-05,  9.8420e-05,  2.1093e-04]],

 …,

 [[ 4.2617e-04,  4.2650e-04,  6.9945e-04,  6.1953e-04,  5.0554e-04],
  [ 5.5662e-04,  5.9684e-04,  7.1121e-04,  7.5911e-04,  7.1633e-04],
  [ 5.8900e-04,  6.2808e-04,  7.8830e-04,  8.7106e-04,  8.1008e-04],
  [ 3.1941e-04,  3.8021e-04,  5.4383e-04,  4.9816e-04,  6.2085e-04],
  [ 1.4531e-04,  3.0970e-04,  5.0685e-04,  4.6099e-04,  6.1411e-04]],

 [[ 3.4971e-04,  3.5549e-04,  6.4501e-04,  6.3488e-04,  6.2563e-04],
  [ 4.2983e-04,  4.6530e-04,  7.0786e-04,  6.9337e-04,  6.4150e-04],
  [ 4.9754e-04,  5.0660e-04,  7.5073e-04,  8.5189e-04,  8.0654e-04],
  [ 3.2580e-04,  2.9268e-04,  5.0106e-04,  4.6618e-04,  5.2099e-04],
  [ 1.8685e-04,  2.2404e-04,  4.1865e-04,  3.9108e-04,  5.1058e-04]],

 [[ 8.0484e-04,  6.5535e-04,  1.0810e-03,  4.1064e-04,  7.3508e-05],
  [ 6.0660e-04,  1.1075e-03,  1.2097e-03,  1.0726e-03,  7.4086e-04],
  [ 6.9735e-04,  8.5230e-04,  1.4594e-03,  1.5027e-03,  8.8247e-04],
  [ 3.9452e-05,  3.9730e-04,  8.0542e-04,  5.1540e-04,  9.7692e-04],
  [ 6.0080e-05,  4.9296e-05,  2.7724e-04,  1.6629e-05,  6.7922e-04]]],


[[[-4.6676e-04, -3.3034e-04, -2.5553e-05,  1.9678e-04,  3.2805e-04],
  [-4.7311e-04, -2.0971e-04, -9.3128e-05,  1.4181e-04,  1.3902e-04],
  [-4.8650e-04, -3.2634e-04, -2.5351e-04,  4.0608e-05,  2.0778e-04],
  [ 3.0977e-06,  1.0135e-05, -2.6984e-05,  4.2832e-05, -1.6897e-04],
  [ 5.8520e-05, -8.8173e-05, -3.9838e-06,  9.1609e-05,  1.7106e-04]],

 [[-1.1610e-04, -1.8126e-04, -4.7977e-04,  7.6558e-05,  1.4069e-04],
  [-1.9060e-04, -4.6176e-04, -7.0102e-04,  7.9737e-05,  2.2415e-04],
  [-2.6875e-04, -6.4143e-04, -6.0490e-04,  1.6749e-04,  5.6513e-05],
  [-5.5630e-04, -5.2172e-04, -4.7473e-04, -1.8438e-04, -3.4264e-04],
  [-6.5342e-04, -4.3706e-04, -2.7343e-04, -3.2326e-04, -5.4220e-04]],

 [[ 2.3474e-06,  1.8251e-05,  8.3132e-05,  2.3905e-04,  6.8122e-04],
  [ 6.4186e-06, -1.5618e-04, -3.6721e-04, -2.5059e-04,  3.4965e-04],
  [ 1.4379e-05, -2.4791e-04, -4.9076e-04, -3.5483e-04,  1.6600e-04],
  [ 3.7658e-06, -2.2895e-04, -3.0546e-04, -3.1744e-04, -8.0041e-05],
  [-6.1458e-05,  4.9396e-05, -2.5381e-05, -1.4857e-05,  4.2724e-05]],

 …,

 [[-2.3513e-04, -2.7028e-04, -2.5451e-04, -9.7733e-05, -6.5872e-05],
  [-2.0656e-04, -2.2995e-04, -1.4939e-04, -7.2438e-05, -1.2865e-04],
  [-2.3851e-04, -1.9079e-04, -7.5651e-05, -7.9648e-05, -4.4119e-05],
  [-2.4508e-04, -2.1418e-04, -1.7299e-04, -1.1616e-04, -1.1862e-04],
  [-1.6148e-04, -1.8625e-04, -1.1504e-04, -1.0890e-04, -1.3788e-04]],
```

```
          [[-2.1547e-04, -1.8363e-04, -1.8968e-04, -1.6313e-04, -1.6117e-04],
           [-1.7570e-04, -1.7138e-04, -1.0334e-04, -9.5146e-05, -1.5339e-04],
           [-2.8121e-04, -1.3572e-04, -7.9098e-05, -5.7072e-05, -1.0883e-04],
           [-2.6056e-04, -1.9923e-04, -2.3395e-04, -1.5479e-04, -1.8760e-04],
           [-1.8384e-04, -1.6927e-04, -1.7135e-04, -1.1533e-04, -1.9542e-04]],

          [[-7.1685e-05, -1.7897e-04, -2.7602e-04,  8.2839e-05,  4.5806e-04],
           [ 5.5627e-05, -2.2715e-04, -2.8804e-04, -6.9623e-05,  2.5537e-04],
           [ 1.0258e-05, -9.4704e-05, -1.3002e-04, -6.4967e-05,  1.1259e-04],
           [ 1.0904e-04,  8.9909e-05,  2.7540e-05, -6.5938e-05,  1.4285e-04],
           [-1.2677e-05,  1.2514e-04,  2.8154e-04,  2.0237e-04,  1.1977e-04]]])
tensor([-7.8035e-10, -9.5196e-10,  1.3824e-09,  2.0739e-09,  1.3929e-10,
        -4.6461e-10, -1.6087e-11,  2.5648e-10,  5.6353e-09, -1.2824e-10,
        -3.3580e-09, -1.2166e-09,  1.2175e-10,  2.6699e-10, -4.3008e-10,
        -1.5362e-09,  1.6727e-09, -1.3253e-10,  5.9712e-10,  4.9508e-10,
        -3.3135e-09, -1.0155e-09,  1.0186e-10,  6.9338e-10, -1.9259e-10,
        -2.8411e-09, -1.0944e-09,  2.2737e-10,  3.9133e-09, -1.1714e-09,
        -4.4338e-11,  1.0885e-09,  4.2573e-09,  5.6495e-10, -5.7101e-10,
         2.9667e-10, -2.4514e-11, -1.2536e-09, -7.4164e-10, -7.2333e-10,
         1.8576e-09,  9.0407e-10, -1.3710e-09,  8.0178e-11, -3.9802e-10,
         6.1868e-10, -9.4022e-10,  8.7965e-12, -1.7810e-09, -1.7433e-09,
         1.4461e-10, -3.2763e-09,  7.3476e-10, -4.1661e-10,  2.4017e-09,
        -4.0941e-10,  7.4488e-10,  1.4725e-09, -3.6168e-10, -3.9302e-11,
         1.1607e-10,  4.2201e-10,  4.2992e-10,  1.8190e-10, -1.0427e-09,
         4.0654e-10,  6.5148e-09, -5.4010e-09, -2.9623e-09, -8.6568e-10,
         3.9773e-09, -1.6118e-09,  9.0802e-10,  7.0548e-10, -6.5288e-10,
        -1.5705e-09,  4.0487e-11, -6.3252e-10,  1.6215e-10, -3.0759e-09,
        -3.3090e-10, -1.1509e-09,  2.1356e-09,  6.7391e-10,  2.7746e-09,
         1.3638e-09,  6.2039e-10, -1.0696e-09,  2.4841e-09,  1.0332e-09,
        -1.6621e-09,  8.4128e-11,  1.8235e-09,  4.4187e-10,  2.8830e-10,
        -4.8112e-09, -8.2389e-10, -3.5925e-10, -3.7699e-10, -1.6680e-09,
         1.3076e-09,  5.9845e-10,  4.5475e-10,  1.1552e-09,  7.4510e-10,
        -9.6409e-10,  1.3570e-09,  2.0721e-09, -2.0459e-09, -1.2050e-09,
        -7.8217e-10,  1.3785e-09, -3.4298e-09,  7.6034e-10,  3.6002e-10,
         6.4999e-10,  1.9973e-09,  4.3730e-10, -7.2077e-10, -1.2292e-09,
         5.6994e-10,  3.3124e-09, -5.4570e-12, -1.2401e-09, -3.4029e-10,
         8.7039e-10, -1.7917e-09,  3.1923e-10,  1.1741e-10,  3.7243e-09,
         4.0734e-11, -2.0021e-09, -1.0621e-09, -1.8094e-09,  2.9932e-09,
        -4.3148e-10,  1.9460e-09,  6.7126e-09, -1.2551e-10, -3.0725e-09,
         1.5620e-09,  1.7485e-10,  1.8945e-09, -4.1123e-10,  9.5224e-10,
         1.5471e-09, -1.5028e-09,  3.2899e-09,  3.0237e-09,  1.8104e-09,
         3.9786e-09,  8.4947e-10,  4.8442e-09,  5.4486e-10, -1.5824e-09,
         1.0679e-09, -1.2603e-09,  1.5374e-09,  1.0010e-09,  2.4378e-10,
         2.0724e-09, -7.4160e-09,  4.4201e-10,  9.0536e-10,  1.1060e-10,
        -1.9991e-09,  3.8835e-10,  2.6651e-10,  1.5217e-09,  8.8508e-11,
         1.5419e-09,  3.3040e-10, -3.1869e-09, -1.1562e-10,  5.3415e-09,
         3.4743e-09, -8.5947e-11, -1.4408e-09, -1.1139e-09, -9.8225e-11,
```

```
         3.2644e-09, -5.0620e-10,  8.0718e-10,  3.2310e-10,  1.5543e-09,
        -8.3421e-10,  1.4293e-09,  1.3156e-09,  1.3136e-10, -1.1546e-09,
        -2.0848e-09, -3.6812e-10,  1.4776e-09, -2.8511e-09, -8.3494e-10,
         4.8658e-10, -1.6376e-09, -3.6707e-11, -1.1041e-09,  3.3358e-09,
        -9.6463e-10,  8.8360e-10, -3.1797e-09, -4.1878e-09,  1.8927e-09,
        -1.0368e-10,  1.6307e-09, -5.9946e-10, -3.1287e-10,  6.3119e-10,
         6.7530e-11,  3.6101e-10, -9.1609e-10, -1.2877e-09,  3.7554e-10,
        -2.3334e-10, -8.9449e-10,  1.2860e-09, -2.2419e-10, -3.1628e-10,
        -1.6662e-09, -3.3692e-09, -3.6820e-09, -1.4328e-09, -1.9715e-09,
         9.7498e-10, -2.2342e-09,  1.0442e-09,  4.2381e-10, -2.1932e-09,
        -9.8282e-11, -2.4420e-09, -8.4991e-10, -4.1487e-09,  1.0424e-09,
         2.1255e-09,  3.2352e-09, -3.6470e-10,  1.2097e-09,  2.1246e-09,
        -8.5194e-11, -4.8438e-10, -6.5940e-10,  4.6879e-10, -1.8700e-09,
        -3.2182e-09, -1.5384e-09,  2.4346e-10,  3.6900e-11, -4.9269e-11,
        -6.0891e-10,  2.1123e-09, -9.1751e-10, -2.0996e-09, -2.9856e-10,
         5.9845e-10])
tensor([-3.8986e-03,  1.4945e-03,  2.5731e-03,  2.0052e-03, -9.6142e-04,
        -6.3291e-03, -8.3329e-03, -9.1264e-03, -6.9377e-03, -1.1869e-03,
        -4.0112e-03, -5.9935e-03,  9.5484e-04,  1.2605e-02, -1.7083e-03,
        -9.4066e-04, -3.4314e-03, -1.0784e-02,  5.3217e-03,  2.5299e-03,
        -2.1551e-03,  2.4943e-02, -2.2172e-03,  1.5063e-02, -8.1814e-05,
         4.6647e-03,  3.9431e-03,  6.8585e-03, -6.4079e-03, -1.3191e-02,
         7.6075e-05, -4.0956e-04, -1.7329e-03, -3.3053e-03,  2.6647e-03,
         2.2028e-03,  9.8469e-04,  6.3675e-03,  4.8811e-03,  8.1782e-04,
        -3.2378e-03, -3.0068e-03, -4.9481e-04,  3.5205e-03, -2.7907e-03,
        -1.8138e-02, -3.0204e-03, -5.4335e-03, -2.1413e-03,  1.5482e-04,
        -2.6574e-03, -6.5593e-04, -1.1001e-02, -2.7784e-03, -2.5458e-03,
         7.6751e-03, -3.4155e-03,  2.9603e-03, -1.0675e-02, -1.4983e-03,
         4.2957e-03, -3.0938e-04,  2.4327e-03,  5.5390e-03, -1.6990e-03,
        -2.5848e-03,  2.2563e-03, -4.7920e-03, -2.7910e-02, -4.1877e-03,
         1.6291e-03,  3.4418e-03, -7.7704e-03, -6.8144e-03, -1.1835e-02,
         1.8658e-02, -4.9774e-03, -2.4984e-04, -2.6018e-03,  4.0282e-03,
         1.8727e-04, -5.9331e-03,  4.5836e-03,  8.4437e-03,  1.0151e-03,
        -6.6689e-03, -1.3976e-02, -7.4795e-03,  4.0821e-04,  5.7819e-03,
        -8.7337e-03, -2.1652e-03, -2.5429e-03,  2.8459e-02, -3.3271e-03,
        -2.9906e-03,  1.8501e-03,  4.4982e-04,  1.4596e-04, -1.3898e-03,
         4.0830e-03, -7.8807e-04, -9.9162e-04, -2.3744e-03, -4.8133e-03,
        -4.9797e-03,  1.3266e-03,  9.1935e-04, -1.6758e-03, -9.9421e-03,
        -2.2801e-03,  2.4047e-04,  4.9124e-03,  8.5496e-04, -2.0701e-03,
        -2.4143e-03,  4.2990e-04, -2.9930e-03,  4.2872e-04, -1.9927e-03,
         1.8014e-02,  3.7795e-03, -4.0677e-03,  2.1227e-03,  1.0786e-04,
        -3.8165e-03,  3.4181e-03, -2.0768e-04,  7.8811e-04, -1.1247e-03,
        -1.6386e-03,  3.7744e-03,  1.1381e-03,  7.7002e-03, -3.7212e-03,
        -2.2507e-04, -1.5803e-02, -3.8254e-03, -6.7616e-04, -8.3223e-03,
        -1.9917e-03,  2.3001e-04,  6.2019e-03, -4.7183e-03, -2.2435e-03,
         3.1849e-03, -1.6747e-02,  9.8823e-03,  9.3029e-03, -1.0607e-02,
        -1.0336e-03, -3.8649e-05, -1.1792e-02,  2.0750e-02, -8.7392e-03,
        -5.1266e-04,  2.5491e-03, -2.7369e-03, -2.2956e-03, -2.8164e-03,
```

```
         6.6464e-03,  3.3100e-03, -5.1524e-03, -1.5433e-03, -7.3968e-05,
        -2.8964e-04,  7.4790e-04, -4.8082e-03,  1.4927e-02,  1.0543e-03,
        -1.6835e-02, -1.7416e-03,  3.2063e-03,  8.9657e-03, -4.1400e-03,
         2.2899e-03,  2.3023e-03, -4.8800e-03,  4.3852e-04, -8.1243e-04,
        -3.6452e-03,  2.6710e-03,  9.3805e-04,  4.7574e-03,  3.8824e-03,
         8.1554e-04, -1.3176e-02, -1.2496e-03, -1.8766e-03, -8.5553e-04,
         3.4231e-03, -1.8003e-03, -8.2283e-04,  2.0538e-03, -3.5917e-03,
         6.4528e-04, -2.4127e-03,  1.3130e-03, -2.6670e-04,  8.2315e-03,
         5.3014e-03, -1.6327e-03, -6.0463e-03, -2.2206e-04, -6.0727e-03,
         4.0026e-03, -4.1517e-03, -1.6483e-03, -3.9739e-03, -7.6874e-04,
        -5.1469e-04, -3.7710e-03, -6.2861e-03,  6.0310e-03, -2.3549e-04,
        -1.2588e-03, -3.5785e-03, -4.4713e-03,  1.7647e-03, -1.8592e-03,
        -3.1165e-05, -6.1110e-03,  5.1406e-03, -2.5132e-03, -1.6648e-03,
         1.5628e-03,  1.9311e-03, -4.0936e-03, -1.4536e-03,  8.8553e-03,
         7.0569e-03, -6.3811e-04,  4.3447e-03,  1.6222e-02,  1.2173e-03,
         1.2571e-02,  8.1568e-04, -5.4856e-04,  1.1623e-02, -1.0969e-03,
        -3.4984e-03, -4.1146e-03,  1.1222e-02, -5.5388e-06, -6.3069e-03,
        -7.4079e-04,  2.0951e-02, -7.2530e-03, -4.0697e-04, -8.6161e-04,
         9.7208e-04,  3.9874e-03,  4.7698e-04,  6.2913e-03, -6.8581e-04,
        -2.9687e-04])
tensor([-2.9957e-03,  3.2991e-03,  4.3644e-03, -6.3355e-03, -1.0738e-03,
        -6.3661e-03, -9.4852e-03, -1.1576e-02, -7.0944e-03,  1.5531e-03,
         3.0700e-03, -8.7313e-03,  9.3740e-04,  5.6101e-03,  8.9850e-05,
         6.6353e-04, -2.6582e-03, -1.3526e-02,  5.7908e-03,  2.6181e-03,
        -3.1052e-03,  5.5800e-03, -3.4673e-03,  5.4478e-03,  1.2145e-03,
         8.7940e-04, -2.9739e-04,  8.6489e-03, -6.4703e-03, -1.0891e-02,
         2.0505e-03, -1.3898e-03, -1.7459e-04, -6.7383e-03,  5.7560e-05,
         4.5921e-03,  2.0834e-03,  6.2316e-03,  4.0072e-03,  1.4321e-03,
        -6.1040e-03, -6.8545e-03,  1.1030e-03,  3.2070e-03, -1.5213e-03,
        -3.2389e-03, -2.4304e-03, -6.5893e-03, -8.5038e-04, -4.0260e-04,
        -3.2622e-03,  5.3209e-04, -1.0232e-02, -5.6708e-03,  3.2695e-03,
        -1.3683e-03, -3.6037e-03,  2.3317e-03, -6.6459e-03, -3.2983e-04,
         6.6641e-03, -5.6473e-04, -2.3208e-04, -1.6412e-03, -1.9212e-04,
        -2.4652e-03, -7.0037e-04, -5.4097e-03, -1.5809e-02, -1.2457e-03,
         7.5471e-04,  1.1477e-03,  1.3306e-04, -5.5205e-03, -2.9343e-03,
         3.4322e-03, -5.1346e-03,  4.9479e-04, -2.0319e-03,  2.9974e-05,
        -2.0477e-04, -5.0588e-03,  3.4810e-03,  5.5609e-03,  1.1693e-03,
        -7.4046e-03, -8.2107e-03, -4.7956e-03,  5.9712e-03,  5.4123e-03,
        -1.1393e-02, -6.1800e-03, -5.2991e-03,  1.4404e-02, -5.4537e-03,
        -5.6984e-03,  5.0284e-03,  4.1546e-03, -1.6838e-04, -5.3165e-04,
         1.6130e-03,  2.6760e-03,  1.7289e-03, -1.3574e-03, -2.3412e-03,
        -4.8095e-03,  5.1819e-03,  5.2204e-04, -1.7143e-04, -1.6432e-02,
        -3.2313e-03,  1.1968e-03,  7.2131e-03,  5.7803e-04, -5.4753e-03,
        -2.7673e-03,  2.8265e-03, -3.5784e-03,  4.0214e-03,  3.8446e-04,
         5.7438e-03, -6.1137e-04, -1.8737e-03,  5.5429e-04, -3.0964e-05,
        -2.2864e-03,  2.6149e-03,  1.5622e-03,  1.5844e-04,  1.0100e-03,
        -8.0174e-05,  3.1496e-03, -1.2976e-03,  8.3471e-03, -6.0608e-03,
         8.7678e-04,  2.7662e-04, -6.5557e-03, -1.3632e-03, -6.0805e-03,
```

```
         -8.0829e-04, -3.9630e-04,  8.2552e-03, -9.7995e-03, -3.3562e-03,
          5.3622e-03, -3.4776e-03,  7.2845e-03,  3.5029e-03, -1.2666e-02,
         -4.6520e-03,  2.2408e-03, -8.1108e-03,  6.1943e-03, -8.8224e-03,
         -7.6276e-04,  7.8552e-03, -2.4998e-03, -6.7975e-03, -2.5651e-03,
          2.5834e-03,  5.4043e-03, -7.8623e-03, -3.3429e-03,  2.3530e-03,
         -6.1829e-04,  1.4006e-03, -1.7140e-03,  2.4718e-03, -5.8330e-04,
         -2.1070e-02, -1.4294e-03,  8.3705e-03,  2.6760e-04, -1.3617e-02,
          4.3438e-03,  3.6081e-04, -6.0492e-03,  2.6049e-03,  1.9962e-04,
         -1.2257e-03,  2.4852e-03, -3.9830e-03,  2.3507e-03,  5.5834e-03,
          1.5324e-03, -2.0312e-03, -8.8335e-04, -2.2193e-03, -2.5488e-03,
          4.3422e-03, -5.6795e-03,  1.4405e-04,  3.2739e-03,  1.6711e-03,
         -1.1240e-04, -2.4323e-03,  1.6726e-03, -8.9865e-04,  2.7804e-03,
          3.8001e-03,  1.7710e-03, -1.0691e-02, -1.5621e-04, -5.7754e-03,
          5.2758e-03, -5.5943e-03,  2.6410e-03, -3.3664e-03,  3.0578e-03,
          1.4643e-03, -1.2871e-03, -6.8284e-03, -1.6017e-03,  2.1897e-03,
          3.6682e-03, -4.5554e-03, -5.0781e-03,  2.3844e-03,  9.1036e-04,
          2.0756e-03, -5.7538e-03, -5.4168e-03, -6.9761e-03, -2.3943e-03,
         -3.7953e-04,  2.9287e-03, -1.5662e-03, -2.4935e-03, -6.4603e-03,
          1.2234e-03,  2.5492e-03,  3.8562e-03,  3.4407e-03,  1.7755e-03,
          4.1530e-03, -1.1279e-03,  2.4975e-03,  3.4883e-03, -5.1812e-03,
         -3.3095e-03, -8.2656e-04,  8.3080e-03,  1.4764e-03, -6.5043e-03,
         -1.5137e-03,  7.5465e-03, -8.8357e-03,  1.0873e-03, -8.5975e-04,
          2.1853e-03,  5.4332e-03,  1.0868e-03,  7.9979e-03, -3.0186e-03,
          1.8121e-03])
tensor([[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,


[[[ 1.9792e-02,  1.8944e-02,  1.7481e-02],
  [ 9.2110e-03,  1.1635e-02,  1.4808e-02],
  [ 2.7389e-03, -1.6844e-03,  2.3286e-03]],

 [[ 4.7058e-03,  4.3664e-03,  8.2634e-03],
  [ 9.0521e-03,  6.1785e-03,  8.3872e-03],
  [ 9.6981e-03,  7.5944e-03,  1.2549e-02]],

 [[-8.3530e-03, -7.5725e-03, -7.4971e-05],
  [-2.8443e-03, -2.7713e-03,  3.8750e-03],
  [ 1.9315e-03, -8.4464e-06,  6.6768e-03]],

 …,

 [[ 3.5950e-03,  4.8220e-03,  8.3805e-03],
  [ 4.0214e-03,  1.6636e-04,  3.4522e-03],
  [ 6.4550e-04, -4.8653e-03, -3.1398e-03]],

 [[-6.1137e-03, -1.0468e-02, -8.8256e-03],
  [ 4.0800e-03, -2.5151e-03,  8.9120e-04],
  [ 4.6663e-03,  2.3586e-04,  2.9938e-03]],

 [[ 1.8170e-02,  1.3849e-02,  1.5937e-02],
  [ 1.7857e-02,  1.4476e-02,  1.7599e-02],
  [ 1.6674e-02,  1.4456e-02,  1.4551e-02]]],


[[[-1.0982e-02, -1.7706e-02, -2.5914e-02],
  [-1.5754e-02, -1.9218e-02, -2.0401e-02],
  [-2.6329e-02, -2.9368e-02, -2.9214e-02]],

 [[-2.8225e-02, -3.4308e-02, -3.3743e-02],
  [-2.1144e-02, -2.1564e-02, -1.3937e-02],
  [-2.0145e-02, -1.3987e-02, -7.6155e-03]],

 [[-2.5831e-02, -2.5019e-02, -2.3647e-02],
```

```
          [-2.5888e-02, -2.4241e-02, -2.3261e-02],
          [-1.7284e-02, -1.2807e-02, -1.7703e-02]],

         …,

         [[-5.8224e-03,  8.6184e-04, -4.4179e-03],
          [-2.3800e-02, -2.2734e-02, -3.0171e-02],
          [-2.4491e-02, -2.9220e-02, -3.7331e-02]],

         [[-5.7767e-02, -4.8255e-02, -4.5417e-02],
          [-5.6522e-02, -4.8551e-02, -4.9066e-02],
          [-2.9973e-02, -1.9263e-02, -3.1864e-02]],

         [[-2.0776e-02, -2.2948e-02, -2.3421e-02],
          [-8.9969e-03, -9.3864e-03, -7.2207e-03],
          [-3.8997e-03,  4.2678e-04,  2.1336e-03]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([ 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -6.8667e-05,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	1.8830e-03,
0.0000e+00,	0.0000e+00,	0.0000e+00,	-5.9354e-03,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	-1.0952e-02,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	-2.0750e-03,	0.0000e+00,	-1.1549e-03,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	2.6216e-02,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	-3.8715e-03,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	6.3405e-03,	0.0000e+00,
0.0000e+00,	5.8381e-03,	0.0000e+00,	0.0000e+00,	3.4494e-10,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,

```
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         -8.1958e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         -8.8232e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  6.5913e-03,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00, -2.5944e-03,  0.0000e+00,
          0.0000e+00,  0.0000e+00, -4.7773e-03,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00, -2.7708e-04,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  1.8774e-03, -2.6057e-04,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -1.7796e-02,  1.5762e-04,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  8.9460e-03, -2.5405e-02,  0.0000e+00])
tensor([[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[-1.2796e-01, -1.2804e-01, -1.0983e-01],
          [-9.8877e-02, -1.3671e-01, -1.3594e-01],
          [-3.1188e-02, -8.5254e-02, -9.7931e-02]],

         [[ 1.9364e-02,  7.3532e-03, -2.8361e-02],
          [-1.1769e-01, -3.6391e-02, -2.1128e-02],
          [-2.8286e-01, -1.5940e-01, -8.8301e-02]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[-1.1908e-05, -2.3358e-04, -1.3902e-03],
  [-1.7711e-07,  1.2492e-04, -1.1588e-04],
  [-5.3834e-06, -5.2625e-07, -2.3939e-05]],

 [[-1.1286e-03, -2.6829e-04,  5.1858e-04],
  [-1.0968e-03, -6.3313e-04,  1.4187e-03],
  [-5.3499e-04, -7.1870e-03, -6.3460e-03]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[-2.5098e-02, -2.0852e-02, -1.5201e-02],
  [-9.6008e-03, -6.0991e-03, -3.1214e-03],
  [-2.3595e-03, -2.1568e-03,  6.1089e-03]],

 [[-2.1041e-03,  1.3255e-02,  1.4892e-02],
  [-7.4796e-03, -1.7767e-03, -8.1930e-04],
  [-2.8992e-02, -3.2886e-02, -3.6862e-02]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  6.8299e-06],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  2.1047e-06]],

         [[ 0.0000e+00,  8.7854e-06,  0.0000e+00],
          [-4.7304e-04, -1.1485e-04, -5.6871e-06],
          [-1.0054e-04,  5.3865e-05, -1.2069e-05]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([-4.4835e-02, -8.2269e-04,  0.0000e+00, -1.0205e-04, -9.5373e-03,
        -1.1639e-01,  7.4429e-04, -1.4386e-02,  1.1529e-02,  0.0000e+00,
         5.7553e-04,  1.6504e-03,  4.3842e-02,  3.2301e-06,  3.7106e-05,
         2.1448e-03,  0.0000e+00, -8.4527e-06,  0.0000e+00, -1.1529e-03,
         4.8221e-04, -7.4422e-04, -1.2985e-03, -4.9261e-04,  0.0000e+00,
```

```
-1.7716e-02,  1.4732e-04,  7.5332e-04,  0.0000e+00,  1.0931e-03,
-1.9286e-02,  1.4373e-02, -8.9155e-04,  0.0000e+00,  3.7434e-04,
-3.5998e-03,  2.0324e-03,  3.2330e-04, -3.8063e-03, -2.5460e-02,
-1.9489e-03, -1.0290e-02, -5.1340e-03, -3.8419e-03,  0.0000e+00,
 2.0737e-01,  2.5672e-03, -1.5530e-03,  8.6433e-07, -6.0417e-03,
 0.0000e+00, -6.2650e-04,  2.5820e-03, -6.4859e-04, -3.1940e-02,
 2.1282e-04,  2.7371e-02,  0.0000e+00,  2.1455e-04,  0.0000e+00,
 0.0000e+00,  0.0000e+00, -1.5315e-03,  6.3923e-04, -1.9057e-02,
-8.3943e-03,  1.2246e-02, -5.3412e-03, -2.0483e-02, -1.0871e-02,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  2.0512e-04, -3.6470e-04,
 1.6315e-04,  9.1975e-05,  2.6977e-03,  4.1368e-04,  1.5029e-03,
 0.0000e+00,  2.1323e-04,  1.1478e-04,  0.0000e+00,  2.0302e-03,
 0.0000e+00, -1.0535e-02, -6.6947e-02, -5.4558e-03,  0.0000e+00,
 7.8844e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  6.5803e-04,
 0.0000e+00,  0.0000e+00,  8.5705e-03,  0.0000e+00,  6.0817e-04,
-8.7523e-02,  2.2813e-02, -4.0867e-03, -1.2615e-03,  0.0000e+00,
-1.4224e-02, -2.5282e-02, -1.1007e-02, -8.3346e-03, -3.5089e-02,
-3.3129e-03, -1.1149e-03, -2.0154e-04,  2.1394e-03,  2.7206e-02,
-1.2741e-02, -2.2715e-02,  1.6229e-03,  0.0000e+00,  2.9480e-03,
 1.1665e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.3380e-04,
 0.0000e+00,  1.7481e-04,  3.1116e-04,  0.0000e+00,  7.3145e-03,
 0.0000e+00,  0.0000e+00, -8.4823e-04, -2.5145e-02, -6.0239e-03,
 2.3120e-02,  0.0000e+00, -7.8275e-04,  0.0000e+00, -5.2640e-04,
-5.7239e-04, -1.7612e-01, -1.0154e-01, -5.2608e-04,  0.0000e+00,
 0.0000e+00,  0.0000e+00, -4.9025e-02,  6.1307e-03, -2.2449e-04,
 0.0000e+00, -2.7262e-03,  4.0645e-03, -7.1600e-05,  1.4805e-05,
-5.6196e-03,  0.0000e+00, -5.0273e-05, -6.6938e-04, -4.8990e-03,
-1.5814e-03,  5.6542e-02,  0.0000e+00,  0.0000e+00, -1.0449e-02,
 3.2213e-06, -2.2396e-03,  7.7616e-03, -7.7524e-04, -1.5002e-02,
-1.9859e-04, -3.7024e-04,  1.4785e-04, -4.5775e-04,  1.9078e-02,
-1.9814e-02,  6.8956e-05, -1.4008e-03,  3.2221e-02, -1.8412e-04,
 0.0000e+00, -1.1604e-02,  7.0708e-04, -5.0958e-04,  1.7366e-03,
 0.0000e+00, -3.6595e-03,  1.0125e-04,  3.7646e-04,  2.5988e-02,
 1.2846e-03,  0.0000e+00,  0.0000e+00,  5.9005e-02,  3.0534e-03,
 1.0795e-03, -7.3429e-02,  2.3819e-02,  0.0000e+00,  0.0000e+00,
 3.7875e-02,  1.6417e-02,  4.8061e-04, -1.3042e-02, -1.8408e-03,
 9.9889e-04, -3.6524e-04,  3.8635e-04, -9.7835e-05,  1.0798e-03,
-6.6645e-04,  7.7756e-04, -2.8963e-03,  1.1265e-02,  5.6243e-04,
 1.0180e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00, -2.8059e-05, -1.3811e-03, -1.0330e-02,  3.1349e-06,
 0.0000e+00,  1.2725e-02, -6.6598e-03,  5.0309e-03,  0.0000e+00,
-4.7774e-03,  0.0000e+00,  0.0000e+00, -1.7269e-02, -3.1877e-03,
-3.3639e-03, -1.6993e-04,  1.8869e-04, -1.0499e-02, -2.3315e-02,
 0.0000e+00, -5.7320e-03,  0.0000e+00,  1.6235e-03, -1.0830e-02,
 0.0000e+00,  8.2480e-03,  0.0000e+00,  3.3721e-05, -1.4691e-02,
 0.0000e+00, -1.7483e-03,  3.1087e-02,  0.0000e+00,  0.0000e+00,
 7.6337e-08,  1.5774e-04, -1.8453e-04, -6.5830e-03,  0.0000e+00,
-1.0548e-04, -1.5354e-03, -9.7096e-03, -9.1351e-04,  0.0000e+00,
```

```
        -4.9773e-03,  3.0238e-03, -4.5130e-03, -2.4738e-03, -2.1631e-02,
         0.0000e+00,  0.0000e+00, -1.3373e-03,  0.0000e+00, -7.5598e-02,
        -1.7103e-04, -2.3097e-02, -1.1814e-03,  2.2262e-04, -1.9393e-03,
         2.1125e-04, -6.7646e-03, -4.3768e-04,  4.4021e-03,  5.1401e-02,
         0.0000e+00,  2.7640e-03,  0.0000e+00, -3.4444e-03, -1.7553e-03,
         5.4422e-04,  5.5801e-05,  3.3393e-04, -1.0717e-03,  0.0000e+00,
        -6.8927e-03, -1.5625e-03, -3.0476e-03,  0.0000e+00, -4.7012e-04,
         9.9366e-03,  4.5822e-02, -2.8412e-04, -1.0556e-02,  0.0000e+00,
        -2.0664e-02,  7.1715e-03,  0.0000e+00, -2.9380e-02,  0.0000e+00,
         0.0000e+00, -2.0922e-03,  1.2987e-03, -4.0210e-03,  1.1874e-03,
         3.7334e-03, -9.2999e-04,  0.0000e+00,  0.0000e+00, -5.2325e-03,
         0.0000e+00,  8.3302e-05,  8.2725e-04, -1.5438e-02, -1.6655e-04,
        -3.9397e-03,  0.0000e+00, -2.9527e-07,  3.0903e-02,  0.0000e+00,
        -5.9302e-04,  5.9380e-02,  1.2683e-02,  0.0000e+00, -1.1208e-02,
        -1.1765e-03,  0.0000e+00, -6.2189e-04,  3.7008e-02,  3.4149e-02,
        -8.2041e-04,  5.0371e-03,  9.9417e-03, -1.3862e-04,  6.1046e-02,
         1.3535e-01, -1.3887e-02,  0.0000e+00,  2.1832e-03,  6.1735e-04,
         1.8581e-04, -2.5400e-03,  1.5011e-03, -1.5495e-03,  5.3763e-04,
         1.6732e-04, -1.1800e-02,  0.0000e+00,  1.6578e-03,  8.9720e-04,
        -4.1742e-03, -8.9232e-05, -4.3099e-04,  0.0000e+00,  0.0000e+00,
         3.9582e-02,  7.7905e-03, -2.3049e-05, -3.0786e-04,  3.5083e-02,
         7.0023e-04,  6.1581e-03,  1.1062e-03,  2.7757e-02, -8.2540e-04,
         0.0000e+00, -4.4566e-03, -1.2764e-03,  0.0000e+00,  1.3659e-03,
         3.3543e-05, -4.5230e-04, -5.9953e-04,  0.0000e+00])
tensor([[[[-4.2797e-04, -6.0166e-04,  9.1756e-04],
          [-4.8923e-04,  1.3966e-04,  1.5801e-03],
          [ 5.5729e-03,  1.6085e-03,  3.0347e-04]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  3.0847e-05],
          [ 0.0000e+00,  0.0000e+00,  3.6848e-05]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00, -8.6148e-05],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00, -5.1853e-05,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([ 8.1902e-03,  0.0000e+00,  0.0000e+00,  2.4470e-07,  0.0000e+00,
         0.0000e+00,  1.8558e-03,  0.0000e+00,  0.0000e+00,  2.4178e-03,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  4.7592e-01,  1.2370e-01,
         4.8638e-02, -5.7917e-02, -1.7985e-02,  0.0000e+00,  1.2339e-03,
         1.8168e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  4.6841e-06,
```

```
          0.0000e+00,  0.0000e+00,  0.0000e+00,  1.2522e-04, -4.3076e-05,
          2.6461e-01, -4.6466e-03,  7.2376e-03,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          2.3871e-04,  0.0000e+00,  0.0000e+00,  2.7215e-02,  3.1522e-02,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  1.3515e-07,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  8.8437e-04, -9.4422e-01,  0.0000e+00,
          2.5169e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.3046e-01,
          0.0000e+00,  4.0751e-02,  0.0000e+00, -2.6676e-03,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  3.8239e-02, -2.4618e-02,  0.0000e+00,
          1.2528e-07,  9.1605e-03,  6.5631e-02, -7.2631e-05,  0.0000e+00,
          7.8597e-04,  1.7179e-02,  0.0000e+00, -1.5471e-06,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  4.6670e-04,  0.0000e+00,  0.0000e+00,
          2.9253e-03,  9.6946e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00, -1.4335e-02,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00, -4.1564e-04, -1.5103e-02,
          0.0000e+00,  0.0000e+00, -2.8395e-02,  0.0000e+00, -1.3929e-04,
         -1.4651e-01,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  1.1698e-02, -1.3746e-01,
          1.2085e-05,  0.0000e+00,  9.2346e-02,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  7.3366e-04, -5.2359e-03,  0.0000e+00,
         -1.5124e-02,  0.0000e+00, -2.1569e-02,  0.0000e+00,  6.7740e-04,
          0.0000e+00,  0.0000e+00, -1.9378e-01, -1.0122e-01, -9.7069e-03,
         -2.6389e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00, -6.3375e-07,
          0.0000e+00,  0.0000e+00,  5.4080e-03,  0.0000e+00,  0.0000e+00,
         -1.6676e-02,  0.0000e+00, -1.7538e-05, -2.0290e-04,  0.0000e+00,
          1.0464e-04,  2.3912e-02,  0.0000e+00,  0.0000e+00,  1.9872e-03,
          7.0189e-04,  0.0000e+00,  1.1895e-01,  0.0000e+00,  0.0000e+00,
         -2.6903e-04,  0.0000e+00,  2.3497e-03, -2.8724e-03, -3.2037e-03,
          1.1636e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -2.5062e-01, -2.1168e-02,  1.8319e-01, -4.9561e-03,
         -6.8174e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00, -6.0996e-02,
         -2.6334e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  1.9901e-03,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  5.3769e-01,  2.8715e-03,  0.0000e+00,
          0.0000e+00, -1.8445e-02,  1.2201e-01,  0.0000e+00, -2.6866e-01,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          2.9269e-01,  0.0000e+00,  0.0000e+00, -4.6228e-02,  0.0000e+00,
          1.3270e-04, -2.2251e-04, -1.4691e-03, -1.1494e-05,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00, -4.4255e-02,  0.0000e+00,
          4.8042e-04, -3.1672e-01,  2.2983e-02,  1.5261e-03,  0.0000e+00,
         -9.7984e-02,  0.0000e+00, -7.9392e-02,  0.0000e+00,  0.0000e+00,
         -5.1334e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,  3.6759e-02,
          0.0000e+00, -5.2776e-06,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         -1.6157e-03,  0.0000e+00,  0.0000e+00,  1.6672e-01,  0.0000e+00,
          0.0000e+00, -6.7255e-03,  1.8480e-02,  0.0000e+00,  0.0000e+00,
          0.0000e+00]])
tensor([[0.0000e+00, 1.1159e-04, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
```

```
         0.0000e+00],
        [0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00],
        [6.1091e-05, 8.9742e-07, 2.6945e-05,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00],
        …,
        [0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00],
        [0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00],
        [0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00]])
tensor([0.0094, 0.0000, 0.0048,  …, 0.0000, 0.0016, 0.0004])
tensor([[0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        …,
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.]])
tensor([0., 0., 0.,  …, 0., 0., 0.])
tensor([[0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        …,
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.]])
tensor([ 5.5263e-02,  6.2145e-03, -6.2052e-02,  1.3076e-02, -9.4425e-03,
        -5.8693e-02,  4.8323e-02,  1.5910e-02, -5.7978e-02,  3.5261e-02,
        -3.5680e-02,  2.3292e-03,  2.4714e-02, -1.7132e-02,  3.9469e-03,
         4.0643e-03,  3.6323e-02, -2.6483e-02,  2.7082e-03,  1.5347e-02,
         5.7529e-06,  4.1781e-06,  4.2152e-06,  4.1085e-06,  4.1501e-06,
         4.1525e-06,  4.0215e-06,  4.1899e-06,  3.9287e-06,  3.9421e-06,
         4.0193e-06,  4.2059e-06,  4.0117e-06,  4.0934e-06,  4.2015e-06,
         4.2040e-06,  4.1238e-06,  3.9395e-06,  4.1981e-06,  3.9555e-06,
         4.1170e-06,  4.1897e-06,  3.7279e-06,  4.1549e-06,  3.7826e-06,
         3.9766e-06,  4.0843e-06,  4.1395e-06,  4.0575e-06,  4.2157e-06,
         3.9485e-06,  4.1270e-06,  3.9723e-06,  4.2161e-06,  4.0183e-06,
         3.9949e-06,  4.0943e-06,  3.9322e-06,  4.2072e-06,  4.0712e-06,
         3.9580e-06,  4.0362e-06,  3.8078e-06,  3.8106e-06,  3.8681e-06,
         4.0167e-06,  3.7989e-06,  3.9981e-06,  4.1650e-06,  3.7593e-06,
         4.0910e-06,  3.9335e-06,  4.2111e-06,  4.1316e-06,  4.0439e-06,
         4.1624e-06,  3.9447e-06,  4.1557e-06,  4.1462e-06,  3.6986e-06,
         4.1574e-06,  4.1355e-06,  3.7709e-06,  3.9787e-06,  4.1119e-06,
         3.7101e-06,  4.0301e-06,  3.9215e-06,  3.8828e-06,  4.2127e-06,
         4.1802e-06,  4.1253e-06,  3.9353e-06,  4.2087e-06,  4.1958e-06,
         3.9500e-06,  3.9741e-06,  4.1511e-06,  3.7404e-06,  4.0045e-06,
```

```
4.0153e-06,  4.2107e-06,  3.9352e-06,  4.1274e-06,  4.1194e-06,
3.8532e-06,  4.0868e-06,  4.1434e-06,  4.2087e-06,  3.7606e-06,
4.0611e-06,  3.9023e-06,  4.0474e-06,  4.1536e-06,  4.0506e-06,
4.2017e-06,  4.0467e-06,  4.1636e-06,  3.8244e-06,  4.2142e-06,
4.1784e-06,  4.1653e-06,  4.1572e-06,  3.8017e-06,  4.1337e-06,
4.2021e-06,  3.9970e-06,  4.1242e-06,  3.9908e-06,  4.1048e-06,
4.2066e-06,  3.7302e-06,  4.2119e-06,  3.9739e-06,  4.0787e-06,
4.2134e-06,  4.0844e-06,  4.1839e-06,  4.1759e-06,  4.1959e-06,
3.9126e-06,  3.7593e-06,  4.0186e-06,  4.2005e-06,  4.2088e-06,
4.0008e-06,  4.2153e-06,  3.9202e-06,  4.1587e-06,  3.9081e-06,
3.9374e-06,  3.9446e-06,  4.0183e-06,  3.7175e-06,  4.2173e-06,
4.2009e-06,  4.2095e-06,  4.0606e-06,  3.8511e-06,  4.1276e-06,
3.9546e-06,  4.1943e-06,  4.1690e-06,  4.0987e-06,  3.8571e-06,
4.1737e-06,  4.1047e-06,  4.1997e-06,  3.8330e-06,  3.9662e-06,
3.9325e-06,  3.9467e-06,  3.8859e-06,  3.7455e-06,  4.0793e-06,
4.1167e-06,  4.0820e-06,  4.1526e-06,  3.9281e-06,  3.9186e-06,
3.9557e-06,  4.2152e-06,  4.1970e-06,  4.0435e-06,  4.1925e-06,
3.8804e-06,  4.2014e-06,  3.7602e-06,  3.9796e-06,  3.9109e-06,
4.2126e-06,  4.0743e-06,  3.9792e-06,  4.0492e-06,  4.1934e-06,
4.2104e-06,  4.2075e-06,  4.0156e-06,  4.1513e-06,  4.2077e-06,
4.2112e-06,  3.8668e-06,  4.2120e-06,  3.8833e-06,  3.8556e-06,
3.9284e-06,  4.0647e-06,  3.9570e-06,  4.2015e-06,  3.9930e-06,
4.2096e-06,  3.9648e-06,  4.2052e-06,  4.1366e-06,  3.7907e-06,
4.1753e-06,  3.9389e-06,  3.9049e-06,  4.0244e-06,  3.7655e-06,
3.8053e-06,  4.1892e-06,  4.1904e-06,  4.2092e-06,  4.1691e-06,
3.9431e-06,  4.2084e-06,  4.1258e-06,  4.2015e-06,  4.2017e-06,
4.0692e-06,  3.8543e-06,  4.1134e-06,  3.9642e-06,  4.0724e-06,
3.9828e-06,  4.1216e-06,  3.9204e-06,  4.1786e-06,  4.0625e-06,
4.1966e-06,  4.2026e-06,  4.0820e-06,  4.0790e-06,  4.0867e-06,
4.0552e-06,  4.1722e-06,  4.1316e-06,  4.1576e-06,  3.9750e-06,
3.9455e-06,  3.9970e-06,  4.1902e-06,  4.1869e-06,  4.2170e-06,
3.9852e-06,  4.1769e-06,  4.1282e-06,  4.1161e-06,  4.1496e-06,
4.2136e-06,  4.1248e-06,  4.2129e-06,  4.0647e-06,  4.0817e-06,
3.8375e-06,  4.0383e-06,  4.0707e-06,  4.1822e-06,  4.0055e-06,
4.1544e-06,  4.2137e-06,  4.0411e-06,  4.1721e-06,  4.2194e-06,
3.8863e-06,  4.1057e-06,  4.1850e-06,  4.1972e-06,  4.0313e-06,
4.1616e-06,  4.0179e-06,  4.1502e-06,  3.8897e-06,  3.9803e-06,
4.1077e-06,  4.1980e-06,  4.2004e-06,  4.2020e-06,  3.9655e-06,
4.1277e-06,  3.8813e-06,  4.1915e-06,  4.1690e-06,  3.9482e-06,
4.1567e-06,  3.9769e-06,  4.1818e-06,  4.1106e-06,  3.8565e-06,
4.0836e-06,  3.9353e-06,  3.7672e-06,  3.8841e-06,  3.8104e-06,
3.8425e-06,  4.0010e-06,  4.2152e-06,  3.9960e-06,  4.2158e-06,
4.1985e-06,  3.8729e-06,  3.7924e-06,  4.2058e-06,  3.8808e-06,
4.1048e-06,  4.0770e-06,  4.2137e-06,  4.1643e-06,  4.1088e-06,
4.1158e-06,  4.0444e-06,  4.1818e-06,  4.2097e-06,  4.0559e-06,
4.0486e-06,  3.9365e-06,  4.2136e-06,  4.0715e-06,  3.9007e-06,
3.8334e-06,  3.8956e-06,  3.7581e-06,  4.0317e-06,  4.2028e-06,
4.0336e-06,  3.8619e-06,  4.1736e-06,  4.0891e-06,  3.8431e-06,
```

```
4.1830e-06,  4.1992e-06,  3.9375e-06,  3.8933e-06,  4.1313e-06,
4.2137e-06,  3.9266e-06,  4.1882e-06,  3.9909e-06,  4.1799e-06,
3.7581e-06,  4.0540e-06,  3.8066e-06,  4.0354e-06,  4.0072e-06,
3.8286e-06,  3.8733e-06,  3.9147e-06,  4.2096e-06,  4.1610e-06,
4.0749e-06,  4.1368e-06,  4.1424e-06,  4.2084e-06,  4.0695e-06,
4.0824e-06,  4.1713e-06,  3.7891e-06,  4.1667e-06,  4.1996e-06,
3.8479e-06,  4.1098e-06,  3.9044e-06,  3.7369e-06,  3.9229e-06,
3.7695e-06,  4.0185e-06,  4.1064e-06,  4.0986e-06,  4.2125e-06,
4.0139e-06,  4.2083e-06,  3.8992e-06,  3.8761e-06,  4.1947e-06,
4.0627e-06,  4.1464e-06,  3.9041e-06,  3.9744e-06,  4.2071e-06,
4.1818e-06,  3.9611e-06,  4.0432e-06,  4.0329e-06,  4.0808e-06,
4.1771e-06,  4.1192e-06,  4.1966e-06,  4.1488e-06,  4.1569e-06,
4.2059e-06,  3.9635e-06,  3.9296e-06,  4.2038e-06,  4.2063e-06,
4.0138e-06,  4.1273e-06,  3.7666e-06,  3.7979e-06,  4.2136e-06,
3.9710e-06,  3.8801e-06,  4.0406e-06,  3.9470e-06,  4.1350e-06,
4.0513e-06,  4.2060e-06,  4.1322e-06,  4.0052e-06,  3.9697e-06,
3.8399e-06,  4.1010e-06,  4.0020e-06,  4.1870e-06,  4.1058e-06,
4.2012e-06,  4.2114e-06,  4.1850e-06,  4.1453e-06,  4.0774e-06,
4.2167e-06,  4.1869e-06,  4.2207e-06,  4.2083e-06,  4.2060e-06,
4.1660e-06,  3.9194e-06,  3.8646e-06,  4.1016e-06,  4.1820e-06,
3.8861e-06,  3.9014e-06,  4.1530e-06,  3.9296e-06,  4.1615e-06,
4.0778e-06,  4.1308e-06,  4.2055e-06,  3.9269e-06,  4.0093e-06,
4.0965e-06,  4.0962e-06,  4.2061e-06,  4.1248e-06,  4.1825e-06,
4.0622e-06,  4.1487e-06,  4.1404e-06,  3.7860e-06,  3.8532e-06,
4.2068e-06,  4.1815e-06,  4.1272e-06,  4.1973e-06,  3.7786e-06,
4.0531e-06,  4.2011e-06,  3.9253e-06,  4.1741e-06,  4.0365e-06,
3.9321e-06,  4.0916e-06,  4.2194e-06,  3.8463e-06,  3.9142e-06,
4.1654e-06,  4.0083e-06,  4.0798e-06,  4.1803e-06,  4.1783e-06,
4.1324e-06,  4.0696e-06,  4.0773e-06,  4.1068e-06,  4.1997e-06,
4.2160e-06,  3.8892e-06,  4.1642e-06,  4.2130e-06,  3.9800e-06,
4.0710e-06,  4.2197e-06,  4.2010e-06,  4.1796e-06,  4.0977e-06,
4.2240e-06,  3.8535e-06,  3.9863e-06,  3.9128e-06,  4.1362e-06,
4.0780e-06,  3.9698e-06,  4.1724e-06,  4.1548e-06,  4.1623e-06,
4.1949e-06,  4.0195e-06,  4.0747e-06,  4.1347e-06,  4.1350e-06,
4.1317e-06,  4.1646e-06,  4.1727e-06,  3.9744e-06,  4.2195e-06,
4.2028e-06,  4.1243e-06,  4.1199e-06,  4.1397e-06,  3.9577e-06,
3.9522e-06,  3.8979e-06,  4.2138e-06,  4.2169e-06,  3.9493e-06,
3.9882e-06,  4.1669e-06,  4.1295e-06,  4.2043e-06,  4.0466e-06,
4.0950e-06,  4.1843e-06,  4.1748e-06,  4.0247e-06,  3.7458e-06,
4.2071e-06,  4.2086e-06,  4.1841e-06,  4.1142e-06,  4.2071e-06,
4.0259e-06,  4.1822e-06,  3.8984e-06,  4.1836e-06,  4.0543e-06,
4.0220e-06,  4.1925e-06,  4.0807e-06,  4.1993e-06,  4.0673e-06,
4.1125e-06,  4.1017e-06,  4.2142e-06,  3.9492e-06,  4.1617e-06,
3.9875e-06,  4.2111e-06,  4.1113e-06,  3.9674e-06,  4.0205e-06,
3.9049e-06,  3.9504e-06,  4.2041e-06,  4.2051e-06,  4.1417e-06,
4.1800e-06,  4.0214e-06,  4.1996e-06,  4.0866e-06,  4.1795e-06,
4.1746e-06,  4.1708e-06,  4.2126e-06,  4.1994e-06,  4.0267e-06,
3.8478e-06,  3.7290e-06,  4.1560e-06,  4.1867e-06,  4.1489e-06,
```

```
4.2193e-06,  3.9711e-06,  3.7516e-06,  4.0663e-06,  4.1638e-06,
4.2010e-06,  4.1826e-06,  4.1701e-06,  4.0892e-06,  4.2149e-06,
4.1200e-06,  4.2100e-06,  3.8808e-06,  3.9045e-06,  4.1393e-06,
4.1835e-06,  3.7623e-06,  4.1357e-06,  4.2102e-06,  4.1599e-06,
3.9215e-06,  4.1539e-06,  4.1747e-06,  3.7581e-06,  4.2090e-06,
3.8063e-06,  4.1997e-06,  3.9709e-06,  4.1282e-06,  3.8874e-06,
4.0582e-06,  4.2001e-06,  4.2167e-06,  4.0730e-06,  4.0575e-06,
4.1448e-06,  4.2003e-06,  3.9638e-06,  3.8895e-06,  4.1068e-06,
4.0602e-06,  4.1451e-06,  4.0834e-06,  4.0869e-06,  4.0786e-06,
4.2145e-06,  4.1250e-06,  4.1376e-06,  4.1666e-06,  4.2090e-06,
4.1799e-06,  3.9549e-06,  4.0616e-06,  4.1940e-06,  4.0419e-06,
4.0807e-06,  4.0414e-06,  4.2144e-06,  4.1354e-06,  4.1987e-06,
3.9229e-06,  4.1886e-06,  4.2050e-06,  4.1952e-06,  4.0202e-06,
4.1500e-06,  4.1858e-06,  3.9373e-06,  4.1712e-06,  3.9844e-06,
3.7422e-06,  3.7606e-06,  4.1894e-06,  3.9458e-06,  4.1680e-06,
3.7529e-06,  4.1105e-06,  4.1271e-06,  4.1965e-06,  3.9238e-06,
3.9375e-06,  4.1361e-06,  4.2073e-06,  4.1899e-06,  4.1627e-06,
3.9535e-06,  4.0910e-06,  4.0097e-06,  3.7019e-06,  4.0921e-06,
4.1230e-06,  3.8567e-06,  4.0909e-06,  4.2095e-06,  4.2049e-06,
3.9399e-06,  4.2092e-06,  3.8356e-06,  3.9307e-06,  4.2038e-06,
4.1707e-06,  4.0849e-06,  4.0684e-06,  4.1049e-06,  4.1403e-06,
4.1977e-06,  4.1051e-06,  4.0558e-06,  4.1849e-06,  4.0642e-06,
3.8749e-06,  4.2196e-06,  3.8381e-06,  4.2084e-06,  4.1728e-06,
4.2144e-06,  3.9585e-06,  4.1643e-06,  3.7441e-06,  4.0976e-06,
4.2095e-06,  4.0224e-06,  4.2091e-06,  4.2109e-06,  4.2121e-06,
3.7510e-06,  4.0521e-06,  3.9191e-06,  4.1989e-06,  4.0736e-06,
3.8859e-06,  3.9363e-06,  3.9955e-06,  4.0139e-06,  4.1711e-06,
4.1836e-06,  4.1846e-06,  3.8672e-06,  4.1047e-06,  3.9291e-06,
3.9536e-06,  4.2140e-06,  4.2104e-06,  4.2161e-06,  4.0559e-06,
4.2002e-06,  4.1137e-06,  4.1571e-06,  4.1380e-06,  4.1365e-06,
4.2045e-06,  3.9697e-06,  4.1798e-06,  4.1196e-06,  4.2186e-06,
4.0649e-06,  4.0783e-06,  3.8668e-06,  4.0723e-06,  3.8489e-06,
3.8660e-06,  4.1743e-06,  4.1225e-06,  4.0322e-06,  4.0671e-06,
3.9292e-06,  3.8949e-06,  4.2155e-06,  4.2047e-06,  4.2214e-06,
3.9932e-06,  3.7418e-06,  4.0233e-06,  4.1848e-06,  3.9174e-06,
4.1935e-06,  4.1616e-06,  3.8984e-06,  3.9934e-06,  3.7835e-06,
3.9732e-06,  4.0893e-06,  3.9845e-06,  4.1397e-06,  3.9503e-06,
4.0706e-06,  4.2011e-06,  4.0552e-06,  3.9253e-06,  4.0389e-06,
4.1361e-06,  3.9433e-06,  3.8077e-06,  4.1921e-06,  4.1134e-06,
4.0484e-06,  4.1047e-06,  4.2010e-06,  3.7184e-06,  3.9517e-06,
4.0632e-06,  4.2074e-06,  4.1392e-06,  3.9463e-06,  4.1354e-06,
3.6535e-06,  3.8130e-06,  4.1637e-06,  4.0431e-06,  4.2005e-06,
4.2018e-06,  3.8669e-06,  4.1688e-06,  4.1613e-06,  4.0464e-06,
4.2018e-06,  4.2126e-06,  4.0598e-06,  4.1457e-06,  4.1965e-06,
4.1993e-06,  4.0460e-06,  3.8090e-06,  4.1929e-06,  4.0846e-06,
4.1835e-06,  4.1144e-06,  4.2102e-06,  4.1451e-06,  4.1929e-06,
3.8523e-06,  4.1728e-06,  4.1434e-06,  3.8138e-06,  3.8398e-06,
4.2151e-06,  4.1721e-06,  4.1205e-06,  4.1270e-06,  4.1885e-06,
```

```
          3.8927e-06,  4.1950e-06,  4.2217e-06,  4.2155e-06,  4.1151e-06,
          4.0782e-06,  4.0396e-06,  4.2020e-06,  3.7490e-06,  4.1872e-06,
          4.2023e-06,  4.1845e-06,  3.8674e-06,  3.8625e-06,  4.0439e-06,
          4.1459e-06,  3.9762e-06,  4.2012e-06,  4.1077e-06,  3.8974e-06,
          4.2143e-06,  4.1873e-06,  4.1698e-06,  4.0867e-06,  3.7348e-06,
          3.9604e-06,  3.8887e-06,  4.0893e-06,  3.9951e-06,  4.1578e-06,
          3.9089e-06,  4.1365e-06,  4.2017e-06,  4.1423e-06,  4.1473e-06,
          4.1986e-06,  3.8231e-06,  4.1732e-06,  3.8007e-06,  4.0566e-06,
          3.7213e-06,  3.8748e-06,  4.2105e-06,  3.6966e-06,  4.1496e-06,
          4.0925e-06,  4.1771e-06,  3.8922e-06,  3.8992e-06,  4.1668e-06,
          4.2041e-06,  3.9658e-06,  4.0824e-06,  4.2151e-06,  3.9466e-06,
          4.0956e-06,  4.1197e-06,  4.0936e-06,  4.1488e-06,  3.8735e-06,
          4.0831e-06,  4.1627e-06,  4.0971e-06,  4.0251e-06,  4.2034e-06,
          4.1818e-06,  4.0340e-06,  3.7924e-06,  4.1203e-06,  3.8334e-06,
          3.7587e-06,  4.1273e-06,  4.1883e-06,  4.2068e-06,  4.2115e-06,
          4.1636e-06,  4.1361e-06,  3.9284e-06,  3.9734e-06,  4.0012e-06,
          4.0570e-06,  3.9690e-06,  4.2178e-06,  3.9334e-06,  3.7109e-06,
          4.0174e-06,  4.1389e-06,  4.0539e-06,  3.9251e-06,  3.9280e-06,
          3.9211e-06,  4.0219e-06,  3.8100e-06,  4.2187e-06,  4.2192e-06,
          3.8116e-06,  3.8744e-06,  3.8746e-06,  3.9309e-06,  4.1704e-06,
          4.0161e-06,  3.7109e-06,  4.1012e-06,  4.2057e-06,  4.1275e-06,
          4.2034e-06,  3.9438e-06,  4.1887e-06,  3.8228e-06,  3.8601e-06,
          3.8859e-06,  3.9369e-06,  3.8847e-06,  3.9653e-06,  3.8809e-06,
          3.9082e-06,  4.1433e-06,  4.1401e-06,  4.1618e-06,  4.1147e-06,
          3.9400e-06,  3.7486e-06,  4.0778e-06,  4.0921e-06,  4.1598e-06,
          4.1187e-06,  4.2181e-06,  4.2047e-06,  4.1380e-06,  4.2095e-06,
          3.7839e-06,  3.9853e-06,  3.9578e-06,  3.9315e-06,  3.7437e-06,
          4.0294e-06,  4.1552e-06,  4.0782e-06,  3.9485e-06,  4.0733e-06,
          4.0911e-06,  3.9412e-06,  4.1645e-06,  3.8817e-06,  3.9225e-06,
          3.9279e-06,  3.7828e-06,  4.2100e-06,  4.2111e-06,  4.1461e-06,
          4.2180e-06,  4.1321e-06,  4.1295e-06,  4.0886e-06,  4.1325e-06,
          3.9824e-06,  4.0133e-06,  3.8695e-06,  3.7954e-06,  4.1511e-06,
          4.1904e-06,  4.1003e-06,  4.1902e-06,  4.2119e-06,  3.8508e-06,
          4.2148e-06,  4.2040e-06,  3.7389e-06,  4.1922e-06,  4.0628e-06,
          3.8865e-06,  4.1933e-06,  4.1258e-06,  4.0957e-06,  4.2042e-06,
          4.2063e-06,  3.9374e-06,  3.9457e-06,  4.0827e-06,  3.6244e-06])
end of p.grad

False
Epoch 4 finished
Epoch [4/10], Loss: 1.0884
************************************************************

Learning rate: 0.001

another way to print learning rate:
tensor([[[[ 0.0051,  0.0051,  0.0047,  …,  0.0023,  0.0025,  0.0021],
          [ 0.0052,  0.0048,  0.0047,  …,  0.0028,  0.0028,  0.0028],
```

```
  [ 0.0050,  0.0049,  0.0051,  …,  0.0035,  0.0036,  0.0030],
  …,
  [ 0.0038,  0.0037,  0.0037,  …,  0.0027,  0.0029,  0.0020],
  [ 0.0035,  0.0036,  0.0037,  …,  0.0032,  0.0029,  0.0022],
  [ 0.0030,  0.0033,  0.0039,  …,  0.0028,  0.0021,  0.0017]],

 [[ 0.0058,  0.0051,  0.0044,  …,  0.0014,  0.0015,  0.0011],
  [ 0.0057,  0.0048,  0.0043,  …,  0.0019,  0.0019,  0.0019],
  [ 0.0054,  0.0048,  0.0047,  …,  0.0026,  0.0026,  0.0024],
  …,
  [ 0.0048,  0.0045,  0.0047,  …,  0.0035,  0.0035,  0.0028],
  [ 0.0049,  0.0047,  0.0048,  …,  0.0041,  0.0036,  0.0031],
  [ 0.0042,  0.0042,  0.0049,  …,  0.0039,  0.0030,  0.0027]],

 [[ 0.0058,  0.0052,  0.0047,  …,  0.0018,  0.0018,  0.0014],
  [ 0.0057,  0.0050,  0.0048,  …,  0.0022,  0.0021,  0.0021],
  [ 0.0056,  0.0050,  0.0053,  …,  0.0029,  0.0030,  0.0026],
  …,
  [ 0.0034,  0.0028,  0.0029,  …,  0.0017,  0.0018,  0.0012],
  [ 0.0037,  0.0031,  0.0032,  …,  0.0025,  0.0021,  0.0018],
  [ 0.0034,  0.0031,  0.0037,  …,  0.0026,  0.0016,  0.0017]]],


[[[ 0.0242,  0.0273,  0.0285,  …,  0.0185,  0.0178,  0.0173],
  [ 0.0243,  0.0280,  0.0273,  …,  0.0164,  0.0176,  0.0172],
  [ 0.0236,  0.0256,  0.0253,  …,  0.0136,  0.0151,  0.0168],
  …,
  [ 0.0100,  0.0089,  0.0085,  …,  0.0018,  0.0009,  0.0021],
  [ 0.0110,  0.0108,  0.0091,  …,  0.0009,  0.0026,  0.0048],
  [ 0.0132,  0.0119,  0.0100,  …,  0.0039,  0.0060,  0.0108]],

 [[-0.0086, -0.0055, -0.0038,  …, -0.0083, -0.0093, -0.0077],
  [-0.0091, -0.0043, -0.0049,  …, -0.0101, -0.0088, -0.0071],
  [-0.0112, -0.0081, -0.0078,  …, -0.0124, -0.0102, -0.0073],
  …,
  [-0.0253, -0.0252, -0.0229,  …, -0.0207, -0.0221, -0.0195],
  [-0.0236, -0.0210, -0.0190,  …, -0.0207, -0.0196, -0.0171],
  [-0.0172, -0.0160, -0.0148,  …, -0.0171, -0.0159, -0.0113]],

 [[ 0.0002,  0.0027,  0.0028,  …,  0.0012, -0.0002, -0.0003],
  [-0.0009,  0.0033,  0.0019,  …, -0.0003,  0.0004,  0.0004],
  [-0.0046, -0.0014, -0.0015,  …, -0.0035, -0.0012,  0.0019],
  …,
  [-0.0164, -0.0160, -0.0145,  …, -0.0121, -0.0132, -0.0108],
  [-0.0154, -0.0122, -0.0114,  …, -0.0122, -0.0109, -0.0087],
  [-0.0092, -0.0093, -0.0092,  …, -0.0090, -0.0076, -0.0038]]],
```

```
[[[ 0.0145,  0.0133,  0.0123,  …,  0.0143,  0.0127,  0.0118],
  [ 0.0143,  0.0141,  0.0129,  …,  0.0133,  0.0127,  0.0120],
  [ 0.0130,  0.0127,  0.0122,  …,  0.0121,  0.0117,  0.0107],
  …,
  [ 0.0101,  0.0109,  0.0107,  …,  0.0060,  0.0064,  0.0059],
  [ 0.0124,  0.0123,  0.0125,  …,  0.0062,  0.0072,  0.0064],
  [ 0.0149,  0.0149,  0.0149,  …,  0.0076,  0.0089,  0.0076]],

 [[-0.0036, -0.0041, -0.0045,  …, -0.0020, -0.0039, -0.0049],
  [-0.0044, -0.0036, -0.0044,  …, -0.0035, -0.0044, -0.0052],
  [-0.0056, -0.0051, -0.0050,  …, -0.0050, -0.0056, -0.0063],
  …,
  [-0.0064, -0.0066, -0.0075,  …, -0.0116, -0.0116, -0.0118],
  [-0.0035, -0.0044, -0.0048,  …, -0.0112, -0.0101, -0.0103],
  [-0.0012, -0.0015, -0.0018,  …, -0.0090, -0.0076, -0.0083]],

 [[ 0.0061,  0.0058,  0.0052,  …,  0.0075,  0.0061,  0.0051],
  [ 0.0054,  0.0062,  0.0055,  …,  0.0063,  0.0056,  0.0045],
  [ 0.0046,  0.0051,  0.0050,  …,  0.0047,  0.0043,  0.0032],
  …,
  [ 0.0051,  0.0050,  0.0039,  …, -0.0009, -0.0008, -0.0013],
  [ 0.0062,  0.0055,  0.0052,  …, -0.0015, -0.0004, -0.0011],
  [ 0.0080,  0.0077,  0.0073,  …,  0.0003,  0.0016,  0.0007]]],


…,


[[[-0.0010, -0.0020, -0.0009,  …,  0.0044,  0.0048,  0.0051],
  [-0.0013, -0.0019, -0.0007,  …,  0.0040,  0.0043,  0.0049],
  [-0.0020, -0.0016, -0.0009,  …,  0.0028,  0.0028,  0.0036],
  …,
  [-0.0020, -0.0020, -0.0017,  …, -0.0039, -0.0039, -0.0033],
  [-0.0003, -0.0014, -0.0011,  …, -0.0034, -0.0035, -0.0028],
  [ 0.0018,  0.0012,  0.0003,  …, -0.0021, -0.0014, -0.0010]],

 [[-0.0003, -0.0018, -0.0017,  …,  0.0036,  0.0045,  0.0051],
  [-0.0014, -0.0028, -0.0019,  …,  0.0029,  0.0035,  0.0044],
  [-0.0034, -0.0034, -0.0027,  …,  0.0017,  0.0017,  0.0028],
  …,
  [-0.0032, -0.0034, -0.0031,  …, -0.0050, -0.0048, -0.0043],
  [-0.0004, -0.0019, -0.0017,  …, -0.0036, -0.0034, -0.0025],
  [ 0.0021,  0.0011,  0.0001,  …, -0.0017, -0.0008, -0.0004]],

 [[-0.0030, -0.0042, -0.0038,  …,  0.0019,  0.0030,  0.0038],
  [-0.0044, -0.0054, -0.0043,  …,  0.0015,  0.0021,  0.0029],
  [-0.0060, -0.0058, -0.0050,  …,  0.0004,  0.0006,  0.0016],
  …,
```

```
  [-0.0056, -0.0057, -0.0055,  …, -0.0075, -0.0071, -0.0064],
  [-0.0033, -0.0049, -0.0048,  …, -0.0067, -0.0066, -0.0059],
  [-0.0012, -0.0024, -0.0034,  …, -0.0051, -0.0043, -0.0037]]],


[[[ 0.0073,  0.0075,  0.0080,  …,  0.0082,  0.0074,  0.0071],
  [ 0.0083,  0.0089,  0.0097,  …,  0.0087,  0.0081,  0.0078],
  [ 0.0090,  0.0099,  0.0106,  …,  0.0092,  0.0090,  0.0084],
  …,
  [ 0.0069,  0.0071,  0.0071,  …,  0.0072,  0.0067,  0.0063],
  [ 0.0058,  0.0059,  0.0064,  …,  0.0062,  0.0056,  0.0054],
  [ 0.0070,  0.0070,  0.0069,  …,  0.0072,  0.0069,  0.0065]],

 [[ 0.0086,  0.0090,  0.0094,  …,  0.0089,  0.0079,  0.0077],
  [ 0.0098,  0.0103,  0.0111,  …,  0.0095,  0.0086,  0.0086],
  [ 0.0106,  0.0111,  0.0118,  …,  0.0099,  0.0097,  0.0094],
  …,
  [ 0.0075,  0.0077,  0.0073,  …,  0.0083,  0.0078,  0.0074],
  [ 0.0064,  0.0064,  0.0067,  …,  0.0073,  0.0066,  0.0064],
  [ 0.0065,  0.0064,  0.0065,  …,  0.0073,  0.0068,  0.0063]],

 [[ 0.0059,  0.0064,  0.0071,  …,  0.0063,  0.0057,  0.0057],
  [ 0.0073,  0.0080,  0.0091,  …,  0.0070,  0.0063,  0.0067],
  [ 0.0083,  0.0090,  0.0098,  …,  0.0075,  0.0073,  0.0074],
  …,
  [ 0.0052,  0.0051,  0.0048,  …,  0.0056,  0.0055,  0.0058],
  [ 0.0040,  0.0038,  0.0042,  …,  0.0053,  0.0051,  0.0054],
  [ 0.0042,  0.0040,  0.0040,  …,  0.0057,  0.0057,  0.0056]]],


[[[-0.0201, -0.0227, -0.0242,  …, -0.0169, -0.0178, -0.0166],
  [-0.0208, -0.0226, -0.0219,  …, -0.0179, -0.0193, -0.0170],
  [-0.0201, -0.0214, -0.0225,  …, -0.0199, -0.0198, -0.0171],
  …,
  [-0.0245, -0.0215, -0.0190,  …, -0.0204, -0.0197, -0.0182],
  [-0.0223, -0.0210, -0.0195,  …, -0.0188, -0.0183, -0.0159],
  [-0.0176, -0.0168, -0.0176,  …, -0.0140, -0.0138, -0.0130]],

 [[-0.0090, -0.0121, -0.0145,  …, -0.0049, -0.0054, -0.0042],
  [-0.0110, -0.0138, -0.0132,  …, -0.0075, -0.0089, -0.0059],
  [-0.0103, -0.0118, -0.0137,  …, -0.0099, -0.0100, -0.0072],
  …,
  [-0.0147, -0.0119, -0.0101,  …, -0.0144, -0.0141, -0.0120],
  [-0.0129, -0.0121, -0.0107,  …, -0.0134, -0.0131, -0.0094],
  [-0.0099, -0.0096, -0.0094,  …, -0.0098, -0.0098, -0.0082]],

 [[-0.0127, -0.0153, -0.0172,  …, -0.0087, -0.0084, -0.0067],
  [-0.0151, -0.0175, -0.0156,  …, -0.0085, -0.0091, -0.0065],
```

```
          [-0.0141, -0.0154, -0.0149,  …, -0.0084, -0.0083, -0.0064],
          …,
          [-0.0147, -0.0112, -0.0091,  …, -0.0148, -0.0142, -0.0129],
          [-0.0129, -0.0116, -0.0099,  …, -0.0149, -0.0148, -0.0118],
          [-0.0098, -0.0086, -0.0085,  …, -0.0126, -0.0124, -0.0110]]])
tensor([-2.1420e-08,  9.9652e-08,  1.3504e-08, -9.7789e-09,  1.1176e-08,
         0.0000e+00,  5.0175e-08, -1.2014e-07, -4.4238e-09, -5.2154e-08,
        -1.0245e-08, -1.0943e-08,  5.5879e-09, -2.0955e-09,  2.8871e-08,
         6.1700e-09, -1.8626e-08, -2.8813e-08,  7.6368e-08,  9.3132e-10,
        -1.0827e-08,  3.0734e-08, -1.6019e-07, -1.4901e-08,  6.9849e-08,
        -1.0245e-08,  1.7535e-09, -4.6566e-10, -1.7695e-08,  9.3132e-08,
         9.1386e-09, -1.4901e-08, -3.4343e-08,  1.9209e-09,  9.3132e-10,
        -6.2282e-09, -1.4901e-08, -1.5832e-08,  1.8626e-08,  6.9849e-09,
         1.1176e-08, -6.0536e-09,  2.9220e-08,  2.3399e-08, -2.7940e-08,
         4.6100e-08, -6.0536e-09,  5.5879e-09, -2.3283e-10, -7.4506e-09,
         1.6589e-09,  1.7695e-08,  3.3993e-08, -1.7812e-08,  4.5169e-08,
         2.6077e-08, -1.5087e-07, -6.9849e-10, -4.4238e-09,  1.2806e-08,
        -7.7998e-09, -6.8103e-09, -2.2352e-08, -9.5461e-09, -6.8103e-09,
        -3.6205e-08, -2.3283e-09, -1.4901e-08, -1.2689e-08, -1.7928e-08,
        -1.1642e-10, -5.5879e-09, -4.8662e-08, -2.9802e-08,  6.6124e-08,
        -5.2620e-08, -2.8522e-09,  2.7707e-08, -3.3993e-08,  3.0268e-08,
        -1.9558e-07, -1.7695e-08, -4.6566e-09,  8.8476e-09, -2.1886e-08,
         1.7695e-08, -4.0396e-08,  1.8626e-09, -6.6357e-09,  4.7497e-08,
         8.1491e-09,  3.0268e-08, -2.4214e-07,  2.1246e-08,  4.5602e-09,
         2.1420e-08])
tensor([-0.0035,  0.0040,  0.0106,  0.0008, -0.0287,  0.0918,  0.0082,  0.1092,
         0.0023, -0.0845, -0.0866, -0.0048,  0.0112, -0.0054,  0.0085, -0.0017,
        -0.1020, -0.0295,  0.0357,  0.0051,  0.0002,  0.0273,  0.0016,  0.0187,
        -0.0923, -0.0039,  0.0015,  0.0116, -0.0247,  0.0828, -0.0087,  0.0166,
         0.0100, -0.0004, -0.0146,  0.0021, -0.0181,  0.0054, -0.0313, -0.0037,
        -0.0370,  0.0085,  0.0167, -0.0089,  0.0255,  0.0163,  0.0121,  0.0081,
        -0.0146, -0.0015,  0.0063,  0.1169,  0.0127, -0.0169,  0.0014, -0.0857,
         0.0757, -0.0029, -0.0028,  0.0094,  0.0019,  0.0018, -0.0650,  0.0011,
         0.0123,  0.0103,  0.0083,  0.0533, -0.0112,  0.0188,  0.0025,  0.0170,
        -0.0111, -0.0252, -0.0154, -0.0723,  0.0017, -0.0014,  0.0012,  0.0007,
         0.0449,  0.0418, -0.0019,  0.0014, -0.0946, -0.0377,  0.0026,  0.0051,
        -0.0031,  0.0111, -0.0074,  0.1123,  0.0183,  0.0086, -0.0085, -0.0137])
tensor([-2.4101e-04,  7.6078e-03, -1.8764e-03,  1.1278e-02, -2.5839e-02,
         3.6011e-02,  3.5148e-03,  3.7290e-02, -9.5083e-03, -2.3053e-02,
        -2.9575e-02,  1.3114e-03, -8.2595e-03,  1.2430e-02,  5.4343e-03,
         8.0349e-03, -2.1923e-02, -1.6832e-02, -6.0967e-03,  4.5207e-04,
         7.7684e-03,  1.6801e-03, -5.7458e-02,  1.0151e-02, -2.0543e-02,
        -1.5493e-02,  9.5506e-03, -7.2157e-05,  4.7876e-03,  2.6197e-02,
        -4.4610e-03, -1.1668e-02, -3.9636e-03,  8.7093e-03, -1.2696e-02,
         7.8094e-03, -1.6107e-02, -1.2891e-02, -1.9975e-02, -7.0019e-04,
        -1.0156e-02, -2.4929e-03, -2.3721e-04, -1.8817e-03,  5.9428e-03,
         1.9012e-03,  1.3104e-05,  1.3794e-03,  1.7336e-02, -4.7156e-03,
         1.0683e-02,  2.4205e-02, -4.5270e-04,  1.6682e-02, -4.2804e-03,
```

```
        -4.7899e-02,  3.6842e-03,  8.4358e-03,  3.3481e-03, -2.2380e-03,
         8.0242e-03,  9.0516e-03, -2.7822e-02,  8.9244e-03,  1.6205e-02,
         1.8740e-03,  5.8701e-03, -4.1897e-02, -2.8477e-03,  6.4618e-03,
         1.1753e-02,  2.9712e-03,  1.7332e-02, -1.9559e-02, -7.8804e-03,
        -2.0082e-02,  9.0529e-03,  1.1006e-02, -2.0477e-03, -3.9696e-03,
         8.0793e-03,  1.0033e-02,  3.7940e-03, -1.0285e-03, -4.4924e-02,
        -2.5954e-02,  6.7244e-03,  1.3748e-02,  3.6963e-03, -8.2636e-03,
        -8.7903e-03,  2.6713e-02, -8.8171e-03,  8.5329e-03,  3.8649e-03,
        -9.2919e-03])
tensor([[[[ 2.8999e-03,  2.6395e-03,  2.8255e-03,  2.6432e-03,  2.3889e-03],
          [ 2.4713e-03,  2.4200e-03,  2.5881e-03,  2.4989e-03,  1.8995e-03],
          [ 1.6655e-03,  1.8056e-03,  1.9924e-03,  1.8649e-03,  1.5966e-03],
          [ 7.7695e-04,  1.2346e-03,  1.4369e-03,  1.1949e-03,  9.4914e-04],
          [ 5.5010e-04,  1.2017e-03,  1.6309e-03,  1.3011e-03,  8.5062e-04]],

         [[-1.2566e-03, -2.2225e-03, -1.3458e-03, -1.9318e-05,  4.9955e-04],
          [-1.2447e-03, -1.1117e-03, -1.3975e-03,  1.1912e-04,  5.9220e-04],
          [-1.4544e-03, -1.7718e-03, -1.6789e-03, -8.1651e-04, -4.6985e-04],
          [-1.7523e-03, -2.2624e-03, -1.8936e-03, -1.8142e-03, -2.0286e-03],
          [-3.3759e-03, -3.7669e-03, -2.3473e-03, -1.9859e-03, -2.3981e-03]],

         [[-1.8175e-03, -2.3662e-03, -2.5101e-03, -2.5405e-03, -2.2718e-03],
          [-2.0196e-03, -2.3438e-03, -2.2453e-03, -2.7575e-03, -2.8251e-03],
          [-2.0469e-03, -2.5373e-03, -2.4926e-03, -3.0156e-03, -3.3389e-03],
          [-2.7373e-03, -2.8972e-03, -2.8180e-03, -3.2933e-03, -3.4150e-03],
          [-2.5178e-03, -2.8985e-03, -2.6972e-03, -2.7571e-03, -2.7771e-03]],

         …,

         [[ 3.3362e-03,  3.1164e-03,  3.1448e-03,  3.1303e-03,  2.9939e-03],
          [ 3.2138e-03,  3.1969e-03,  3.3202e-03,  2.9956e-03,  2.8187e-03],
          [ 3.3521e-03,  3.3902e-03,  3.6223e-03,  3.4948e-03,  2.8613e-03],
          [ 2.3996e-03,  2.6163e-03,  3.0156e-03,  2.7274e-03,  2.4358e-03],
          [ 2.5756e-03,  2.7896e-03,  2.9642e-03,  2.8687e-03,  2.5271e-03]],

         [[ 3.5749e-03,  3.5285e-03,  3.5916e-03,  3.4913e-03,  3.4164e-03],
          [ 3.4355e-03,  3.5685e-03,  3.7864e-03,  3.6142e-03,  3.1951e-03],
          [ 2.9124e-03,  3.1900e-03,  3.5054e-03,  3.3417e-03,  2.9151e-03],
          [ 1.9983e-03,  2.2408e-03,  2.4912e-03,  2.3601e-03,  2.0420e-03],
          [ 1.9835e-03,  2.1464e-03,  2.3646e-03,  2.1461e-03,  1.8320e-03]],

         [[-5.2699e-04, -7.1768e-04, -5.1244e-04, -8.7240e-04, -9.1973e-04],
          [-6.6269e-04, -2.5646e-04,  1.1863e-04, -7.3838e-04, -5.1988e-04],
          [-1.1298e-03, -2.2970e-04, -1.4958e-04, -4.9350e-04, -5.9101e-04],
          [-2.1378e-03, -1.2835e-03, -1.2213e-03, -9.4407e-04, -9.7775e-04],
          [-2.4487e-03, -1.9756e-03, -2.1878e-03, -1.6020e-03, -2.0983e-03]]],
```

```
[[[ 2.4259e-05,  5.3580e-04,  8.4906e-04,  8.1946e-04,  9.1564e-04],
  [ 3.8311e-04,  6.5442e-04,  8.3364e-04,  7.6059e-04,  7.1655e-04],
  [ 1.6857e-04,  4.7560e-04,  3.9395e-04,  4.7624e-04,  7.4619e-04],
  [ 5.4674e-04, -3.3052e-05, -1.4615e-04,  3.1684e-04,  5.4403e-04],
  [ 1.7879e-04,  2.0078e-04, -4.0081e-04,  5.8470e-04,  4.8152e-04]],

 [[-5.8898e-04, -1.4558e-03,  6.3175e-04, -1.5010e-04, -5.4792e-04],
  [-1.2760e-03, -1.7312e-03, -4.4219e-05,  5.4797e-05,  8.8829e-05],
  [-2.5955e-03, -2.1222e-03, -5.6596e-04, -2.3882e-04, -5.8814e-04],
  [-2.6672e-03, -2.0413e-03, -8.8111e-04, -7.5402e-04, -8.9288e-04],
  [-1.6895e-03, -1.8509e-03, -1.9956e-03, -1.0908e-03, -9.8667e-04]],

 [[-5.0930e-04, -3.8584e-04, -1.6749e-04,  7.0459e-04,  2.9611e-04],
  [-8.0965e-04, -2.7697e-04, -4.2840e-04,  5.3236e-04,  1.4827e-04],
  [-8.5062e-04, -2.5426e-04, -2.2899e-04,  7.2213e-04,  4.1256e-04],
  [-6.7957e-04, -3.7076e-04, -6.4563e-04,  5.1047e-04,  3.1841e-04],
  [-7.9077e-04, -5.9440e-04, -2.7441e-04,  3.5071e-04,  3.6671e-04]],

 …,

 [[ 1.1934e-04,  2.3320e-04,  5.5581e-04,  7.2478e-04,  5.1682e-04],
  [ 4.4630e-05,  3.5445e-04,  4.9210e-04,  8.6427e-04,  6.3789e-04],
  [ 1.7939e-04,  4.2599e-04,  8.5096e-04,  1.0413e-03,  7.9632e-04],
  [ 3.7793e-04,  6.5410e-04,  4.9592e-04,  8.3529e-04,  6.4173e-04],
  [ 4.3659e-04,  6.2246e-04,  7.0888e-04,  6.8493e-04,  5.4370e-04]],

 [[ 6.1043e-04,  5.2739e-04,  5.2343e-04,  6.3449e-04,  6.2577e-04],
  [ 7.3076e-04,  6.6613e-04,  6.9331e-04,  8.0961e-04,  7.1601e-04],
  [ 8.9806e-04,  7.9961e-04,  8.1749e-04,  9.1506e-04,  8.6114e-04],
  [ 8.1660e-04,  8.1660e-04,  6.6823e-04,  7.9697e-04,  7.7114e-04],
  [ 7.5454e-04,  5.7108e-04,  5.4355e-04,  7.7252e-04,  6.4117e-04]],

 [[-1.1265e-05,  5.7748e-05,  1.4466e-04,  6.4537e-04,  3.4406e-04],
  [-1.7430e-04,  1.9796e-04, -1.6778e-05,  7.9301e-04,  7.6047e-04],
  [-5.3009e-04, -1.3867e-04,  6.9938e-05,  5.7714e-04,  7.3709e-04],
  [-7.1589e-04, -3.0638e-04, -7.7470e-04,  3.6998e-04,  3.9396e-04],
  [-1.2480e-03, -4.8980e-04, -7.0440e-04,  9.1360e-05,  3.4154e-04]]],


[[[ 6.6147e-04, -4.4935e-04, -7.7703e-04, -8.9541e-04, -6.3858e-04],
  [-1.2422e-04, -5.6928e-04, -6.4101e-04, -8.1823e-04, -5.9873e-04],
  [-6.1719e-04, -5.2506e-04, -5.8494e-04, -2.9215e-04, -2.8799e-04],
  [-1.5948e-03, -1.5507e-03, -1.4092e-03, -1.4328e-03, -1.5739e-03],
  [-1.6451e-03, -1.4488e-03, -1.4837e-03, -1.5276e-03, -1.3877e-03]],

 [[ 1.7798e-03,  1.5249e-03,  1.7045e-03, -2.0178e-03, -1.6021e-03],
  [ 2.0842e-03,  2.2949e-04, -6.2717e-04, -1.6140e-03, -1.6493e-03],
  [ 1.5568e-03, -2.5673e-04, -4.9769e-04, -2.4186e-03, -2.4936e-03],
```

```
  [ 3.3961e-03,  1.5862e-03,  1.4634e-03, -1.2938e-03, -1.3645e-03],
  [ 3.0936e-03,  1.7237e-03,  1.4557e-03, -2.4896e-04, -7.4884e-04]],

 [[ 1.9350e-03,  1.4032e-03,  7.5394e-04, -2.5496e-05, -8.0025e-04],
  [ 2.3983e-03,  1.6820e-03,  7.7481e-04, -6.7642e-05, -8.5287e-05],
  [ 2.1744e-03,  1.9369e-03,  9.3934e-04,  5.6282e-05, -7.0123e-04],
  [ 1.4424e-03,  1.2792e-03,  3.4972e-04, -6.7473e-04, -1.5403e-03],
  [ 1.3064e-03,  1.2101e-03,  4.9907e-04, -8.4953e-04, -1.5221e-03]],

 …,

 [[-1.6171e-04, -5.4294e-04, -1.0861e-03, -1.4101e-03, -1.4060e-03],
  [ 4.0726e-04,  1.2370e-04, -5.0205e-04, -8.4003e-04, -7.3686e-04],
  [-2.4076e-04, -3.2570e-04, -1.0700e-03, -1.4208e-03, -1.9079e-03],
  [-6.1992e-04, -9.8015e-04, -1.5338e-03, -1.9060e-03, -2.4451e-03],
  [-5.8120e-04, -9.2168e-04, -1.3985e-03, -1.8828e-03, -2.2367e-03]],

 [[-1.6572e-03, -1.7541e-03, -1.9714e-03, -2.0873e-03, -1.5342e-03],
  [-1.6448e-03, -1.7488e-03, -1.8439e-03, -1.8560e-03, -1.5078e-03],
  [-1.7823e-03, -1.8484e-03, -2.0903e-03, -1.9883e-03, -1.8991e-03],
  [-2.0081e-03, -2.0844e-03, -2.2441e-03, -2.0322e-03, -1.9030e-03],
  [-1.8825e-03, -2.0055e-03, -2.1217e-03, -1.9815e-03, -1.8429e-03]],

 [[ 2.2970e-03,  1.9109e-03,  1.1842e-03,  7.4249e-04,  8.7764e-04],
  [ 3.1751e-03,  2.2659e-03,  1.2119e-03,  1.1081e-03,  1.0487e-03],
  [ 1.0909e-03,  8.3970e-04,  3.6952e-05, -5.0990e-04, -9.1073e-04],
  [ 8.0161e-05,  1.1267e-04, -1.6975e-04, -4.7332e-04, -2.6871e-04],
  [-9.5519e-05, -1.6451e-04, -3.1176e-04, -4.8375e-04, -1.6427e-04]]],


…,


[[[ 2.2021e-03,  1.5582e-03,  1.4027e-03,  8.8972e-04,  7.6800e-04],
  [ 2.2077e-03,  2.2512e-03,  2.1504e-03,  1.4850e-03,  9.9545e-04],
  [ 2.3681e-03,  2.4292e-03,  2.0963e-03,  1.8159e-03,  1.6924e-03],
  [ 1.4565e-03,  1.5032e-03,  1.4548e-03,  1.4413e-03,  1.1847e-03],
  [ 1.5088e-03,  1.3748e-03,  7.9908e-04,  8.9599e-04,  1.2172e-03]],

 [[ 6.0062e-04,  1.2196e-03,  7.0046e-04, -1.4454e-04, -8.2038e-04],
  [ 2.3124e-04,  9.5244e-04,  1.3985e-03,  5.7360e-04,  6.6717e-05],
  [-7.2761e-04,  1.2019e-04,  7.3429e-04,  1.3598e-03,  1.1449e-03],
  [-1.1005e-03, -1.8790e-05,  4.4718e-04,  1.3574e-03,  4.7245e-04],
  [-6.9008e-04, -5.6045e-04,  5.8851e-06,  3.4227e-04, -3.3915e-04]],

 [[ 1.6881e-03,  1.4428e-03,  1.1089e-03,  1.1128e-03,  1.0339e-03],
  [ 1.9364e-03,  1.7913e-03,  1.3845e-03,  1.5880e-03,  1.4182e-03],
  [ 1.8071e-03,  1.7069e-03,  1.2650e-03,  1.2158e-03,  1.2926e-03],
```

```
  [ 1.8747e-03,  1.8108e-03,  1.2840e-03,  1.0715e-03,  1.0153e-03],
  [ 2.2422e-03,  2.5003e-03,  1.8580e-03,  1.7050e-03,  1.1093e-03]],

 …,

 [[ 2.3252e-03,  1.9810e-03,  1.6976e-03,  1.8133e-03,  1.7934e-03],
  [ 2.6727e-03,  2.3293e-03,  1.9164e-03,  2.0312e-03,  1.9034e-03],
  [ 2.6913e-03,  2.3618e-03,  2.0659e-03,  2.0553e-03,  1.8336e-03],
  [ 2.9962e-03,  2.9569e-03,  2.3799e-03,  2.2085e-03,  1.8887e-03],
  [ 3.0586e-03,  3.1982e-03,  2.6047e-03,  2.6655e-03,  2.2263e-03]],

 [[ 1.7978e-03,  1.7487e-03,  1.7485e-03,  1.7296e-03,  1.6269e-03],
  [ 1.9597e-03,  1.9981e-03,  1.8906e-03,  1.9215e-03,  1.7336e-03],
  [ 2.1151e-03,  2.2222e-03,  2.0873e-03,  2.0168e-03,  1.8495e-03],
  [ 2.0458e-03,  2.1605e-03,  2.1034e-03,  2.0636e-03,  1.8418e-03],
  [ 1.9142e-03,  1.9622e-03,  1.9310e-03,  2.0036e-03,  1.7868e-03]],

 [[ 2.5946e-03,  2.2865e-03,  1.5384e-03,  1.4668e-03,  1.3035e-03],
  [ 2.9198e-03,  2.5051e-03,  1.7563e-03,  1.1590e-03,  1.4979e-03],
  [ 3.2840e-03,  3.1905e-03,  2.1599e-03,  1.6229e-03,  1.3794e-03],
  [ 3.3455e-03,  3.5696e-03,  2.7840e-03,  2.0220e-03,  1.7025e-03],
  [ 2.9987e-03,  3.3378e-03,  2.9699e-03,  2.9007e-03,  2.1406e-03]]],


[[[ 8.6758e-04,  8.4458e-04,  7.1831e-04,  3.5591e-04,  3.8817e-05],
  [ 4.5233e-04,  6.8608e-04,  3.8609e-04, -4.9611e-04, -8.0755e-04],
  [-6.3414e-04, -3.1517e-04, -4.0875e-04, -2.6842e-04, -3.1715e-04],
  [-2.4219e-04, -1.1923e-04,  2.5545e-04,  6.9858e-04,  7.3381e-04],
  [ 1.8411e-04,  5.1464e-05, -1.6465e-04, -8.3227e-05, -8.8367e-05]],

 [[-7.2270e-04,  2.0058e-03,  7.9353e-04,  8.0987e-04,  3.4831e-04],
  [-7.8892e-04,  1.3682e-03,  5.3737e-04,  1.9637e-03,  1.0891e-03],
  [-9.8493e-04,  6.1819e-04,  1.0847e-03,  3.3904e-03,  2.0948e-03],
  [-3.6508e-04,  4.3299e-05,  1.8600e-04,  1.5710e-03,  1.4029e-03],
  [-5.5077e-05, -3.0135e-04, -1.5199e-03,  9.2724e-04,  1.2905e-03]],

 [[ 1.8135e-04,  7.9489e-04,  9.3434e-04,  1.0972e-03,  1.3003e-03],
  [-1.7188e-05,  2.5557e-04,  4.0360e-04,  7.6506e-04,  9.9554e-04],
  [-3.3151e-04, -6.8056e-05, -2.8687e-04,  2.3735e-04,  9.7931e-04],
  [-3.0199e-05, -1.8391e-05, -3.7223e-04, -1.1212e-04,  4.0520e-04],
  [-7.9732e-05, -1.5860e-04, -6.1655e-04, -6.2019e-04, -3.2300e-05]],

 …,

 [[-3.1923e-04, -1.3180e-04, -1.8628e-04,  1.0054e-04, -1.3772e-04],
  [-3.7629e-04, -4.6878e-04, -6.3113e-04, -4.1003e-04, -2.7531e-04],
  [-7.0189e-04, -6.2397e-04, -7.1047e-04, -3.8952e-04, -1.9626e-04],
  [ 3.3161e-06, -1.6542e-04, -3.8469e-04, -1.0580e-04, -1.0157e-05],
```

```
  [-1.5867e-04, -2.6566e-04, -4.9825e-04, -2.0012e-04, -1.9372e-04]],

 [[-2.7703e-04, -3.4672e-04, -3.9786e-04, -2.9162e-04, -3.4259e-04],
  [-2.8878e-04, -4.0694e-04, -4.3956e-04, -3.0694e-04, -4.0015e-04],
  [-2.8444e-04, -4.0570e-04, -4.3580e-04, -3.0087e-04, -4.4504e-04],
  [ 8.9796e-05,  2.4230e-06,  6.4092e-05,  8.4343e-05,  1.4433e-05],
  [ 7.9538e-06, -4.8976e-05,  9.2813e-05,  1.4652e-04,  6.6263e-05]],

 [[ 1.4749e-03,  1.4853e-03,  1.7836e-03,  1.2525e-03,  7.7069e-04],
  [ 1.6095e-03,  1.4312e-03,  9.7740e-04,  6.1674e-04,  7.1075e-04],
  [ 1.1132e-03,  1.0735e-03,  7.2543e-04,  8.7441e-04,  6.2575e-04],
  [ 1.1665e-03,  1.0235e-03,  6.5400e-04,  5.9413e-04,  2.1402e-04],
  [ 7.7456e-04,  8.6902e-04,  7.8895e-04,  6.0133e-04,  3.3710e-04]]],


[[[-2.1804e-04, -3.9987e-04, -5.4026e-04, -4.1390e-04, -6.1187e-04],
  [-1.0197e-03, -8.9265e-04, -7.8599e-04, -5.4932e-04, -5.1727e-04],
  [-5.7476e-04, -5.1527e-04, -6.5797e-04, -5.0639e-04, -7.5774e-04],
  [-8.2404e-04, -6.0027e-04, -4.4894e-04,  9.5037e-05,  4.5914e-04],
  [-1.3535e-05,  1.6865e-04,  4.0469e-04,  4.0024e-04,  2.9919e-04]],

 [[ 1.7328e-03,  1.8380e-04, -7.4463e-04,  4.5450e-04,  1.1113e-03],
  [ 1.6690e-03, -2.7912e-04, -1.2783e-03,  8.1500e-05,  9.6806e-04],
  [ 5.1389e-04, -1.1332e-04, -1.3919e-03,  2.9543e-04,  2.6029e-04],
  [-2.3645e-04,  3.4104e-04, -1.0300e-03, -9.4770e-04, -3.2660e-04],
  [-8.3015e-05, -2.8146e-04, -6.7055e-04, -8.0611e-04, -6.5117e-04]],

 [[ 4.6678e-04,  6.2837e-04,  1.4191e-04,  1.9902e-06,  4.7116e-04],
  [ 2.3195e-04,  3.4766e-04,  3.8420e-05, -4.4527e-05,  4.0193e-04],
  [ 1.1163e-03,  9.9105e-04,  7.1080e-04,  6.6685e-04,  5.7039e-04],
  [ 9.1643e-04,  9.2700e-04,  7.1146e-04,  6.7019e-04,  5.1364e-04],
  [ 8.2015e-04,  6.0736e-04,  5.6139e-04,  4.9157e-04,  3.9841e-04]],

 …,

 [[-5.8334e-04, -5.1648e-04, -8.0790e-04, -6.8860e-04, -5.9885e-04],
  [-6.9714e-04, -6.1785e-04, -8.1385e-04, -6.6558e-04, -6.3578e-04],
  [-4.1860e-04, -5.3856e-04, -7.0751e-04, -3.8961e-04, -3.5931e-04],
  [-5.1598e-04, -4.0033e-04, -5.1649e-04, -3.2145e-04, -4.0345e-04],
  [-5.3600e-04, -7.9033e-04, -6.4069e-04, -4.9627e-04, -3.9744e-04]],

 [[-4.2481e-04, -4.5264e-04, -5.1349e-04, -3.9902e-04, -4.3942e-04],
  [-5.3914e-04, -5.2390e-04, -5.2897e-04, -4.3733e-04, -4.3864e-04],
  [-5.9598e-04, -5.7986e-04, -6.3443e-04, -5.4068e-04, -5.2555e-04],
  [-5.2380e-04, -4.9532e-04, -5.5990e-04, -3.9658e-04, -3.4752e-04],
  [-5.0955e-04, -5.0994e-04, -5.7879e-04, -3.5096e-04, -3.2765e-04]],

 [[ 1.0621e-04,  3.0011e-04,  2.5685e-04,  3.1974e-04,  2.4042e-04],
```

```
          [-1.4192e-04,  6.7780e-05, -9.0530e-05,  1.7879e-04,  5.4551e-05],
          [ 2.6051e-04,  4.4998e-04,  2.5991e-04,  2.1772e-04,  2.0069e-04],
          [ 1.7584e-04,  1.1506e-04,  1.2450e-04, -5.8910e-05,  1.0583e-04],
          [-6.6982e-05, -2.0054e-04, -3.0735e-04, -4.5137e-04, -3.6644e-06]]])
tensor([ 6.1101e-10, -1.4734e-09,  3.6744e-10, -2.1209e-09,  4.5402e-09,
         2.0876e-09, -4.9331e-09,  2.0825e-09, -6.4122e-09,  1.4347e-09,
        -1.0539e-08,  3.9254e-09, -1.8602e-09,  6.1291e-09, -2.9468e-10,
        -1.7244e-09,  4.6665e-09,  8.9278e-10,  1.5084e-08, -2.0277e-09,
         6.7303e-11,  9.6180e-10,  1.5352e-09, -1.0145e-08,  7.7489e-10,
        -1.0092e-09,  2.2955e-08,  6.6293e-09, -3.3554e-09, -5.6627e-09,
        -4.9840e-10,  2.0249e-09,  3.1223e-09, -8.7670e-10, -2.9449e-09,
         1.0914e-10,  3.1433e-09,  3.3469e-10, -3.5457e-09,  7.7125e-10,
         2.5280e-09, -2.6138e-09,  3.5652e-09, -7.4215e-10, -2.7612e-09,
        -9.6213e-09,  1.1014e-09, -5.2714e-09, -1.9311e-09,  1.5672e-08,
        -1.9581e-09,  1.6316e-09, -4.0163e-09,  1.9022e-10, -1.1279e-09,
        -2.5545e-09,  1.8458e-09,  1.1969e-09,  1.1123e-09, -8.0007e-09,
         4.3992e-09, -4.9695e-09,  3.5652e-10,  1.5289e-08,  1.7099e-10,
        -1.3042e-09, -8.7175e-10, -2.6873e-09, -1.2018e-08,  2.3829e-09,
        -7.6193e-10, -6.0754e-10, -1.4734e-10, -4.0654e-09, -4.3898e-09,
         5.0156e-09,  3.4177e-09, -2.0609e-09, -1.5261e-09,  4.6819e-09,
         1.2469e-09,  1.7171e-09,  3.3995e-09, -1.9600e-08, -2.8922e-10,
         7.0304e-10, -1.7224e-08, -1.0259e-09, -1.6778e-09, -4.0163e-09,
        -5.5752e-10,  1.4656e-09,  1.5227e-08, -4.3366e-09, -9.6577e-11,
         8.3129e-09, -2.1919e-10,  4.7294e-10, -4.0245e-09,  3.8832e-09,
        -1.0587e-08, -2.9904e-09, -1.1730e-09, -1.9136e-09, -1.3898e-10,
         3.2887e-09, -6.3847e-09,  3.4734e-09,  7.4016e-09,  2.3865e-09,
         9.2518e-10,  2.5684e-09, -1.4134e-09,  6.8212e-11, -1.0827e-08,
         9.1222e-10, -2.7867e-09, -2.1828e-11, -8.2740e-10, -1.3071e-09,
         5.9434e-09, -8.4583e-10, -1.2922e-08,  1.2119e-09,  4.7053e-09,
         1.6394e-08,  6.8879e-09,  1.5061e-09,  2.4520e-09, -7.3960e-09,
        -2.6757e-09, -4.0852e-10,  2.9659e-09,  3.1231e-09,  1.7418e-09,
         7.7059e-09,  2.4538e-08, -3.7434e-09, -3.2232e-09,  3.9167e-09,
        -3.6089e-09, -1.5498e-09,  2.0790e-09,  4.6215e-10, -3.5943e-09,
        -2.1828e-10,  4.2310e-09,  1.9580e-09,  1.7126e-08, -3.4019e-09,
         7.9672e-09, -4.6930e-10,  1.8733e-09,  1.5339e-09,  8.0769e-09,
         8.8767e-10, -1.8224e-10,  5.5661e-09,  1.7872e-09, -3.3598e-09,
         2.2101e-08, -1.8044e-09, -2.0482e-09,  1.8044e-09,  4.4679e-10,
        -1.5177e-08, -7.2787e-09,  1.3482e-08,  4.1741e-09, -1.4699e-08,
        -3.5361e-09,  1.0150e-09,  4.5075e-09, -7.8155e-09, -1.6543e-08,
        -6.1700e-09, -2.2919e-10,  3.5943e-09, -1.3042e-09, -1.2915e-09,
         3.9950e-09, -2.4377e-10, -1.1406e-09, -1.7335e-09, -7.9535e-10,
         7.4797e-09,  8.2196e-11,  6.5134e-09,  8.6550e-10, -3.8999e-09,
        -1.1417e-09, -3.6800e-10, -2.3865e-09,  1.8152e-07,  4.5693e-09,
        -2.8267e-09,  4.0084e-09,  1.2594e-08,  1.4599e-09,  2.3546e-08,
         4.6058e-09,  1.1092e-08, -2.7459e-09, -3.8926e-10,  2.7485e-09,
         2.8049e-09,  7.9308e-10, -2.8813e-09, -3.3033e-09, -3.0519e-10,
        -3.6598e-09, -1.3085e-09,  1.7829e-09,  1.0800e-09,  1.9683e-10,
         3.2742e-11,  7.6398e-10, -2.2486e-09, -9.0222e-10, -1.3388e-09,
```

```
         1.1059e-09,  1.2105e-09,  4.9110e-08,  1.4754e-08, -4.3923e-09,
         1.3715e-09, -8.9913e-09, -1.4294e-09, -1.6444e-09, -7.0179e-09,
         2.8055e-09,  2.1100e-09,  3.8692e-10, -4.0746e-09, -9.6043e-10,
        -3.7480e-09, -1.5264e-08, -1.4922e-08, -2.8626e-09, -6.5775e-09,
        -3.6380e-11, -5.4274e-10, -1.1676e-08, -6.2118e-10, -2.5617e-09,
        -3.6407e-09, -1.2224e-08,  1.1318e-08,  5.9663e-10, -4.6566e-10,
         2.8158e-09,  5.6957e-10,  1.3006e-09, -5.7321e-10,  3.9654e-09,
         2.2737e-09])
tensor([-1.6675e-02, -5.9876e-03,  5.2761e-03, -1.4705e-02, -6.8855e-04,
         1.0468e-02,  1.4900e-02, -5.1709e-03,  3.6681e-02,  8.4783e-04,
         9.7887e-03,  1.9641e-02, -1.8693e-02,  2.6897e-02,  6.2705e-03,
        -5.9322e-03, -9.8358e-05, -3.3938e-03,  1.8324e-02,  1.0865e-02,
        -6.2046e-04,  5.0529e-03, -3.2153e-03, -2.0778e-02, -6.4145e-03,
        -3.0142e-03,  1.1997e-02,  3.8164e-02,  2.9435e-02,  5.7641e-02,
        -4.7552e-03,  1.2755e-02, -2.3510e-02, -2.3278e-02, -8.4902e-04,
        -2.3346e-03,  1.9368e-02, -1.3728e-02,  1.7363e-02, -4.5066e-03,
        -1.5996e-03, -8.1678e-03,  3.4977e-03, -4.7798e-03, -7.2069e-03,
        -2.8333e-02,  1.6976e-03,  5.0745e-03, -2.4204e-02, -2.9931e-02,
         1.4269e-02,  1.1108e-02,  6.6432e-03,  7.7234e-04,  1.1074e-02,
         2.2515e-02,  1.3739e-02, -1.1748e-02,  6.3297e-02, -2.3051e-02,
         2.1793e-03,  2.9531e-03,  1.4550e-03, -2.7370e-02,  2.8213e-03,
         2.3578e-03, -4.2853e-03,  2.8291e-02,  4.3796e-04,  9.3959e-03,
         1.2542e-02, -9.4434e-03, -1.6637e-03,  4.1936e-03,  4.4684e-02,
         3.5830e-02,  4.1571e-02,  5.2385e-03, -8.5897e-03,  1.5681e-02,
        -7.0946e-03,  1.2905e-03, -9.5842e-04, -6.7126e-02,  1.3059e-03,
         2.8779e-02,  4.4048e-02, -2.5047e-03, -6.9185e-03, -4.6382e-03,
         1.5994e-02,  4.4681e-03,  3.8283e-02, -8.3110e-03, -3.1759e-02,
        -7.4336e-03, -8.7741e-03,  4.2404e-03,  1.5949e-03, -2.3438e-02,
        -5.0400e-02, -8.4677e-03, -8.8192e-03, -8.8341e-03,  2.1191e-02,
         2.4164e-03,  2.6815e-05, -3.7002e-03,  1.6110e-02, -4.3943e-03,
        -5.1532e-03,  1.1054e-03, -3.5420e-03, -4.7793e-03,  1.0038e-02,
         1.9010e-03, -5.4414e-03, -8.4433e-03,  7.3183e-03,  4.7310e-03,
        -4.6487e-02, -3.8631e-03,  2.0429e-02, -4.8985e-03, -7.0614e-04,
         1.6921e-02, -2.4981e-02, -8.2526e-03,  2.2224e-03,  1.5159e-02,
         3.0614e-03, -1.2429e-02, -3.0219e-03, -1.5438e-02,  1.3252e-02,
         3.9596e-02,  3.1083e-02, -1.6647e-02,  6.2195e-03, -6.7349e-03,
         1.1701e-02, -2.0710e-03,  2.4968e-02, -1.2998e-03, -3.4413e-03,
        -9.4584e-03,  1.4099e-01, -3.5952e-02, -6.5345e-02,  3.2873e-02,
         1.0016e-02,  1.4348e-04,  7.0556e-02,  1.0110e-02,  4.8343e-02,
        -1.1663e-02,  1.9671e-03,  3.4551e-03, -3.0843e-03,  4.8265e-03,
        -6.4320e-03, -2.7422e-03, -3.5808e-04,  4.3079e-03,  1.8907e-02,
         2.9411e-02,  2.2674e-03,  3.6026e-03, -2.3414e-02,  3.3193e-02,
        -3.0902e-02,  1.8615e-04, -1.4234e-02, -4.2826e-02, -8.7661e-02,
        -1.1733e-02,  1.2026e-03,  5.5935e-03, -5.7500e-03,  5.8946e-03,
        -4.8523e-03, -3.7372e-03,  9.2092e-03, -6.3247e-04, -1.1686e-02,
        -3.3297e-03,  7.8709e-03,  6.4497e-03,  1.1349e-02, -1.2079e-02,
         1.7475e-02, -2.7904e-02, -1.1077e-02, -4.5971e-06,  1.7759e-02,
        -4.7762e-03, -5.2697e-05, -6.2281e-03,  2.7801e-03, -8.6036e-02,
```

```
        -6.8884e-03, -7.8653e-03,  1.6310e-02,  3.4071e-03, -4.4221e-03,
        -1.1426e-03, -1.0696e-02,  8.8116e-03,  1.8226e-02,  1.0240e-02,
         1.5160e-02,  3.7658e-02,  4.5357e-03, -3.4001e-02, -6.6497e-04,
        -8.8573e-04, -9.8316e-04,  7.5916e-03,  1.3766e-04, -6.5205e-04,
        -8.8312e-03, -5.7250e-03, -5.5077e-02, -2.5281e-02, -2.4680e-02,
         5.5328e-04, -4.1156e-03,  2.3395e-02,  9.7631e-04, -9.2223e-02,
         9.5132e-02, -7.7729e-03,  6.0877e-03,  4.4436e-03, -3.2945e-03,
         2.2606e-02,  7.0691e-02,  2.0193e-02,  3.4473e-02,  3.2171e-03,
        -5.8406e-04,  1.2189e-02,  5.0262e-02,  4.9110e-03,  2.2098e-02,
        -1.8767e-03,  5.7470e-02, -2.3113e-04, -5.8062e-03, -1.2063e-02,
         3.2282e-03, -1.1951e-02,  5.9142e-03,  4.6181e-02,  6.5696e-04,
         2.8151e-03])
tensor([-9.4822e-03, -2.7055e-03,  1.2105e-02, -2.6036e-02, -5.0386e-04,
         9.5823e-03,  1.9068e-02, -5.5323e-03,  3.2361e-02,  4.5750e-03,
         9.3031e-03,  2.5513e-02, -1.8702e-02,  3.3151e-02,  8.0095e-03,
        -9.0175e-03,  5.4279e-03, -2.7309e-02, -4.5439e-03, -3.1467e-03,
         2.7038e-03,  6.4544e-03, -2.0996e-03, -7.8135e-03, -3.8567e-03,
        -1.6251e-02, -1.9135e-03, -6.2809e-03,  2.2557e-02,  1.4363e-02,
        -4.6707e-03, -6.2517e-03, -2.4332e-02, -2.1666e-02,  5.7225e-03,
        -1.0643e-04,  1.8054e-02, -9.8555e-03, -1.2819e-02, -7.9072e-03,
        -2.3339e-03, -1.0209e-02,  1.5689e-03, -4.4197e-03, -1.4302e-02,
        -8.1553e-02,  3.5250e-03,  2.5800e-03, -1.2224e-02, -1.9805e-02,
         1.8483e-02,  1.7046e-02,  8.0271e-03,  4.3588e-03,  8.1300e-03,
         4.1060e-03,  1.7269e-02, -1.3612e-02,  4.3039e-02, -2.5237e-02,
        -2.1862e-04,  1.1342e-02,  3.5274e-03, -3.1416e-04, -3.0095e-03,
         3.8050e-03, -9.6260e-04,  2.1978e-02, -4.4498e-02,  8.1318e-03,
         1.9480e-02, -1.0485e-02,  5.7024e-03,  7.3845e-03,  1.1156e-02,
         3.2493e-02,  2.7092e-02,  8.2133e-03, -8.4128e-03,  1.4801e-02,
        -1.3401e-02,  5.4801e-03, -3.1314e-03, -3.4178e-02, -1.4503e-03,
         2.1258e-02,  2.2778e-02, -8.8264e-03, -4.2384e-03, -4.5662e-03,
         9.3570e-03,  4.5154e-03,  3.2872e-02,  4.0046e-03, -3.1488e-02,
         2.8831e-03, -8.8748e-03,  3.3338e-04,  1.0461e-02, -1.1741e-02,
        -2.7523e-02, -8.6797e-03, -5.6987e-03, -1.2420e-02,  2.6979e-02,
         5.3005e-03, -9.2539e-04, -5.7647e-03,  3.6727e-02, -8.9450e-03,
        -3.4502e-03,  2.7216e-03,  1.4181e-03, -1.6387e-03,  2.1577e-02,
         1.9330e-03, -2.9700e-03, -7.5876e-04,  9.3100e-03,  9.4367e-03,
        -4.5556e-02, -4.8120e-03,  1.6736e-02, -8.0415e-03,  1.4844e-02,
         2.9363e-02, -1.7689e-02, -1.1258e-02,  1.6710e-03,  3.1837e-02,
         3.8213e-03, -1.5597e-02, -4.0526e-03, -1.9869e-02,  1.2109e-02,
         3.5690e-02, -7.0963e-03, -2.9827e-02,  8.3238e-03, -4.7863e-03,
         1.4657e-02,  2.9220e-03,  1.3461e-02, -2.6228e-03, -4.9488e-03,
        -8.5360e-03,  4.1124e-02, -3.9294e-02, -3.0060e-02, -3.9337e-05,
         1.3037e-02,  3.1560e-04,  8.5747e-03,  9.5671e-04,  2.8599e-03,
        -6.2606e-03,  8.3873e-03,  8.8070e-03, -2.0516e-02,  1.2049e-02,
        -2.4116e-02, -3.0031e-03, -1.1025e-03,  4.8217e-03,  2.0350e-02,
         1.9650e-02, -9.7365e-04, -1.1468e-03, -1.3764e-02,  2.9844e-02,
        -1.8182e-02,  2.1098e-03, -1.2056e-02, -6.1337e-02, -3.7652e-02,
        -1.4846e-02,  2.3369e-03,  1.0360e-02, -6.2734e-03,  4.5750e-03,
```

```
        -1.7691e-03, -9.9122e-03,  4.9259e-03,  4.6564e-03, -9.1571e-03,
        -3.4261e-02, -2.7005e-02,  1.0795e-02,  3.1817e-02, -1.8301e-02,
         1.7504e-02, -7.0655e-03, -1.4988e-02, -7.2471e-03,  1.7893e-02,
        -1.9120e-03, -2.7553e-03, -5.2579e-03, -5.9014e-04, -4.5764e-02,
        -1.9423e-02, -1.5492e-02,  2.7501e-02,  5.9519e-03,  2.2289e-03,
        -4.2656e-03, -1.0632e-02,  2.2935e-02,  1.3048e-02,  9.4214e-03,
         1.2569e-02,  2.7725e-02,  1.6013e-02, -3.8637e-03, -3.7752e-03,
        -1.4118e-03,  2.1897e-03,  6.5006e-03,  3.2552e-03,  2.5448e-03,
        -1.8054e-02, -1.5009e-02, -3.5196e-02, -3.7141e-02, -7.1087e-02,
         4.7301e-03, -7.3938e-03,  2.2905e-02,  1.6663e-03, -4.5769e-02,
         5.5503e-02, -7.1332e-03,  3.2341e-03,  4.7492e-03, -9.2926e-04,
         1.8313e-02,  4.8833e-02, -1.9869e-02,  1.5383e-02,  1.0534e-02,
         3.8571e-03,  7.5357e-03,  2.8370e-02,  8.7205e-03, -1.5048e-02,
         1.9015e-03,  1.7353e-02, -1.4112e-02, -1.0144e-02, -2.5249e-02,
         1.3108e-02, -1.4168e-02,  1.4434e-02,  3.8863e-02, -2.2454e-03,
         9.9943e-03]])
tensor([[[[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         …,

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]]],


        [[[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
```

```
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],
```

```
…,


[[[-0.0598, -0.0699, -0.0727],
  [-0.0550, -0.0501, -0.0483],
  [-0.0566, -0.0589, -0.0534]],

 [[-0.0268, -0.0348, -0.0342],
  [-0.0265, -0.0264, -0.0200],
  [-0.0228, -0.0106, -0.0070]],

 [[-0.0401, -0.0429, -0.0385],
  [-0.0627, -0.0563, -0.0439],
  [-0.0634, -0.0547, -0.0409]],

 …,

 [[-0.0911, -0.0926, -0.0825],
  [-0.0962, -0.0991, -0.0884],
  [-0.1041, -0.1059, -0.0946]],

 [[-0.0594, -0.0580, -0.0612],
  [-0.0762, -0.0637, -0.0677],
  [-0.0847, -0.0678, -0.0605]],

 [[ 0.0175,  0.0264,  0.0320],
  [-0.0043,  0.0096,  0.0250],
  [-0.0018,  0.0080,  0.0246]]],


[[[ 0.0017,  0.0001,  0.0019],
  [-0.0230, -0.0250, -0.0217],
  [-0.0409, -0.0260, -0.0186]],

 [[ 0.0069,  0.0178,  0.0074],
  [ 0.0107,  0.0218,  0.0036],
  [ 0.0117,  0.0242,  0.0087]],

 [[ 0.0054,  0.0134, -0.0027],
  [ 0.0089,  0.0199, -0.0004],
  [ 0.0002,  0.0178,  0.0030]],

 …,

 [[-0.0250, -0.0125, -0.0006],
  [-0.0283, -0.0147, -0.0050],
  [-0.0298, -0.0161, -0.0132]],
```

```
          [[ 0.0150,  0.0190,  0.0044],
           [ 0.0086,  0.0270,  0.0082],
           [-0.0058,  0.0216,  0.0078]],

          [[-0.0142, -0.0065, -0.0111],
           [-0.0045,  0.0029, -0.0057],
           [-0.0027,  0.0025, -0.0076]]],


         [[[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          …,

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]]]])
tensor([ 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  8.7779e-04,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  7.6719e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -2.4675e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -2.4131e-04,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -1.2260e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

```
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00, -6.4652e-02,  0.0000e+00,  2.7971e-03,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  2.3481e-02,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00, -9.0877e-02,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00, -2.3705e-02,  0.0000e+00,
0.0000e+00,  5.7032e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00, -7.9152e-08,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
6.2262e-03,  0.0000e+00, -2.0477e-07,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
3.0376e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

```
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  7.2900e-02,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00, -3.3963e-05,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  3.4051e-02,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  1.8031e-03,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -1.4820e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  2.8919e-03,  1.5286e-05,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -9.0174e-02, -2.7971e-02,  0.0000e+00])
tensor([[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 1.7532e-01,  9.1567e-02,  2.8952e-02],
          [ 2.6506e-01,  1.4819e-01,  7.6959e-02],
          [-7.8241e-03, -3.0571e-02,  2.1549e-02]],

         [[-7.1976e-03, -1.9396e-02,  5.7809e-03],
          [ 7.0704e-02, -2.3962e-03,  3.4656e-03],
          [ 6.5026e-02, -3.1550e-02, -4.3902e-02]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 3.9008e-02,  2.5981e-02,  1.2715e-02],
  [ 1.6895e-02,  8.3094e-03, -1.7949e-03],
  [ 1.0518e-02,  6.4923e-03, -4.0211e-03]],

 [[ 5.3175e-02,  5.5300e-02,  5.1919e-02],
  [ 4.0931e-02,  4.9137e-02,  5.0704e-02],
  [ 2.6214e-02,  4.6723e-02,  6.5797e-02]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],
```

```
…,


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 1.2274e-01,  9.5499e-02,  7.0904e-02],
  [ 4.5519e-02,  5.8217e-02,  3.1469e-02],
  [-2.6265e-02,  1.4694e-02,  1.1321e-02]],

 [[ 1.5912e-01,  1.3427e-01,  8.7222e-02],
  [ 1.1264e-01,  1.2257e-01,  1.0534e-01],
  [ 3.7979e-02,  4.6600e-02,  6.7351e-02]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[-5.1541e-04,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 2.6377e-05,  0.0000e+00,  0.0000e+00]],
```

```
          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 1.7077e-04,  2.1683e-04,  1.8877e-04]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


         [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([-1.4230e-01,  3.7595e-02,  0.0000e+00,  2.2282e-02, -8.4996e-02,
         1.4806e-01,  4.9857e-03,  3.8497e-03, -1.6109e-02,  0.0000e+00,
        -1.0732e-03, -1.0887e-02, -2.9583e-02,  0.0000e+00,  0.0000e+00,
        -6.5469e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  6.2683e-05,  5.6940e-03,  0.0000e+00,
         2.9242e-02,  0.0000e+00, -6.0326e-03,  0.0000e+00, -7.7194e-04,
        -9.7421e-03, -7.1729e-04,  7.6047e-03,  0.0000e+00,  5.3905e-03,
         2.7088e-02, -3.8142e-03,  6.3668e-02, -6.6842e-04, -4.3365e-02,
         5.2626e-03,  3.1513e-02,  0.0000e+00, -6.7172e-03,  0.0000e+00,
         2.2041e-01, -2.5469e-03, -5.5456e-03,  3.2252e-02, -7.8463e-04,
         0.0000e+00, -5.3823e-04, -2.5311e-03, -1.4107e-03,  1.0800e-02,
        -4.6660e-03, -1.0760e-02, -3.7323e-04,  4.9670e-03,  0.0000e+00,
        -8.2205e-04,  0.0000e+00, -1.9057e-03,  0.0000e+00,  5.2571e-03,
```

```
 4.6238e-02, -7.6561e-02, -1.2971e-02, -1.7666e-02, -1.2858e-03,
 0.0000e+00,  0.0000e+00, -2.5175e-04,  2.5216e-05, -1.1921e-04,
-2.9344e-05,  1.4078e-03,  2.1440e-03, -1.7477e-04,  4.8726e-03,
 0.0000e+00,  0.0000e+00,  5.2593e-04,  2.9226e-03, -3.0997e-03,
 0.0000e+00, -7.2193e-03, -2.6177e-02,  3.3895e-04,  0.0000e+00,
-8.1772e-03,  0.0000e+00,  5.4253e-04,  0.0000e+00, -4.1264e-04,
 0.0000e+00,  1.2983e-04, -7.9082e-03,  0.0000e+00, -2.5760e-03,
 6.5684e-02,  4.2298e-03,  4.6621e-02,  1.9059e-03,  0.0000e+00,
 3.0947e-02,  2.0277e-03, -2.5818e-03,  6.1825e-03, -2.0920e-02,
-6.8245e-04, -3.0405e-03, -2.0156e-04, -8.6150e-03,  8.0893e-03,
-3.5807e-04, -4.9754e-04,  0.0000e+00,  0.0000e+00, -3.6474e-02,
 4.6583e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 7.4821e-05, -1.0347e-03, -1.1528e-05,  0.0000e+00, -2.4620e-03,
 0.0000e+00,  1.4461e-04, -6.7346e-04, -7.9498e-02,  3.5179e-03,
 7.8598e-02,  0.0000e+00,  0.0000e+00,  3.2613e-04,  1.9641e-04,
-1.3727e-03,  7.9073e-03,  6.8841e-02,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00, -9.3143e-02, -3.7797e-02,  0.0000e+00,
 0.0000e+00,  1.2221e-03,  9.0244e-03,  3.0421e-04,  0.0000e+00,
-5.3538e-02,  0.0000e+00,  0.0000e+00,  1.2790e-03, -1.3043e-02,
 1.6285e-02, -1.1165e-02,  0.0000e+00,  0.0000e+00,  6.5656e-02,
 0.0000e+00, -1.4730e-03,  1.5411e-04,  8.8766e-04, -2.1147e-02,
 3.9155e-03,  1.0476e-03,  5.3400e-03,  0.0000e+00, -1.2980e-02,
 3.0112e-02,  1.0584e-03,  8.3952e-03, -6.5469e-03, -2.8459e-02,
 0.0000e+00,  9.4822e-03,  3.1567e-04,  2.1397e-05, -1.0721e-02,
 0.0000e+00,  5.2441e-03,  2.6978e-03,  4.7012e-04, -8.1125e-03,
 5.0003e-04,  0.0000e+00,  0.0000e+00,  2.7427e-02, -1.2691e-02,
-4.1702e-04,  1.3829e-02, -5.7367e-03,  0.0000e+00,  0.0000e+00,
 5.9531e-02, -8.0152e-02,  0.0000e+00,  4.7687e-02, -1.1777e-03,
 1.1367e-03, -1.3298e-03,  0.0000e+00, -9.0891e-04,  1.2252e-02,
-3.1958e-03, -1.4539e-04,  3.1740e-03,  4.2613e-03, -2.5028e-02,
-5.9549e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  3.2687e-05,
 0.0000e+00,  0.0000e+00,  5.3673e-04, -1.0027e-02,  0.0000e+00,
 6.3700e-06,  2.0345e-02, -1.1805e-01,  5.3603e-03,  0.0000e+00,
 2.1698e-02,  0.0000e+00,  2.9317e-05, -3.1026e-03,  3.0907e-03,
 2.7262e-02,  0.0000e+00,  2.2361e-03, -2.8873e-03, -5.6445e-02,
-9.9747e-06, -2.8087e-02,  0.0000e+00,  6.9807e-03,  3.1146e-02,
 0.0000e+00,  1.7669e-02,  0.0000e+00,  1.0424e-02,  1.0905e-02,
-2.2422e-04,  3.2376e-02,  8.9100e-03,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  7.8404e-04, -1.8729e-02,  0.0000e+00,
 0.0000e+00, -7.2096e-02,  3.9287e-02, -2.9417e-03,  0.0000e+00,
 1.2936e-03, -1.8401e-01, -8.8700e-03,  1.3934e-03,  1.3135e-02,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  1.7924e-01,
 2.4650e-03,  5.4906e-02, -4.4002e-03,  0.0000e+00,  8.0242e-03,
-1.3429e-02,  1.0855e-03,  3.6966e-04, -1.3608e-02, -4.5895e-02,
 0.0000e+00,  4.8496e-02,  0.0000e+00,  2.0818e-04, -8.5990e-03,
-8.1306e-04, -9.0872e-04,  1.4213e-03,  2.1913e-03,  0.0000e+00,
 2.3403e-02,  2.4471e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 3.1863e-02, -5.3062e-02, -8.1963e-04,  1.7501e-02,  0.0000e+00,
```

```
         3.2284e-03, -6.6018e-03,  4.3007e-04,  6.1240e-03,  0.0000e+00,
         0.0000e+00, -9.2381e-04, -9.9435e-04,  9.3837e-03, -8.5511e-03,
        -8.5982e-04, -3.8643e-04,  0.0000e+00,  0.0000e+00,  1.9361e-03,
         0.0000e+00, -4.0022e-05,  9.0729e-04,  9.1562e-02, -2.2244e-02,
        -4.5260e-03,  0.0000e+00,  0.0000e+00,  1.5321e-02,  0.0000e+00,
         3.0455e-04,  6.6290e-02,  1.6165e-02,  0.0000e+00,  3.8069e-02,
         4.3377e-04,  0.0000e+00,  8.2164e-03,  8.7953e-04,  8.2683e-03,
         1.3893e-03, -8.2246e-02, -2.6515e-03, -1.0348e-03, -1.8736e-02,
        -5.3994e-03, -4.4074e-02, -3.7656e-05,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  1.1671e-04, -4.2042e-03, -6.7676e-05,
        -7.3643e-03, -1.1179e-03,  0.0000e+00,  2.5792e-02,  0.0000e+00,
         5.2384e-05,  1.4235e-03,  3.9807e-02,  0.0000e+00,  0.0000e+00,
        -1.2514e-02, -4.5975e-03,  0.0000e+00, -8.6621e-04, -4.1610e-03,
        -1.4110e-03,  0.0000e+00,  0.0000e+00,  3.8901e-02, -3.7505e-04,
        -1.6189e-03,  6.9295e-03,  9.2433e-03,  0.0000e+00,  1.5428e-02,
         0.0000e+00,  4.9478e-03, -1.4284e-04,  0.0000e+00])
tensor([[[[ 2.6330e-02,  4.1582e-02,  5.6063e-02],
          [ 2.3421e-02,  3.4691e-02,  4.8682e-02],
          [ 2.6946e-02,  3.4193e-02,  4.1463e-02]],

         [[ 0.0000e+00,  0.0000e+00, -2.9530e-05],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00, -6.9948e-06],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],
```

```
…,


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


         [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([ 1.1272e-01,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.3435e-03,
         2.8729e-05,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  1.0199e-02,  3.2479e-01, -1.4932e-01,
        -1.8548e-02,  7.7551e-03,  4.9796e-02,  0.0000e+00, -3.7232e-04,
        -3.2007e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -2.6470e-02,
        -3.7828e-01,  3.9569e-02,  1.7464e-03,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  1.6785e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -5.9628e-03,  0.0000e+00, -1.3942e-05,  1.8314e-01,  7.7404e-02,
         0.0000e+00,  0.0000e+00, -7.4281e-04,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  3.8012e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -1.1625e-03,  6.5228e-01,  0.0000e+00,
        -1.8371e-01,  0.0000e+00,  0.0000e+00,  5.8483e-05, -1.4436e-02,
```

```
          0.0000e+00,  3.5780e-02,  0.0000e+00,  5.2153e-03, -1.0614e-05,
          0.0000e+00, -5.7228e-03,  2.2089e-02,  1.2900e-02,  0.0000e+00,
          0.0000e+00, -1.0312e-02,  1.4040e-02,  0.0000e+00,  7.3561e-04,
          0.0000e+00,  2.6288e-02,  0.0000e+00,  2.5346e-04,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  1.4235e-02,  0.0000e+00,  0.0000e+00, -1.3768e-05,
         -6.7304e-04, -6.1606e-04,  0.0000e+00, -1.8013e-01,  8.8525e-02,
          0.0000e+00,  0.0000e+00,  0.0000e+00, -3.6865e-03,  9.5767e-02,
          0.0000e+00,  0.0000e+00, -4.0123e-02,  0.0000e+00,  0.0000e+00,
          9.3379e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00, -8.9875e-02, -2.4396e-02,
          0.0000e+00,  0.0000e+00,  1.7389e-02,  0.0000e+00,  2.2154e-02,
          0.0000e+00,  0.0000e+00,  4.3046e-05,  0.0000e+00,  0.0000e+00,
         -2.4083e-01,  0.0000e+00,  0.0000e+00,  0.0000e+00,  1.1653e-02,
          0.0000e+00,  0.0000e+00,  5.4159e-03,  1.0649e-01,  1.7587e-02,
         -1.3007e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00, -4.8442e-04,  0.0000e+00,  0.0000e+00,
         -3.2329e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -2.2697e-01,  7.6602e-04,  0.0000e+00,  0.0000e+00,
         -1.2793e-04,  0.0000e+00,  2.2037e-01,  0.0000e+00,  0.0000e+00,
         -8.4630e-02,  0.0000e+00,  0.0000e+00,  8.8683e-05,  2.2201e-02,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -7.8666e-02,  0.0000e+00,  5.0835e-03,  6.5212e-02,
         -3.9953e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  2.0836e-02,
          0.0000e+00,  0.0000e+00,  5.2645e-04,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  1.4298e-01,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  3.5139e-01, -3.8365e-04,  0.0000e+00,
          0.0000e+00,  6.5552e-02, -7.1008e-02,  0.0000e+00,  2.4530e-01,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          1.3511e-01,  0.0000e+00,  1.0803e-03,  4.6036e-02,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00, -2.5316e-04, -3.6459e-01,  0.0000e+00,
          1.3525e-02,  1.1421e-01, -1.0504e-01, -4.2315e-03,  0.0000e+00,
          6.5658e-03,  0.0000e+00,  3.8222e-02,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  9.5926e-03,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -2.1791e-04,
          1.1258e-01,  0.0000e+00,  6.3760e-03, -3.6064e-01,  0.0000e+00,
          0.0000e+00,  5.8628e-03,  3.6371e-03,  0.0000e+00,  0.0000e+00,
          0.0000e+00]])
tensor([[ 0.0000e+00,  0.0000e+00, -1.9855e-06,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  2.5443e-05,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        …,
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
```

```
        [ 0.0000e+00,  0.0000e+00, -4.0042e-06,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00]])
tensor([-0.0008,  0.0000,  0.0036,  …,  0.0000,  0.0059,  0.0000])
tensor([[0.0000, 0.0000, 0.0000,  …, 0.0000, 0.0009, 0.0000],
        [0.0000, 0.0000, 0.0000,  …, 0.0000, 0.0000, 0.0000],
        [0.0000, 0.0000, 0.0000,  …, 0.0000, 0.0000, 0.0000],
        …,
        [0.0000, 0.0000, 0.0000,  …, 0.0000, 0.0000, 0.0000],
        [0.0000, 0.0000, 0.0000,  …, 0.0000, 0.0000, 0.0000],
        [0.0000, 0.0000, 0.0000,  …, 0.0000, 0.0000, 0.0000]])
tensor([ 2.2254e-03,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
         0.0000e+00, -7.6680e-07])
tensor([[1.1831e-05, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         8.6416e-10],
        [1.4042e-06, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         2.0657e-10],
        [8.3442e-07, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         7.6602e-12],
        …,
        [5.0418e-09, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         5.4887e-18],
        [5.2068e-09, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         5.9496e-18],
        [4.8151e-09, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         4.8703e-18]])
tensor([ 9.6335e-03, -5.9717e-02,  5.5365e-02,  6.5251e-02, -5.6070e-03,
         3.5008e-02, -7.9631e-02,  6.3311e-02, -2.3970e-02,  4.8787e-02,
        -5.0032e-02,  5.4626e-02, -1.5811e-02,  2.4497e-02, -5.9435e-02,
        -1.1161e-01, -1.2001e-02,  7.8812e-03, -8.8386e-03,  5.8417e-02,
         1.3275e-05,  3.8324e-06,  3.9973e-06,  4.0083e-06,  3.7119e-06,
         4.0349e-06,  3.9623e-06,  3.8924e-06,  3.8956e-06,  3.9095e-06,
         3.9532e-06,  4.0131e-06,  3.9488e-06,  3.9971e-06,  4.0208e-06,
         3.9797e-06,  4.0174e-06,  3.9029e-06,  4.0343e-06,  3.9100e-06,
         4.0081e-06,  4.0241e-06,  3.7678e-06,  4.0283e-06,  3.8008e-06,
         3.9281e-06,  3.9892e-06,  4.0193e-06,  3.9844e-06,  3.9583e-06,
         3.9162e-06,  4.0183e-06,  3.9284e-06,  3.9611e-06,  3.9506e-06,
         3.9437e-06,  4.0029e-06,  3.9035e-06,  4.0346e-06,  3.9855e-06,
         3.9122e-06,  3.9620e-06,  3.8176e-06,  3.8122e-06,  3.8533e-06,
         3.9485e-06,  3.8042e-06,  3.9422e-06,  3.7827e-06,  3.7767e-06,
         4.0079e-06,  3.9025e-06,  3.9822e-06,  4.0331e-06,  3.9714e-06,
         3.7705e-06,  3.9033e-06,  4.0367e-06,  4.0298e-06,  3.7383e-06,
         4.0386e-06,  4.0260e-06,  3.7882e-06,  3.9269e-06,  4.0079e-06,
         3.7480e-06,  3.9612e-06,  3.8887e-06,  3.8619e-06,  3.9707e-06,
         4.0376e-06,  4.0120e-06,  3.8962e-06,  3.9839e-06,  4.0400e-06,
         3.9100e-06,  3.9233e-06,  4.0194e-06,  3.7751e-06,  3.9429e-06,
         3.9570e-06,  4.0094e-06,  3.8963e-06,  4.0136e-06,  4.0160e-06,
```

```
3.8395e-06,  4.0014e-06,  4.0311e-06,  4.0257e-06,  3.7861e-06,
3.9715e-06,  3.8756e-06,  3.9738e-06,  3.7273e-06,  3.9787e-06,
4.0332e-06,  3.9689e-06,  4.0384e-06,  3.8209e-06,  4.0066e-06,
3.8215e-06,  4.0298e-06,  3.7728e-06,  3.8093e-06,  4.0199e-06,
4.0465e-06,  3.9424e-06,  4.0156e-06,  3.9404e-06,  4.0095e-06,
3.9977e-06,  3.7753e-06,  4.0177e-06,  3.9287e-06,  3.9904e-06,
3.9801e-06,  4.0000e-06,  4.0224e-06,  3.8170e-06,  4.0232e-06,
3.8828e-06,  3.7813e-06,  3.9634e-06,  3.9919e-06,  3.9321e-06,
3.9517e-06,  3.9859e-06,  3.8856e-06,  4.0225e-06,  3.8820e-06,
3.9018e-06,  3.9055e-06,  3.9554e-06,  3.7598e-06,  3.9735e-06,
4.0006e-06,  3.9760e-06,  3.9871e-06,  3.8422e-06,  4.0185e-06,
3.9165e-06,  3.8790e-06,  4.0374e-06,  4.0187e-06,  3.8427e-06,
3.8306e-06,  4.0101e-06,  4.0249e-06,  3.8303e-06,  3.9320e-06,
3.8913e-06,  3.9075e-06,  3.8667e-06,  3.7842e-06,  3.9933e-06,
4.0189e-06,  4.0020e-06,  4.0314e-06,  3.8982e-06,  3.8912e-06,
3.9127e-06,  3.9991e-06,  4.0457e-06,  3.9635e-06,  3.8702e-06,
3.8619e-06,  4.0290e-06,  3.7890e-06,  3.9284e-06,  3.8783e-06,
4.0175e-06,  3.9881e-06,  3.9268e-06,  3.9784e-06,  4.0348e-06,
3.9982e-06,  4.0094e-06,  3.9511e-06,  4.0382e-06,  4.0429e-06,
4.0257e-06,  3.8549e-06,  3.9615e-06,  3.8634e-06,  3.8409e-06,
3.9004e-06,  3.9862e-06,  3.9120e-06,  4.0340e-06,  3.9360e-06,
3.9978e-06,  3.9235e-06,  3.9134e-06,  4.0317e-06,  3.8058e-06,
4.0331e-06,  3.8997e-06,  3.8752e-06,  3.9555e-06,  3.7865e-06,
3.8137e-06,  4.0415e-06,  4.0268e-06,  4.0229e-06,  3.8119e-06,
3.8972e-06,  3.9990e-06,  4.0220e-06,  3.9031e-06,  4.0278e-06,
3.9887e-06,  3.8443e-06,  4.0140e-06,  3.9208e-06,  3.9866e-06,
3.9279e-06,  4.0238e-06,  3.8886e-06,  4.0374e-06,  3.9818e-06,
4.0121e-06,  4.0238e-06,  3.9997e-06,  4.0034e-06,  3.9967e-06,
3.9767e-06,  4.0271e-06,  4.0215e-06,  4.0312e-06,  3.9331e-06,
3.9032e-06,  3.9333e-06,  3.8750e-06,  3.8666e-06,  3.9979e-06,
3.9348e-06,  4.0388e-06,  4.0310e-06,  4.0193e-06,  4.0231e-06,
3.9765e-06,  4.0185e-06,  3.9822e-06,  3.9907e-06,  3.9981e-06,
3.8354e-06,  3.9720e-06,  3.9921e-06,  4.0454e-06,  3.9497e-06,
4.0147e-06,  4.0158e-06,  3.9699e-06,  4.0228e-06,  3.9718e-06,
3.8651e-06,  4.0077e-06,  3.8523e-06,  4.0287e-06,  3.9667e-06,
4.0354e-06,  3.9534e-06,  4.0422e-06,  3.8595e-06,  3.9335e-06,
3.5685e-06,  4.0208e-06,  3.9152e-06,  4.0266e-06,  3.9220e-06,
3.6406e-06,  3.8634e-06,  4.0396e-06,  4.0392e-06,  3.9053e-06,
4.0360e-06,  3.9279e-06,  4.0346e-06,  4.0092e-06,  3.8457e-06,
3.9898e-06,  3.9017e-06,  3.7907e-06,  3.8644e-06,  3.8075e-06,
3.8371e-06,  3.9378e-06,  3.9944e-06,  3.9420e-06,  3.9949e-06,
4.0240e-06,  3.8555e-06,  3.8083e-06,  4.0262e-06,  3.8574e-06,
4.0125e-06,  3.9893e-06,  4.0111e-06,  4.0325e-06,  4.0065e-06,
4.0105e-06,  3.9725e-06,  3.8193e-06,  4.0404e-06,  3.9737e-06,
3.9825e-06,  3.8987e-06,  3.9562e-06,  3.9876e-06,  3.8759e-06,
3.8278e-06,  3.8694e-06,  3.7910e-06,  3.9724e-06,  3.9219e-06,
3.9683e-06,  3.8450e-06,  4.0310e-06,  3.9941e-06,  3.8378e-06,
4.0368e-06,  3.8940e-06,  3.9034e-06,  3.8664e-06,  3.6727e-06,
```

```
3.9869e-06,  3.8979e-06,  4.0371e-06,  3.9349e-06,  4.0472e-06,
3.7709e-06,  3.9763e-06,  3.8105e-06,  3.9661e-06,  3.9546e-06,
3.8333e-06,  3.8533e-06,  3.8783e-06,  4.0018e-06,  4.0321e-06,
3.9897e-06,  4.0323e-06,  3.6865e-06,  3.9780e-06,  3.9921e-06,
3.9950e-06,  3.7948e-06,  3.7910e-06,  4.0341e-06,  4.0255e-06,
3.8363e-06,  4.0126e-06,  3.8791e-06,  3.7567e-06,  3.8854e-06,
3.7888e-06,  3.9601e-06,  3.5689e-06,  4.0020e-06,  3.9878e-06,
3.9572e-06,  4.0168e-06,  3.8726e-06,  3.8554e-06,  4.0268e-06,
3.9794e-06,  4.0271e-06,  3.8797e-06,  3.9291e-06,  3.9238e-06,
4.0357e-06,  3.9200e-06,  3.9827e-06,  3.9714e-06,  3.9982e-06,
4.0379e-06,  4.0093e-06,  4.0335e-06,  4.0399e-06,  4.0342e-06,
4.0117e-06,  3.9180e-06,  3.8948e-06,  3.9817e-06,  3.9699e-06,
3.9587e-06,  4.0155e-06,  3.7936e-06,  3.8086e-06,  4.0146e-06,
3.9262e-06,  3.8652e-06,  3.9784e-06,  3.9061e-06,  4.0332e-06,
3.9759e-06,  4.0364e-06,  4.0230e-06,  3.9512e-06,  3.9232e-06,
3.8364e-06,  4.0111e-06,  3.9417e-06,  4.0392e-06,  4.0146e-06,
4.0269e-06,  4.0240e-06,  3.8319e-06,  4.0161e-06,  3.9889e-06,
4.0207e-06,  4.0267e-06,  3.9996e-06,  4.0169e-06,  4.0269e-06,
4.0236e-06,  3.8836e-06,  3.8480e-06,  4.0133e-06,  4.0191e-06,
3.8638e-06,  3.8831e-06,  4.0374e-06,  3.8916e-06,  4.0328e-06,
3.9841e-06,  4.0179e-06,  3.9118e-06,  3.8943e-06,  3.9410e-06,
4.0145e-06,  4.0051e-06,  3.9132e-06,  4.0200e-06,  4.0343e-06,
3.9734e-06,  4.0290e-06,  4.0306e-06,  3.7980e-06,  3.8427e-06,
3.9641e-06,  4.0315e-06,  4.0152e-06,  3.8955e-06,  3.7961e-06,
3.9844e-06,  3.9000e-06,  3.8933e-06,  4.0316e-06,  3.9660e-06,
3.8972e-06,  4.0027e-06,  4.0084e-06,  3.8376e-06,  3.8891e-06,
4.0295e-06,  3.9473e-06,  3.9915e-06,  4.0263e-06,  4.0357e-06,
4.0227e-06,  3.9843e-06,  3.9971e-06,  4.0124e-06,  3.9002e-06,
3.9973e-06,  3.8645e-06,  4.0442e-06,  3.9890e-06,  3.9311e-06,
3.4683e-06,  4.0168e-06,  3.9410e-06,  4.0390e-06,  4.0015e-06,
3.9886e-06,  3.8383e-06,  3.9350e-06,  3.8800e-06,  4.0250e-06,
3.9871e-06,  3.9338e-06,  4.0274e-06,  4.0259e-06,  4.0231e-06,
3.8632e-06,  3.9591e-06,  3.9861e-06,  4.0251e-06,  4.0275e-06,
4.0186e-06,  4.0305e-06,  3.8350e-06,  3.9255e-06,  4.0058e-06,
3.9790e-06,  4.0140e-06,  4.0159e-06,  4.0236e-06,  3.9125e-06,
3.9054e-06,  3.8717e-06,  3.9610e-06,  3.9922e-06,  3.9122e-06,
3.9323e-06,  4.0348e-06,  3.6511e-06,  4.0181e-06,  3.9789e-06,
4.0039e-06,  4.0328e-06,  3.8132e-06,  3.9590e-06,  3.7739e-06,
4.0226e-06,  3.9715e-06,  4.0386e-06,  4.0079e-06,  3.9529e-06,
3.9608e-06,  3.8303e-06,  3.8748e-06,  4.0290e-06,  3.9687e-06,
3.9596e-06,  4.0268e-06,  3.9899e-06,  4.0183e-06,  3.9862e-06,
4.0245e-06,  4.0050e-06,  4.0239e-06,  3.9073e-06,  3.7772e-06,
3.9400e-06,  3.9751e-06,  4.0100e-06,  3.9175e-06,  3.9612e-06,
3.8818e-06,  3.9109e-06,  4.0135e-06,  3.9736e-06,  4.0262e-06,
3.8239e-06,  3.9595e-06,  4.0150e-06,  4.0036e-06,  4.0428e-06,
4.0269e-06,  4.0325e-06,  4.0044e-06,  4.0245e-06,  3.9639e-06,
3.8395e-06,  3.7685e-06,  4.0351e-06,  3.8422e-06,  4.0265e-06,
3.9829e-06,  3.9202e-06,  3.7875e-06,  3.9862e-06,  4.0224e-06,
```

```
3.9353e-06,  3.8350e-06,  4.0363e-06,  3.9965e-06,  4.0214e-06,
4.0086e-06,  3.9437e-06,  3.8630e-06,  3.8807e-06,  4.0190e-06,
4.0345e-06,  3.7851e-06,  4.0267e-06,  3.9947e-06,  4.0274e-06,
3.8924e-06,  4.0300e-06,  4.0321e-06,  3.7832e-06,  3.9757e-06,
3.8110e-06,  4.0219e-06,  3.9287e-06,  4.0190e-06,  3.8634e-06,
3.9872e-06,  3.9220e-06,  4.0015e-06,  3.9901e-06,  3.9839e-06,
4.0312e-06,  4.0334e-06,  3.9213e-06,  3.8680e-06,  4.0152e-06,
3.9790e-06,  4.0175e-06,  4.0056e-06,  4.0062e-06,  4.0018e-06,
3.9809e-06,  4.0215e-06,  4.0361e-06,  4.0354e-06,  4.0104e-06,
4.0360e-06,  3.9154e-06,  3.9828e-06,  3.8728e-06,  3.9694e-06,
3.9908e-06,  3.9663e-06,  3.9864e-06,  4.0207e-06,  4.0356e-06,
3.8890e-06,  4.0408e-06,  3.9961e-06,  4.0422e-06,  3.9586e-06,
4.0225e-06,  4.0338e-06,  3.9050e-06,  3.8010e-06,  3.9419e-06,
3.7427e-06,  3.7905e-06,  4.0372e-06,  3.9031e-06,  4.0313e-06,
3.7794e-06,  4.0242e-06,  4.0147e-06,  4.0298e-06,  3.8996e-06,
3.8987e-06,  4.0312e-06,  3.9974e-06,  4.0356e-06,  4.0194e-06,
3.9095e-06,  4.0078e-06,  3.9427e-06,  3.7367e-06,  4.0025e-06,
4.0145e-06,  3.8449e-06,  3.9973e-06,  4.0192e-06,  4.0038e-06,
3.9053e-06,  4.0046e-06,  3.8304e-06,  3.8975e-06,  4.0149e-06,
4.0340e-06,  4.0000e-06,  3.9953e-06,  4.0013e-06,  4.0223e-06,
4.0261e-06,  4.0042e-06,  3.9823e-06,  4.0411e-06,  3.9812e-06,
3.8643e-06,  3.9908e-06,  3.8307e-06,  3.9445e-06,  4.0244e-06,
3.9871e-06,  3.9153e-06,  4.0353e-06,  3.7801e-06,  4.0095e-06,
3.9461e-06,  3.9619e-06,  3.9591e-06,  3.9707e-06,  3.9820e-06,
3.7868e-06,  3.9887e-06,  3.8905e-06,  4.0259e-06,  3.9919e-06,
3.8655e-06,  3.9035e-06,  3.9450e-06,  3.9539e-06,  3.8099e-06,
4.0264e-06,  4.0418e-06,  3.8546e-06,  4.0067e-06,  3.8940e-06,
3.9165e-06,  4.0305e-06,  4.0105e-06,  4.0100e-06,  3.9835e-06,
4.0007e-06,  4.0116e-06,  4.0224e-06,  4.0227e-06,  4.0316e-06,
4.0300e-06,  3.9213e-06,  4.0429e-06,  3.6407e-06,  3.9716e-06,
3.9874e-06,  3.9923e-06,  3.8536e-06,  3.9886e-06,  3.8398e-06,
3.8478e-06,  4.0321e-06,  4.0216e-06,  3.9671e-06,  3.9958e-06,
3.8991e-06,  3.8673e-06,  3.9739e-06,  4.0226e-06,  4.0055e-06,
3.9372e-06,  3.7807e-06,  3.9632e-06,  3.8619e-06,  3.8946e-06,
3.8950e-06,  4.0346e-06,  3.8721e-06,  3.9392e-06,  3.8008e-06,
3.9267e-06,  3.9913e-06,  3.9300e-06,  4.0326e-06,  3.9085e-06,
3.9906e-06,  3.9099e-06,  3.9774e-06,  3.8867e-06,  3.9738e-06,
4.0197e-06,  3.9021e-06,  3.8188e-06,  3.8890e-06,  3.5821e-06,
3.9768e-06,  4.0071e-06,  4.0313e-06,  3.7714e-06,  3.9063e-06,
3.9945e-06,  4.0048e-06,  4.0261e-06,  3.9062e-06,  4.0222e-06,
3.7166e-06,  3.8166e-06,  4.0290e-06,  3.9704e-06,  3.9096e-06,
3.8961e-06,  3.8516e-06,  4.0308e-06,  3.7886e-06,  3.9660e-06,
3.9049e-06,  3.9662e-06,  3.9775e-06,  4.0342e-06,  4.0362e-06,
3.8787e-06,  3.9754e-06,  3.8128e-06,  4.0270e-06,  4.0010e-06,
4.0394e-06,  4.0212e-06,  3.9669e-06,  4.0308e-06,  4.0345e-06,
3.8454e-06,  4.0362e-06,  4.0293e-06,  3.8127e-06,  3.8317e-06,
3.9953e-06,  4.0349e-06,  3.6225e-06,  4.0202e-06,  4.0151e-06,
3.8682e-06,  4.0292e-06,  4.0051e-06,  3.9745e-06,  4.0155e-06,
```

```
          3.9975e-06,  3.9713e-06,  4.0412e-06,  3.7740e-06,  4.0411e-06,
          4.0220e-06,  3.8579e-06,  3.8512e-06,  3.8476e-06,  3.9654e-06,
          4.0359e-06,  3.9298e-06,  4.0266e-06,  4.0114e-06,  3.8781e-06,
          4.0180e-06,  3.8450e-06,  4.0323e-06,  3.9919e-06,  3.7756e-06,
          3.9109e-06,  3.8739e-06,  4.0036e-06,  3.9444e-06,  4.0210e-06,
          3.8866e-06,  4.0329e-06,  4.0314e-06,  3.7002e-06,  3.7254e-06,
          4.0302e-06,  3.8236e-06,  4.0310e-06,  3.8101e-06,  3.9757e-06,
          3.7699e-06,  3.8599e-06,  3.9578e-06,  3.7466e-06,  4.0304e-06,
          4.0067e-06,  3.8314e-06,  3.8701e-06,  3.8718e-06,  3.8205e-06,
          3.9425e-06,  3.9206e-06,  3.9902e-06,  4.0055e-06,  3.9038e-06,
          3.9988e-06,  4.0065e-06,  4.0039e-06,  4.0341e-06,  3.8549e-06,
          3.9951e-06,  4.0328e-06,  4.0020e-06,  3.9608e-06,  3.9370e-06,
          4.0368e-06,  3.9658e-06,  3.7993e-06,  4.0195e-06,  3.8315e-06,
          3.7813e-06,  4.0126e-06,  4.0342e-06,  4.0357e-06,  3.9906e-06,
          4.0349e-06,  4.0308e-06,  3.8979e-06,  3.9308e-06,  3.9499e-06,
          3.9839e-06,  3.9248e-06,  3.9841e-06,  3.8966e-06,  3.7582e-06,
          3.9602e-06,  4.0250e-06,  3.9766e-06,  3.8904e-06,  3.9006e-06,
          3.8872e-06,  3.9652e-06,  3.8133e-06,  3.9827e-06,  3.9998e-06,
          3.8233e-06,  3.8530e-06,  3.8597e-06,  3.8952e-06,  4.0276e-06,
          3.9595e-06,  3.7633e-06,  3.9966e-06,  4.0180e-06,  4.0136e-06,
          3.9144e-06,  3.9090e-06,  3.8668e-06,  3.8201e-06,  3.8470e-06,
          3.8608e-06,  3.8985e-06,  3.8644e-06,  3.9315e-06,  3.8625e-06,
          3.8851e-06,  4.0182e-06,  3.6900e-06,  4.0295e-06,  4.0098e-06,
          3.9049e-06,  3.7819e-06,  3.9972e-06,  4.0066e-06,  4.0411e-06,
          4.0101e-06,  4.0490e-06,  4.0122e-06,  3.7080e-06,  3.9327e-06,
          3.8018e-06,  3.9311e-06,  3.9098e-06,  3.8927e-06,  3.7820e-06,
          3.9577e-06,  4.0236e-06,  3.9983e-06,  3.9136e-06,  3.9888e-06,
          3.9986e-06,  3.9007e-06,  4.0392e-06,  3.8569e-06,  3.8986e-06,
          3.9040e-06,  3.8011e-06,  4.0223e-06,  4.0151e-06,  4.0161e-06,
          4.0010e-06,  4.0338e-06,  4.0206e-06,  4.0066e-06,  4.0238e-06,
          3.9383e-06,  3.9536e-06,  3.8532e-06,  3.8041e-06,  4.0349e-06,
          4.0494e-06,  4.0047e-06,  4.0323e-06,  3.9818e-06,  3.8415e-06,
          3.9960e-06,  3.9528e-06,  3.7754e-06,  4.0304e-06,  3.9826e-06,
          3.8679e-06,  4.0368e-06,  4.0258e-06,  4.0044e-06,  3.9119e-06,
          3.9527e-06,  3.8966e-06,  3.9045e-06,  3.9945e-06,  3.7069e-06])
end of p.grad

False
Epoch 5 finished
Epoch [5/10], Loss: 2.2006
************************************************************

Learning rate: 0.001

another way to print learning rate:
tensor([[[[-6.6103e-03, -7.2798e-03, -7.5098e-03,  …, -6.7229e-03,
           -6.2667e-03, -6.2526e-03],
          [-6.6504e-03, -6.6051e-03, -7.3299e-03,  …, -6.5998e-03,
```

```
  -6.6768e-03, -6.5474e-03],
 [-6.2185e-03, -6.4761e-03, -6.8497e-03,  …, -7.0155e-03,
  -6.7772e-03, -6.4984e-03],
 …,
 [-4.4061e-03, -4.7937e-03, -5.5361e-03,  …, -4.5800e-03,
  -4.5694e-03, -5.0760e-03],
 [-4.5179e-03, -5.5767e-03, -5.9066e-03,  …, -5.5465e-03,
  -5.4325e-03, -5.8461e-03],
 [-5.5773e-03, -5.8871e-03, -6.4512e-03,  …, -5.9118e-03,
  -5.7735e-03, -5.8105e-03]],

[[-5.8690e-03, -6.5819e-03, -7.0341e-03,  …, -6.3911e-03,
  -5.7005e-03, -5.4851e-03],
 [-5.9609e-03, -6.0552e-03, -6.9202e-03,  …, -6.1315e-03,
  -6.0649e-03, -5.5644e-03],
 [-5.6511e-03, -6.0973e-03, -6.6415e-03,  …, -6.2848e-03,
  -5.8817e-03, -5.5115e-03],
 …,
 [-2.9366e-03, -3.3224e-03, -3.8508e-03,  …, -2.4199e-03,
  -2.4186e-03, -2.9577e-03],
 [-2.7117e-03, -3.8002e-03, -3.9917e-03,  …, -3.5103e-03,
  -3.5651e-03, -3.9054e-03],
 [-3.7663e-03, -4.1967e-03, -4.5511e-03,  …, -4.0735e-03,
  -4.0740e-03, -4.2284e-03]],

[[-3.8329e-03, -4.4343e-03, -4.9519e-03,  …, -4.8216e-03,
  -4.2096e-03, -4.1270e-03],
 [-3.5299e-03, -3.5904e-03, -4.5837e-03,  …, -4.7094e-03,
  -4.7510e-03, -4.1332e-03],
 [-3.0340e-03, -3.6776e-03, -4.5302e-03,  …, -4.8699e-03,
  -4.4643e-03, -3.9822e-03],
 …,
 [-4.0860e-04, -1.0368e-03, -1.8084e-03,  …, -9.1457e-04,
  -8.9521e-04, -1.4800e-03],
 [-2.2538e-05, -1.3599e-03, -1.8325e-03,  …, -1.8210e-03,
  -2.0063e-03, -2.3428e-03],
 [-1.0130e-03, -1.6114e-03, -2.2600e-03,  …, -2.3852e-03,
  -2.4679e-03, -2.4754e-03]]],


[[[ 5.9393e-03,  2.8437e-03, -1.7902e-03,  …,  1.1385e-02,
    1.1514e-02,  1.1599e-02],
 [-3.1355e-03, -1.1681e-03, -4.7849e-03,  …,  1.2220e-02,
    1.1778e-02,  1.0639e-02],
 [-2.0023e-03, -1.2800e-03, -5.7717e-04,  …,  9.1068e-03,
    1.0017e-02,  7.5886e-03],
 …,
 [ 1.0925e-02,  1.2673e-02,  8.6481e-03,  …,  1.3875e-02,
```

```
    1.0391e-02,  1.0788e-02],
   [ 5.5999e-03,  6.4859e-03,  7.1449e-03,  …,  1.2821e-02,
     8.5853e-03,  8.1735e-03],
   [ 5.3674e-03,  7.1309e-03,  6.5803e-03,  …,  1.1727e-02,
     8.1570e-03,  5.7994e-03]],

  [[ 5.2457e-03,  1.9025e-03, -1.1961e-03,  …,  8.4145e-03,
     8.3072e-03,  8.2356e-03],
   [-3.1700e-03, -2.5140e-03, -6.2514e-03,  …,  1.0045e-02,
     8.2867e-03,  5.9936e-03],
   [-4.8161e-03, -4.5564e-03, -2.9533e-03,  …,  6.3641e-03,
     5.7721e-03,  3.1603e-03],
   …,
   [ 4.5786e-03,  6.1620e-03,  3.2788e-03,  …,  8.4180e-03,
     3.5517e-03,  3.1953e-03],
   [-2.1857e-03, -2.0292e-03,  5.6594e-04,  …,  6.0921e-03,
     1.6356e-03,  1.2769e-03],
   [-3.0186e-03, -5.8094e-04,  1.2203e-04,  …,  5.1132e-03,
     1.6063e-03, -2.5467e-04]],

  [[-1.1774e-02, -1.4233e-02, -1.7397e-02,  …, -4.8856e-03,
    -2.8175e-03, -2.1435e-03],
   [-1.7727e-02, -1.7911e-02, -2.2310e-02,  …, -3.6006e-03,
    -1.7907e-03, -4.1648e-03],
   [-2.0128e-02, -2.0076e-02, -1.9018e-02,  …, -7.3515e-03,
    -7.0676e-03, -8.8293e-03],
   …,
   [-1.1180e-02, -8.1999e-03, -9.2559e-03,  …, -2.4927e-03,
    -6.8167e-03, -5.8944e-03],
   [-1.7163e-02, -1.4821e-02, -1.2554e-02,  …, -6.2043e-03,
    -8.2382e-03, -8.1792e-03],
   [-1.9484e-02, -1.4129e-02, -1.3576e-02,  …, -8.0844e-03,
    -1.0195e-02, -1.2162e-02]]],


 [[[ 3.9345e-03,  4.0945e-03,  4.4281e-03,  …,  6.5725e-03,
     6.3594e-03,  6.9720e-03],
   [ 3.6514e-03,  3.9598e-03,  3.7746e-03,  …,  7.1571e-03,
     6.3031e-03,  6.9588e-03],
   [ 4.5715e-03,  4.1724e-03,  4.7678e-03,  …,  7.0349e-03,
     6.8578e-03,  7.4733e-03],
   …,
   [ 5.9998e-03,  6.2843e-03,  6.2727e-03,  …,  5.4545e-03,
     6.4154e-03,  7.0402e-03],
   [ 4.5392e-03,  4.9909e-03,  5.4082e-03,  …,  4.6297e-03,
     4.4796e-03,  4.7652e-03],
   [ 3.2341e-03,  4.7097e-03,  5.1713e-03,  …,  3.5741e-03,
     2.6557e-03,  3.4057e-03]],
```

```
 [[ 5.1968e-04,  4.9302e-04,  4.0496e-04,  …,  2.1086e-03,
    1.9084e-03,  2.3128e-03],
  [ 4.2558e-04,  3.5906e-04, -5.3390e-04,  …,  2.5855e-03,
    1.2453e-03,  2.0971e-03],
  [ 1.6552e-03,  8.1435e-04,  1.0901e-03,  …,  2.5484e-03,
    2.2307e-03,  3.1311e-03],
  …,
  [ 3.3009e-03,  3.1961e-03,  3.2000e-03,  …,  1.8348e-03,
    2.7375e-03,  3.3235e-03],
  [ 2.1288e-03,  2.4030e-03,  2.7763e-03,  …,  1.1712e-03,
    1.1864e-03,  1.4463e-03],
  [ 7.0710e-04,  2.1571e-03,  2.5490e-03,  …,  3.3751e-04,
   -2.5467e-04,  7.9251e-04]],

 [[-3.2392e-03, -3.2975e-03, -3.5353e-03,  …, -4.7108e-03,
   -4.4780e-03, -3.8293e-03],
  [-3.7623e-03, -3.5095e-03, -4.5626e-03,  …, -3.9772e-03,
   -4.7126e-03, -4.0104e-03],
  [-2.9662e-03, -3.8179e-03, -3.7785e-03,  …, -3.7467e-03,
   -4.0012e-03, -2.8859e-03],
  …,
  [-1.1080e-03, -1.4986e-03, -1.6338e-03,  …, -3.3481e-03,
   -2.8929e-03, -2.0846e-03],
  [-2.5257e-03, -2.3362e-03, -2.3132e-03,  …, -3.5840e-03,
   -3.6044e-03, -3.6146e-03],
  [-4.8191e-03, -3.2350e-03, -2.8438e-03,  …, -4.6391e-03,
   -5.0099e-03, -3.9042e-03]]],


…,


[[[ 9.3348e-03,  1.0414e-02,  8.5194e-03,  …,  3.3510e-03,
    1.5854e-03,  4.6092e-04],
  [ 9.5602e-03,  8.7930e-03,  5.9446e-03,  …,  3.0524e-03,
    8.2855e-04, -9.9042e-05],
  [ 9.0354e-03,  8.1740e-03,  5.1232e-03,  …,  3.6389e-03,
    1.8612e-03,  4.8222e-04],
  …,
  [ 2.3378e-03,  2.2979e-03,  3.3804e-03,  …,  4.2957e-03,
    4.2130e-03,  4.3721e-03],
  [ 2.8718e-03,  2.3424e-03,  3.2428e-03,  …,  4.6977e-03,
    3.4324e-03,  3.5288e-03],
  [ 2.2388e-03,  1.8908e-03,  3.0302e-03,  …,  4.0103e-03,
    3.8352e-03,  3.6530e-03]],

 [[ 7.4176e-03,  8.6201e-03,  7.0117e-03,  …,  2.6506e-03,
```

```
    1.0487e-03,  2.1586e-04],
   [ 7.7099e-03,  6.9471e-03,  4.4240e-03,  …,  2.5184e-03,
    -1.7528e-05, -7.9411e-04],
   [ 7.0611e-03,  6.2357e-03,  2.9804e-03,  …,  2.4842e-03,
     6.0622e-04, -4.8799e-04],
   …,
   [ 1.3356e-05, -2.3973e-05,  6.5809e-04,  …,  1.4819e-03,
     1.6685e-03,  1.8594e-03],
   [ 9.8400e-04, -6.5603e-05,  6.8964e-04,  …,  2.1568e-03,
     5.8899e-04,  5.9206e-04],
   [ 9.3235e-04, -8.6951e-05,  7.8630e-04,  …,  1.9620e-03,
     1.1548e-03,  8.6992e-04]],

  [[ 1.1194e-02,  1.2488e-02,  1.1001e-02,  …,  8.8240e-03,
     7.8130e-03,  6.7112e-03],
   [ 1.1250e-02,  1.0809e-02,  8.5436e-03,  …,  7.9399e-03,
     6.1674e-03,  5.6926e-03],
   [ 1.1258e-02,  1.0664e-02,  7.5282e-03,  …,  7.8152e-03,
     6.2178e-03,  5.4352e-03],
   …,
   [ 6.5804e-03,  6.0612e-03,  6.7040e-03,  …,  6.2566e-03,
     5.9500e-03,  5.8199e-03],
   [ 6.9719e-03,  5.9984e-03,  6.7358e-03,  …,  5.9882e-03,
     4.5208e-03,  4.7938e-03],
   [ 7.1348e-03,  6.3614e-03,  7.7273e-03,  …,  6.0811e-03,
     4.7122e-03,  4.7642e-03]]],


 [[[ 9.8901e-03,  9.9454e-03,  9.7447e-03,  …,  1.0834e-02,
     1.2343e-02,  9.5107e-03],
   [ 1.0136e-02,  8.6442e-03,  7.6724e-03,  …,  1.0408e-02,
     1.0581e-02,  8.8440e-03],
   [ 1.0574e-02,  8.7702e-03,  8.4492e-03,  …,  1.0983e-02,
     1.0356e-02,  7.0090e-03],
   …,
   [ 1.0727e-02,  1.0927e-02,  1.3500e-02,  …,  1.2622e-02,
     1.0848e-02,  1.3020e-02],
   [ 1.5572e-02,  1.4122e-02,  1.4398e-02,  …,  1.2805e-02,
     1.2212e-02,  1.4044e-02],
   [ 1.5818e-02,  1.5054e-02,  1.3888e-02,  …,  1.6422e-02,
     1.6330e-02,  1.6993e-02]],

  [[ 6.9348e-04,  5.1430e-04,  9.4547e-04,  …,  4.7399e-03,
     7.4907e-03,  5.1996e-03],
   [ 8.6731e-04, -9.5707e-04, -1.1285e-03,  …,  6.0721e-03,
     5.9966e-03,  4.0439e-03],
   [ 1.7323e-03, -8.1017e-04, -7.5466e-04,  …,  5.3961e-03,
     4.5789e-03,  1.5529e-03],
```

```
  …,
   [ 2.0807e-03,  2.7419e-03,  4.1962e-03,  …,  4.2457e-03,
     2.9932e-03,  5.3854e-03],
   [ 5.7038e-03,  4.7861e-03,  5.1830e-03,  …,  2.7555e-03,
     2.9202e-03,  4.8868e-03],
   [ 6.5857e-03,  5.4785e-03,  4.1750e-03,  …,  7.1047e-03,
     6.3115e-03,  6.6695e-03]],

  [[ 9.3363e-03,  9.9290e-03,  1.0596e-02,  …,  1.7595e-02,
     2.1156e-02,  1.8791e-02],
   [ 9.4810e-03,  7.7266e-03,  9.0119e-03,  …,  1.7712e-02,
     1.8137e-02,  1.6641e-02],
   [ 1.0926e-02,  8.9062e-03,  1.0255e-02,  …,  1.7301e-02,
     1.6339e-02,  1.3993e-02],
   …,
   [ 1.1314e-02,  1.2445e-02,  1.4620e-02,  …,  1.3939e-02,
     1.2235e-02,  1.4104e-02],
   [ 1.5093e-02,  1.4148e-02,  1.5506e-02,  …,  1.1674e-02,
     1.2080e-02,  1.4137e-02],
   [ 1.5745e-02,  1.5144e-02,  1.5648e-02,  …,  1.7329e-02,
     1.6716e-02,  1.7456e-02]]],


[[[-2.8750e-03, -2.3968e-04, -2.8688e-03,  …, -4.1883e-03,
    -3.7784e-03, -6.1623e-03],
   [-3.6769e-03, -3.6479e-03, -4.5978e-03,  …, -3.4686e-03,
    -6.6559e-03, -5.5116e-03],
   [-3.2114e-03, -1.8200e-03, -1.5998e-03,  …, -4.2493e-03,
    -5.8899e-03, -4.7676e-03],
   …,
   [-3.0896e-03, -3.7610e-03, -3.7419e-03,  …, -2.2219e-03,
    -1.5288e-03, -1.2200e-03],
   [-2.4669e-03, -3.3180e-03, -3.5080e-03,  …, -3.7488e-03,
    -2.5999e-03, -2.4157e-03],
   [-2.7015e-03, -3.8195e-03, -3.6367e-03,  …, -3.9723e-03,
    -1.2247e-03,  3.3243e-04]],

  [[-6.9410e-03, -4.1724e-03, -6.9344e-03,  …, -5.5186e-03,
    -5.3463e-03, -7.9002e-03],
   [-6.6873e-03, -6.4891e-03, -7.5998e-03,  …, -6.3779e-03,
    -9.4283e-03, -7.7892e-03],
   [-5.9817e-03, -4.6484e-03, -4.5861e-03,  …, -7.3557e-03,
    -8.3216e-03, -7.3629e-03],
   …,
   [-5.8528e-03, -6.4444e-03, -7.2027e-03,  …, -4.5119e-03,
    -3.3782e-03, -3.7669e-03],
   [-4.4002e-03, -5.6382e-03, -6.4007e-03,  …, -6.4936e-03,
    -5.6707e-03, -5.0984e-03],
```

```
          [-4.2679e-03, -6.2600e-03, -6.2776e-03,  …, -6.2990e-03,
           -4.1635e-03, -2.5961e-03]],

         [[-1.1350e-02, -8.0628e-03, -9.8902e-03,  …, -9.5766e-03,
           -8.8592e-03, -1.0156e-02],
          [-1.0029e-02, -9.4616e-03, -9.3529e-03,  …, -1.0150e-02,
           -1.1854e-02, -9.1529e-03],
          [-8.4815e-03, -6.9669e-03, -6.5095e-03,  …, -9.1537e-03,
           -1.0069e-02, -8.8572e-03],
          …,
          [-6.7444e-03, -7.0379e-03, -8.2921e-03,  …, -6.2444e-03,
           -5.8895e-03, -6.4675e-03],
          [-4.1391e-03, -5.0517e-03, -5.6706e-03,  …, -8.2274e-03,
           -6.6263e-03, -6.2187e-03],
          [-2.4564e-03, -4.9037e-03, -4.6347e-03,  …, -7.2801e-03,
           -5.5406e-03, -4.1617e-03]]])
tensor([ 4.6566e-09,  3.9116e-08,  2.3749e-08,  5.6752e-09, -1.1176e-08,
        -9.3132e-09, -6.5193e-09, -3.9116e-08,  1.8626e-09,  1.3737e-08,
        -1.8161e-08, -1.0827e-08, -5.1223e-09, -3.6787e-08, -2.3283e-09,
         5.2387e-09,  8.0327e-09,  1.3039e-08,  1.3039e-08, -1.8626e-08,
         2.1741e-08, -1.8626e-09, -3.9116e-08, -4.6566e-09,  4.5984e-09,
         1.8510e-08,  7.6834e-09,  2.0023e-08, -1.6764e-08, -4.8429e-08,
         6.5193e-09,  9.4878e-09,  5.7626e-09,  3.5798e-09, -4.1910e-08,
        -7.3196e-09, -9.3132e-09, -6.9849e-09, -4.6566e-10, -1.3621e-08,
        -1.9791e-09, -1.4435e-08, -2.5146e-08, -2.5611e-09,  0.0000e+00,
        -2.8871e-08,  2.4913e-08,  5.3551e-09, -6.5193e-09, -9.3132e-09,
         1.6298e-09,  1.2107e-08,  5.5588e-09,  3.7253e-09,  1.9558e-08,
         1.1176e-08,  8.5682e-08,  1.0943e-08, -4.4238e-09,  1.0594e-08,
        -5.5879e-08,  9.5461e-09, -1.0710e-08, -8.6147e-09,  4.6566e-09,
        -2.2817e-08, -4.0978e-08, -1.3039e-08, -3.0268e-09,  8.3819e-09,
         2.0023e-08,  5.5879e-09,  1.0710e-08, -2.6776e-08,  9.3132e-09,
         6.2864e-09, -1.3970e-09, -6.5193e-09, -7.2643e-08,  8.3819e-09,
        -7.4506e-08,  6.5193e-09, -3.4459e-08, -4.6566e-09,  5.5879e-08,
        -9.2201e-08, -2.9337e-08, -2.7940e-09,  6.2864e-09, -9.3132e-10,
        -1.4203e-08, -4.0513e-08,  1.1176e-08,  4.4238e-09, -1.2689e-08,
        -6.4727e-08])
tensor([ 0.0343,  0.0370, -0.0182,  0.0085, -0.0624,  0.0032,  0.0092,  0.0206,
         0.0122, -0.0126, -0.0204,  0.0058, -0.0038,  0.0238,  0.0170,  0.0151,
        -0.0160,  0.0496, -0.0002,  0.0249,  0.0078, -0.0222, -0.0615, -0.0044,
        -0.0117, -0.0056,  0.0141, -0.0183, -0.0017,  0.0060,  0.0036, -0.0003,
        -0.0260,  0.0083, -0.0113,  0.0044, -0.0091,  0.0356, -0.0332,  0.0315,
        -0.0350, -0.0200, -0.0178,  0.0077, -0.0064, -0.0009, -0.0152, -0.0117,
         0.0396, -0.0054,  0.0232,  0.0127, -0.0178,  0.0448, -0.0235, -0.0576,
        -0.0943,  0.0198,  0.0089, -0.0132,  0.0351,  0.0135,  0.0104,  0.0154,
         0.0240,  0.0034,  0.0208, -0.0094,  0.0080, -0.0049,  0.0115, -0.0046,
         0.0358, -0.0108, -0.0149, -0.0048,  0.0091,  0.0299, -0.0121,  0.0210,
        -0.0054, -0.0440,  0.0106,  0.0151,  0.0089, -0.0168,  0.0143,  0.0009,
         0.0065, -0.0224,  0.0552,  0.0196, -0.0394,  0.0205,  0.0493, -0.0105])
```

```
tensor([ 0.0302, -0.0271, -0.0081,  0.0029, -0.0261,  0.0023,  0.0023,  0.0017,
         0.0031, -0.0068, -0.0037,  0.0114,  0.0068,  0.0220,  0.0160,  0.0081,
        -0.0066,  0.0151, -0.0006,  0.0063,  0.0073, -0.0031, -0.0396, -0.0057,
        -0.0071,  0.0045,  0.0076, -0.0117,  0.0025, -0.0100,  0.0041,  0.0008,
        -0.0163,  0.0035,  0.0030,  0.0038,  0.0055,  0.0115,  0.0009,  0.0232,
         0.0088, -0.0148, -0.0109,  0.0048, -0.0096,  0.0021, -0.0083, -0.0069,
         0.0301,  0.0072,  0.0155, -0.0125, -0.0113,  0.0308, -0.0124, -0.0308,
        -0.0283,  0.0167,  0.0064, -0.0093,  0.0239,  0.0079,  0.0009,  0.0117,
         0.0193,  0.0007,  0.0191, -0.0235,  0.0029, -0.0038,  0.0055,  0.0005,
         0.0229,  0.0048, -0.0012, -0.0085,  0.0050,  0.0250, -0.0035,  0.0193,
        -0.0130, -0.0348,  0.0109,  0.0136, -0.0254, -0.0037,  0.0135, -0.0150,
         0.0048, -0.0111,  0.0317, -0.0158, -0.0050,  0.0179,  0.0318,  0.0116])
tensor([[[[ 2.3131e-03,  2.5926e-03,  2.5423e-03,  2.4660e-03,  2.7757e-03],
          [ 2.0732e-03,  2.0269e-03,  2.3049e-03,  2.9195e-03,  3.2142e-03],
          [ 5.8013e-04,  1.3462e-03,  1.7548e-03,  2.2272e-03,  2.2165e-03],
          [ 2.9903e-04,  8.2561e-04,  1.2333e-03,  1.7381e-03,  1.3721e-03],
          [ 2.7919e-04,  4.5396e-04,  9.0029e-04,  1.2709e-03,  1.1295e-03]],

         [[ 2.3663e-03,  2.2103e-03,  1.1909e-03, -1.5328e-04,  1.0340e-05],
          [ 1.9343e-03,  1.4908e-03,  1.9134e-04,  1.4050e-03,  4.5092e-04],
          [ 9.8472e-04,  4.4769e-04, -1.0549e-03,  1.0898e-03,  6.5292e-04],
          [-3.4547e-04, -1.2829e-03, -5.6873e-04,  5.6856e-04,  2.0664e-04],
          [ 5.9288e-04, -8.6037e-04, -5.6605e-04, -6.5562e-04, -1.5123e-04]],

         [[ 9.4375e-04,  1.2460e-03,  1.0010e-03,  1.2637e-03,  1.0496e-03],
          [ 1.0840e-03,  1.2570e-03,  1.0381e-03,  8.2556e-04,  1.1815e-03],
          [ 1.2458e-03,  1.2595e-03,  9.7817e-04,  7.6188e-04,  9.0074e-04],
          [ 1.1201e-03,  1.0467e-03,  9.1317e-04,  9.7234e-04,  8.7136e-04],
          [ 1.6223e-03,  1.5285e-03,  1.2667e-03,  1.2006e-03,  1.2646e-03]],

         …,

         [[ 2.9881e-03,  3.4428e-03,  3.6215e-03,  4.3347e-03,  4.0219e-03],
          [ 3.5300e-03,  4.1600e-03,  4.3490e-03,  4.4140e-03,  4.6711e-03],
          [ 3.5782e-03,  3.8607e-03,  4.0387e-03,  4.3717e-03,  4.0742e-03],
          [ 2.7831e-03,  3.0701e-03,  3.1595e-03,  3.9018e-03,  3.2203e-03],
          [ 2.6417e-03,  2.9091e-03,  2.6999e-03,  2.7961e-03,  2.9451e-03]],

         [[ 2.7066e-03,  2.7913e-03,  2.6925e-03,  3.3451e-03,  2.8836e-03],
          [ 2.8607e-03,  2.9851e-03,  3.0535e-03,  3.3891e-03,  3.0353e-03],
          [ 2.6383e-03,  2.9674e-03,  3.1969e-03,  3.5855e-03,  2.9680e-03],
          [ 1.5997e-03,  1.9411e-03,  2.4720e-03,  2.7958e-03,  2.1123e-03],
          [ 9.0797e-04,  1.1207e-03,  1.0219e-03,  1.2893e-03,  1.2380e-03]],

         [[ 4.1629e-03,  4.5357e-03,  4.6564e-03,  5.2061e-03,  4.0914e-03],
          [ 4.2813e-03,  4.6390e-03,  4.5340e-03,  4.4816e-03,  4.1358e-03],
          [ 3.4832e-03,  3.6070e-03,  3.7385e-03,  4.0239e-03,  3.5581e-03],
          [ 2.4772e-03,  2.4633e-03,  2.5053e-03,  2.9604e-03,  2.3169e-03],
```

```
  [ 3.0780e-03,  3.2443e-03,  2.9129e-03,  2.4705e-03,  2.2114e-03]]],


[[[-9.6671e-04, -1.1751e-03, -1.6763e-03, -1.1533e-03, -1.3339e-03],
  [-9.2377e-04, -9.6999e-04, -1.1621e-03, -7.8722e-04, -5.8985e-04],
  [ 1.6499e-05,  3.4845e-04, -3.9557e-04, -3.6202e-04, -4.8245e-04],
  [ 8.7899e-05,  1.0572e-03,  1.4980e-04, -1.1916e-04, -3.9274e-04],
  [-7.3078e-04,  1.7155e-04, -4.8381e-04, -1.5001e-04, -1.8116e-04]],

 [[-1.4498e-04, -3.6030e-05, -8.0921e-04, -1.0448e-03, -7.0234e-04],
  [ 3.1425e-04, -4.8316e-05, -2.0589e-03, -1.5075e-03, -6.3686e-04],
  [-4.3562e-04,  3.8575e-04, -1.5445e-03, -9.8016e-04, -6.5068e-04],
  [-3.6497e-04,  1.1808e-03, -3.8323e-04, -1.3932e-04, -3.5133e-04],
  [-5.4527e-04,  9.9051e-04, -4.6275e-04, -6.7751e-04,  7.1229e-06]],

 [[ 5.4049e-04,  1.1312e-03,  1.7873e-03,  1.9398e-03,  2.0167e-03],
  [ 3.0524e-04,  8.0558e-04,  1.2751e-03,  1.3257e-03,  1.5503e-03],
  [ 2.4542e-04,  4.4791e-04,  7.7658e-04,  8.6471e-04,  9.8517e-04],
  [ 2.6328e-04,  3.0406e-04,  4.3655e-04,  3.8896e-04,  2.9104e-04],
  [ 5.5006e-05,  9.7958e-06,  2.3820e-04,  3.4184e-04,  1.8269e-04]],

 …,

 [[ 4.2433e-04,  6.9771e-04,  1.0727e-03,  1.4905e-03,  1.2384e-03],
  [-2.8424e-04,  3.0275e-05,  4.3857e-04,  4.1043e-04,  5.8860e-04],
  [-2.0392e-04, -2.8847e-04, -1.1037e-04, -2.9809e-05, -1.5990e-04],
  [-1.5941e-04, -1.1850e-04,  6.1389e-05, -3.2230e-04, -6.9299e-04],
  [-4.5312e-04, -1.9199e-04,  5.1302e-05,  1.8611e-04, -9.5386e-05]],

 [[ 6.1040e-05,  1.3805e-04,  8.6282e-05,  3.8809e-04,  3.0046e-04],
  [-5.1338e-04, -2.1065e-04, -1.8412e-04, -1.0118e-04, -3.7432e-05],
  [-6.8497e-04, -5.0375e-04, -3.6500e-04, -2.5842e-04, -2.3609e-04],
  [-5.7694e-04, -4.8989e-04, -3.3183e-04, -2.2195e-04, -3.0125e-04],
  [-5.9033e-04, -3.8250e-04, -2.1830e-04, -2.0514e-04, -3.9050e-04]],

 [[ 3.8540e-04,  7.0527e-04,  1.0522e-03,  1.5928e-03,  1.5661e-03],
  [-1.2472e-04,  3.4819e-04,  7.4295e-04,  5.6189e-04,  5.6392e-04],
  [-2.7331e-04, -1.3346e-04,  1.0890e-04,  3.0752e-04,  7.5049e-05],
  [ 1.0679e-04,  3.5008e-04,  4.9618e-04, -2.0543e-04, -7.2407e-04],
  [ 2.6305e-04,  4.0788e-04,  7.3146e-04,  5.4271e-04,  4.0878e-05]]],


[[[-5.8458e-04, -9.8516e-04, -1.4896e-03, -4.7313e-04, -8.2746e-04],
  [-5.4353e-04, -2.9960e-04, -8.6044e-04, -2.4546e-04, -4.0347e-04],
  [-3.2296e-04, -1.8194e-04, -2.6446e-04, -6.9666e-05, -3.0468e-05],
  [-7.9031e-04, -8.5908e-04, -1.3001e-03, -6.7780e-04, -3.5817e-04],
  [-4.7715e-04, -5.0092e-04, -5.5452e-04, -1.3266e-04,  6.1714e-05]],
```

```
 [[-2.5438e-03, -2.9851e-03, -1.4621e-03, -3.0355e-03, -9.3649e-04],
  [-2.7938e-03, -3.4903e-03, -2.9524e-03, -2.7839e-03, -1.3265e-03],
  [-2.7612e-03, -3.4209e-03, -1.3049e-03, -1.1781e-03, -1.1624e-03],
  [-3.7204e-03, -3.8062e-03, -2.6587e-03, -2.0364e-03, -2.0987e-03],
  [-3.4887e-03, -3.3378e-03, -2.7059e-03, -2.3766e-03, -2.7730e-03]],

 [[-3.2320e-05, -3.1516e-04, -5.1531e-04, -5.9334e-04, -7.1408e-04],
  [-6.2929e-05, -4.4446e-04, -5.8788e-04, -8.2030e-04, -8.4504e-04],
  [ 3.6342e-04, -2.1553e-04, -4.7601e-04, -1.6130e-04, -1.6588e-04],
  [-1.4662e-04, -6.3088e-04, -8.7655e-04, -6.5164e-04, -5.8804e-04],
  [ 2.7489e-04, -3.5382e-04, -3.7506e-04, -2.0273e-04, -3.4376e-04]],

 …,

 [[ 2.5639e-03,  2.4762e-03,  1.9544e-03,  1.8182e-03,  1.7151e-03],
  [ 2.4436e-03,  2.0541e-03,  1.9393e-03,  1.2143e-03,  1.2448e-03],
  [ 2.1127e-03,  1.6369e-03,  1.4344e-03,  1.6174e-03,  1.8672e-03],
  [ 2.0172e-03,  1.4508e-03,  9.9738e-04,  8.8797e-04,  9.7558e-04],
  [ 1.4971e-03,  9.4333e-04,  9.3403e-04,  8.8392e-04,  8.8860e-04]],

 [[ 3.0070e-03,  3.0950e-03,  2.9397e-03,  2.5525e-03,  2.5009e-03],
  [ 2.5573e-03,  2.6594e-03,  2.2466e-03,  1.8343e-03,  1.8582e-03],
  [ 2.1133e-03,  2.1158e-03,  1.8862e-03,  1.6622e-03,  1.8340e-03],
  [ 1.5758e-03,  1.5005e-03,  1.1203e-03,  6.7035e-04,  9.5487e-04],
  [ 1.1899e-03,  1.0152e-03,  6.3911e-04,  7.1857e-04,  1.0940e-03]],

 [[ 3.1827e-03,  2.8475e-03,  1.9562e-03,  1.7685e-03,  1.8474e-03],
  [ 2.7085e-03,  2.1854e-03,  1.9311e-03,  1.3260e-03,  1.3975e-03],
  [ 2.3283e-03,  1.6928e-03,  1.4535e-03,  2.0848e-03,  2.1509e-03],
  [ 2.2065e-03,  1.6747e-03,  1.5608e-03,  1.4108e-03,  1.0023e-03],
  [ 1.6562e-03,  1.6460e-03,  1.2837e-03,  1.5287e-03,  1.6461e-03]]],


…,


[[[-6.4782e-04, -4.9055e-04, -1.3915e-04,  3.8151e-04, -1.8433e-04],
  [-9.3217e-04, -8.4453e-04, -5.6782e-04, -1.6591e-04,  2.4670e-04],
  [-1.4465e-03, -1.5638e-03, -1.0506e-03, -3.9221e-04, -7.4499e-05],
  [-1.4201e-03, -1.4883e-03, -1.7387e-03, -1.3714e-03, -9.0845e-04],
  [-1.4445e-03, -1.6322e-03, -1.5772e-03, -1.6259e-03, -1.1830e-03]],

 [[-1.1392e-03, -5.0664e-04, -6.2782e-04,  3.0934e-04,  1.7713e-04],
  [-1.4547e-03, -1.3302e-03, -1.1106e-03, -4.8254e-04,  6.4709e-04],
  [-1.3790e-03, -1.3034e-03, -1.4889e-03, -6.0475e-05,  6.4952e-04],
  [-9.4374e-04, -7.4783e-04, -3.2082e-04,  3.6134e-04,  4.5796e-04],
  [-5.1254e-04, -4.1799e-04, -1.5093e-04,  3.8175e-04,  7.7141e-05]],
```

```
 [[ 9.2031e-04,  8.4586e-04,  1.1319e-03,  9.8905e-04,  1.1925e-03],
  [ 1.2345e-03,  1.2131e-03,  1.3956e-03,  1.1690e-03,  1.3691e-03],
  [ 1.6363e-03,  1.4932e-03,  1.8193e-03,  1.4879e-03,  1.2898e-03],
  [ 1.8236e-03,  1.7319e-03,  2.0114e-03,  1.9001e-03,  1.5118e-03],
  [ 1.8742e-03,  1.8216e-03,  2.1335e-03,  1.9619e-03,  1.5052e-03]],

 …,

 [[ 2.4008e-03,  2.1885e-03,  2.2529e-03,  2.0424e-03,  2.5949e-03],
  [ 2.5341e-03,  2.3607e-03,  2.5146e-03,  1.9446e-03,  1.9909e-03],
  [ 2.5213e-03,  2.5984e-03,  2.7259e-03,  2.0924e-03,  1.5475e-03],
  [ 2.1736e-03,  2.3595e-03,  2.3405e-03,  2.0156e-03,  1.5469e-03],
  [ 2.1944e-03,  2.3422e-03,  2.2835e-03,  2.0516e-03,  1.4790e-03]],

 [[ 2.2801e-03,  1.9707e-03,  1.8470e-03,  1.7510e-03,  1.8654e-03],
  [ 2.0540e-03,  1.9555e-03,  1.8615e-03,  1.4832e-03,  1.6407e-03],
  [ 1.9076e-03,  1.9549e-03,  1.8773e-03,  1.4661e-03,  1.2679e-03],
  [ 1.6590e-03,  1.8425e-03,  1.6308e-03,  1.2568e-03,  9.6403e-04],
  [ 1.4510e-03,  1.6247e-03,  1.5025e-03,  1.0906e-03,  8.2470e-04]],

 [[ 2.4778e-03,  2.7303e-03,  2.8840e-03,  2.4407e-03,  2.6015e-03],
  [ 2.8949e-03,  2.7809e-03,  3.3679e-03,  2.6434e-03,  2.2672e-03],
  [ 2.8642e-03,  3.0736e-03,  3.6355e-03,  3.0041e-03,  2.4020e-03],
  [ 2.8099e-03,  2.9637e-03,  3.0706e-03,  2.9951e-03,  2.5833e-03],
  [ 3.0493e-03,  3.0862e-03,  3.3154e-03,  2.9752e-03,  2.4535e-03]]],


[[[ 5.1002e-04,  5.2716e-04,  3.8527e-04, -6.2940e-04, -3.8799e-04],
  [ 7.3600e-04,  2.2600e-04, -1.2496e-04,  3.1037e-05, -4.5200e-04],
  [-6.0594e-04, -8.6268e-04, -5.8952e-04,  5.3673e-04,  5.4167e-04],
  [-7.6481e-04, -5.5939e-04, -3.8795e-04, -1.2786e-05,  2.1870e-05],
  [-7.8448e-04, -5.0064e-04, -4.4504e-04, -2.7924e-05, -2.2987e-04]],

 [[-1.3837e-03,  1.4014e-04,  5.3248e-04, -5.6148e-04, -7.9159e-04],
  [-9.3818e-04, -1.4830e-04,  9.7718e-04,  8.4170e-04,  3.4435e-05],
  [-2.8867e-04, -1.1309e-03, -5.1260e-04,  7.2088e-04,  7.7403e-04],
  [ 1.7641e-04, -9.9044e-04, -6.2456e-04,  1.1063e-03,  2.2531e-03],
  [-4.5242e-04, -1.1159e-03, -1.1700e-03, -3.8701e-05,  2.9960e-04]],

 [[ 3.8659e-04,  3.0532e-04,  8.3039e-04,  8.3958e-04,  8.9328e-04],
  [ 2.5313e-04, -1.5011e-04,  4.5149e-05,  5.4191e-04,  8.3436e-04],
  [-4.0022e-04, -3.0110e-04, -1.0892e-04,  2.9927e-04,  5.6600e-04],
  [-9.1827e-05,  3.2110e-04,  5.0538e-04,  3.1443e-04,  1.0778e-04],
  [-2.7389e-04, -4.5492e-05,  3.8900e-04,  3.9302e-04,  3.2193e-04]],

 …,

 [[ 8.3721e-04,  6.3146e-04,  9.8162e-04,  1.2134e-03,  8.9631e-04],
```

```
  [ 2.8669e-04,  9.7313e-05,  3.8360e-06,  6.0116e-04,  3.3288e-04],
  [ 6.2779e-05,  1.5820e-04, -9.3360e-05,  9.5692e-06,  1.0797e-05],
  [-1.7788e-04,  2.3934e-04,  3.7698e-06, -4.0925e-04, -4.0158e-04],
  [-2.2868e-04, -8.3495e-05, -6.1052e-05,  4.7222e-05, -3.9558e-04]],

 [[ 6.9201e-04,  5.2961e-04,  5.8589e-04,  5.7775e-04,  1.1175e-04],
  [ 4.9219e-04,  4.8465e-04,  3.3385e-04,  1.1347e-04, -1.4270e-04],
  [ 3.2668e-04,  2.6028e-04, -1.1132e-04, -2.3804e-04, -3.9587e-04],
  [-6.3439e-05, -7.1690e-05, -6.0735e-04, -7.7660e-04, -6.2673e-04],
  [-9.4495e-05, -1.3932e-04, -4.5685e-04, -7.1448e-04, -8.3376e-04]],

 [[ 4.6830e-04,  4.4110e-04,  6.7519e-04,  1.0535e-03,  1.0460e-03],
  [-1.6543e-04,  9.4567e-05, -2.7452e-04,  2.5956e-04,  2.0546e-04],
  [-6.7115e-04, -5.4222e-05, -3.1129e-04, -3.4283e-04, -2.8123e-04],
  [-4.0339e-04,  3.0586e-05,  1.0729e-05, -4.5892e-04, -6.1288e-04],
  [-4.6795e-04, -2.2225e-04, -2.8400e-06, -2.7990e-04, -8.2521e-04]]],


[[[-4.7741e-04, -6.9860e-04, -7.2589e-04, -2.9225e-04, -6.2876e-04],
  [-1.9166e-04, -4.4866e-04, -7.6332e-04, -7.9708e-04, -2.2991e-04],
  [-1.6627e-04, -1.9201e-04, -5.4685e-04, -3.3583e-04, -2.0940e-04],
  [ 5.2290e-04,  7.2426e-04,  1.8964e-04,  1.0816e-04,  1.0200e-04],
  [ 7.1111e-04,  1.0984e-03,  7.0120e-04,  1.1512e-03,  6.1709e-04]],

 [[ 4.8694e-03,  3.0664e-03,  2.1200e-03,  3.4923e-03,  3.3686e-03],
  [ 4.8538e-03,  3.8415e-03,  2.5229e-03,  3.0549e-03,  3.2175e-03],
  [ 3.6935e-03,  4.4646e-03,  2.4184e-03,  3.2198e-03,  4.2995e-03],
  [ 2.8568e-03,  3.9630e-03,  3.3049e-03,  4.3344e-03,  5.7455e-03],
  [ 3.4429e-03,  4.2828e-03,  5.0298e-03,  5.9582e-03,  6.5057e-03]],

 [[ 2.3200e-04,  5.3710e-04,  2.8541e-04,  5.5478e-05,  3.1842e-04],
  [ 1.7245e-04,  5.1800e-04,  2.5001e-04,  1.8898e-04,  5.4142e-04],
  [ 8.2002e-04,  1.0194e-03,  9.4409e-04,  8.7359e-04,  9.9803e-04],
  [ 1.3280e-03,  9.3105e-04,  7.6246e-04,  6.8199e-04,  9.4318e-04],
  [ 1.5245e-03,  1.0707e-03,  6.0665e-04,  4.6895e-04,  6.3377e-04]],

 …,

 [[-3.9885e-04, -2.1281e-04, -3.6146e-04, -6.9641e-04, -4.9050e-04],
  [-5.2990e-04, -1.3818e-04, -6.5258e-05, -2.7808e-04, -2.3751e-04],
  [-1.7502e-04,  4.3937e-05,  1.9009e-05,  7.8837e-05,  2.8502e-04],
  [-5.9740e-06,  1.0127e-05, -2.5382e-04, -9.3376e-05,  4.7842e-04],
  [ 3.6160e-04,  2.2430e-04, -2.0550e-04, -2.5834e-04,  1.4093e-04]],

 [[-3.8551e-04, -2.9170e-04, -4.7035e-04, -6.1419e-04, -4.4207e-04],
  [-4.3354e-04, -2.5433e-04, -4.7892e-04, -6.2983e-04, -5.1533e-04],
  [-3.5046e-04, -3.0327e-04, -4.4930e-04, -4.8724e-04, -3.6571e-04],
  [-2.3355e-04, -2.0233e-04, -4.3278e-04, -3.5680e-04, -8.0181e-05],
```

```
          [-1.0572e-04, -1.5636e-04, -4.2819e-04, -4.8831e-04, -3.0335e-04]],

         [[-5.5074e-04, -3.0158e-04, -4.4414e-04, -5.0465e-04,  1.8802e-04],
          [-4.6580e-04,  5.3547e-06,  2.3442e-05,  1.0499e-04,  4.7010e-04],
          [-1.1924e-04,  2.8416e-04,  5.0175e-04,  4.6172e-04,  8.0641e-04],
          [-9.9442e-05,  1.3134e-04,  3.7011e-05,  3.1481e-04,  8.6023e-04],
          [ 3.6312e-04,  1.6587e-04,  2.0497e-04,  3.9323e-04,  8.6977e-04]]]])
tensor([-3.9695e-09,  2.5257e-09, -4.1948e-09, -4.6657e-09, -1.2072e-09,
         2.2808e-09,  1.7535e-09, -1.1818e-09,  8.7402e-10, -4.7780e-09,
        -7.4095e-10,  7.1225e-11,  3.7854e-09, -7.6568e-11, -3.8194e-09,
         4.3201e-10,  9.1357e-10, -1.0963e-08,  4.5867e-09, -4.6242e-11,
         6.8394e-10,  9.3394e-11,  4.0239e-10,  1.0606e-09, -2.2084e-10,
        -2.4426e-09,  2.8887e-09, -8.9536e-09,  2.0364e-08,  3.1484e-09,
        -1.9057e-09, -1.5404e-09,  2.6221e-09, -4.5574e-09,  1.0593e-09,
        -2.8862e-09,  7.3074e-09,  3.1309e-09, -1.4766e-10, -1.2341e-09,
        -6.2300e-11,  2.2479e-09, -1.0901e-09,  1.6862e-09,  6.7303e-11,
         3.1597e-09, -1.7450e-09,  5.2322e-09,  4.6636e-09,  2.4525e-09,
        -5.8526e-09, -4.7149e-09, -1.2446e-08,  1.2789e-09, -7.6704e-09,
         5.9508e-09, -2.0231e-10, -1.3075e-09, -3.9222e-11, -9.3559e-09,
         6.0845e-10, -1.1605e-09,  2.2090e-09, -3.0499e-10, -5.5891e-10,
        -6.7034e-09,  1.4566e-09, -8.9066e-09, -1.1446e-08, -3.3041e-09,
        -1.0986e-09, -4.2269e-10, -5.7211e-09, -1.1507e-09, -7.3744e-09,
        -9.5825e-09,  2.3050e-09, -7.9672e-10, -9.3723e-10, -1.5824e-09,
        -1.7460e-09, -1.5314e-09, -6.6672e-10,  8.8482e-10, -6.5948e-10,
         2.5955e-09, -3.7990e-09,  6.7507e-10,  4.8467e-09, -4.5830e-10,
         9.7874e-10, -1.8298e-09,  1.4199e-09, -6.2580e-09, -2.4862e-09,
         1.1194e-09, -5.2660e-10,  1.4276e-09, -4.9795e-10,  1.2818e-09,
        -1.6503e-09,  3.8061e-09,  9.3678e-10,  2.9629e-09,  1.8124e-09,
         7.0759e-10, -1.3434e-09,  2.4735e-09, -4.0386e-09, -1.4543e-09,
        -2.6466e-10, -3.7548e-09, -2.3715e-09, -1.5848e-09,  1.1164e-09,
        -3.0737e-09,  8.0490e-10,  5.0943e-10,  1.7016e-09,  2.3944e-09,
         9.9759e-10, -3.7970e-09,  1.6298e-09,  2.4686e-09, -4.2449e-10,
        -1.1621e-09, -7.2218e-09, -3.1609e-09, -9.1609e-10, -5.8251e-09,
         7.2805e-10,  3.4235e-09, -5.4487e-10, -2.0324e-09, -1.3802e-08,
         1.9184e-09,  4.6259e-09,  4.4502e-09, -7.7137e-10,  1.9711e-09,
        -3.5519e-09,  4.4861e-10, -1.7496e-09,  1.0868e-10,  3.0786e-10,
         2.8612e-09,  1.3929e-09,  8.9020e-09,  6.9945e-09, -3.0588e-09,
        -4.1402e-09, -1.3810e-09,  9.3161e-10,  3.6723e-09, -3.7608e-09,
        -1.4317e-09,  2.3306e-10, -6.9429e-10, -2.3108e-09, -4.6014e-09,
         2.9261e-09,  1.2087e-09, -1.6723e-09,  3.1832e-10,  1.1703e-09,
         7.8694e-10,  3.4474e-09,  2.2083e-09,  1.0738e-09,  3.3806e-09,
         4.9631e-09, -3.4652e-10, -2.6434e-09, -1.1183e-08,  6.7848e-10,
         6.3367e-09,  4.1211e-10, -3.8901e-09, -9.0917e-11,  5.7594e-10,
         6.5802e-10, -1.1811e-09,  8.7903e-10, -2.9795e-09, -8.4992e-10,
        -1.1914e-09, -8.8455e-09, -8.2792e-10,  1.3814e-09, -2.2458e-09,
        -8.6121e-10,  7.7904e-10,  1.5280e-09, -1.8740e-09, -7.2032e-10,
         5.8404e-09,  1.2203e-09,  7.5283e-10, -8.0260e-10,  2.9037e-09,
        -2.5135e-09,  2.0898e-09, -6.7791e-10,  5.4877e-09, -1.2728e-09,
```

```
         1.6995e-09,  3.6807e-09,  2.5962e-09, -4.2897e-10,  4.5208e-10,
        -1.2893e-09, -1.2619e-09,  1.5499e-09,  3.5233e-09, -8.6777e-09,
         3.8075e-09, -8.8329e-10, -1.3983e-11,  1.2014e-09, -1.5307e-09,
        -4.5796e-10, -2.1505e-09, -6.2905e-09, -7.1446e-10, -3.2367e-09,
        -1.4529e-09,  9.9104e-10, -2.5201e-09, -3.5834e-10,  8.8806e-10,
         4.8387e-10,  1.3482e-09,  2.4261e-09,  5.8732e-09,  1.6139e-09,
         1.3587e-08, -2.4424e-09,  1.4243e-09, -1.3331e-09, -9.8748e-10,
        -9.9135e-11, -3.5207e-09, -4.6967e-09, -4.2812e-10, -7.3078e-09,
         4.3099e-10,  1.9306e-09, -6.2707e-09,  1.8217e-09,  6.9151e-09,
         5.6741e-10, -9.8752e-10, -1.5280e-09, -4.7066e-10, -1.3644e-09,
        -3.2977e-09])
tensor([ 2.7277e-02, -1.5349e-03,  9.3041e-03, -1.4790e-02, -3.2848e-03,
         1.2297e-02,  3.5606e-02,  4.5507e-02,  2.4510e-02, -2.9488e-03,
         9.3607e-03, -3.0083e-02, -5.1791e-02, -4.5313e-02,  3.3095e-03,
         1.1064e-02,  3.5446e-03, -4.7910e-02, -5.1270e-02, -3.5876e-02,
         1.0394e-02, -7.2935e-02, -7.7675e-03, -6.1409e-03, -6.2378e-03,
        -4.9720e-02,  3.4843e-04, -2.9047e-02, -1.5199e-02, -3.6454e-02,
         1.6506e-03,  1.3442e-02, -8.6619e-03, -3.1953e-02,  9.3011e-03,
        -3.0305e-04, -1.4666e-03, -8.4677e-03, -3.2692e-02,  2.1591e-03,
        -4.8136e-03,  2.6745e-03,  1.0465e-02,  3.5076e-03,  1.2788e-02,
        -2.5598e-02, -5.6356e-03,  4.9484e-02,  4.4535e-03, -2.2032e-02,
        -8.9427e-03,  3.1417e-02,  2.9089e-02,  5.1659e-02,  3.5249e-03,
        -2.5710e-02,  7.1074e-03, -1.3778e-02,  2.2365e-02, -3.1342e-02,
        -1.4535e-02,  1.0045e-02,  8.5741e-03, -7.5019e-03,  2.1228e-04,
         8.6521e-04, -1.4413e-03, -1.5796e-03, -8.5186e-02,  1.6786e-02,
         2.5689e-02,  2.3803e-02,  2.3566e-02,  1.2425e-02,  1.4655e-02,
        -3.2064e-02,  1.8373e-02,  2.0755e-03, -5.9231e-03,  2.4305e-02,
         8.6564e-03,  3.6206e-02, -4.0701e-03, -7.6346e-02,  2.3423e-02,
         3.3687e-02,  5.0210e-03, -1.5342e-02,  3.2086e-02, -4.2325e-03,
         1.7740e-02, -1.1244e-02,  2.0189e-02,  3.6672e-02, -7.2388e-02,
         6.5306e-03, -3.9289e-03,  7.5227e-03, -5.9092e-03, -5.8976e-03,
        -3.8962e-02,  4.2394e-03, -1.2697e-02,  7.1903e-03,  4.7079e-03,
         3.0006e-02, -1.0778e-02,  7.3161e-03,  1.6984e-02,  2.7804e-02,
        -9.9974e-03,  2.2839e-03, -1.7448e-02, -1.2544e-02,  1.2502e-02,
         1.0259e-02, -2.3192e-03,  1.6334e-03, -2.0454e-02,  3.1482e-02,
        -5.5854e-02,  1.1868e-02,  1.8199e-02,  8.5249e-03,  4.7954e-03,
         3.7318e-03, -2.9542e-02, -2.6642e-03,  2.6589e-03,  5.8292e-02,
         1.4303e-02,  1.9776e-02, -1.1400e-02,  6.4291e-03, -1.4870e-02,
         6.0573e-02,  2.3893e-02,  2.4229e-02,  8.4878e-03,  3.4743e-02,
         1.5875e-02, -1.7727e-04,  1.1912e-02,  3.4174e-02,  2.2147e-03,
        -1.3699e-02,  7.6812e-02, -6.2737e-02,  9.2949e-03, -3.1834e-02,
         8.4093e-03,  6.3829e-03, -3.4273e-03, -3.8925e-02,  4.0624e-02,
        -7.7520e-03, -8.1468e-03,  5.3185e-03,  2.6856e-02, -7.9088e-03,
        -2.9562e-02, -7.3196e-03, -9.5489e-03,  1.8343e-02,  3.5141e-02,
         2.7017e-02, -2.0174e-02,  1.6257e-02, -8.4368e-03,  2.5605e-02,
         1.9781e-02,  7.9879e-03, -2.7533e-02, -5.9911e-02, -2.4516e-02,
        -1.8920e-02, -4.5708e-03,  1.9466e-02, -1.9974e-03,  5.1703e-03,
        -3.9294e-03, -2.1976e-03,  4.6336e-03, -6.4269e-03, -3.2750e-05,
```

```
        -7.1842e-03,  2.6660e-02,  5.2086e-03,  1.7066e-02, -1.0803e-02,
         5.4659e-05, -1.5296e-02, -3.5460e-03, -2.8640e-02,  1.4056e-02,
        -3.3122e-03,  3.8200e-03, -1.4386e-03,  4.4032e-03, -4.7339e-02,
        -3.9755e-02, -2.8562e-03,  8.0722e-03, -7.0002e-03, -3.4299e-03,
        -1.5316e-02, -7.0315e-03,  1.1550e-02,  1.3235e-02,  2.0712e-02,
         9.7666e-03,  1.2324e-03,  3.5241e-02,  5.5747e-04,  1.4695e-02,
         1.3612e-02, -5.0026e-03,  1.5611e-02, -8.1241e-03, -4.8336e-03,
        -2.2594e-02,  2.0029e-02, -3.1047e-02, -1.3884e-03, -6.9041e-02,
        -4.6463e-03,  2.2326e-03,  1.5280e-02,  7.5656e-03, -2.5596e-02,
         7.7233e-03, -8.4711e-03, -1.9278e-02, -8.7801e-03, -1.4945e-03,
         2.0361e-02,  3.4875e-02,  1.6165e-03, -2.0551e-02, -2.5575e-03,
         1.4514e-02,  4.7962e-02, -3.9307e-02,  8.8673e-03, -2.0589e-02,
        -4.4700e-03, -3.8529e-02,  5.4806e-03,  5.1465e-03, -7.1062e-02,
        -5.1878e-03,  3.5334e-03, -2.4979e-03,  2.0359e-03,  1.1057e-02,
         2.0496e-03])
tensor([ 1.8772e-02, -4.7189e-05,  7.0835e-03, -3.6640e-03, -4.9151e-03,
         1.1548e-02,  2.7263e-02,  3.7657e-02,  1.7409e-02, -2.1508e-03,
        -1.6666e-02, -2.2707e-02, -2.6646e-02, -1.1433e-02,  4.2634e-03,
         1.0757e-02,  5.1478e-03, -4.3086e-02, -5.2882e-02, -3.3886e-02,
         6.3440e-03, -2.2952e-02, -7.7385e-03,  7.1875e-05, -8.4854e-03,
        -2.7989e-02, -3.1111e-03, -3.1412e-02, -1.8347e-02, -3.4577e-02,
         1.7927e-03, -2.6239e-04, -1.2939e-02, -3.7107e-02,  1.2698e-02,
        -4.1404e-04,  4.3468e-03, -1.2597e-02, -2.1542e-02,  8.1368e-03,
        -8.3907e-03,  8.0217e-03,  4.4248e-03,  3.6422e-03,  1.3647e-02,
        -2.8367e-02,  3.2608e-03,  2.6631e-02, -5.8862e-03, -1.9340e-02,
        -9.3629e-03,  3.4907e-02,  2.3520e-02,  3.7936e-02,  1.9594e-02,
        -2.4948e-02,  7.8942e-03, -3.0483e-03,  4.3544e-03, -3.8126e-02,
        -1.5348e-02,  9.6446e-03,  1.0782e-02, -6.3420e-03,  1.3135e-03,
         6.5378e-04, -3.0221e-03, -8.0633e-03, -6.7810e-02,  1.6138e-02,
         2.7677e-02,  3.6926e-02,  2.4552e-02,  1.2245e-02,  4.2020e-03,
        -1.3147e-02,  7.5935e-03, -3.5496e-03, -7.4655e-03,  2.6753e-02,
         1.1099e-02,  3.5630e-02, -7.1757e-03, -3.3593e-02,  1.1961e-02,
         2.5970e-02, -8.1513e-03, -2.0503e-02,  2.7413e-02, -2.9312e-03,
         6.1655e-03, -6.5994e-03,  1.3897e-02,  3.0798e-02, -6.6934e-02,
        -2.7041e-03, -7.1783e-03,  9.8082e-03, -4.5395e-03, -6.8841e-03,
        -3.3646e-02, -1.0171e-04, -1.4103e-02,  6.2209e-03,  6.4511e-03,
         2.5838e-02, -9.2272e-03,  1.0901e-02,  2.2996e-02,  3.0087e-02,
        -1.2193e-02, -6.6903e-05, -1.9132e-02, -1.3840e-02,  7.3722e-03,
         8.3439e-03, -2.9448e-03,  5.7446e-03, -2.0055e-02,  2.0975e-02,
        -3.1402e-02,  2.2696e-02,  1.9365e-02,  1.2472e-02, -2.1643e-03,
         2.7957e-03, -3.0664e-02, -3.7820e-03,  6.9917e-03,  7.0384e-02,
         1.3595e-02,  1.0684e-02, -1.0443e-02,  9.5363e-03, -1.9263e-02,
         5.1614e-02,  2.2070e-02,  2.4327e-02,  2.7046e-03,  2.9608e-02,
         1.6835e-02, -2.9169e-03,  1.3576e-02,  2.7884e-02,  2.9385e-03,
        -1.1741e-02,  2.5936e-02, -6.5192e-02,  7.6873e-03, -3.6725e-02,
         5.8745e-03,  6.3792e-03, -1.7368e-02, -1.0680e-02,  6.3255e-03,
        -1.6966e-03, -5.1645e-03,  1.0258e-02,  2.7279e-02,  6.0010e-03,
        -1.4683e-02, -5.8110e-03, -9.9057e-03,  1.9769e-02,  3.1884e-02,
```

```
         1.9659e-02, -2.7180e-02,  1.9337e-02,  3.3320e-03,  2.0152e-02,
         1.9139e-02,  9.5296e-03, -2.7874e-02, -4.2967e-02, -6.4601e-03,
        -2.3316e-02, -5.2495e-03,  2.0605e-02,  3.2368e-03,  4.6175e-03,
        -2.2051e-03,  3.4687e-03, -4.0204e-03, -6.8176e-03, -2.7844e-03,
        -4.8193e-03,  6.8516e-03,  6.0466e-04,  2.4133e-02, -2.2634e-02,
        -4.4633e-03, -1.6878e-02, -8.6586e-03, -6.7413e-03,  1.0901e-02,
        -5.9342e-03,  8.0954e-03, -2.2001e-03,  3.4527e-03, -3.7966e-02,
        -3.4361e-02, -8.0764e-03,  2.8367e-02, -6.9839e-03, -3.8112e-04,
        -1.6484e-02, -1.2351e-02,  1.4471e-02,  1.4148e-02,  1.7132e-02,
         2.4591e-03,  1.5410e-03,  3.4153e-02,  2.8227e-03,  1.0046e-02,
         1.5406e-02, -5.0890e-03,  1.8738e-02, -8.3184e-03, -1.5163e-03,
        -2.2439e-02,  2.1404e-02, -3.4355e-02, -1.5047e-02, -6.5139e-02,
        -6.5406e-03,  1.6775e-03,  6.6933e-03,  3.3919e-03, -3.2948e-02,
        -9.3184e-03, -9.6537e-03, -1.8771e-02,  7.7208e-03, -2.6676e-03,
         2.9807e-02,  2.7497e-02, -5.7163e-03, -1.5568e-02, -4.4994e-03,
         2.2572e-02,  3.5924e-02, -2.4237e-02,  1.0416e-02, -3.2251e-02,
        -2.2433e-03, -2.7043e-02,  2.8359e-03,  4.5627e-03, -3.7780e-02,
        -3.1685e-03,  7.6368e-03,  8.4702e-04, -2.4719e-03,  8.2753e-03,
         2.5151e-03])
tensor([[[[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         …,

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]]],


        [[[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
```

```
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
```

```
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


…,


[[[-0.0095, -0.0173, -0.0210],
  [-0.0023, -0.0075, -0.0065],
  [-0.0004, -0.0037, -0.0027]],

 [[-0.0038, -0.0043, -0.0086],
  [ 0.0083,  0.0115,  0.0099],
  [ 0.0005,  0.0082,  0.0062]],

 [[-0.0005, -0.0014, -0.0006],
  [-0.0036,  0.0010,  0.0054],
  [ 0.0037,  0.0103,  0.0098]],

 …,

 [[-0.0137, -0.0157, -0.0184],
  [-0.0115, -0.0168, -0.0195],
  [-0.0106, -0.0193, -0.0192]],

 [[-0.0110, -0.0048, -0.0044],
  [ 0.0004,  0.0063,  0.0048],
  [ 0.0052,  0.0084,  0.0069]],

 [[-0.0333, -0.0281, -0.0290],
  [-0.0272, -0.0237, -0.0226],
  [-0.0183, -0.0210, -0.0257]]],


[[[-0.0463, -0.0378, -0.0310],
  [-0.0636, -0.0617, -0.0538],
  [-0.0670, -0.0640, -0.0512]],

 [[ 0.0114,  0.0025, -0.0107],
  [-0.0007, -0.0017, -0.0105],
  [-0.0030, -0.0001, -0.0039]],

 [[-0.0083, -0.0032, -0.0135],
  [-0.0029, -0.0002, -0.0059],
  [-0.0088, -0.0097, -0.0133]],

 …,
```

```
          [[ 0.0459,  0.0501,  0.0463],
           [ 0.0416,  0.0463,  0.0471],
           [ 0.0298,  0.0355,  0.0377]],

          [[ 0.0565,  0.0653,  0.0658],
           [ 0.0829,  0.0958,  0.0843],
           [ 0.0873,  0.0874,  0.0661]],

          [[ 0.0276,  0.0223,  0.0231],
           [ 0.0238,  0.0237,  0.0185],
           [ 0.0052,  0.0044, -0.0013]]],


         [[[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          …,

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]]]])
tensor([ 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -2.4917e-03,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  2.6981e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  8.9703e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  1.4850e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

```
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00, -2.2397e-02,  0.0000e+00,  6.4779e-03,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.0621e-04,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.1609e-02,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00, -1.3656e-01,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00, -1.6102e-02,  0.0000e+00,
0.0000e+00,  1.3912e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
6.5028e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

```
          7.7733e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00, -4.7156e-03,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  3.4421e-03,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  1.2360e-02,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  3.3991e-03,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  3.5555e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -2.0132e-02, -4.9833e-04,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -2.9283e-02, -4.8869e-03,  0.0000e+00])
tensor([[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 2.1180e-02,  1.5524e-02, -5.8355e-03],
          [ 2.7880e-02,  2.9105e-02, -1.8007e-02],
          [ 2.7357e-02,  1.9943e-02, -2.5931e-02]],

         [[ 5.3156e-03,  2.1203e-02, -7.5900e-03],
          [-1.1978e-02,  3.6068e-02,  2.3616e-02],
          [-5.7992e-03,  3.2798e-02,  2.2706e-02]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 4.4499e-03,  1.3706e-03, -4.6150e-03],
  [ 2.1838e-03, -3.5209e-03, -3.1414e-03],
  [ 2.3284e-03, -2.1125e-03, -4.4233e-03]],

 [[-2.4531e-03, -6.4189e-04, -5.0238e-03],
  [ 2.5849e-02,  2.8301e-02,  3.8398e-02],
  [ 3.7015e-02,  5.3532e-02,  6.4922e-02]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  …,

  [[-9.1977e-03, -2.8359e-02, -5.1334e-03],
   [-3.8339e-03, -1.8733e-02, -3.2220e-03],
   [-3.3002e-03, -1.0948e-02, -6.6728e-04]],

  [[-1.0303e-01, -9.6465e-02, -4.6688e-02],
   [-1.3550e-01, -1.1275e-01, -4.8817e-02],
   [-8.9135e-02, -7.1695e-02, -4.9582e-02]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  …,
```

```
          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[-3.6081e-04,  4.5836e-05,  1.5572e-05],
           [ 0.0000e+00,  2.8545e-05,  0.0000e+00],
           [ 0.0000e+00,  1.8991e-05,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


         [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([-1.4157e-02,  2.4052e-02,  0.0000e+00,  5.8741e-03, -8.7336e-02,
         1.8482e-01,  2.9045e-04,  7.5877e-02, -2.0598e-02,  0.0000e+00,
         6.1820e-04,  4.8963e-04, -2.4032e-02,  0.0000e+00,  1.9362e-05,
         3.5435e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  8.3591e-05,
         0.0000e+00,  0.0000e+00, -4.1308e-04, -1.8244e-03,  0.0000e+00,
         4.5274e-02,  0.0000e+00,  5.2723e-05, -3.9454e-04, -6.1796e-03,
        -5.7572e-04, -4.4921e-03,  8.0808e-03,  0.0000e+00,  4.7424e-03,
         8.5505e-03,  1.7341e-03,  4.6974e-02,  3.3800e-03, -1.4523e-02,
        -1.1750e-03,  1.0103e-02,  5.5764e-04, -2.5929e-02,  0.0000e+00,
        -1.7688e-02, -1.1250e-03,  0.0000e+00,  3.2812e-02,  0.0000e+00,
```

```
 0.0000e+00, -4.6554e-03, -1.2090e-02,  3.8396e-04, -6.9645e-03,
 0.0000e+00, -2.3785e-02, -1.8952e-05, -3.9705e-03,  0.0000e+00,
 0.0000e+00,  0.0000e+00, -5.0053e-03,  0.0000e+00, -6.5592e-03,
-2.4718e-02,  5.4201e-02, -6.3896e-03, -1.4923e-01,  4.5166e-02,
 0.0000e+00,  0.0000e+00, -2.1715e-04,  0.0000e+00,  0.0000e+00,
 8.0042e-06, -3.6820e-04, -8.6048e-04,  4.2792e-03,  3.0859e-04,
-5.0753e-06,  0.0000e+00, -2.2377e-04, -3.5667e-03,  2.2904e-03,
 0.0000e+00,  3.5244e-02, -1.1075e-02, -1.9432e-03,  0.0000e+00,
 8.3714e-03,  0.0000e+00,  3.7818e-03,  0.0000e+00, -4.3490e-04,
 0.0000e+00, -3.3244e-04,  1.3082e-02,  0.0000e+00, -6.6015e-04,
-6.4313e-02, -7.9119e-03, -1.1286e-01,  1.2057e-02,  0.0000e+00,
-5.7607e-03,  5.4290e-03, -1.6476e-02, -1.0774e-03, -1.0605e-01,
-7.5502e-03, -4.3471e-04, -1.1203e-03, -2.5440e-03,  3.3982e-02,
-6.5489e-03,  6.4625e-02,  2.2970e-03, -9.3889e-04,  3.6871e-03,
 8.3055e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00, -1.5160e-02, -9.9663e-04,  0.0000e+00, -2.2291e-03,
 0.0000e+00,  0.0000e+00, -9.8031e-04, -3.2533e-02,  1.5547e-02,
 6.7354e-03, -2.2892e-03,  0.0000e+00,  7.0270e-04,  0.0000e+00,
 1.1131e-03, -2.1517e-02,  5.6774e-02, -4.2419e-05,  0.0000e+00,
 0.0000e+00,  0.0000e+00, -6.0148e-02,  8.1503e-03,  5.5031e-05,
 0.0000e+00,  0.0000e+00, -1.7312e-02,  3.7060e-04,  0.0000e+00,
-7.2288e-02,  0.0000e+00, -3.5053e-04, -9.7669e-05, -1.3278e-02,
 4.1922e-03, -1.8586e-02,  0.0000e+00,  0.0000e+00,  1.9273e-02,
-1.7681e-05, -6.6708e-04, -2.7164e-04,  4.6551e-03, -8.9636e-03,
 1.1808e-04,  0.0000e+00, -1.0716e-02,  0.0000e+00, -1.4934e-02,
 7.8994e-03,  0.0000e+00, -1.3876e-02, -1.3756e-02,  1.4499e-03,
 8.5908e-05,  6.0711e-02,  1.0172e-03, -2.4636e-04,  2.2625e-03,
 0.0000e+00,  1.7685e-03,  2.0897e-02, -2.9690e-03, -3.3352e-02,
 0.0000e+00,  0.0000e+00, -1.6633e-03, -3.5091e-02, -8.6508e-04,
-4.2819e-04, -2.3559e-02, -4.5951e-03,  0.0000e+00,  0.0000e+00,
 1.7782e-02, -9.1521e-03, -1.7609e-03,  2.1633e-03,  2.9740e-03,
 7.9586e-04, -1.4651e-02,  0.0000e+00,  3.3565e-03, -5.4921e-05,
 1.7091e-03,  7.9125e-03,  2.4127e-03,  3.0545e-03, -5.3232e-03,
 4.0602e-02,  2.9847e-04,  0.0000e+00, -1.4883e-04,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  1.9701e-05, -2.7255e-02,  0.0000e+00,
 1.5118e-04,  4.9571e-03, -4.9361e-02,  1.8803e-03,  0.0000e+00,
 1.5484e-02,  0.0000e+00,  6.8025e-06,  2.6975e-03,  6.0939e-03,
 6.4774e-03,  0.0000e+00, -8.4655e-03,  5.3137e-02, -2.6543e-02,
-2.9765e-04, -1.0543e-02,  0.0000e+00, -6.9470e-04, -5.1680e-03,
 0.0000e+00,  3.7170e-03,  0.0000e+00,  0.0000e+00,  1.0979e-02,
 0.0000e+00, -9.1648e-02, -2.0142e-02,  0.0000e+00,  0.0000e+00,
-4.3594e-04,  0.0000e+00,  1.6708e-03,  3.0604e-02,  0.0000e+00,
 0.0000e+00, -1.0690e-03, -2.2111e-02, -3.6874e-03,  0.0000e+00,
 1.4113e-03, -5.3586e-02, -3.0754e-02, -1.9709e-03, -5.2360e-02,
 0.0000e+00, -5.5880e-03, -2.7288e-04,  0.0000e+00,  3.5647e-03,
-2.6403e-03,  5.3446e-02, -4.9758e-03,  0.0000e+00, -3.1092e-02,
-6.4637e-04, -4.5538e-04,  0.0000e+00, -2.2298e-02, -1.9171e-02,
 0.0000e+00, -8.6711e-03,  0.0000e+00,  1.5376e-03, -7.7899e-04,
```

```
          1.1283e-02, -2.0284e-05,  1.9646e-04, -9.3685e-04,  0.0000e+00,
          5.1791e-03,  3.0686e-03, -1.4244e-04, -5.9473e-05,  0.0000e+00,
          2.4066e-02, -3.3510e-02, -4.7379e-04,  4.3750e-02,  0.0000e+00,
          2.2501e-02,  8.1834e-04,  0.0000e+00,  3.5742e-02,  0.0000e+00,
         -1.4865e-05,  3.5532e-02, -2.1436e-04, -1.7726e-02,  9.3385e-03,
          3.1930e-04,  3.4312e-05,  0.0000e+00,  0.0000e+00,  2.1564e-04,
         -1.7478e-03,  8.6150e-04,  1.0048e-02, -3.0108e-02, -5.4446e-03,
          1.1325e-02,  0.0000e+00, -7.7612e-04,  8.1111e-03,  0.0000e+00,
         -7.2367e-04,  8.2544e-02,  2.5474e-02,  2.1092e-04, -1.4896e-03,
          5.8495e-04,  0.0000e+00,  4.5936e-03, -1.5198e-02,  7.0374e-02,
          6.6028e-03, -4.5525e-02, -2.0319e-03,  1.1910e-02, -6.4814e-02,
         -7.3065e-02,  2.3052e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  2.5561e-05, -1.3660e-02, -8.4750e-03,  0.0000e+00,
          4.0702e-03,  7.9486e-03,  0.0000e+00, -4.4028e-03,  4.9755e-03,
          1.0874e-03, -1.1190e-03, -9.6119e-03,  0.0000e+00,  0.0000e+00,
          8.2587e-02, -5.3135e-03,  3.4596e-03,  3.4673e-05,  3.4839e-02,
         -1.2525e-04, -1.0765e-03,  0.0000e+00,  6.1451e-02,  1.1980e-03,
          3.2699e-03, -3.3775e-03,  4.2035e-03,  0.0000e+00, -1.2327e-03,
          2.1578e-06, -3.9206e-02,  4.0892e-04,  0.0000e+00])
tensor([[[[ 0.0000e+00, -2.4454e-04, -2.9027e-06],
          [ 0.0000e+00, -3.6884e-04,  0.0000e+00],
          [ 0.0000e+00, -1.4392e-04,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  …,

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  …,
```

```
          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


         [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([-2.2679e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  1.3970e-05,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  8.2688e-04, -1.4820e-01,  8.8075e-02,
         7.1775e-02,  5.3533e-03,  5.3086e-02,  0.0000e+00,  0.0000e+00,
        -5.4181e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         1.3614e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00, -5.2316e-02,
        -6.7348e-02,  1.8665e-03,  9.5500e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         5.2138e-05,  0.0000e+00,  0.0000e+00,  1.5722e-01,  1.0537e-01,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

```
         0.0000e+00,  7.7014e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -4.6365e-02,  2.9866e-01,  0.0000e+00,
         5.3898e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00, -3.3138e-02,
         0.0000e+00, -4.3077e-02,  0.0000e+00,  4.4704e-04,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -1.7165e-01, -3.8512e-02,  0.0000e+00,
         0.0000e+00, -1.4329e-01, -1.2645e-01,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  2.3627e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  1.5331e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -8.2779e-04,  9.5651e-04,  0.0000e+00, -8.5386e-02,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  2.5971e-02,
         0.0000e+00,  3.3497e-03,  1.8245e-01,  0.0000e+00, -3.8462e-04,
         1.9837e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -3.1345e-02,  3.0035e-01,
         0.0000e+00,  0.0000e+00, -7.9330e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         7.6190e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  4.8232e-04,
         0.0000e+00,  0.0000e+00,  6.1943e-02,  8.0605e-02, -5.5037e-02,
        -2.5915e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -3.4231e-02, -3.7644e-03,  0.0000e+00,
         2.8285e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -6.6121e-03,  0.0000e+00,  0.0000e+00,  6.8775e-04,
         0.0000e+00,  0.0000e+00,  3.3133e-03,  0.0000e+00,  0.0000e+00,
        -2.6910e-02,  0.0000e+00,  0.0000e+00,  7.7170e-05,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -9.4107e-03,
        -1.2028e-03, -2.5912e-01,  0.0000e+00, -1.2094e-02,  6.8955e-03,
         1.5078e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.1959e-01,
         0.0000e+00,  9.3397e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  4.5880e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -1.0269e-01,  6.3884e-05, -2.4272e-05,
         0.0000e+00, -1.0626e-01, -1.5918e-02,  0.0000e+00,  3.7001e-01,
        -1.2744e-04,  0.0000e+00,  6.8719e-03,  0.0000e+00,  0.0000e+00,
        -3.3699e-01,  0.0000e+00,  0.0000e+00, -2.5242e-01,  0.0000e+00,
         0.0000e+00, -9.0802e-04,  5.1749e-03,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  6.0886e-04,  7.4228e-03,  0.0000e+00,
         0.0000e+00,  1.5831e-01, -1.7731e-01,  1.4022e-02,  0.0000e+00,
         1.1707e-01,  0.0000e+00, -1.3288e-01,  0.0000e+00,  1.9529e-05,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  1.3237e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  1.9828e-03,
        -2.6671e-02,  0.0000e+00,  0.0000e+00,  3.3469e-02,  0.0000e+00,
         0.0000e+00, -1.1158e-02, -6.9230e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00])
tensor([[0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        …,
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
```

```
        [0., 0., 0.,  …, 0., 0., 0.]])
tensor([-0.0033,  0.0000,  0.0002,  …,  0.0000, -0.0012,  0.0000])
tensor([[0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        …,
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.]])
tensor([-0.0008,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000])
tensor([[ 1.8561e-05,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [-2.0189e-03,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 7.9980e-07,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        …,
        [ 1.4423e-10,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 1.4461e-10,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 1.3530e-10,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00]])
tensor([-3.7117e-02, -3.7064e-02,  3.0927e-02, -2.2257e-02,  7.8316e-02,
        -1.1709e-02,  3.0690e-02,  4.7593e-02, -1.5793e-02, -5.4028e-02,
        -3.8896e-02, -8.9118e-02,  3.9474e-02,  5.8589e-02,  6.4594e-02,
         2.1131e-02, -3.9196e-02, -3.0114e-02,  6.7870e-03, -8.9740e-03,
         4.7258e-05,  6.0619e-06,  6.1439e-06,  6.2841e-06,  6.0697e-06,
         6.2433e-06,  6.3210e-06,  6.0936e-06,  6.3421e-06,  6.3388e-06,
         6.3210e-06,  6.1673e-06,  6.3249e-06,  6.2872e-06,  6.1750e-06,
         6.1280e-06,  6.2709e-06,  6.3378e-06,  6.1784e-06,  6.3369e-06,
         6.2789e-06,  6.2103e-06,  6.1851e-06,  6.2385e-06,  6.2939e-06,
         6.3341e-06,  6.2955e-06,  6.2514e-06,  6.3006e-06,  6.1076e-06,
         6.3361e-06,  6.2674e-06,  6.3308e-06,  6.1085e-06,  6.3210e-06,
         6.3245e-06,  6.2909e-06,  6.3410e-06,  6.1754e-06,  6.3026e-06,
         6.3361e-06,  6.3191e-06,  6.3088e-06,  6.3105e-06,  6.3377e-06,
         6.3224e-06,  6.3066e-06,  6.3292e-06,  6.0701e-06,  6.2414e-06,
         6.2800e-06,  6.3417e-06,  6.1320e-06,  6.2545e-06,  6.3140e-06,
         6.0586e-06,  6.3422e-06,  6.2384e-06,  6.2517e-06,  6.1057e-06,
         6.2378e-06,  6.2520e-06,  6.2676e-06,  6.3354e-06,  6.2748e-06,
         6.1140e-06,  6.3180e-06,  6.3443e-06,  6.3395e-06,  6.1137e-06,
         6.2183e-06,  6.2656e-06,  6.3418e-06,  6.1360e-06,  6.1868e-06,
         6.3378e-06,  6.3302e-06,  6.2570e-06,  6.2571e-06,  6.3282e-06,
         6.3208e-06,  6.1537e-06,  6.3426e-06,  6.2655e-06,  6.2740e-06,
         6.3382e-06,  6.2872e-06,  6.2525e-06,  6.1634e-06,  6.2676e-06,
         6.3036e-06,  6.3453e-06,  6.3047e-06,  6.0654e-06,  6.3082e-06,
         6.1839e-06,  6.3138e-06,  6.2318e-06,  6.3207e-06,  6.1383e-06,
         6.0557e-06,  6.2308e-06,  6.0610e-06,  6.3094e-06,  6.2634e-06,
```

```
6.1768e-06,  6.3268e-06,  6.2641e-06,  6.3292e-06,  6.2733e-06,
6.1595e-06,  6.2255e-06,  6.1752e-06,  6.3322e-06,  6.2935e-06,
6.1192e-06,  6.2912e-06,  6.2136e-06,  6.0734e-06,  6.2007e-06,
6.3372e-06,  6.2727e-06,  6.3204e-06,  6.1671e-06,  6.0969e-06,
6.3248e-06,  6.1239e-06,  6.3396e-06,  6.2413e-06,  6.3400e-06,
6.3352e-06,  6.3407e-06,  6.3182e-06,  6.1966e-06,  6.1280e-06,
6.1710e-06,  6.1416e-06,  6.2973e-06,  6.3355e-06,  6.2671e-06,
6.3361e-06,  6.0719e-06,  6.2249e-06,  6.2718e-06,  6.3360e-06,
6.0545e-06,  6.2799e-06,  6.1914e-06,  6.3296e-06,  6.3300e-06,
6.3427e-06,  6.3420e-06,  6.3364e-06,  6.2638e-06,  6.2962e-06,
6.2763e-06,  6.2898e-06,  6.2448e-06,  6.3417e-06,  6.3397e-06,
6.3363e-06,  6.1226e-06,  6.1900e-06,  6.3148e-06,  6.0818e-06,
6.3438e-06,  6.1847e-06,  6.2808e-06,  6.3373e-06,  6.3433e-06,
6.1583e-06,  6.2918e-06,  6.3363e-06,  6.3082e-06,  6.1936e-06,
6.1283e-06,  6.1585e-06,  6.3228e-06,  6.2384e-06,  6.1749e-06,
6.1456e-06,  6.3368e-06,  6.1207e-06,  6.3363e-06,  6.3312e-06,
6.3376e-06,  6.2996e-06,  6.3311e-06,  6.1884e-06,  6.3284e-06,
6.1569e-06,  6.3361e-06,  6.0859e-06,  6.2568e-06,  6.3065e-06,
6.2285e-06,  6.3416e-06,  6.3440e-06,  6.3169e-06,  6.2677e-06,
6.3079e-06,  6.2058e-06,  6.2055e-06,  6.1576e-06,  6.0551e-06,
6.3396e-06,  6.1540e-06,  6.2616e-06,  6.0798e-06,  6.1891e-06,
6.2959e-06,  6.3336e-06,  6.2692e-06,  6.3393e-06,  6.2985e-06,
6.3360e-06,  6.2612e-06,  6.3410e-06,  6.2120e-06,  6.3036e-06,
6.1844e-06,  6.1811e-06,  6.2915e-06,  6.2872e-06,  6.2913e-06,
6.3107e-06,  6.2214e-06,  6.2598e-06,  6.2474e-06,  6.3257e-06,
6.3444e-06,  6.3307e-06,  6.0614e-06,  6.0872e-06,  6.1332e-06,
6.3304e-06,  6.2196e-06,  6.2530e-06,  6.2715e-06,  6.2489e-06,
6.1264e-06,  6.2661e-06,  6.1279e-06,  6.3001e-06,  6.2941e-06,
6.3320e-06,  6.3088e-06,  6.2994e-06,  6.2119e-06,  6.3285e-06,
6.2463e-06,  6.1380e-06,  6.3119e-06,  6.2225e-06,  6.1016e-06,
6.3435e-06,  6.2801e-06,  6.0668e-06,  6.1951e-06,  6.3130e-06,
6.2378e-06,  6.3234e-06,  6.2420e-06,  6.3401e-06,  6.3361e-06,
6.0656e-06,  6.1930e-06,  6.0722e-06,  6.1866e-06,  6.3360e-06,
6.0482e-06,  6.3399e-06,  6.2061e-06,  6.2259e-06,  6.3380e-06,
6.2366e-06,  6.3312e-06,  6.2125e-06,  6.2730e-06,  6.3328e-06,
6.2933e-06,  6.3440e-06,  6.2725e-06,  6.3408e-06,  6.3067e-06,
6.3365e-06,  6.3258e-06,  6.1364e-06,  6.3321e-06,  6.1481e-06,
6.1771e-06,  6.3400e-06,  6.2982e-06,  6.1746e-06,  6.3392e-06,
6.2789e-06,  6.3027e-06,  6.1386e-06,  6.2247e-06,  6.2836e-06,
6.2771e-06,  6.3082e-06,  6.0795e-06,  6.1680e-06,  6.3090e-06,
6.3096e-06,  6.3404e-06,  6.0916e-06,  6.3033e-06,  6.3406e-06,
6.3190e-06,  6.3439e-06,  6.2724e-06,  6.3140e-06,  6.0961e-06,
6.3127e-06,  6.3390e-06,  6.2260e-06,  6.2931e-06,  6.3336e-06,
6.2104e-06,  6.0916e-06,  6.3415e-06,  6.3461e-06,  6.0418e-06,
6.1390e-06,  6.3385e-06,  6.1943e-06,  6.3331e-06,  6.2092e-06,
6.2381e-06,  6.3056e-06,  6.3096e-06,  6.3183e-06,  6.3235e-06,
6.3179e-06,  6.3373e-06,  6.3437e-06,  6.1298e-06,  6.2345e-06,
6.2952e-06,  6.2573e-06,  6.0661e-06,  6.1405e-06,  6.2984e-06,
```

```
6.2984e-06,  6.0865e-06,  6.2848e-06,  6.2299e-06,  6.1869e-06,
6.3313e-06,  6.2729e-06,  6.3383e-06,  6.1745e-06,  6.3368e-06,
6.2797e-06,  6.3188e-06,  6.0740e-06,  6.2878e-06,  6.1211e-06,
6.3245e-06,  6.1540e-06,  6.3431e-06,  6.3379e-06,  6.1895e-06,
6.2972e-06,  6.2548e-06,  6.3405e-06,  6.3327e-06,  6.1043e-06,
6.2133e-06,  6.3379e-06,  6.3062e-06,  6.3132e-06,  6.2861e-06,
6.2158e-06,  6.2749e-06,  6.1879e-06,  6.2404e-06,  6.2409e-06,
6.1574e-06,  6.3377e-06,  6.3398e-06,  6.1243e-06,  6.1310e-06,
6.3193e-06,  6.2687e-06,  6.2872e-06,  6.3077e-06,  6.1577e-06,
6.3358e-06,  6.3446e-06,  6.3100e-06,  6.3392e-06,  6.2485e-06,
6.3070e-06,  6.1769e-06,  6.2565e-06,  6.3276e-06,  6.3363e-06,
6.3305e-06,  6.2853e-06,  6.3241e-06,  6.2071e-06,  6.2783e-06,
6.1763e-06,  6.1566e-06,  6.0499e-06,  6.2495e-06,  6.2942e-06,
6.1579e-06,  6.2001e-06,  6.1170e-06,  6.1605e-06,  6.1797e-06,
6.2373e-06,  6.3418e-06,  6.3383e-06,  6.2808e-06,  6.2153e-06,
6.3410e-06,  6.3402e-06,  6.2383e-06,  6.3416e-06,  6.2392e-06,
6.3011e-06,  6.2644e-06,  6.0633e-06,  6.3436e-06,  6.3212e-06,
6.2818e-06,  6.2864e-06,  6.0739e-06,  6.2663e-06,  6.2160e-06,
6.3056e-06,  6.2503e-06,  6.2503e-06,  6.2946e-06,  6.3340e-06,
6.1096e-06,  6.2146e-06,  6.2688e-06,  6.0575e-06,  6.2879e-06,
6.3047e-06,  6.0713e-06,  6.3439e-06,  6.2071e-06,  6.3169e-06,
6.3436e-06,  6.2921e-06,  6.1225e-06,  6.3322e-06,  6.3428e-06,
6.2365e-06,  6.3279e-06,  6.2928e-06,  6.2217e-06,  6.2243e-06,
6.2646e-06,  6.2985e-06,  6.2926e-06,  6.2791e-06,  6.0760e-06,
6.1401e-06,  6.3405e-06,  6.2295e-06,  6.1167e-06,  6.3324e-06,
6.0982e-06,  6.1424e-06,  6.1079e-06,  6.2194e-06,  6.2821e-06,
6.0864e-06,  6.3343e-06,  6.3291e-06,  6.3428e-06,  6.2625e-06,
6.3004e-06,  6.3296e-06,  6.2282e-06,  6.2451e-06,  6.2416e-06,
6.0714e-06,  6.3217e-06,  6.2948e-06,  6.2525e-06,  6.2592e-06,
6.2641e-06,  6.2368e-06,  6.0723e-06,  6.3352e-06,  6.1431e-06,
6.1616e-06,  6.2707e-06,  6.2638e-06,  6.2575e-06,  6.3386e-06,
6.3392e-06,  6.3378e-06,  6.0683e-06,  6.1123e-06,  6.3385e-06,
6.3334e-06,  6.2298e-06,  6.0691e-06,  6.1797e-06,  6.3040e-06,
6.2830e-06,  6.2078e-06,  6.0680e-06,  6.3227e-06,  6.2447e-06,
6.1767e-06,  6.1417e-06,  6.2155e-06,  6.2828e-06,  6.1055e-06,
6.3200e-06,  6.0655e-06,  6.3364e-06,  6.2197e-06,  6.3019e-06,
6.3223e-06,  6.1964e-06,  6.2960e-06,  6.1889e-06,  6.2979e-06,
6.2655e-06,  6.2819e-06,  6.1518e-06,  6.3435e-06,  6.0525e-06,
6.3321e-06,  6.1431e-06,  6.2780e-06,  6.3365e-06,  6.3151e-06,
6.3380e-06,  6.3408e-06,  6.1560e-06,  6.1428e-06,  6.2528e-06,
6.0625e-06,  6.3175e-06,  6.1776e-06,  6.2874e-06,  6.2223e-06,
6.2270e-06,  6.2302e-06,  6.1484e-06,  6.1956e-06,  6.3117e-06,
6.3295e-06,  6.2064e-06,  6.2365e-06,  6.0679e-06,  6.2549e-06,
6.1178e-06,  6.3312e-06,  6.2754e-06,  6.2896e-06,  6.2315e-06,
6.0998e-06,  6.0429e-06,  6.2313e-06,  6.2905e-06,  6.1359e-06,
6.2757e-06,  6.1105e-06,  6.3355e-06,  6.3398e-06,  6.2549e-06,
6.2169e-06,  6.2706e-06,  6.2571e-06,  6.1379e-06,  6.2424e-06,
6.3415e-06,  6.2397e-06,  6.2253e-06,  6.2695e-06,  6.1047e-06,
```

```
6.3138e-06,  6.1762e-06,  6.3333e-06,  6.2668e-06,  6.3352e-06,
6.3034e-06,  6.0834e-06,  6.1387e-06,  6.2996e-06,  6.3056e-06,
6.2523e-06,  6.1913e-06,  6.3383e-06,  6.3403e-06,  6.2773e-06,
6.3079e-06,  6.2551e-06,  6.2876e-06,  6.2788e-06,  6.2896e-06,
6.1055e-06,  6.2625e-06,  6.2558e-06,  6.2328e-06,  6.1661e-06,
6.2197e-06,  6.3346e-06,  6.3037e-06,  6.0610e-06,  6.3128e-06,
6.2963e-06,  6.3171e-06,  6.1083e-06,  6.2605e-06,  6.2019e-06,
6.3429e-06,  6.2048e-06,  6.1461e-06,  6.1859e-06,  6.3178e-06,
6.2475e-06,  6.2177e-06,  6.3366e-06,  6.0585e-06,  6.3308e-06,
6.0318e-06,  6.2679e-06,  6.1972e-06,  6.3390e-06,  6.2275e-06,
6.2557e-06,  6.2659e-06,  6.2627e-06,  6.1899e-06,  6.3408e-06,
6.3394e-06,  6.2483e-06,  6.1551e-06,  6.2060e-06,  6.2428e-06,
6.3397e-06,  6.2854e-06,  6.3249e-06,  6.1654e-06,  6.2876e-06,
6.2669e-06,  6.3284e-06,  6.2874e-06,  6.1463e-06,  6.1597e-06,
6.3392e-06,  6.1471e-06,  6.3217e-06,  6.3415e-06,  6.1709e-06,
6.2238e-06,  6.2884e-06,  6.2924e-06,  6.2812e-06,  6.2550e-06,
6.1884e-06,  6.2743e-06,  6.3059e-06,  6.2090e-06,  6.2996e-06,
6.3364e-06,  6.1243e-06,  6.3232e-06,  6.1355e-06,  6.2306e-06,
6.1210e-06,  6.3373e-06,  6.2325e-06,  6.2578e-06,  6.2835e-06,
6.0815e-06,  6.3192e-06,  6.0869e-06,  6.1213e-06,  6.1256e-06,
6.2630e-06,  6.3023e-06,  6.3388e-06,  6.1898e-06,  6.2990e-06,
6.3363e-06,  6.3447e-06,  6.3289e-06,  6.3208e-06,  6.0574e-06,
6.2191e-06,  6.1988e-06,  6.3341e-06,  6.2780e-06,  6.3411e-06,
6.3353e-06,  6.1491e-06,  6.1616e-06,  6.1346e-06,  6.3055e-06,
6.1782e-06,  6.2755e-06,  6.2499e-06,  6.2618e-06,  6.2577e-06,
6.1724e-06,  6.3334e-06,  6.2096e-06,  6.0557e-06,  6.1019e-06,
6.2979e-06,  6.2956e-06,  6.3369e-06,  6.3002e-06,  6.3367e-06,
6.3351e-06,  6.2244e-06,  6.2646e-06,  6.3153e-06,  6.2983e-06,
6.3380e-06,  6.3408e-06,  6.1256e-06,  6.1687e-06,  6.1227e-06,
6.3282e-06,  6.2590e-06,  6.3160e-06,  6.0710e-06,  6.3382e-06,
6.0690e-06,  6.2424e-06,  6.3367e-06,  6.3227e-06,  6.2988e-06,
6.3351e-06,  6.2897e-06,  6.3336e-06,  6.2497e-06,  6.3405e-06,
6.2991e-06,  6.0818e-06,  6.3097e-06,  6.3456e-06,  6.3100e-06,
6.2634e-06,  6.3417e-06,  6.3122e-06,  6.0976e-06,  6.0829e-06,
6.3128e-06,  6.2779e-06,  6.1822e-06,  6.2138e-06,  6.3401e-06,
6.2990e-06,  6.1776e-06,  6.2533e-06,  6.3405e-06,  6.2603e-06,
6.0677e-06,  6.3123e-06,  6.2398e-06,  6.3106e-06,  6.0872e-06,
6.0622e-06,  6.3375e-06,  6.2209e-06,  6.0740e-06,  6.3127e-06,
6.0794e-06,  6.1151e-06,  6.3087e-06,  6.2519e-06,  6.1801e-06,
6.0786e-06,  6.3130e-06,  6.3133e-06,  6.1956e-06,  6.2889e-06,
6.2100e-06,  6.2613e-06,  6.1331e-06,  6.2546e-06,  6.1977e-06,
6.3296e-06,  6.2323e-06,  6.2472e-06,  6.3066e-06,  6.3324e-06,
6.1439e-06,  6.2259e-06,  6.0869e-06,  6.2661e-06,  6.2035e-06,
6.3409e-06,  6.1902e-06,  6.1360e-06,  6.1179e-06,  6.2788e-06,
6.2894e-06,  6.3107e-06,  6.1856e-06,  6.2279e-06,  6.2094e-06,
6.1938e-06,  6.0735e-06,  6.3394e-06,  6.3352e-06,  6.3163e-06,
6.2449e-06,  6.3325e-06,  6.1941e-06,  6.2800e-06,  6.3405e-06,
6.1433e-06,  6.0785e-06,  6.2297e-06,  6.2959e-06,  6.2333e-06,
```

```
         6.3392e-06,  6.3386e-06,  6.2877e-06,  6.3300e-06,  6.2449e-06,
         6.3397e-06,  6.2524e-06,  6.1816e-06,  6.0311e-06,  6.0517e-06,
         6.1900e-06,  6.3283e-06,  6.2287e-06,  6.3113e-06,  6.3084e-06,
         6.2257e-06,  6.3372e-06,  6.1166e-06,  6.1883e-06,  6.2423e-06,
         6.2814e-06,  6.0412e-06,  6.3409e-06,  6.3442e-06,  6.0655e-06,
         6.1227e-06,  6.3381e-06,  6.2952e-06,  6.1296e-06,  6.3466e-06,
         6.2846e-06,  6.2718e-06,  6.2859e-06,  6.2418e-06,  6.3411e-06,
         6.2940e-06,  6.2297e-06,  6.2876e-06,  6.3196e-06,  6.0932e-06,
         6.2115e-06,  6.3135e-06,  6.2965e-06,  6.2704e-06,  6.3254e-06,
         6.2710e-06,  6.2673e-06,  6.2067e-06,  6.1755e-06,  6.1181e-06,
         6.2215e-06,  6.2524e-06,  6.3463e-06,  6.3352e-06,  6.3264e-06,
         6.3071e-06,  6.3372e-06,  6.1390e-06,  6.3409e-06,  6.1680e-06,
         6.3225e-06,  6.2620e-06,  6.3070e-06,  6.3390e-06,  6.3423e-06,
         6.3383e-06,  6.3182e-06,  6.3168e-06,  6.1170e-06,  6.1335e-06,
         6.3157e-06,  6.3396e-06,  6.3439e-06,  6.3401e-06,  6.2215e-06,
         6.3159e-06,  6.2074e-06,  6.2846e-06,  6.1834e-06,  6.2614e-06,
         6.0902e-06,  6.3406e-06,  6.0658e-06,  6.3207e-06,  6.3326e-06,
         6.3450e-06,  6.3375e-06,  6.3417e-06,  6.3321e-06,  6.3415e-06,
         6.3387e-06,  6.2574e-06,  6.0624e-06,  6.2280e-06,  6.2745e-06,
         6.3409e-06,  6.2520e-06,  6.2945e-06,  6.2785e-06,  6.2340e-06,
         6.2728e-06,  6.1608e-06,  6.1747e-06,  6.0485e-06,  6.0683e-06,
         6.2975e-06,  6.3334e-06,  6.3400e-06,  6.3399e-06,  6.2534e-06,
         6.3199e-06,  6.2419e-06,  6.2861e-06,  6.3411e-06,  6.2977e-06,
         6.2892e-06,  6.3414e-06,  6.2256e-06,  6.3331e-06,  6.3381e-06,
         6.3315e-06,  6.3015e-06,  6.1786e-06,  6.1478e-06,  6.2615e-06,
         6.1288e-06,  6.2497e-06,  6.2691e-06,  6.2868e-06,  6.2641e-06,
         6.3331e-06,  6.3246e-06,  6.3361e-06,  6.3070e-06,  6.2362e-06,
         6.1901e-06,  6.2767e-06,  6.1920e-06,  6.1213e-06,  6.3321e-06,
         6.1359e-06,  6.1193e-06,  6.2411e-06,  6.1992e-06,  6.3058e-06,
         6.3360e-06,  6.1990e-06,  6.2640e-06,  6.2821e-06,  6.0683e-06,
         6.1157e-06,  6.3404e-06,  6.3444e-06,  6.2932e-06,  6.0704e-06])
end of p.grad

False
Epoch 6 finished
Epoch [6/10], Loss: 1.4521
************************************************************

Learning rate: 0.001

another way to print learning rate:
tensor([[[[ 2.2847e-03,  2.1860e-03,  1.3853e-03,  …,  5.0111e-04,
            8.8820e-04,  1.2330e-03],
          [ 2.6514e-03,  1.8205e-03,  1.0071e-03,  …,  1.6120e-03,
            6.0123e-04,  1.1748e-03],
          [ 1.3725e-03,  4.6453e-04,  4.8231e-04,  …,  1.3412e-03,
            1.3668e-03,  1.6965e-03],
         …,
```

```
  [ 5.8024e-03,  6.0333e-03,  6.4902e-03,  …,  1.0275e-02,
    6.7977e-03,  2.8388e-03],
  [ 8.3674e-03,  7.9674e-03,  7.3076e-03,  …,  9.7602e-03,
    6.4899e-03,  2.6109e-03],
  [ 4.7572e-03,  7.0437e-03,  6.4724e-03,  …,  5.9226e-03,
    4.3873e-03,  1.6419e-03]],

 [[-3.2130e-03, -3.5899e-03, -4.4805e-03,  …, -6.0020e-03,
   -5.9417e-03, -5.6574e-03],
  [-2.8138e-03, -3.7086e-03, -4.9111e-03,  …, -5.0693e-03,
   -6.5558e-03, -6.0162e-03],
  [-4.3900e-03, -5.3918e-03, -5.6942e-03,  …, -5.6348e-03,
   -6.0731e-03, -5.7345e-03],
  …,
  [-8.9741e-04, -9.5221e-04, -8.3912e-04,  …,  8.8138e-04,
   -2.0901e-03, -5.4318e-03],
  [ 1.6667e-03,  1.1319e-03, -1.1382e-04,  …,  2.1220e-04,
   -2.6361e-03, -5.9160e-03],
  [-1.5062e-03,  1.9980e-04, -1.0906e-03,  …, -3.0578e-03,
   -4.6146e-03, -6.8903e-03]],

 [[-9.5591e-03, -9.4711e-03, -1.0311e-02,  …, -1.1175e-02,
   -1.1498e-02, -1.1051e-02],
  [-8.8702e-03, -9.6335e-03, -1.0711e-02,  …, -1.0484e-02,
   -1.2284e-02, -1.1690e-02],
  [-1.0003e-02, -1.1026e-02, -1.1356e-02,  …, -1.1011e-02,
   -1.1724e-02, -1.1397e-02],
  …,
  [-2.4815e-03, -2.3839e-03, -2.7316e-03,  …, -2.6337e-04,
   -3.7166e-03, -7.6650e-03],
  [ 5.2317e-05, -4.4139e-04, -2.1471e-03,  …, -7.5534e-04,
   -4.2856e-03, -7.8767e-03],
  [-3.5386e-03, -1.8114e-03, -3.0489e-03,  …, -3.9374e-03,
   -5.8414e-03, -8.5136e-03]]],


[[[ 4.3662e-03,  5.2433e-03,  5.5774e-03,  …, -8.4435e-03,
   -7.9850e-03, -8.7333e-03],
  [ 5.9142e-03,  4.4357e-03,  2.9184e-03,  …, -1.1110e-02,
   -9.5021e-03, -7.0862e-03],
  [ 4.8943e-03,  5.5489e-03,  2.3051e-03,  …, -9.9278e-03,
   -8.1652e-03, -6.7318e-03],
  …,
  [ 6.4663e-03,  5.1621e-03,  7.1404e-03,  …,  5.9351e-04,
   -4.2561e-07, -7.2013e-04],
  [ 4.9950e-03,  4.2970e-03,  4.4515e-03,  …,  1.1620e-03,
   -2.2207e-04, -2.3830e-03],
  [ 2.6284e-03,  2.8400e-03,  2.6584e-03,  …,  1.4166e-03,
```

```
  -1.4918e-03, -3.4042e-03]],

 [[ 4.1959e-03,  5.2994e-03,  5.9951e-03,  …, -6.4163e-03,
   -5.9921e-03, -6.1482e-03],
  [ 5.6377e-03,  4.5009e-03,  4.2683e-03,  …, -8.6524e-03,
   -7.2709e-03, -4.7705e-03],
  [ 5.4364e-03,  6.1517e-03,  3.8499e-03,  …, -6.5348e-03,
   -5.2183e-03, -4.0935e-03],
  …,
  [ 6.6914e-03,  6.1644e-03,  9.0628e-03,  …,  3.4997e-04,
   -5.8593e-04, -1.4257e-03],
  [ 5.5069e-03,  5.3332e-03,  5.9167e-03,  …,  3.1559e-04,
   -1.5650e-03, -3.5411e-03],
  [ 3.6815e-03,  3.7994e-03,  3.8372e-03,  …,  6.5171e-04,
   -2.1587e-03, -3.6311e-03]],

 [[ 1.4363e-02,  1.4923e-02,  1.5495e-02,  …,  3.9961e-03,
    5.8144e-03,  7.5754e-03],
  [ 1.7358e-02,  1.5777e-02,  1.5158e-02,  …,  3.6703e-04,
    3.0032e-03,  7.7327e-03],
  [ 1.7109e-02,  1.8186e-02,  1.5760e-02,  …,  2.5070e-03,
    4.8107e-03,  7.9817e-03],
  …,
  [ 1.7959e-02,  1.7248e-02,  2.0192e-02,  …,  1.0973e-02,
    1.0349e-02,  9.5079e-03],
  [ 1.6941e-02,  1.6876e-02,  1.7593e-02,  …,  1.1493e-02,
    9.7487e-03,  7.0579e-03],
  [ 1.4818e-02,  1.5271e-02,  1.6042e-02,  …,  1.1358e-02,
    8.9968e-03,  6.7089e-03]]],


[[[-8.8540e-03, -9.3146e-03, -1.0367e-02,  …, -1.1181e-02,
   -1.0389e-02, -1.0598e-02],
  [-7.9075e-03, -8.3108e-03, -9.3832e-03,  …, -1.1265e-02,
   -1.1612e-02, -1.1270e-02],
  [-8.1315e-03, -8.5743e-03, -9.0424e-03,  …, -1.0654e-02,
   -1.1739e-02, -1.1931e-02],
  …,
  [-6.5017e-03, -6.8785e-03, -7.3042e-03,  …, -6.9823e-03,
   -7.7229e-03, -9.6102e-03],
  [-5.7562e-03, -6.2925e-03, -6.4188e-03,  …, -5.4929e-03,
   -5.4657e-03, -7.0422e-03],
  [-5.9240e-03, -5.5036e-03, -4.7150e-03,  …, -5.5743e-03,
   -4.7618e-03, -5.7450e-03]],

 [[-5.5600e-03, -6.0687e-03, -6.6777e-03,  …, -5.6710e-03,
   -5.0563e-03, -5.3497e-03],
  [-4.4123e-03, -4.5060e-03, -5.1153e-03,  …, -6.2742e-03,
```

```
   -6.5992e-03, -5.8831e-03],
  [-4.0129e-03, -4.5099e-03, -4.8338e-03,  …, -6.1616e-03,
   -7.0949e-03, -6.9869e-03],
  …,
  [-3.8987e-03, -4.1586e-03, -4.2791e-03,  …, -3.3637e-03,
   -4.3952e-03, -6.9158e-03],
  [-3.5847e-03, -4.0777e-03, -4.4852e-03,  …, -2.1673e-03,
   -2.4655e-03, -4.5069e-03],
  [-3.8541e-03, -3.4131e-03, -3.0269e-03,  …, -3.1188e-03,
   -2.3548e-03, -3.2896e-03]],

 [[ 7.4815e-04,  6.1996e-04, -2.1474e-04,  …, -4.3242e-05,
    4.9704e-04,  6.2748e-05],
  [ 1.6762e-03,  1.8139e-03,  1.0967e-03,  …, -6.8119e-04,
   -1.1368e-03, -5.2338e-04],
  [ 1.9425e-03,  1.9301e-03,  1.5751e-03,  …, -4.9205e-04,
   -1.4312e-03, -1.5820e-03],
  …,
  [ 9.8304e-04,  4.8409e-04, -9.7537e-05,  …,  1.9862e-03,
    4.2164e-04, -1.8755e-03],
  [ 8.1379e-04,  3.5208e-04, -2.2780e-04,  …,  2.8661e-03,
    2.1225e-03,  4.3286e-04],
  [ 3.7610e-04,  9.8977e-04,  1.1519e-03,  …,  1.9381e-03,
    3.0335e-03,  2.6452e-03]]],


…,


[[[-8.8962e-04, -1.1900e-03, -2.3003e-03,  …, -1.8884e-03,
   -9.3215e-04, -1.0755e-03],
  [-9.5402e-04, -7.7545e-04, -1.7445e-03,  …, -1.4424e-03,
   -4.6372e-04, -4.3824e-04],
  [-2.2777e-03, -8.7660e-04, -1.4094e-03,  …, -1.1768e-03,
   -7.1848e-05,  4.5618e-04],
  …,
  [-1.5714e-03, -2.3855e-03, -2.5695e-03,  …, -1.1798e-03,
   -7.8079e-04,  1.3764e-04],
  [-1.4020e-03, -2.0654e-03, -1.9090e-03,  …, -7.9632e-04,
   -3.0037e-05,  9.6690e-04],
  [ 8.9240e-05, -5.9652e-04, -8.3089e-04,  …, -4.5808e-04,
    8.6888e-04,  9.8624e-04]],

 [[-2.1638e-03, -2.9817e-03, -4.3223e-03,  …, -4.2910e-03,
   -2.6779e-03, -1.9785e-03],
  [-2.5164e-03, -2.7217e-03, -3.7590e-03,  …, -3.7623e-03,
   -2.2370e-03, -1.4889e-03],
  [-4.0621e-03, -2.8776e-03, -3.3193e-03,  …, -3.3339e-03,
```

```
   -1.7258e-03, -7.3777e-04],
  …,
  [-2.4654e-03, -3.2856e-03, -3.2689e-03,  …, -1.5563e-03,
   -9.6363e-04,  1.4222e-04],
  [-1.9467e-03, -2.6296e-03, -2.1944e-03,  …, -8.6887e-04,
   -6.2469e-05,  1.0277e-03],
  [-5.8720e-04, -1.3479e-03, -1.3674e-03,  …, -6.2126e-04,
    6.2564e-04,  8.0574e-04]],

 [[-1.6159e-03, -4.0083e-03, -6.1993e-03,  …, -4.8231e-03,
   -3.0845e-03, -2.1458e-03],
  [-2.4425e-03, -3.9278e-03, -5.6746e-03,  …, -4.4151e-03,
   -2.6066e-03, -1.7789e-03],
  [-4.2743e-03, -4.2113e-03, -5.3133e-03,  …, -4.1946e-03,
   -2.3273e-03, -1.0538e-03],
  …,
  [-2.6470e-03, -3.8235e-03, -3.9839e-03,  …, -1.8080e-03,
   -1.2880e-03, -4.3029e-04],
  [-1.9891e-03, -3.1542e-03, -2.8814e-03,  …, -8.8512e-04,
   -4.2181e-05,  8.6144e-04],
  [-3.4994e-04, -1.5411e-03, -1.7542e-03,  …, -4.8633e-04,
    7.1240e-04,  6.7650e-04]]],


[[[ 3.3139e-03,  3.6321e-03,  4.3599e-03,  …,  7.8974e-03,
    8.7635e-03,  1.0542e-02],
  [ 4.9467e-03,  5.2924e-03,  6.3717e-03,  …,  8.1763e-03,
    9.0149e-03,  1.0998e-02],
  [ 4.5751e-03,  5.5239e-03,  6.7014e-03,  …,  8.7657e-03,
    9.7240e-03,  1.1773e-02],
  …,
  [ 1.0102e-02,  1.1652e-02,  1.2549e-02,  …,  1.6244e-02,
    1.4456e-02,  1.3286e-02],
  [ 9.1919e-03,  1.0969e-02,  1.1708e-02,  …,  1.5564e-02,
    1.4115e-02,  1.3448e-02],
  [ 1.0151e-02,  1.2510e-02,  1.3626e-02,  …,  1.6661e-02,
    1.5276e-02,  1.3973e-02]],

 [[ 5.9949e-03,  6.4383e-03,  7.8816e-03,  …,  8.6241e-03,
    1.0743e-02,  1.3911e-02],
  [ 7.4257e-03,  8.3659e-03,  9.5648e-03,  …,  8.7326e-03,
    1.0536e-02,  1.3993e-02],
  [ 7.3958e-03,  8.3589e-03,  9.0848e-03,  …,  9.2809e-03,
    1.1162e-02,  1.4438e-02],
  …,
  [ 1.0859e-02,  1.1968e-02,  1.3447e-02,  …,  1.8576e-02,
    1.7158e-02,  1.6326e-02],
  [ 1.0341e-02,  1.1939e-02,  1.3163e-02,  …,  1.8821e-02,
```

```
    1.7521e-02,  1.6912e-02],
  [ 1.1126e-02,  1.3602e-02,  1.5029e-02,  …,  1.9330e-02,
    1.8111e-02,  1.6924e-02]],

 [[ 1.0692e-02,  9.5677e-03,  1.0038e-02,  …,  6.7480e-03,
    9.1972e-03,  1.3276e-02],
  [ 1.2857e-02,  1.1363e-02,  1.0830e-02,  …,  6.5524e-03,
    8.7094e-03,  1.3244e-02],
  [ 1.2385e-02,  1.0420e-02,  9.1800e-03,  …,  6.5389e-03,
    8.6183e-03,  1.3238e-02],
  …,
  [ 1.3517e-02,  1.3073e-02,  1.4072e-02,  …,  1.8342e-02,
    1.7098e-02,  1.6432e-02],
  [ 1.3711e-02,  1.3606e-02,  1.4139e-02,  …,  1.9226e-02,
    1.8377e-02,  1.7858e-02],
  [ 1.4001e-02,  1.5152e-02,  1.5572e-02,  …,  1.9552e-02,
    1.9120e-02,  1.8219e-02]]],


[[[-7.7046e-03, -8.1967e-03, -6.9369e-03,  …, -4.6460e-03,
   -3.6973e-03, -3.1921e-03],
  [-3.7796e-03, -5.0359e-03, -3.1617e-03,  …, -1.9398e-03,
   -2.9830e-03, -3.3734e-03],
  [-4.1790e-03, -4.2997e-03, -3.7814e-04,  …, -7.0683e-04,
   -1.2392e-03, -1.4160e-03],
  …,
  [ 3.5909e-03,  2.1367e-03, -2.1527e-04,  …,  6.1409e-03,
    7.7142e-03,  8.2697e-03],
  [ 3.6876e-03,  3.6637e-03,  3.2318e-03,  …,  6.1168e-03,
    7.6081e-03,  7.2043e-03],
  [ 4.7009e-03,  5.0696e-03,  4.4890e-03,  …,  6.4060e-03,
    7.5940e-03,  5.9070e-03]],

 [[-1.0164e-02, -1.0800e-02, -9.8299e-03,  …, -8.8633e-03,
   -7.0393e-03, -5.4091e-03],
  [-6.8248e-03, -7.8682e-03, -6.1753e-03,  …, -6.0028e-03,
   -6.4713e-03, -5.6401e-03],
  [-7.3508e-03, -7.4416e-03, -3.9381e-03,  …, -5.1214e-03,
   -5.1487e-03, -4.1008e-03],
  …,
  [ 1.9109e-04, -1.2643e-03, -3.8744e-03,  …,  4.7375e-03,
    6.9858e-03,  7.4385e-03],
  [ 4.5205e-04,  2.5237e-04,  1.3297e-04,  …,  5.8518e-03,
    7.9901e-03,  7.1929e-03],
  [ 1.6753e-03,  1.7488e-03,  1.6672e-03,  …,  6.3342e-03,
    7.8457e-03,  6.1435e-03]],

 [[-2.4119e-03, -4.4368e-03, -5.6975e-03,  …, -9.2298e-03,
```

```
           -6.8999e-03, -4.9072e-03],
          [ 1.2266e-03, -1.8739e-03, -3.3516e-03,  …, -6.1421e-03,
           -6.6162e-03, -5.2340e-03],
          [-5.6984e-04, -2.9006e-03, -1.8389e-03,  …, -5.1198e-03,
           -4.9863e-03, -3.1885e-03],
          …,
          [ 1.7609e-03, -1.4419e-03, -4.6586e-03,  …,  5.5712e-03,
            7.6844e-03,  8.0222e-03],
          [ 1.9925e-03,  3.4286e-05, -8.5228e-04,  …,  6.4824e-03,
            8.5417e-03,  7.7221e-03],
          [ 3.1249e-03,  2.0790e-03,  1.0594e-03,  …,  6.9585e-03,
            8.1682e-03,  6.3147e-03]]]])
tensor([ 1.3970e-08,  6.3330e-08,  5.1223e-09, -5.8208e-10, -3.3528e-08,
         2.9802e-08, -8.8476e-09,  6.8918e-08, -7.5670e-10, -1.5367e-08,
         1.9558e-08,  2.7940e-09, -2.7940e-08,  3.0268e-09,  9.3132e-10,
        -2.0955e-09,  5.7742e-08,  6.1467e-08, -3.3528e-08,  1.2107e-08,
         1.2573e-08,  4.0978e-08, -1.4901e-08, -8.3819e-09, -1.3970e-09,
        -1.0245e-08, -2.4302e-09,  6.5484e-11,  4.5402e-09, -1.0245e-08,
        -1.3039e-08,  4.6566e-09, -1.0477e-08, -6.4028e-10, -2.5728e-08,
        -4.4238e-09, -4.8894e-09,  1.2573e-08,  1.3970e-08, -9.0804e-09,
        -1.5367e-08, -1.4435e-08,  2.2670e-08,  1.3737e-08, -3.0734e-08,
         2.5146e-08,  1.4399e-08,  1.1642e-09,  4.1910e-09, -4.6566e-10,
        -6.2864e-09, -2.0955e-09, -8.4983e-09,  1.3504e-08, -1.6298e-09,
        -2.0489e-08, -1.1176e-08,  5.5297e-10,  1.0477e-09, -1.5250e-08,
        -2.3283e-09,  1.5716e-09,  5.3551e-08, -8.5565e-09, -9.3132e-10,
         2.7940e-08,  2.3283e-10,  3.4459e-08,  1.6298e-09,  8.8476e-09,
        -1.2806e-09,  3.2131e-08,  6.5193e-09, -2.8871e-08,  2.3283e-09,
         1.3970e-08, -9.8953e-10,  4.6566e-09, -1.9558e-08,  7.9162e-09,
         2.2352e-08, -1.3970e-09,  5.8208e-09,  9.8953e-09, -1.6764e-07,
        -9.7789e-09,  6.0583e-07,  1.1292e-08,  8.7311e-09, -8.7311e-09,
         0.0000e+00, -3.3062e-08,  2.1420e-08, -5.5879e-09,  3.2131e-08,
         2.9104e-09])
tensor([ 1.7335e-02,  2.2620e-02,  1.2650e-02,  5.0085e-03,  1.8026e-02,
        -1.2118e-02,  7.6520e-03,  8.3127e-04,  6.1883e-03, -3.6103e-02,
        -3.0392e-02,  5.9812e-03,  1.7779e-02, -2.7774e-02,  7.6714e-03,
         2.2642e-03, -4.3786e-02, -3.5800e-03,  1.8792e-02, -5.7205e-03,
         4.6045e-03,  2.4340e-02,  9.4467e-03,  9.8303e-03, -3.1934e-02,
         3.5556e-03,  5.3372e-03,  1.7776e-02, -4.3666e-02, -8.6568e-03,
         4.7767e-03, -4.0054e-03,  1.4382e-02,  3.1821e-03,  7.2496e-03,
         2.0206e-03, -2.6295e-02,  5.4190e-03,  4.8262e-03, -1.9775e-02,
         1.0739e-02,  1.4734e-02,  1.4100e-02,  1.9271e-03,  4.3806e-02,
         5.9110e-03,  1.4041e-02,  7.0120e-03, -3.3821e-02, -6.6076e-03,
        -5.1316e-03,  4.4707e-02,  8.8838e-03, -2.9530e-02,  1.3998e-02,
        -1.1304e-02,  8.4840e-02,  5.2216e-03,  2.8680e-03,  8.8588e-03,
        -5.8060e-03,  1.1416e-03, -3.6897e-02,  4.9075e-03, -1.1466e-02,
         1.3345e-02,  3.3366e-03,  3.2291e-03, -3.1925e-03,  1.4130e-02,
         3.6152e-03,  5.1215e-03, -3.1312e-02,  5.1321e-03,  2.0750e-02,
        -2.4320e-02,  3.1416e-03, -2.5531e-02,  1.3188e-02, -7.8167e-03,
```

```
         1.0689e-02,  2.2549e-02,  5.2503e-03, -2.5093e-03, -4.1499e-02,
        -3.1779e-02, -2.9208e-06, -1.1994e-02,  1.5697e-03,  2.8721e-02,
        -9.8465e-03, -6.2158e-04,  2.7155e-02, -1.3655e-02, -2.1636e-02,
         6.3388e-03])
tensor([ 1.3700e-02,  3.6330e-02,  1.0308e-02,  5.0438e-03,  1.2387e-02,
        -1.1380e-02,  4.0456e-03, -6.3300e-03,  3.4033e-03, -3.5637e-03,
        -1.2746e-02,  6.4361e-03,  1.7428e-02, -2.0331e-05,  5.7017e-03,
         7.3351e-03, -1.8921e-02,  4.8038e-04,  1.5821e-02,  2.8252e-03,
         7.3676e-03,  9.0772e-03, -2.9997e-02,  2.2323e-03, -1.3695e-02,
         3.4880e-03, -4.3616e-04,  7.4737e-03, -8.7995e-03, -1.0933e-02,
         9.5889e-03,  1.5603e-02,  5.8558e-03,  6.4832e-03,  4.4422e-03,
         8.8566e-03,  6.9208e-03,  1.2436e-02,  1.8197e-02,  7.0881e-03,
         1.9159e-02,  7.3764e-03,  1.4435e-02,  9.3038e-03,  3.0460e-02,
         1.7089e-02,  7.2741e-03,  3.1680e-03, -4.2502e-03,  9.0121e-03,
        -3.6361e-03,  1.7590e-02,  3.4750e-03, -5.8237e-03,  6.6271e-03,
        -2.3189e-02,  3.6827e-02,  1.1563e-03,  9.4702e-03,  4.3065e-03,
         8.2252e-03, -5.0796e-03, -1.7924e-02,  3.6408e-04,  3.2397e-03,
         9.6358e-03,  3.5959e-03, -2.8711e-02,  6.2534e-03,  8.2673e-03,
         1.5008e-03,  1.1037e-02,  1.2465e-05,  1.3700e-02,  1.1644e-02,
        -6.9125e-03,  2.8483e-04,  9.8743e-04,  2.1566e-03,  5.5706e-04,
        -1.1139e-02,  1.0526e-02,  5.0476e-03,  4.8852e-03,  2.3812e-02,
         8.9182e-03,  6.0038e-03, -1.0099e-02,  2.6418e-03,  1.3159e-02,
         5.5611e-03, -1.9624e-02,  4.9860e-03,  5.1552e-03, -1.0660e-02,
         6.4676e-03])
tensor([[[[ 1.7895e-03,  1.8867e-03,  1.7326e-03,  1.5859e-03,  1.2193e-03],
          [ 1.6225e-03,  1.5604e-03,  1.6559e-03,  1.3116e-03,  8.2410e-04],
          [ 1.5874e-03,  1.5450e-03,  1.2861e-03,  8.3109e-04,  5.3684e-04],
          [ 1.7948e-03,  1.4455e-03,  9.1288e-04,  5.7730e-04,  6.1002e-04],
          [ 1.5482e-03,  1.0548e-03,  5.0453e-04,  2.1075e-04, -3.1399e-04]],

         [[ 1.0209e-03,  7.9355e-04,  2.5129e-04,  7.8128e-04,  1.0513e-03],
          [ 4.7293e-05,  8.7913e-05, -4.7778e-05, -8.0438e-05,  1.0567e-04],
          [-1.4792e-03, -1.0706e-03, -1.2142e-03, -6.6288e-04, -1.2146e-03],
          [-1.6969e-03, -3.4682e-04, -1.5405e-03, -1.1118e-03, -1.2618e-03],
          [-1.9825e-03, -8.2072e-04, -1.4861e-03, -2.7612e-03, -1.3627e-03]],

         [[ 1.9216e-04,  4.5491e-04,  7.3965e-04,  8.4886e-04,  7.9737e-04],
          [-2.8743e-04, -2.0833e-04,  1.1671e-04,  2.0906e-04, -8.7012e-05],
          [-7.7979e-04, -9.0355e-04, -1.0481e-03, -1.3829e-03, -1.2742e-03],
          [-1.1846e-03, -1.6904e-03, -1.9776e-03, -2.0287e-03, -2.0221e-03],
          [-1.8164e-03, -1.9283e-03, -2.1325e-03, -2.1793e-03, -2.2293e-03]],

         …,

         [[ 2.2847e-03,  2.1270e-03,  2.1278e-03,  1.9648e-03,  1.8067e-03],
          [ 1.8559e-03,  1.5442e-03,  1.5651e-03,  1.3152e-03,  1.1768e-03],
          [ 1.6515e-03,  1.5006e-03,  1.0655e-03,  7.7700e-04,  7.2365e-04],
          [ 1.4239e-03,  1.5787e-03,  1.2340e-03,  8.9175e-04,  7.7498e-04],
```

```
  [ 1.1440e-03,  1.7777e-03,  1.5150e-03,  9.3984e-04,  6.6989e-04]],

 [[ 2.1747e-03,  1.8614e-03,  1.7407e-03,  1.4598e-03,  1.2626e-03],
  [ 2.0281e-03,  1.7340e-03,  1.6968e-03,  1.3501e-03,  1.0809e-03],
  [ 2.0073e-03,  2.2925e-03,  2.1824e-03,  1.8955e-03,  1.3670e-03],
  [ 2.0210e-03,  2.1741e-03,  2.1586e-03,  1.8306e-03,  1.5916e-03],
  [ 1.8795e-03,  2.2392e-03,  2.0518e-03,  1.5210e-03,  1.5698e-03]],

 [[ 2.1021e-03,  1.9307e-03,  1.7928e-03,  1.2260e-03,  8.1205e-04],
  [ 1.8359e-03,  1.3573e-03,  1.3300e-03,  8.5989e-04,  6.0419e-04],
  [ 2.0204e-03,  1.9299e-03,  1.2321e-03,  6.1382e-04,  3.3783e-04],
  [ 1.6429e-03,  1.8711e-03,  1.1206e-03,  5.4371e-04,  1.3479e-04],
  [ 1.4364e-03,  1.7461e-03,  1.0759e-03,  5.9975e-04,  4.4012e-04]]],


[[[-1.8381e-04, -2.6795e-04, -4.2501e-04, -5.1455e-04, -4.0884e-04],
  [-5.6770e-04, -4.9767e-04, -6.4072e-04, -7.6454e-04, -8.3030e-04],
  [-1.1006e-03, -1.1516e-03, -8.7717e-04, -7.6826e-04, -7.3795e-04],
  [-5.0027e-04, -4.7577e-04, -3.6338e-04, -4.7071e-04, -1.7404e-04],
  [ 1.2701e-04, -5.0031e-05,  3.6390e-04,  4.3361e-04,  4.0581e-04]],

 [[ 5.5612e-06,  5.7816e-04,  2.5889e-04, -1.3926e-04, -1.6216e-04],
  [-8.5355e-04,  4.9062e-04,  4.8125e-04,  3.0879e-04,  3.3403e-04],
  [-3.6156e-04, -2.2504e-04,  1.2406e-04,  3.3232e-04,  1.0497e-03],
  [ 1.2798e-03,  4.6849e-04,  4.3575e-04,  8.0257e-05,  8.5372e-04],
  [ 1.9322e-03,  9.6189e-04,  3.1621e-04,  1.9763e-04,  9.2679e-04]],

 [[ 6.6893e-04,  4.3129e-04,  3.6132e-04,  4.4622e-04,  4.6421e-04],
  [ 6.3962e-04,  3.4404e-04,  1.7214e-04,  2.0331e-04,  2.3916e-04],
  [ 7.1701e-04,  4.6755e-04,  2.3578e-04,  1.8666e-04,  1.9969e-04],
  [ 9.8523e-04,  7.7078e-04,  4.6196e-04,  3.1292e-04,  1.5503e-04],
  [ 9.2678e-04,  9.1857e-04,  7.1192e-04,  4.6711e-04,  4.1883e-04]],

 …,

 [[-4.3054e-04, -5.9528e-04, -8.4068e-04, -8.7151e-04, -7.5828e-04],
  [-5.1506e-04, -5.8061e-04, -7.9663e-04, -9.4267e-04, -7.3867e-04],
  [-3.3981e-04, -4.6504e-04, -6.5368e-04, -8.2557e-04, -8.3178e-04],
  [ 2.0643e-04, -3.7195e-05, -3.9669e-05, -4.0682e-04, -5.1269e-04],
  [ 2.3115e-04,  1.0677e-04,  1.5004e-05, -3.3043e-04, -3.8213e-04]],

 [[-9.7080e-04, -1.1123e-03, -1.2311e-03, -1.1559e-03, -1.0761e-03],
  [-1.0145e-03, -1.0197e-03, -1.1738e-03, -1.2361e-03, -1.1443e-03],
  [-9.6054e-04, -1.0795e-03, -1.1092e-03, -1.2508e-03, -1.0915e-03],
  [-8.1056e-04, -9.3967e-04, -7.7676e-04, -9.5065e-04, -7.8063e-04],
  [-7.3221e-04, -8.8894e-04, -5.7613e-04, -7.2862e-04, -5.9887e-04]],

 [[ 2.7114e-04,  5.0220e-05, -6.7824e-04, -8.8826e-04, -7.4760e-04],
```

```
  [ 5.4868e-05, -1.3652e-06, -6.4437e-04, -7.4784e-04, -6.1305e-04],
  [ 2.5407e-04,  3.6125e-04, -1.7831e-04, -4.6531e-04, -3.3642e-04],
  [ 6.0973e-04,  5.2048e-04,  5.1035e-04, -3.0336e-05,  4.5061e-05],
  [ 6.7756e-04,  5.7893e-04,  4.9721e-04,  1.4895e-04,  2.6923e-04]]],


[[[-4.9236e-05,  2.1655e-04,  3.6177e-04,  7.5619e-04,  9.7773e-04],
  [-3.2509e-04, -8.7244e-04,  1.9083e-04,  5.2528e-04,  8.1163e-04],
  [ 1.3221e-04, -2.0668e-04,  2.1276e-04,  4.1578e-04,  7.1667e-04],
  [ 5.1699e-04,  1.8454e-04,  2.2422e-04,  6.0262e-04,  5.5552e-04],
  [ 2.2529e-04, -2.9565e-04, -9.5151e-05,  4.3933e-04,  5.7222e-04]],

 [[ 2.4613e-03,  2.4445e-03,  4.7516e-03,  5.1126e-03,  6.2036e-03],
  [ 2.9619e-03,  2.3725e-03,  4.4274e-03,  5.4293e-03,  5.7303e-03],
  [ 3.0341e-03,  1.2986e-03,  4.1525e-03,  5.4578e-03,  4.5540e-03],
  [ 2.0733e-03,  1.2916e-03,  3.8551e-03,  4.8071e-03,  3.8867e-03],
  [ 1.7736e-03,  1.9363e-03,  3.3957e-03,  4.0165e-03,  3.3597e-03]],

 [[-9.8338e-04, -1.1704e-03, -9.8976e-04, -3.8717e-04,  2.2318e-04],
  [-9.5621e-04, -1.3978e-03, -1.2999e-03, -7.0279e-04,  3.6467e-04],
  [-8.2142e-04, -1.0035e-03, -9.4724e-04, -2.3241e-04,  5.3866e-04],
  [-9.3534e-04, -1.3561e-03, -1.1517e-03, -4.2670e-04,  8.0682e-05],
  [-1.1587e-03, -8.4225e-04, -4.2592e-04, -2.4111e-05,  2.3809e-04]],

 …,

 [[-8.1961e-04, -9.4342e-04, -5.8319e-04,  5.4833e-04,  1.2136e-03],
  [-5.9457e-04, -6.6123e-04, -2.2125e-04,  8.7191e-04,  1.4181e-03],
  [-2.9329e-04, -8.4158e-05,  1.0015e-04,  1.1073e-03,  1.3839e-03],
  [-2.4941e-04, -8.4761e-05,  1.8577e-04,  8.9275e-04,  1.0170e-03],
  [-1.1770e-04,  8.0066e-04,  9.5678e-04,  1.2844e-03,  1.2823e-03]],

 [[ 4.4345e-04,  5.1030e-04,  8.2454e-04,  1.0988e-03,  1.3958e-03],
  [ 6.9153e-04,  5.6218e-04,  9.6182e-04,  1.4230e-03,  1.4378e-03],
  [ 9.5005e-04,  1.2163e-03,  1.7344e-03,  1.6560e-03,  1.5810e-03],
  [ 1.3689e-03,  1.5566e-03,  2.0899e-03,  1.6067e-03,  1.3123e-03],
  [ 1.3450e-03,  1.6783e-03,  2.0265e-03,  1.6079e-03,  1.3290e-03]],

 [[-2.0798e-03, -2.7373e-03, -2.2736e-03, -1.4302e-03, -8.4590e-04],
  [-1.3374e-03, -1.8368e-03, -1.5906e-03, -9.2004e-04, -6.6640e-04],
  [-1.1775e-03, -1.3537e-03, -1.2169e-03, -2.4019e-04, -6.0556e-04],
  [-1.1135e-03, -7.6712e-04, -5.6297e-04, -5.8520e-04, -7.5783e-04],
  [-5.3075e-04,  1.6835e-04,  2.0642e-04,  1.0596e-04, -3.0393e-04]]],


…,
```

```
[[[ 6.7231e-04, -1.6903e-04, -1.7366e-03, -2.4899e-03, -1.3712e-03],
  [ 8.7099e-04,  1.6557e-04, -6.2269e-04, -1.1091e-03, -7.3232e-04],
  [ 2.4153e-04,  1.1763e-04,  3.6163e-04,  6.8705e-04,  5.9561e-04],
  [-6.9865e-04, -6.1075e-04, -3.7274e-04, -7.7369e-06,  2.8770e-04],
  [-9.9607e-04, -1.2653e-03, -1.7566e-03, -5.5387e-04, -2.0517e-04]],

 [[ 1.7429e-03,  3.8355e-03,  1.5414e-03, -3.4694e-03, -2.6465e-04],
  [ 1.5625e-03,  4.0591e-03,  1.0524e-03, -3.4309e-03,  3.9424e-05],
  [ 1.7714e-03,  2.0684e-03,  4.1280e-04, -1.2392e-03,  8.0153e-04],
  [ 9.9404e-04,  9.5380e-04,  2.0371e-04,  7.1329e-04,  1.6474e-03],
  [-5.1742e-04,  1.2305e-04, -1.7464e-03, -1.1041e-03,  8.2048e-04]],

 [[ 3.8292e-04,  8.2586e-04,  8.9769e-04,  7.4759e-04,  4.1088e-04],
  [ 4.3177e-04,  7.6785e-04,  1.0601e-03,  8.9859e-04,  6.2108e-04],
  [ 1.7971e-04,  4.8021e-04,  6.2745e-04,  5.4050e-04,  4.2789e-04],
  [ 5.1141e-04,  4.1792e-04,  3.6451e-04,  4.2312e-04,  2.1990e-04],
  [ 5.5401e-04,  4.5536e-04,  2.0073e-04,  3.6843e-04, -7.7680e-05]],

 …,

 [[-1.4053e-05,  6.3707e-04,  8.9853e-04,  1.0627e-03,  8.6160e-04],
  [-1.6478e-04,  3.4178e-04,  5.1042e-04,  4.4019e-04,  2.4532e-04],
  [-3.0322e-04,  6.4004e-05, -1.1176e-04,  3.6106e-05, -1.0787e-04],
  [-7.7568e-05, -8.5929e-05, -2.1874e-04, -6.2858e-05,  6.1919e-05],
  [ 2.4775e-04,  4.7319e-05, -9.5999e-05, -5.1037e-05, -1.8606e-04]],

 [[ 3.3233e-04,  5.4028e-04,  2.3735e-04,  3.2077e-04,  2.5689e-04],
  [ 2.4076e-04,  3.0378e-04,  1.0218e-04,  1.6763e-04,  2.1475e-04],
  [ 6.7278e-04,  2.4051e-04,  5.9330e-04,  6.9037e-04,  4.5541e-04],
  [ 5.8526e-04, -9.9901e-06,  2.3915e-04,  3.8044e-04,  3.1234e-04],
  [ 3.5776e-04,  7.0600e-05,  1.2737e-04,  3.2226e-04,  2.7857e-04]],

 [[-1.5497e-04,  1.4383e-04,  6.7909e-04,  1.0945e-03,  1.0485e-03],
  [ 5.6313e-04,  6.4322e-04, -1.5713e-04,  1.2828e-04,  1.1992e-04],
  [ 1.1402e-03,  8.3699e-04,  1.6220e-04,  4.1760e-04,  8.9612e-05],
  [ 1.8778e-03,  7.9515e-04,  4.4145e-04,  8.0969e-04,  5.9220e-04],
  [ 1.3717e-03,  7.9185e-04,  6.1680e-04,  4.7635e-04,  4.2564e-04]]],


[[[ 5.4282e-05, -5.8172e-04, -1.1861e-03, -1.1226e-03, -1.3361e-03],
  [ 6.6158e-04, -5.0571e-04, -1.3356e-03, -1.9966e-03, -2.1732e-03],
  [ 2.0382e-04,  1.5112e-04, -5.7772e-04, -2.0848e-03, -2.2313e-03],
  [ 9.9820e-05,  5.1128e-04,  4.0004e-04, -1.3262e-03, -2.2375e-03],
  [-6.5150e-04,  1.5972e-04,  6.0510e-04, -6.0990e-04, -1.2718e-03]],

 [[-2.8869e-03, -4.1579e-03, -3.9716e-03, -2.9484e-03, -7.7982e-04],
  [-2.1699e-03, -3.1959e-03, -3.5167e-03, -3.3477e-03, -7.6616e-04],
  [-1.4565e-03, -1.6870e-03, -3.2171e-03, -3.5965e-03, -1.4772e-03],
```

```
  [-1.7716e-03, -9.3873e-04, -1.7978e-03, -3.4485e-03, -2.4789e-03],
  [-1.8550e-03, -1.6568e-03, -1.7434e-03, -2.1688e-03, -1.4283e-03]],

 [[ 7.9571e-04,  1.2815e-04, -8.2375e-04, -1.6813e-03, -2.2350e-03],
  [ 1.2003e-04,  4.4580e-04, -1.8053e-04, -1.3442e-03, -1.8321e-03],
  [-3.5877e-05,  4.0190e-04,  4.7031e-04, -5.2276e-04, -1.1434e-03],
  [ 1.6625e-04,  5.2927e-04,  8.4666e-04,  7.9457e-05, -2.6677e-04],
  [ 5.5802e-04,  7.0028e-04,  6.2299e-04,  1.3178e-04, -1.5781e-04]],

 …,

 [[ 7.8973e-04,  2.9100e-04, -5.7113e-04, -1.4349e-03, -2.0741e-03],
  [ 6.2409e-04,  5.1243e-04, -1.6299e-05, -7.8742e-04, -1.4062e-03],
  [ 1.1192e-03,  1.3946e-03,  9.7705e-04, -2.4827e-04, -8.0524e-04],
  [ 1.1611e-03,  1.3811e-03,  1.2815e-03, -4.6673e-05, -5.0870e-04],
  [ 1.2253e-03,  1.1694e-03,  1.0303e-03, -3.7935e-05, -5.4026e-04]],

 [[ 7.2826e-04,  7.4581e-04,  6.9760e-04,  6.4129e-04,  5.9658e-04],
  [ 9.5774e-04,  9.7847e-04,  8.2305e-04,  8.0058e-04,  6.5563e-04],
  [ 1.5230e-03,  1.6246e-03,  1.3914e-03,  7.8525e-04,  6.0196e-04],
  [ 1.3446e-03,  1.3904e-03,  1.2518e-03,  4.4443e-04,  1.4801e-04],
  [ 1.1920e-03,  1.2798e-03,  1.2526e-03,  4.0655e-04,  2.4776e-04]],

 [[-4.4258e-04, -8.1224e-04, -1.7569e-03, -2.5641e-03, -3.3613e-03],
  [-6.2759e-04, -5.7823e-04, -1.0305e-03, -1.7515e-03, -2.3556e-03],
  [ 8.6327e-05,  3.2153e-04, -6.8818e-05, -1.1314e-03, -1.5824e-03],
  [ 3.8145e-04,  4.7105e-04,  2.1506e-04, -9.1866e-04, -1.1959e-03],
  [ 4.8974e-04,  3.9601e-04, -1.3677e-05, -1.0004e-03, -1.2231e-03]]],


[[[ 6.9914e-04,  6.3632e-04,  8.3349e-04,  1.1645e-03,  1.8408e-03],
  [ 4.6938e-04,  1.0104e-04,  2.7424e-04,  8.0553e-04,  1.2615e-03],
  [ 5.1964e-04,  1.9323e-04, -2.2170e-04,  5.4397e-04,  1.4819e-03],
  [ 4.7005e-04,  3.3821e-04,  3.3556e-04,  1.0509e-03,  1.6758e-03],
  [ 6.9986e-04,  1.0581e-03,  1.7908e-03,  1.9677e-03,  2.0777e-03]],

 [[ 2.1082e-03,  2.3189e-04,  1.1093e-03,  2.3647e-03,  3.4344e-03],
  [ 2.0273e-03,  4.6060e-04,  5.5204e-04,  1.7323e-03,  3.5443e-03],
  [ 9.4050e-04,  6.3303e-05,  2.8591e-04,  2.3499e-03,  3.1538e-03],
  [ 2.3408e-04, -2.9607e-04, -6.5404e-05,  1.0751e-03,  1.2688e-03],
  [ 2.9036e-04,  5.4148e-05, -1.3772e-04,  1.2968e-03,  9.0359e-04]],

 [[-1.3984e-04, -2.4001e-04, -3.6442e-04, -1.7957e-06,  2.7272e-04],
  [-3.2476e-04, -5.0029e-04, -4.3519e-04, -2.5828e-04,  3.0627e-04],
  [-3.2344e-04, -4.0698e-04, -4.9514e-05,  6.5549e-05,  4.9182e-04],
  [-2.8610e-04, -3.0674e-04, -4.0660e-05,  5.1830e-04,  5.3930e-04],
  [ 5.8670e-06, -2.4907e-04,  4.4670e-05,  3.8346e-04,  5.3778e-04]],
```

```
         …,

         [[-8.8467e-05, -1.6375e-04, -1.0420e-04,  1.5295e-04,  6.7202e-04],
          [-3.2273e-04, -2.4822e-04, -1.4845e-04,  2.3002e-04,  9.2147e-04],
          [-2.9728e-04, -1.7354e-04,  1.2432e-04,  5.2160e-04,  9.9727e-04],
          [-2.7597e-04, -3.4742e-05,  2.8253e-04,  7.3995e-04,  8.7389e-04],
          [-2.5977e-04, -1.5093e-04,  1.4549e-04,  6.0749e-04,  7.0814e-04]],

         [[ 1.9779e-04,  2.0167e-04,  3.5137e-04,  5.8976e-04,  6.6879e-04],
          [ 1.0572e-04,  7.1515e-05,  2.9664e-04,  5.6344e-04,  7.2762e-04],
          [ 1.1495e-04,  4.1647e-06,  6.0916e-05,  4.8955e-04,  6.8461e-04],
          [ 1.1497e-04,  1.1576e-04,  9.9723e-05,  4.8587e-04,  7.7597e-04],
          [ 4.6154e-05,  5.5064e-05,  1.9292e-04,  6.3560e-04,  6.8780e-04]],

         [[ 1.8895e-04,  1.2055e-04,  6.6570e-04,  9.4906e-04,  1.4584e-03],
          [-7.8462e-05, -8.4365e-05,  2.3487e-04,  7.6161e-04,  1.4880e-03],
          [-1.1680e-04, -6.5324e-05,  4.4803e-04,  1.2377e-03,  1.6515e-03],
          [-3.6096e-05,  1.8402e-04,  5.1254e-04,  1.1925e-03,  1.3385e-03],
          [ 4.2005e-05,  1.1812e-05,  4.1899e-04,  1.2226e-03,  1.3493e-03]]])
tensor([ 1.0276e-09,  1.0815e-09, -9.2768e-11,  1.9187e-09, -9.6793e-10,
        -1.2169e-09, -1.0691e-09, -1.5762e-09,  7.3500e-09, -4.5180e-09,
         1.4774e-09,  7.7654e-10, -3.1229e-10, -1.9653e-09,  1.0721e-10,
         6.0299e-09,  5.4414e-09, -3.6742e-09,  1.6481e-09, -1.8082e-10,
         5.7662e-10, -5.9621e-09,  1.2429e-10, -5.3194e-10, -3.8131e-10,
        -8.5500e-10, -2.2150e-09,  1.0439e-08, -8.6533e-10,  3.2585e-09,
         1.0077e-09, -5.2410e-10,  2.0766e-09,  7.2899e-09, -1.5034e-10,
         6.4256e-10,  3.0351e-09,  1.9299e-09,  5.2547e-09, -2.6312e-10,
         5.2835e-09, -2.8472e-09, -3.2628e-10,  3.5834e-10, -8.3583e-10,
        -1.2542e-08, -1.7711e-09, -2.9409e-09,  9.7896e-10,  2.7275e-09,
        -4.5106e-09,  1.4688e-09,  1.7708e-09,  4.8129e-10, -8.1616e-10,
        -1.3849e-09,  9.7415e-11, -1.0006e-09, -5.0924e-09,  4.0639e-09,
         1.7365e-09, -9.5309e-10, -4.0063e-10, -4.2513e-10,  2.8267e-09,
        -1.6047e-09, -2.4090e-09, -3.5773e-09,  1.5442e-08,  3.9383e-09,
        -1.0833e-08, -2.7416e-11,  5.5501e-09, -1.5677e-10, -5.5638e-10,
        -5.4595e-09, -5.4129e-11,  2.7500e-09, -2.8679e-09, -3.6712e-09,
        -3.8020e-10, -4.7329e-10, -2.6247e-09, -6.3869e-11, -4.3613e-11,
         1.6107e-09, -2.2828e-09, -5.7705e-09,  3.8475e-09, -1.6474e-09,
        -5.1466e-10, -1.9619e-09, -1.1490e-08,  4.0314e-09,  6.2655e-09,
         4.4952e-09,  2.1439e-09, -5.5813e-09, -4.1277e-09,  5.1463e-09,
         9.8756e-10,  8.1286e-10,  8.1855e-12,  2.1654e-09,  2.5904e-09,
         2.1844e-09, -8.0826e-09,  2.9922e-10,  1.2888e-09,  7.7267e-10,
        -2.0736e-09,  5.8076e-09, -2.1305e-10,  8.4174e-10,  1.4422e-09,
        -1.5768e-10, -2.2696e-09, -1.7126e-09,  2.6897e-10, -8.3219e-10,
         1.2784e-09,  7.1264e-10,  1.3476e-09,  1.5377e-09,  4.5961e-09,
        -3.5905e-09, -3.9082e-09,  2.6739e-10,  4.6782e-10, -7.8596e-09,
         3.5661e-09,  7.6881e-10,  1.0346e-09,  2.1682e-09,  5.2383e-09,
         9.2746e-09,  2.9810e-09,  2.4145e-09, -4.2314e-09,  5.1045e-10,
         6.6345e-10, -7.7611e-10, -6.6071e-10, -4.5041e-09, -1.4868e-09,
```

```
        -5.6777e-10, -8.1362e-09, -5.2218e-09,  9.5565e-10, -4.9881e-09,
         2.3056e-10, -3.4816e-09,  3.7783e-09,  7.8511e-09,  8.0828e-09,
         8.0934e-10,  3.4327e-09,  8.9040e-10,  6.1684e-09,  3.3546e-10,
        -1.6475e-09, -1.8172e-09,  4.2940e-10,  1.9256e-10,  3.2367e-10,
        -8.8370e-09, -1.0312e-09, -1.1862e-09, -6.8195e-09, -1.1362e-08,
         3.3009e-09,  8.9093e-10, -1.6371e-11, -1.5062e-08,  9.1059e-10,
         6.4665e-10, -2.7313e-11,  1.7112e-09,  1.4640e-09,  2.5375e-10,
        -3.6718e-09,  5.3756e-10, -6.8820e-09, -1.0414e-10,  9.1131e-10,
         1.7877e-09,  9.1359e-09, -6.9022e-10, -6.8079e-10,  1.1072e-09,
         2.0204e-09,  1.6242e-09, -1.6758e-09,  3.0345e-09, -2.3305e-09,
        -1.7653e-09,  9.5042e-10, -2.4897e-10,  1.9097e-09,  1.1761e-09,
         4.2075e-10, -3.0741e-10,  5.5689e-10, -8.6038e-10, -3.0832e-10,
        -5.7230e-10, -2.1469e-09, -1.1505e-09, -1.7827e-09,  1.1395e-09,
         2.1232e-09, -3.8122e-10, -1.2054e-09, -4.8539e-09,  1.9636e-09,
        -6.0845e-10,  1.6760e-09, -8.1599e-09,  3.1282e-09,  5.6568e-09,
        -1.8144e-10, -9.7725e-10,  2.3834e-09,  7.5909e-09, -1.1068e-08,
        -2.4584e-09, -1.3397e-09,  3.4404e-09,  6.0880e-09,  3.6399e-09,
         2.0731e-09, -1.3692e-09, -3.0770e-09,  2.7967e-09,  1.4595e-09,
         1.6162e-09, -2.6645e-08,  4.1882e-09,  2.3591e-09, -1.8467e-09,
        -1.3438e-10, -2.2126e-09,  1.3642e-09, -1.3083e-09,  3.4026e-09,
         6.1681e-10, -4.4939e-09,  1.7045e-08,  1.7940e-10,  8.5493e-11,
        -3.4908e-09, -1.8619e-10,  2.5593e-09, -1.4205e-09, -8.7859e-09,
         6.5688e-10])
tensor([-1.7832e-02,  2.2518e-03, -1.9957e-03,  2.1103e-02,  2.1938e-03,
        -1.3434e-03, -4.4775e-03, -8.9622e-04, -1.1854e-02,  9.9238e-03,
        -9.9273e-03, -1.5491e-02, -7.8572e-03,  3.1698e-02,  3.0741e-03,
        -6.3311e-04,  1.0069e-02, -3.3448e-03,  1.3641e-02,  3.3125e-02,
        -1.6038e-04,  9.3981e-02,  3.5554e-03,  1.0345e-02,  1.6807e-03,
        -1.3082e-02, -1.2079e-02,  7.9072e-02, -5.9136e-03, -1.1491e-02,
         6.6860e-03,  2.2915e-02, -1.2164e-03, -2.7726e-02, -8.3271e-03,
         8.6511e-05, -4.5614e-03,  3.6385e-04,  6.0152e-02,  1.7716e-03,
        -2.4989e-03, -6.8440e-03,  1.4287e-03, -5.9903e-03,  2.3885e-03,
        -1.7547e-03, -5.4532e-03, -2.8062e-02,  3.6038e-02,  9.2102e-04,
         1.8147e-03, -9.3509e-03, -2.7288e-02, -7.7386e-03,  1.5297e-02,
         3.8475e-02,  6.5387e-03, -8.1514e-03, -2.1557e-03, -2.8578e-03,
         9.8328e-03,  1.3820e-02,  3.0118e-03, -4.9434e-03, -8.0610e-04,
         5.9539e-04, -5.0624e-03, -2.8214e-02,  1.7978e-02,  1.3492e-03,
        -9.6791e-03, -9.1240e-03, -1.5556e-02, -5.2206e-03, -2.6013e-02,
         7.7013e-02,  6.2387e-03,  3.0727e-04, -3.1127e-04, -4.2965e-03,
        -3.7163e-03, -2.9881e-03, -8.7892e-04, -1.6473e-02,  5.3467e-03,
        -9.3061e-03,  7.1365e-03, -4.8840e-03,  3.7713e-03, -1.5017e-04,
        -5.8032e-03, -1.1374e-02, -1.0619e-02, -3.1566e-02,  1.3552e-02,
        -4.0556e-02,  3.5147e-03,  2.7168e-03,  6.0542e-03, -7.7516e-03,
        -4.0932e-03,  1.6645e-03,  5.2681e-03,  1.0743e-03, -1.8570e-02,
        -5.9077e-04,  1.8467e-02,  3.9000e-04,  1.9473e-04, -2.8137e-02,
        -1.1038e-03,  3.5763e-03,  1.4473e-02,  1.9226e-05,  6.3125e-03,
         4.4736e-03,  6.7247e-04,  4.7286e-03,  1.2582e-02, -6.4737e-03,
        -1.9961e-04, -2.3640e-03, -1.6676e-03, -2.5075e-03,  3.5316e-03,
```

```
         8.8651e-03, -8.3263e-03, -8.2018e-04,  3.9853e-05, -1.7919e-02,
        -5.0165e-03,  5.0339e-03,  4.4677e-03, -9.7487e-03, -1.6541e-02,
        -1.8067e-02, -3.7636e-02, -3.2014e-02,  4.0863e-04,  1.1359e-02,
         3.7964e-03, -2.1028e-03, -8.8380e-03, -9.5163e-03,  7.2948e-05,
         4.9562e-03, -1.3923e-02, -1.1166e-02, -3.0793e-02, -1.7141e-02,
        -1.8825e-03,  2.3828e-03,  3.7062e-02,  2.5886e-02, -5.4030e-02,
        -1.9275e-03,  1.1815e-02,  2.0552e-03,  2.1343e-02, -3.1003e-04,
         8.4817e-03,  5.0214e-03, -2.7207e-03,  1.3392e-03, -8.2816e-03,
        -3.1931e-03,  1.1245e-02, -2.4932e-03,  1.8554e-02, -4.8289e-03,
        -4.2118e-02, -2.8605e-03,  4.7305e-03, -1.9095e-02, -2.3154e-02,
         4.4320e-03,  1.7515e-04, -2.2556e-04,  3.9423e-03,  3.4405e-03,
         1.0077e-02, -1.0200e-03, -1.6765e-02,  2.1852e-03,  1.7804e-03,
         7.1261e-04, -6.4142e-02,  1.0397e-02, -1.0614e-02, -2.8956e-04,
         2.6756e-03, -3.5522e-04,  5.8343e-03,  1.3963e-02, -2.5572e-04,
         4.6604e-03,  3.7979e-06,  1.7894e-03,  1.3953e-03, -1.7053e-02,
        -3.8027e-04,  4.2838e-03,  5.3217e-03, -5.9611e-03, -4.8804e-03,
         3.7712e-03, -3.2815e-03, -7.8975e-03,  2.9710e-05, -4.3018e-03,
        -1.2476e-04, -6.0447e-03, -4.4061e-03, -3.5495e-02,  3.0772e-03,
         2.7454e-03, -1.2819e-03, -9.6425e-03,  1.1092e-02,  6.0207e-03,
         8.6081e-03, -3.0837e-02, -1.4766e-03, -3.0017e-02,  3.8974e-02,
         2.4407e-03,  4.1539e-03, -2.2666e-03,  1.3843e-02, -3.3818e-02,
         1.6536e-02,  2.1264e-04,  3.3519e-02,  4.6251e-02,  2.5824e-03,
        -2.8767e-02, -1.7771e-02,  1.0725e-02,  3.7836e-03, -4.3429e-03,
        -4.0016e-03, -9.2459e-03,  4.7179e-02,  1.1430e-03,  2.0531e-02,
        -1.0412e-03,  4.1403e-02, -1.9063e-02,  6.5938e-04,  3.3582e-02,
         1.3198e-02, -2.7047e-04,  7.1560e-03, -1.2902e-02, -1.1915e-02,
         1.2056e-02])
tensor([-1.5021e-02,  4.0998e-03, -1.4537e-03,  3.6635e-03,  1.8098e-03,
        -2.1908e-03, -2.7054e-03,  1.2332e-02, -1.5676e-02,  1.9480e-02,
         1.7324e-03, -4.2490e-03, -3.1916e-03,  3.8177e-02,  1.0677e-02,
        -7.8244e-03,  1.5849e-02, -1.5348e-02,  1.4227e-02,  2.2586e-02,
        -3.6196e-04,  4.7247e-02,  1.7023e-03,  7.3659e-03,  3.4134e-03,
         6.7970e-03, -8.3875e-03,  3.4339e-02, -1.3136e-02, -9.3946e-03,
         7.5143e-03,  1.1030e-02,  2.3482e-03, -1.5034e-02, -1.2625e-02,
        -3.2093e-03, -2.1316e-03, -4.1912e-03,  2.7611e-02, -2.4063e-04,
        -7.1918e-03,  3.0018e-03, -6.9551e-04, -1.4788e-02,  1.0545e-03,
        -1.3040e-02, -2.6555e-03, -2.2608e-02,  1.3522e-02, -1.4882e-03,
        -3.7200e-03, -8.6770e-03, -2.1540e-02,  1.8613e-03,  2.3519e-02,
         2.4644e-02,  5.9526e-03, -1.1961e-02, -7.3124e-03,  8.5619e-03,
         1.5286e-02,  1.3723e-02,  1.5500e-02, -6.8765e-03, -5.4410e-03,
         3.0054e-06, -1.6739e-02, -3.0600e-02,  3.4850e-03,  1.9435e-03,
        -1.6702e-02, -9.6181e-03, -1.1195e-02, -9.1811e-03, -1.4741e-02,
         3.6674e-02,  4.4512e-03, -3.5945e-03, -1.1017e-03, -2.2209e-03,
        -6.5498e-03, -7.8359e-03, -1.1154e-03, -8.3144e-03, -4.9963e-03,
        -1.4430e-02,  7.5632e-03, -6.3565e-03, -5.3324e-03,  2.1836e-03,
        -7.7202e-03, -1.5618e-02, -1.2932e-02, -9.9216e-03,  3.2072e-02,
        -2.3954e-02,  4.0169e-03,  5.2049e-03,  8.4676e-03, -6.7266e-03,
        -5.0896e-03,  6.5485e-03,  9.8857e-03,  1.4611e-03, -1.2883e-02,
```

```
         5.3474e-03,  2.3478e-02, -1.0665e-02,  9.3246e-04, -2.4343e-02,
        -4.1575e-03,  1.4519e-02,  2.3886e-02, -4.2434e-03,  5.5017e-03,
         6.1145e-04,  1.6159e-03,  9.8530e-03,  1.5067e-02, -8.4072e-03,
         2.7029e-03, -4.0405e-03, -4.2418e-03, -3.2271e-03,  5.1176e-03,
         9.6498e-03, -8.8315e-03, -5.9698e-03, -1.1986e-03, -1.7999e-02,
        -1.6616e-02,  5.4989e-03,  2.4684e-03, -8.3620e-04, -1.7706e-02,
        -2.0234e-02, -1.4318e-02, -1.9436e-02, -3.5640e-03,  9.6534e-03,
         3.0764e-03, -3.8106e-03,  1.0116e-03, -2.8349e-03,  2.0599e-03,
         1.5007e-02,  3.9855e-03, -1.2130e-02, -1.9400e-02, -3.0699e-02,
        -2.6122e-03,  7.5331e-03,  2.6673e-03,  3.2849e-02, -2.2347e-02,
        -2.4412e-03,  2.3320e-02,  6.3914e-03,  2.3599e-02,  6.4762e-03,
        -4.7785e-03,  3.1204e-03, -7.9609e-03,  5.1030e-03, -8.0091e-03,
         4.2224e-05,  1.8180e-02, -4.9343e-03,  9.4068e-03, -8.9964e-03,
        -2.0610e-02, -2.6197e-03,  1.2584e-02, -1.9748e-02, -3.8180e-03,
         6.8107e-03, -3.0114e-03, -2.4740e-03,  9.3532e-03,  9.1389e-03,
         1.9194e-02, -4.5823e-04, -1.4916e-02, -1.2337e-03,  3.9918e-03,
         2.7152e-03, -2.1527e-02,  1.2447e-02, -9.6117e-03, -5.8822e-03,
         6.5088e-03,  1.2640e-03,  6.9415e-03,  3.3197e-02, -6.7256e-04,
         5.6309e-03,  2.6194e-03, -3.3962e-03, -4.5249e-03, -8.7522e-03,
        -4.9251e-03,  7.5898e-03,  1.4503e-02, -8.4063e-03, -4.3531e-03,
         6.8185e-03, -3.6327e-03, -9.0079e-03, -3.6405e-03,  1.2854e-03,
        -8.1688e-03, -4.0937e-03,  3.3536e-03, -1.6597e-02, -1.0495e-03,
         9.4291e-03,  2.1232e-03, -1.7120e-02,  1.6985e-02,  1.1571e-02,
         3.5217e-03, -1.1392e-02,  3.9760e-03, -7.3365e-03,  3.7733e-02,
        -3.3825e-03,  4.8753e-03, -7.5015e-03,  1.6035e-02, -2.1226e-02,
         1.6163e-02, -3.1412e-03,  3.0411e-02,  2.5304e-02, -1.3619e-03,
        -2.1825e-02, -1.1453e-02,  1.2185e-03, -2.6839e-03, -1.5943e-02,
        -9.0130e-03, -9.3985e-03,  3.3067e-02,  4.5352e-03,  3.5354e-03,
         1.9988e-03,  2.8815e-02, -8.0114e-03, -5.8911e-03,  3.6037e-02,
         1.6135e-02, -1.2836e-03,  1.1172e-02, -8.9111e-03, -1.4254e-02,
         1.6770e-02]])
tensor([[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,


[[[-1.6340e-02, -2.3892e-02, -2.3903e-02],
  [-1.2718e-02, -2.1620e-02, -2.2919e-02],
  [-3.8508e-03, -1.6867e-03, -2.3717e-03]],

 [[ 5.6947e-03,  1.0001e-03, -2.6142e-03],
  [-3.6130e-04, -3.2339e-03, -1.2979e-03],
  [-2.4383e-03, -5.7051e-03, -6.2218e-03]],

 [[-5.3419e-03, -8.9827e-03, -6.0374e-03],
  [-1.5287e-03, -2.6111e-03,  3.6914e-04],
  [ 6.0913e-03,  7.4689e-03,  9.4072e-03]],

 …,

 [[-2.0335e-03,  2.2574e-03,  3.2833e-06],
  [-1.4243e-02, -1.6109e-02, -1.3604e-02],
  [-1.0019e-02, -8.9757e-03, -7.1960e-03]],

 [[-1.2874e-02, -1.0551e-02, -9.4741e-03],
  [-5.7440e-03, -2.7365e-04, -4.3550e-04],
  [-8.8490e-03, -7.1760e-03, -9.0541e-03]],

 [[-1.6885e-02, -2.5507e-02, -2.4959e-02],
  [-7.5022e-04, -9.3410e-03, -8.4588e-03],
  [ 1.1964e-02,  3.9559e-03, -6.4173e-04]]],


[[[ 7.0906e-03, -3.6538e-03, -8.4660e-03],
  [ 1.0209e-02,  9.4275e-03,  1.1659e-02],
```

```
  [-2.0055e-03,  5.3711e-03,  1.3843e-02]],

 [[ 1.0266e-02,  3.8156e-03,  6.4308e-03],
  [ 1.1001e-02,  5.2258e-03,  8.9776e-03],
  [ 1.6825e-03, -1.0199e-03, -1.5872e-03]],

 [[ 2.3255e-03,  4.9826e-03,  2.2115e-02],
  [ 3.9459e-04,  3.0142e-03,  1.7162e-02],
  [ 3.2037e-03,  4.0722e-03,  9.0104e-03]],

 …,

 [[ 4.4190e-04,  3.6734e-03, -1.6424e-03],
  [ 9.9306e-03,  1.2795e-02,  5.2698e-03],
  [-2.0509e-03, -5.9230e-05, -3.7961e-03]],

 [[-1.1021e-02, -2.5327e-04,  6.0436e-03],
  [-6.3556e-03,  5.1267e-03,  1.3512e-02],
  [ 2.5015e-03,  5.8040e-03,  6.4208e-03]],

 [[-4.2228e-03,  3.6347e-03,  2.0543e-02],
  [ 2.6704e-03,  9.8735e-03,  2.0792e-02],
  [ 1.0524e-02,  1.2809e-02,  1.4449e-02]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([ 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -2.1661e-05,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.4033e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -3.3511e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -2.9967e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -1.0335e-02,  0.0000e+00, -5.5645e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  6.0571e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -5.2450e-03,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  6.1129e-02,  0.0000e+00,
         0.0000e+00, -2.9189e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

```
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -3.5203e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -1.1217e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  2.2049e-02,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  6.5556e-04,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  3.1638e-03,  0.0000e+00, -2.6581e-04,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  9.2569e-05,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -5.5847e-03,  7.1386e-04,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -4.3185e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -8.8440e-03,  9.9041e-03,  0.0000e+00])
tensor([[[[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         …,

         [[ 0.0627,  0.0654,  0.0441],
          [ 0.0702,  0.0600,  0.0562],
          [ 0.0580,  0.0423,  0.0360]],
```

```
 [[ 0.0038,  0.0013, -0.0307],
  [ 0.0612,  0.0120, -0.0372],
  [ 0.1123,  0.0761, -0.0048]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[-0.0097,  0.0035,  0.0224],
  [-0.0079, -0.0174, -0.0144],
  [ 0.0040, -0.0024, -0.0064]],

 [[ 0.0047, -0.0027,  0.0306],
  [-0.0127, -0.0106,  0.0375],
  [ 0.0049, -0.0125,  0.0140]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],
```

```
 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


…,


[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0020, -0.0098, -0.0247],
  [ 0.0149,  0.0020, -0.0074],
  [-0.0140, -0.0126, -0.0129]],

 [[ 0.0705,  0.0909,  0.0519],
  [ 0.0911,  0.0718,  0.0356],
  [ 0.0691,  0.0310,  0.0252]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
```

```
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0038,  0.0004,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
```

```
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]]]])
tensor([ 4.0307e-02,  1.6511e-02,  0.0000e+00, -8.1976e-03, -2.3083e-02,
        -4.7330e-02, -3.1767e-03, -4.8458e-02, -5.2808e-02,  0.0000e+00,
         0.0000e+00, -7.9067e-05,  3.6254e-02,  0.0000e+00, -6.2512e-03,
        -1.7409e-02,  4.9509e-06,  0.0000e+00,  0.0000e+00, -5.6125e-05,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  1.2993e-02,  0.0000e+00,
         8.4426e-03,  0.0000e+00,  3.2290e-03, -1.0567e-05,  1.0759e-03,
         3.7902e-04, -1.4023e-03,  4.2579e-03,  1.1668e-05, -7.6588e-04,
         4.6029e-03, -3.3190e-03, -1.2618e-02,  2.4069e-03, -3.0212e-02,
        -2.8722e-02, -6.3480e-03,  6.5487e-03,  4.4468e-02,  0.0000e+00,
         6.6463e-02, -2.7126e-04,  0.0000e+00,  1.7941e-02,  0.0000e+00,
         0.0000e+00, -9.3290e-03,  2.6594e-04,  4.5445e-03, -1.3300e-02,
         0.0000e+00, -1.5369e-02,  7.2554e-04,  8.2689e-04,  0.0000e+00,
         4.2181e-05,  0.0000e+00,  3.8755e-04,  2.3787e-05, -1.0920e-02,
         6.0939e-02, -4.8642e-02, -1.5112e-02,  3.3886e-03, -1.2456e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -2.9196e-02,  0.0000e+00,  1.4885e-04, -2.1455e-03,
         0.0000e+00,  0.0000e+00, -2.0606e-04,  6.7257e-04,  6.9660e-04,
         0.0000e+00, -1.1293e-02, -2.6761e-02, -6.0465e-03,  0.0000e+00,
         1.3688e-02, -2.4747e-04,  0.0000e+00, -8.9682e-06,  0.0000e+00,
         0.0000e+00,  2.7812e-04, -8.6227e-02,  0.0000e+00, -2.0262e-03,
         5.0135e-02,  7.8116e-03, -5.0259e-02,  2.2800e-05,  0.0000e+00,
         4.0059e-02,  5.9109e-03, -6.1258e-04,  6.0588e-04, -9.3291e-02,
         9.9988e-03,  2.2716e-04,  0.0000e+00, -2.1110e-03,  4.0810e-02,
        -5.8566e-03, -3.9536e-02,  3.6586e-05,  1.6483e-04,  1.3394e-02,
        -1.0823e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  3.9256e-04,
         0.0000e+00,  1.3023e-02,  1.0775e-02, -3.3148e-05, -7.6013e-03,
         0.0000e+00,  0.0000e+00,  2.4194e-04, -4.0899e-02,  5.6275e-03,
        -1.7349e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  8.6376e-05,
         2.9127e-04,  1.4408e-02, -1.2153e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -2.4504e-02,  2.6033e-04,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  5.0790e-03,  0.0000e+00,  0.0000e+00,
         1.2381e-01,  0.0000e+00, -8.1637e-05, -1.0824e-03, -2.4761e-02,
         2.5215e-04,  1.8940e-02,  0.0000e+00, -5.6437e-05, -1.0915e-02,
         0.0000e+00,  0.0000e+00,  4.1694e-06, -5.6979e-04,  8.4793e-04,
        -7.0276e-04,  4.5144e-06, -2.9316e-04,  0.0000e+00,  3.1884e-03,
         9.7887e-02,  0.0000e+00,  1.9978e-03,  1.2596e-02,  1.6311e-03,
         0.0000e+00, -1.7965e-03,  2.7877e-04,  4.6409e-03, -1.5191e-02,
         0.0000e+00, -2.7883e-04,  4.1780e-04, -5.5768e-04, -4.8315e-02,
         0.0000e+00,  3.9545e-04, -1.9691e-04,  8.0509e-02,  2.1878e-02,
         7.2346e-04,  2.9504e-03, -1.3916e-03,  0.0000e+00,  0.0000e+00,
        -3.9713e-02, -1.3368e-01, -2.5821e-04, -9.4936e-03,  4.0894e-03,
         0.0000e+00, -3.7001e-03,  0.0000e+00,  0.0000e+00, -1.7369e-04,
         6.7004e-04,  3.3709e-04, -1.9103e-03,  2.9651e-03, -1.7570e-02,
        -5.3026e-02, -1.5984e-05,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  3.5433e-04,  4.8132e-02,  0.0000e+00,
        -1.2181e-04,  2.7533e-02, -4.2342e-02, -5.7414e-05,  0.0000e+00,
```

```
         6.8210e-03,  0.0000e+00, -3.5155e-04,  6.5742e-03,  1.1391e-03,
         9.3303e-03,  0.0000e+00,  2.4206e-03, -1.1300e-02, -3.6676e-02,
         0.0000e+00,  7.0276e-02,  0.0000e+00,  7.1050e-04, -1.5316e-02,
         0.0000e+00,  9.4328e-03,  0.0000e+00,  0.0000e+00,  1.3409e-02,
         0.0000e+00, -1.7018e-02,  5.7376e-03,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  1.3578e-03, -1.4918e-02,  0.0000e+00,
         0.0000e+00,  2.7625e-02,  1.1820e-01, -1.0313e-03,  0.0000e+00,
        -2.0269e-03, -1.5215e-01,  5.0193e-03,  1.2675e-03,  7.3731e-03,
         0.0000e+00,  0.0000e+00, -1.4249e-05,  0.0000e+00, -3.5639e-02,
        -3.3514e-03, -9.9866e-02,  2.6551e-02,  0.0000e+00, -3.1423e-03,
         2.4768e-04,  3.7289e-04,  0.0000e+00, -1.2840e-03,  7.0422e-03,
         0.0000e+00, -8.4618e-04,  0.0000e+00,  3.9868e-04, -1.6372e-03,
        -5.7068e-04,  0.0000e+00,  1.5299e-04,  1.6781e-03,  0.0000e+00,
         2.6613e-02,  9.3936e-04,  4.4318e-05, -1.5076e-05, -2.8269e-04,
         2.1163e-02, -5.4843e-02, -8.2491e-04, -6.9189e-03,  0.0000e+00,
         1.0848e-02, -6.7574e-04,  0.0000e+00, -4.6214e-03,  0.0000e+00,
         1.1846e-06, -2.7035e-04, -7.5022e-04, -1.4745e-02, -5.1796e-03,
        -4.3829e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,  1.4195e-03,
         0.0000e+00, -4.2253e-04, -5.3180e-04,  1.4746e-02, -2.0725e-04,
        -8.9808e-03,  0.0000e+00,  0.0000e+00, -4.4208e-02,  0.0000e+00,
        -3.8543e-05,  2.6227e-02, -3.4393e-02,  0.0000e+00,  1.7082e-02,
        -6.0014e-04,  0.0000e+00, -3.0698e-03,  9.6545e-02,  4.6599e-02,
        -2.2221e-02,  5.2240e-03,  5.2253e-03,  0.0000e+00,  4.7549e-04,
        -6.4463e-03,  7.8850e-03,  0.0000e+00, -8.2318e-05,  1.3125e-05,
         0.0000e+00,  9.2311e-05, -6.2406e-04,  5.9511e-04,  1.2453e-04,
         7.3092e-04,  4.4643e-03,  0.0000e+00, -5.0671e-03,  0.0000e+00,
         2.3295e-03,  0.0000e+00, -5.3205e-03,  0.0000e+00,  0.0000e+00,
        -7.8637e-02, -6.4051e-03,  0.0000e+00,  0.0000e+00,  3.3358e-02,
         0.0000e+00,  8.3377e-04,  3.8723e-04, -5.8293e-02,  3.0547e-03,
        -4.9989e-05,  7.8317e-03,  4.2143e-03,  0.0000e+00, -2.6938e-03,
        -3.5041e-05,  2.1165e-02,  1.5074e-03,  0.0000e+00])
tensor([[[[ 1.8488e-03,  5.4334e-04,  5.6412e-04],
          [ 2.8681e-03,  2.9724e-03,  2.9214e-03],
          [ 1.6724e-03,  1.0081e-03,  1.4615e-03]],

         [[ 0.0000e+00,  0.0000e+00, -1.7549e-03],
          [ 1.7359e-04,  0.0000e+00, -7.5548e-04],
          [ 1.6510e-04,  3.6818e-04,  1.2724e-03]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[-7.6251e-04, -3.0689e-04,  0.0000e+00],
          [ 0.0000e+00,  2.5907e-05,  5.1597e-05],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([ 1.1662e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  5.8710e-05, -1.9493e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  1.4440e-03,  2.6717e-01,  2.7926e-01,
        -1.9129e-02,  0.0000e+00,  3.7714e-02,  0.0000e+00,  7.8642e-04,
        -3.7959e-02, -2.8890e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -9.8900e-03,  0.0000e+00,  0.0000e+00,  7.7011e-04,  1.9357e-02,
        -2.5337e-01, -3.0243e-02, -4.5538e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  1.2047e-05,  0.0000e+00,  0.0000e+00, -2.5469e-06,
         1.3692e-02,  0.0000e+00,  0.0000e+00, -3.4749e-02,  1.8938e-01,
         0.0000e+00,  1.1194e-02,  0.0000e+00,  0.0000e+00, -8.3051e-04,
         0.0000e+00,  3.5103e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  9.6875e-02, -3.2523e-01,  0.0000e+00,
        -2.6036e-01,  0.0000e+00, -2.6080e-04,  0.0000e+00,  7.4233e-02,
         0.0000e+00, -5.9914e-02,  0.0000e+00, -8.1301e-03,  3.6651e-05,
        -2.1670e-03, -5.9769e-04,  5.7845e-02, -1.3760e-01,  0.0000e+00,
         0.0000e+00, -1.0216e-01,  3.0735e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -2.8808e-02, -2.6213e-05, -5.2265e-05,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  1.7063e-01,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -5.9063e-03,  0.0000e+00, -4.3259e-02,  2.6511e-03,
        -1.5508e-03, -2.3134e-03,  7.0943e-04,  1.3943e-04,  8.5397e-02,
         0.0000e+00, -1.0953e-05, -6.1505e-02, -2.7608e-04,  0.0000e+00,
         2.1814e-02,  0.0000e+00,  2.8734e-02,  0.0000e+00,  2.5177e-03,
         0.0000e+00,  0.0000e+00, -3.2945e-03, -7.1489e-02, -2.2844e-02,
         3.0015e-03,  1.6569e-06,  8.8965e-03, -2.8453e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  8.9942e-04,  0.0000e+00,  1.1481e-05,
         1.1493e-01,  0.0000e+00, -4.0652e-05,  0.0000e+00,  1.3154e-02,
         0.0000e+00,  0.0000e+00,  9.5651e-02,  9.6246e-02,  8.9600e-03,
        -2.0488e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  4.4297e-03,  0.0000e+00,
         1.1550e-01,  0.0000e+00,  0.0000e+00,  1.4408e-04,  0.0000e+00,
         0.0000e+00, -2.9566e-02,  2.6698e-05,  0.0000e+00, -4.2118e-03,
         0.0000e+00,  0.0000e+00, -1.7730e-02,  0.0000e+00,  1.5395e-04,
         2.4608e-02,  0.0000e+00,  0.0000e+00,  3.3056e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -1.0027e-01,  0.0000e+00,  2.2358e-01, -3.8844e-02,
        -7.0241e-03,  4.2726e-05,  0.0000e+00,  2.0548e-03, -5.8676e-03,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  5.2728e-02,  0.0000e+00,
         0.0000e+00, -6.3949e-05,  2.1125e-01,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  2.2110e-02, -3.6003e-02,  0.0000e+00, -1.5456e-01,
         0.0000e+00,  3.2201e-03, -9.4563e-05,  0.0000e+00,  0.0000e+00,
         2.4554e-01,  0.0000e+00,  4.2088e-03,  4.3473e-02,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -1.2114e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -2.2235e-01,  0.0000e+00,
         0.0000e+00, -2.4562e-02,  5.5401e-02,  0.0000e+00,  0.0000e+00,
```

```
        -2.9121e-02,  0.0000e+00,  1.2767e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  3.9869e-03,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -9.8446e-03,  0.0000e+00,  0.0000e+00, -1.1276e-01,  0.0000e+00,
         0.0000e+00,  1.8012e-02, -1.8108e-01,  0.0000e+00,  0.0000e+00,
         0.0000e+00])
tensor([[0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        …,
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.]])
tensor([-1.0226e-05,  0.0000e+00,  2.2294e-03,  …,  0.0000e+00,
         5.4511e-04,  0.0000e+00])
tensor([[ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        …,
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [-1.8367e-07,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00]])
tensor([ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
         0.0000e+00, -1.0263e-05])
tensor([[0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         1.3566e-07],
        [0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         4.0315e-08],
        [0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         1.9619e-08],
        …,
        [0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         6.5055e-12],
        [0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         6.3529e-12],
        [0.0000e+00, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         6.4714e-12]])
tensor([ 2.1356e-02, -3.5724e-02,  2.6507e-02,  2.4462e-02, -8.3932e-03,
         2.0923e-02, -3.9032e-02, -3.6664e-02,  5.3589e-02,  1.5301e-02,
         1.1070e-02,  3.6058e-04, -3.3909e-02,  4.0411e-03,  7.7991e-03,
        -9.6371e-02,  9.7256e-02, -9.0737e-03, -4.7579e-03,  5.9906e-02,
         1.6470e-05,  2.6971e-06,  2.9219e-06,  3.1065e-06,  2.5998e-06,
```

```
3.0636e-06,  3.1488e-06,  2.7878e-06,  3.1846e-06,  3.1774e-06,
3.1551e-06,  2.9552e-06,  3.1528e-06,  3.1188e-06,  2.9652e-06,
2.8930e-06,  3.0904e-06,  3.1773e-06,  2.9924e-06,  3.1761e-06,
3.1028e-06,  3.0187e-06,  3.2180e-06,  3.0530e-06,  3.2180e-06,
3.1663e-06,  3.1162e-06,  3.0820e-06,  3.1254e-06,  2.8478e-06,
3.1747e-06,  3.0897e-06,  3.1642e-06,  2.8541e-06,  3.1564e-06,
3.1550e-06,  3.1078e-06,  3.1798e-06,  2.9794e-06,  3.1281e-06,
3.1768e-06,  3.1469e-06,  3.2132e-06,  3.2168e-06,  3.2034e-06,
3.1591e-06,  3.2195e-06,  3.1560e-06,  2.6644e-06,  3.2214e-06,
3.1104e-06,  3.1822e-06,  2.8953e-06,  3.0761e-06,  3.1377e-06,
2.6430e-06,  3.1854e-06,  3.0634e-06,  3.0752e-06,  3.2107e-06,
3.0654e-06,  3.0736e-06,  3.2184e-06,  3.1710e-06,  3.0965e-06,
3.2119e-06,  3.1462e-06,  3.1855e-06,  3.1972e-06,  2.8632e-06,
3.0389e-06,  3.0910e-06,  3.1853e-06,  2.9055e-06,  3.0031e-06,
3.1792e-06,  3.1696e-06,  3.0722e-06,  3.2235e-06,  3.1612e-06,
3.1471e-06,  2.9452e-06,  3.1845e-06,  3.0878e-06,  3.0981e-06,
3.2077e-06,  3.1100e-06,  3.0744e-06,  2.9656e-06,  3.2213e-06,
3.1383e-06,  3.1927e-06,  3.1376e-06,  2.6045e-06,  3.1318e-06,
2.9932e-06,  3.1431e-06,  3.0567e-06,  3.2144e-06,  2.9188e-06,
2.6810e-06,  3.0368e-06,  2.6476e-06,  3.2172e-06,  3.0875e-06,
2.9876e-06,  3.1639e-06,  3.0872e-06,  3.1591e-06,  3.0936e-06,
2.9390e-06,  3.2247e-06,  2.9626e-06,  3.1656e-06,  3.1194e-06,
2.8817e-06,  3.1177e-06,  3.0288e-06,  2.7047e-06,  3.0079e-06,
3.1901e-06,  3.2226e-06,  3.1471e-06,  2.9569e-06,  2.8246e-06,
3.1576e-06,  2.8795e-06,  3.1894e-06,  3.0522e-06,  3.1910e-06,
3.1834e-06,  3.1769e-06,  3.1502e-06,  3.2231e-06,  2.8835e-06,
2.9566e-06,  2.9116e-06,  3.1255e-06,  3.2079e-06,  3.0882e-06,
3.1756e-06,  2.7528e-06,  3.0469e-06,  3.1067e-06,  3.2067e-06,
2.6909e-06,  3.1022e-06,  2.9953e-06,  3.2110e-06,  3.1645e-06,
3.1879e-06,  3.1785e-06,  3.1914e-06,  3.2251e-06,  3.1212e-06,
3.0963e-06,  3.1180e-06,  3.0648e-06,  3.1808e-06,  3.1866e-06,
3.1757e-06,  2.9076e-06,  3.0082e-06,  3.1457e-06,  2.7557e-06,
3.2002e-06,  2.9924e-06,  3.2190e-06,  3.1686e-06,  3.1932e-06,
2.9563e-06,  3.1166e-06,  3.1679e-06,  3.1371e-06,  3.0082e-06,
2.9011e-06,  2.9516e-06,  3.1513e-06,  3.0641e-06,  2.9845e-06,
2.9443e-06,  3.1993e-06,  2.8731e-06,  3.1984e-06,  3.2044e-06,
3.1839e-06,  3.1246e-06,  3.1727e-06,  3.0077e-06,  3.1632e-06,
2.9368e-06,  3.1738e-06,  2.7982e-06,  3.0805e-06,  3.2186e-06,
3.0485e-06,  3.1828e-06,  3.1955e-06,  3.1483e-06,  3.2240e-06,
3.2144e-06,  3.0220e-06,  3.0182e-06,  2.9566e-06,  2.6807e-06,
3.1884e-06,  2.9378e-06,  3.0886e-06,  2.7765e-06,  2.9895e-06,
3.1215e-06,  3.2044e-06,  3.0964e-06,  3.1737e-06,  3.1258e-06,
3.1687e-06,  3.0888e-06,  3.1869e-06,  3.0303e-06,  3.1300e-06,
2.9795e-06,  2.9802e-06,  3.1187e-06,  3.1158e-06,  3.1157e-06,
3.1346e-06,  3.0308e-06,  3.0814e-06,  3.0689e-06,  3.1622e-06,
3.1813e-06,  3.1643e-06,  2.7399e-06,  2.7601e-06,  2.9047e-06,
3.1666e-06,  3.0411e-06,  3.0846e-06,  3.0953e-06,  3.0674e-06,
2.8832e-06,  3.0878e-06,  2.8881e-06,  3.1290e-06,  3.1199e-06,
```

```
3.2075e-06,  3.1368e-06,  3.1212e-06,  3.0319e-06,  3.1575e-06,
3.0683e-06,  2.9295e-06,  3.1409e-06,  3.0388e-06,  2.8643e-06,
3.1992e-06,  3.1076e-06,  2.7266e-06,  3.0035e-06,  3.1420e-06,
3.0649e-06,  3.1480e-06,  3.0710e-06,  3.2035e-06,  3.1679e-06,
2.4432e-06,  2.9899e-06,  2.7842e-06,  2.9853e-06,  3.1726e-06,
2.4976e-06,  3.1999e-06,  3.0239e-06,  3.0470e-06,  3.1834e-06,
3.0619e-06,  3.1686e-06,  3.0332e-06,  3.1035e-06,  3.2050e-06,
3.1205e-06,  3.1822e-06,  3.2197e-06,  3.1944e-06,  3.2193e-06,
3.2062e-06,  3.1553e-06,  2.9124e-06,  3.1625e-06,  2.9213e-06,
2.9742e-06,  3.1999e-06,  3.2145e-06,  2.9810e-06,  3.2012e-06,
3.1052e-06,  3.1235e-06,  2.9171e-06,  3.0396e-06,  3.1061e-06,
3.1029e-06,  3.1351e-06,  2.7159e-06,  2.9749e-06,  3.1376e-06,
3.1352e-06,  3.1830e-06,  2.8433e-06,  3.1293e-06,  3.1934e-06,
3.2096e-06,  3.1987e-06,  3.2216e-06,  3.1419e-06,  2.8098e-06,
3.1474e-06,  3.2074e-06,  3.0413e-06,  3.1181e-06,  3.2070e-06,
3.0264e-06,  2.7818e-06,  3.1805e-06,  3.1978e-06,  2.5242e-06,
2.9110e-06,  3.1835e-06,  3.0045e-06,  3.1657e-06,  3.0321e-06,
3.2226e-06,  3.1354e-06,  3.2155e-06,  3.1435e-06,  3.1507e-06,
3.2067e-06,  3.2012e-06,  3.1891e-06,  2.9094e-06,  3.0526e-06,
3.1178e-06,  3.0840e-06,  2.5617e-06,  2.8977e-06,  3.1230e-06,
3.1227e-06,  2.7025e-06,  3.2210e-06,  3.0484e-06,  2.9897e-06,
3.2102e-06,  3.1004e-06,  3.1913e-06,  3.2212e-06,  3.1864e-06,
3.2227e-06,  3.1511e-06,  2.4489e-06,  3.1096e-06,  2.8849e-06,
3.1488e-06,  2.9480e-06,  3.1935e-06,  3.1999e-06,  2.9948e-06,
3.1271e-06,  3.0745e-06,  3.1910e-06,  3.1661e-06,  2.8240e-06,
3.0279e-06,  3.1698e-06,  3.1341e-06,  3.1416e-06,  3.1154e-06,
3.0342e-06,  3.0933e-06,  2.9980e-06,  3.0645e-06,  3.0637e-06,
2.9426e-06,  3.1724e-06,  3.1837e-06,  2.8782e-06,  2.8871e-06,
3.1487e-06,  3.0888e-06,  3.2206e-06,  3.2182e-06,  2.9493e-06,
3.1706e-06,  3.1960e-06,  3.1399e-06,  3.1814e-06,  3.0771e-06,
3.1331e-06,  2.9777e-06,  3.0823e-06,  3.1556e-06,  3.1678e-06,
3.2112e-06,  3.1091e-06,  3.1569e-06,  3.0167e-06,  3.1041e-06,
2.9869e-06,  2.9539e-06,  2.6847e-06,  3.0699e-06,  3.1194e-06,
2.9528e-06,  3.0117e-06,  2.8928e-06,  2.9518e-06,  2.9927e-06,
3.0540e-06,  3.1901e-06,  3.2059e-06,  3.1070e-06,  3.0278e-06,
3.2006e-06,  3.1927e-06,  3.0601e-06,  3.1858e-06,  3.0627e-06,
3.1228e-06,  3.0855e-06,  2.7704e-06,  3.1842e-06,  3.1578e-06,
3.1057e-06,  3.1130e-06,  2.7772e-06,  3.0920e-06,  3.0311e-06,
3.1357e-06,  3.0723e-06,  3.0766e-06,  3.2213e-06,  3.2054e-06,
2.8596e-06,  3.0265e-06,  3.0886e-06,  2.7432e-06,  3.2216e-06,
3.1323e-06,  2.7693e-06,  3.1834e-06,  3.0270e-06,  3.1430e-06,
3.1826e-06,  3.1195e-06,  2.9087e-06,  3.2063e-06,  3.1867e-06,
3.0535e-06,  3.1580e-06,  3.1206e-06,  3.0318e-06,  3.0387e-06,
3.0855e-06,  3.1242e-06,  3.1183e-06,  3.1073e-06,  2.7814e-06,
2.9136e-06,  3.1973e-06,  3.0500e-06,  2.8790e-06,  3.1636e-06,
2.3677e-06,  2.9274e-06,  2.8403e-06,  3.0376e-06,  3.1076e-06,
2.8616e-06,  3.2104e-06,  3.1620e-06,  3.1941e-06,  3.0794e-06,
3.1207e-06,  3.1614e-06,  3.0434e-06,  3.0677e-06,  3.0583e-06,
```

2.7379e-06, 3.1452e-06, 3.1222e-06, 3.0810e-06, 3.0836e-06,
3.0901e-06, 3.0525e-06, 2.7144e-06, 3.1680e-06, 2.9255e-06,
2.9281e-06, 3.0928e-06, 3.0885e-06, 3.0794e-06, 3.1753e-06,
3.1767e-06, 3.1961e-06, 2.8182e-06, 2.8883e-06, 3.1760e-06,
3.1663e-06, 3.0499e-06, 2.5310e-06, 2.9758e-06, 3.1346e-06,
3.1109e-06, 3.0176e-06, 2.6841e-06, 3.1474e-06, 3.2215e-06,
2.9799e-06, 2.9032e-06, 3.0377e-06, 3.1027e-06, 2.8467e-06,
3.1537e-06, 2.6995e-06, 3.1937e-06, 3.0333e-06, 3.1407e-06,
3.1470e-06, 2.9947e-06, 3.1207e-06, 2.9819e-06, 3.1207e-06,
3.0934e-06, 3.1106e-06, 2.9428e-06, 3.1778e-06, 2.6431e-06,
3.1607e-06, 2.9087e-06, 3.0968e-06, 3.1753e-06, 3.1478e-06,
3.1877e-06, 3.1760e-06, 2.9510e-06, 2.9027e-06, 3.0782e-06,
2.6889e-06, 3.1522e-06, 2.9678e-06, 3.1097e-06, 3.0365e-06,
3.0420e-06, 3.0461e-06, 2.9325e-06, 2.9972e-06, 3.1454e-06,
3.2111e-06, 3.2244e-06, 3.0565e-06, 2.7055e-06, 3.0789e-06,
2.8884e-06, 3.1714e-06, 3.2236e-06, 3.1200e-06, 3.0481e-06,
2.8162e-06, 2.6856e-06, 3.0478e-06, 3.1168e-06, 2.9346e-06,
3.1011e-06, 2.8376e-06, 3.1997e-06, 3.1877e-06, 3.0760e-06,
3.0270e-06, 3.2240e-06, 3.0784e-06, 2.9053e-06, 3.0602e-06,
3.1851e-06, 3.0574e-06, 3.0454e-06, 3.2211e-06, 2.8654e-06,
3.2204e-06, 2.9757e-06, 3.1688e-06, 3.0845e-06, 3.1953e-06,
3.1313e-06, 2.7932e-06, 2.9118e-06, 3.1271e-06, 3.1328e-06,
3.0817e-06, 3.0002e-06, 3.1688e-06, 3.1936e-06, 3.1016e-06,
3.1344e-06, 3.0779e-06, 3.1169e-06, 3.1084e-06, 3.1193e-06,
2.8665e-06, 3.0902e-06, 3.0769e-06, 3.0482e-06, 2.9544e-06,
3.0329e-06, 3.1755e-06, 3.1305e-06, 2.7354e-06, 3.1413e-06,
3.1199e-06, 3.1465e-06, 2.8807e-06, 3.0885e-06, 3.0181e-06,
3.1861e-06, 3.0182e-06, 2.9276e-06, 2.9967e-06, 3.1513e-06,
3.0624e-06, 3.0338e-06, 3.1806e-06, 2.6715e-06, 3.1621e-06,
3.2009e-06, 3.2241e-06, 3.0103e-06, 3.1843e-06, 3.0423e-06,
3.2199e-06, 3.0999e-06, 3.0813e-06, 3.0001e-06, 3.1816e-06,
3.1820e-06, 3.0709e-06, 2.9382e-06, 3.0231e-06, 3.0517e-06,
3.1799e-06, 3.1138e-06, 3.1571e-06, 3.2163e-06, 3.1143e-06,
3.0936e-06, 3.2016e-06, 3.1166e-06, 2.9348e-06, 2.9459e-06,
3.1780e-06, 2.9315e-06, 3.2097e-06, 3.1831e-06, 2.9582e-06,
3.0448e-06, 3.1172e-06, 3.1212e-06, 3.1141e-06, 3.0725e-06,
2.9969e-06, 3.1043e-06, 3.1300e-06, 3.0266e-06, 3.1307e-06,
3.1959e-06, 2.9030e-06, 3.2086e-06, 2.8768e-06, 3.0384e-06,
2.8839e-06, 3.1773e-06, 3.0499e-06, 3.2240e-06, 3.1150e-06,
2.8101e-06, 3.1486e-06, 2.8272e-06, 2.8803e-06, 2.8979e-06,
3.2215e-06, 3.1295e-06, 3.1884e-06, 2.9907e-06, 3.1256e-06,
3.1982e-06, 3.1810e-06, 3.1583e-06, 3.1538e-06, 2.6732e-06,
3.0275e-06, 3.0127e-06, 3.1997e-06, 3.1001e-06, 3.1869e-06,
3.1757e-06, 2.9526e-06, 2.9505e-06, 2.9219e-06, 3.1345e-06,
2.9618e-06, 3.1010e-06, 3.0653e-06, 3.0820e-06, 3.0859e-06,
2.9798e-06, 3.1737e-06, 3.0270e-06, 2.5017e-06, 2.8585e-06,
3.1299e-06, 3.1259e-06, 3.2002e-06, 3.1271e-06, 3.2057e-06,
3.2050e-06, 3.0311e-06, 3.0900e-06, 3.1478e-06, 3.1260e-06,

```
3.1848e-06,  3.1997e-06,  2.8844e-06,  2.9745e-06,  2.9037e-06,
3.1616e-06,  3.2211e-06,  3.1474e-06,  2.7379e-06,  3.1863e-06,
2.7639e-06,  3.0675e-06,  3.1928e-06,  3.1569e-06,  3.2177e-06,
3.1691e-06,  3.1182e-06,  3.1647e-06,  3.0795e-06,  3.1760e-06,
3.1243e-06,  2.7921e-06,  3.1380e-06,  3.1881e-06,  3.1384e-06,
3.0861e-06,  3.1812e-06,  3.2101e-06,  2.7897e-06,  2.4775e-06,
3.1362e-06,  3.1076e-06,  2.9816e-06,  3.2264e-06,  3.1805e-06,
3.1215e-06,  2.9616e-06,  3.0802e-06,  3.1824e-06,  3.0871e-06,
3.2063e-06,  3.2161e-06,  3.0619e-06,  3.1367e-06,  2.7922e-06,
2.7503e-06,  3.2024e-06,  3.0357e-06,  2.6715e-06,  3.1420e-06,
2.7925e-06,  2.8693e-06,  3.1319e-06,  3.0791e-06,  2.9928e-06,
2.7578e-06,  3.1409e-06,  3.2149e-06,  3.0029e-06,  3.1157e-06,
3.0279e-06,  3.0897e-06,  2.8864e-06,  3.0761e-06,  3.0017e-06,
3.2026e-06,  3.0514e-06,  3.0760e-06,  3.2118e-06,  3.2112e-06,
2.9196e-06,  3.0434e-06,  2.5359e-06,  3.0923e-06,  3.0053e-06,
3.1968e-06,  2.9933e-06,  2.9190e-06,  2.8751e-06,  3.0991e-06,
3.1183e-06,  3.1384e-06,  2.9964e-06,  3.2238e-06,  3.0243e-06,
3.0059e-06,  2.7318e-06,  3.2014e-06,  3.2049e-06,  3.1434e-06,
3.0650e-06,  3.1705e-06,  3.0052e-06,  3.1021e-06,  3.1959e-06,
2.9334e-06,  2.7345e-06,  3.0459e-06,  3.1193e-06,  3.2204e-06,
3.1796e-06,  3.1921e-06,  3.1127e-06,  3.1556e-06,  3.0570e-06,
3.1896e-06,  3.0757e-06,  2.9861e-06,  2.5388e-06,  2.5848e-06,
2.9988e-06,  3.2117e-06,  3.0464e-06,  3.2144e-06,  3.1313e-06,
3.2183e-06,  3.1986e-06,  2.8689e-06,  3.2173e-06,  3.0682e-06,
3.1078e-06,  2.6765e-06,  3.1927e-06,  3.1956e-06,  2.6994e-06,
2.8528e-06,  3.1748e-06,  3.1177e-06,  2.9101e-06,  3.1816e-06,
3.1079e-06,  3.0959e-06,  3.1126e-06,  3.0702e-06,  3.2020e-06,
3.1173e-06,  3.0498e-06,  3.1135e-06,  3.1439e-06,  2.8196e-06,
3.0358e-06,  3.1410e-06,  3.2219e-06,  3.0970e-06,  3.2113e-06,
3.2234e-06,  3.0901e-06,  3.0180e-06,  2.9868e-06,  2.8786e-06,
3.0417e-06,  3.0811e-06,  3.1836e-06,  3.1655e-06,  3.1554e-06,
3.1327e-06,  3.1666e-06,  2.9012e-06,  3.1847e-06,  3.2188e-06,
3.1538e-06,  3.0839e-06,  3.1391e-06,  3.1851e-06,  3.1821e-06,
3.1882e-06,  3.1523e-06,  3.2154e-06,  2.8890e-06,  2.9056e-06,
3.2168e-06,  3.2044e-06,  3.1984e-06,  3.1839e-06,  3.0393e-06,
3.1508e-06,  3.2182e-06,  3.1148e-06,  2.9806e-06,  3.0867e-06,
2.7951e-06,  3.1769e-06,  2.7329e-06,  3.2113e-06,  3.2054e-06,
3.2003e-06,  3.1836e-06,  3.1953e-06,  3.1651e-06,  3.1965e-06,
3.1885e-06,  3.0745e-06,  2.5663e-06,  3.0437e-06,  3.1004e-06,
3.1811e-06,  3.2196e-06,  3.1215e-06,  3.1102e-06,  3.0612e-06,
3.0949e-06,  2.9766e-06,  2.9612e-06,  2.5628e-06,  2.7950e-06,
3.2198e-06,  3.1685e-06,  3.1770e-06,  3.1873e-06,  3.2213e-06,
3.1494e-06,  3.0633e-06,  3.1173e-06,  3.1773e-06,  3.1203e-06,
3.1175e-06,  3.1830e-06,  3.0516e-06,  3.2006e-06,  3.1862e-06,
3.1800e-06,  3.2201e-06,  2.9793e-06,  2.9408e-06,  3.0758e-06,
2.9147e-06,  3.0781e-06,  3.0896e-06,  3.1147e-06,  3.0811e-06,
3.1620e-06,  3.1513e-06,  3.2047e-06,  3.2186e-06,  3.0611e-06,
3.0140e-06,  3.1066e-06,  3.0049e-06,  2.8822e-06,  3.2057e-06,
```

```
         2.9094e-06,  2.8566e-06,  3.2246e-06,  3.0082e-06,  3.1293e-06,
         3.1974e-06,  3.0123e-06,  3.0870e-06,  3.1110e-06,  2.7726e-06,
         2.8567e-06,  3.1832e-06,  3.1816e-06,  3.1192e-06,  3.2016e-06])
end of p.grad

False
Epoch 7 finished
Epoch [7/10], Loss: 1.4033
************************************************************

Learning rate: 0.001

another way to print learning rate:
tensor([[[[ 1.1616e-02,  1.1091e-02,  1.0421e-02,  …,  8.3738e-03,
            7.8936e-03,  7.1475e-03],
          [ 1.0982e-02,  1.0589e-02,  1.0300e-02,  …,  7.9400e-03,
            7.1284e-03,  5.9371e-03],
          [ 1.0667e-02,  1.0479e-02,  1.0290e-02,  …,  7.7031e-03,
            6.9928e-03,  4.9421e-03],
          …,
          [ 7.9361e-03,  7.8475e-03,  7.3203e-03,  …,  3.0037e-03,
            3.9523e-03,  2.6730e-03],
          [ 8.2245e-03,  8.3890e-03,  7.9214e-03,  …,  3.6909e-03,
            4.1552e-03,  3.4554e-03],
          [ 8.6429e-03,  8.1928e-03,  7.5204e-03,  …,  3.9868e-03,
            3.0288e-03,  2.6554e-03]],

         [[ 1.3823e-02,  1.3303e-02,  1.2521e-02,  …,  1.0572e-02,
            1.0278e-02,  9.4789e-03],
          [ 1.3135e-02,  1.2689e-02,  1.2218e-02,  …,  1.0025e-02,
            9.4138e-03,  8.2135e-03],
          [ 1.2911e-02,  1.2570e-02,  1.2148e-02,  …,  9.9281e-03,
            9.2687e-03,  7.1126e-03],
          …,
          [ 1.0549e-02,  1.0610e-02,  1.0098e-02,  …,  6.0183e-03,
            7.0206e-03,  5.5156e-03],
          [ 1.0962e-02,  1.1150e-02,  1.0775e-02,  …,  7.0453e-03,
            7.5536e-03,  6.4849e-03],
          [ 1.1193e-02,  1.0920e-02,  1.0380e-02,  …,  7.1810e-03,
            6.3436e-03,  5.7173e-03]],

         [[ 1.5062e-02,  1.4310e-02,  1.3662e-02,  …,  1.1932e-02,
            1.1785e-02,  1.1205e-02],
          [ 1.4340e-02,  1.3777e-02,  1.3356e-02,  …,  1.1643e-02,
            1.1381e-02,  1.0273e-02],
          [ 1.4189e-02,  1.3958e-02,  1.3587e-02,  …,  1.2015e-02,
            1.1498e-02,  9.4172e-03],
          …,
```

```
  [ 1.2075e-02,  1.1732e-02,  1.1303e-02,  …,  8.4232e-03,
    9.4452e-03,  8.2821e-03],
  [ 1.2215e-02,  1.2423e-02,  1.2187e-02,  …,  9.3140e-03,
    1.0050e-02,  9.2912e-03],
  [ 1.2594e-02,  1.2261e-02,  1.2089e-02,  …,  9.5330e-03,
    9.1609e-03,  8.8549e-03]]],


[[[ 1.1065e-02,  1.0998e-02,  1.1285e-02,  …,  1.8356e-02,
    1.6261e-02,  1.5570e-02],
  [ 1.0920e-02,  1.1215e-02,  1.0387e-02,  …,  1.5751e-02,
    1.6013e-02,  1.5399e-02],
  [ 1.1077e-02,  1.0417e-02,  8.8589e-03,  …,  1.3320e-02,
    1.3818e-02,  1.3213e-02],
  …,
  [ 1.1410e-02,  1.0500e-02,  9.4762e-03,  …,  1.1969e-02,
    1.2775e-02,  1.3797e-02],
  [ 1.0986e-02,  8.8977e-03,  7.5750e-03,  …,  1.2056e-02,
    1.2702e-02,  1.5319e-02],
  [ 1.2385e-02,  1.1100e-02,  1.0915e-02,  …,  1.3450e-02,
    1.3757e-02,  1.4641e-02]],

 [[ 6.2063e-03,  6.4971e-03,  7.1770e-03,  …,  1.4078e-02,
    1.1970e-02,  1.1497e-02],
  [ 7.1069e-03,  7.7199e-03,  6.5684e-03,  …,  1.1489e-02,
    1.1748e-02,  1.1187e-02],
  [ 7.9930e-03,  7.4327e-03,  5.4815e-03,  …,  9.2539e-03,
    9.9598e-03,  9.4483e-03],
  …,
  [ 7.4277e-03,  6.4455e-03,  5.3067e-03,  …,  8.5108e-03,
    9.0193e-03,  9.6141e-03],
  [ 6.7800e-03,  5.2775e-03,  3.9994e-03,  …,  8.6391e-03,
    9.4728e-03,  1.1826e-02],
  [ 8.1866e-03,  7.4547e-03,  7.1436e-03,  …,  1.0312e-02,
    1.0599e-02,  1.1585e-02]],

 [[ 1.8648e-03,  2.3690e-03,  3.0744e-03,  …,  6.5526e-03,
    4.5028e-03,  3.8021e-03],
  [ 3.0268e-03,  3.8575e-03,  2.9827e-03,  …,  4.2159e-03,
    4.1794e-03,  3.5323e-03],
  [ 4.8648e-03,  4.3089e-03,  2.1401e-03,  …,  1.8870e-03,
    2.2943e-03,  1.6382e-03],
  …,
  [ 4.5781e-03,  3.6053e-03,  2.2682e-03,  …,  1.0777e-03,
    8.6662e-04,  1.7673e-03],
  [ 4.3167e-03,  3.3663e-03,  1.8672e-03,  …,  1.2024e-03,
    1.6275e-03,  4.4394e-03],
  [ 5.7434e-03,  5.2879e-03,  5.0919e-03,  …,  3.1610e-03,
```

```
    3.1682e-03,  4.3532e-03]]],


[[[ 2.2210e-03,  2.4700e-03,  1.8394e-03,  …,  2.1276e-03,
    1.0452e-03, -1.2042e-04],
  [ 1.7508e-03,  1.9372e-03,  1.7801e-03,  …,  2.0327e-03,
    7.5065e-04, -2.1203e-04],
  [ 1.1114e-03,  1.3011e-03,  7.3277e-04,  …,  1.2484e-03,
    7.2119e-04, -1.4587e-04],
  …,
  [ 8.9857e-04,  1.2802e-03,  8.1518e-04,  …, -3.4090e-05,
   -2.4599e-03, -3.4426e-03],
  [-7.7476e-04, -2.3972e-04, -5.2511e-05,  …, -2.8083e-03,
   -3.6484e-03, -3.8284e-03],
  [-1.2091e-03, -1.5866e-03, -1.7524e-03,  …, -3.3448e-03,
   -3.5579e-03, -4.5241e-03]],

 [[ 1.9296e-03,  2.3943e-03,  1.7340e-03,  …,  1.6599e-03,
    6.1629e-04, -3.6181e-04],
  [ 1.5090e-03,  1.7507e-03,  1.7264e-03,  …,  1.8282e-03,
    5.5791e-04, -3.4260e-04],
  [ 9.8540e-04,  1.3028e-03,  7.4338e-04,  …,  9.8937e-04,
    6.1606e-04, -2.4200e-04],
  …,
  [ 6.0568e-04,  1.3096e-03,  8.6456e-04,  …, -1.0773e-03,
   -3.4032e-03, -4.2144e-03],
  [-1.1571e-03, -2.8706e-04, -2.3320e-05,  …, -4.1824e-03,
   -4.6706e-03, -4.8686e-03],
  [-1.2281e-03, -1.3758e-03, -1.4787e-03,  …, -4.2668e-03,
   -4.2062e-03, -5.2032e-03]],

 [[ 4.5755e-04,  3.7824e-04, -4.6900e-04,  …, -7.2136e-04,
   -1.7840e-03, -2.6839e-03],
  [-1.5880e-04, -2.8595e-04, -4.2345e-04,  …, -5.8359e-04,
   -1.5495e-03, -2.4123e-03],
  [-5.7379e-04, -4.4499e-04, -1.1023e-03,  …, -1.2463e-03,
   -1.6128e-03, -2.0957e-03],
  …,
  [-1.5584e-03, -1.0452e-03, -1.4638e-03,  …, -3.2433e-03,
   -5.3836e-03, -6.2893e-03],
  [-3.0956e-03, -2.5704e-03, -2.3051e-03,  …, -6.6912e-03,
   -6.8706e-03, -6.8507e-03],
  [-2.9390e-03, -3.3388e-03, -3.5590e-03,  …, -6.8876e-03,
   -6.8396e-03, -7.5644e-03]]],


…,
```

```
[[[-9.0088e-03, -7.6569e-03, -7.2471e-03,  …,  9.3270e-04,
    1.3595e-03,  1.1283e-03],
  [-8.6762e-03, -7.0607e-03, -6.6533e-03,  …,  5.4815e-04,
    6.3548e-04, -5.1511e-04],
  [-4.3300e-03, -3.4092e-03, -1.6854e-03,  …,  1.3332e-03,
    4.9552e-04, -8.5692e-04],
  …,
  [ 3.3683e-03,  2.3153e-03,  1.2002e-03,  …,  1.5008e-03,
    1.9415e-03,  2.4241e-03],
  [ 3.5842e-03,  2.6134e-03,  1.8861e-03,  …,  1.4926e-03,
    8.1762e-04,  1.1620e-03],
  [ 2.3979e-03,  2.5455e-03,  2.6325e-03,  …,  2.3096e-03,
    1.1117e-03,  3.6902e-04]],

 [[-8.7319e-03, -7.3720e-03, -7.1267e-03,  …,  1.4362e-03,
    1.6101e-03,  1.1486e-03],
  [-8.0062e-03, -6.5078e-03, -6.1913e-03,  …,  1.3229e-03,
    1.0942e-03, -4.1486e-04],
  [-3.5978e-03, -3.0360e-03, -1.3418e-03,  …,  1.6446e-03,
    5.5535e-04, -1.1340e-03],
  …,
  [ 4.0734e-03,  3.0770e-03,  1.7884e-03,  …,  1.2930e-03,
    1.2773e-03,  1.7086e-03],
  [ 3.6050e-03,  2.4563e-03,  1.5132e-03,  …,  1.1433e-03,
    1.9482e-04,  2.9950e-04],
  [ 2.0712e-03,  2.0645e-03,  1.7428e-03,  …,  1.8936e-03,
    7.1394e-04, -2.8605e-04]],

 [[-5.3483e-03, -3.6608e-03, -3.2401e-03,  …,  3.8418e-03,
    3.9673e-03,  3.2110e-03],
  [-4.2548e-03, -2.7137e-03, -2.7353e-03,  …,  3.7583e-03,
    3.3220e-03,  1.7094e-03],
  [-2.1180e-04, -1.1486e-04,  1.0034e-03,  …,  3.4177e-03,
    2.3819e-03,  7.0566e-04],
  …,
  [ 5.7502e-03,  4.8916e-03,  3.7380e-03,  …,  2.5516e-03,
    2.2235e-03,  1.9125e-03],
  [ 4.8385e-03,  3.9441e-03,  2.8452e-03,  …,  2.2399e-03,
    1.2235e-03,  7.0513e-04],
  [ 3.0866e-03,  3.3330e-03,  2.6099e-03,  …,  2.6378e-03,
    1.5415e-03,  8.9380e-05]]],


[[[-1.1419e-02, -9.3132e-03, -9.7640e-03,  …, -6.9773e-03,
   -6.9942e-03, -7.8152e-03],
  [-1.1889e-02, -1.0137e-02, -9.7139e-03,  …, -6.5190e-03,
   -5.5019e-03, -6.3541e-03],
```

```
  [-1.1681e-02, -1.0673e-02, -9.7501e-03,  …, -6.4192e-03,
   -6.3710e-03, -5.8182e-03],
  …,
  [-8.7687e-03, -9.1856e-03, -9.1470e-03,  …, -7.8030e-03,
   -8.7703e-03, -6.1228e-03],
  [-7.7116e-03, -8.5105e-03, -6.6407e-03,  …, -7.2029e-03,
   -7.9452e-03, -6.5055e-03],
  [-6.9240e-03, -7.1247e-03, -4.7770e-03,  …, -6.9351e-03,
   -5.7654e-03, -7.2686e-03]],

 [[-1.5049e-02, -1.3453e-02, -1.4504e-02,  …, -9.7649e-03,
   -9.3493e-03, -9.8768e-03],
  [-1.5649e-02, -1.4542e-02, -1.4570e-02,  …, -9.9730e-03,
   -8.3942e-03, -8.6586e-03],
  [-1.4956e-02, -1.4838e-02, -1.4157e-02,  …, -1.0384e-02,
   -9.6319e-03, -8.2581e-03],
  …,
  [-7.2704e-03, -7.7721e-03, -8.1610e-03,  …, -1.2010e-02,
   -1.3237e-02, -1.0165e-02],
  [-7.2691e-03, -8.0099e-03, -6.7035e-03,  …, -1.0950e-02,
   -1.1710e-02, -9.7712e-03],
  [-7.3638e-03, -7.3728e-03, -5.7782e-03,  …, -9.6651e-03,
   -8.4999e-03, -9.4206e-03]],

 [[-1.3288e-02, -1.1838e-02, -1.2931e-02,  …, -6.3178e-03,
   -5.3234e-03, -5.1214e-03],
  [-1.2997e-02, -1.2050e-02, -1.2686e-02,  …, -6.7001e-03,
   -4.6937e-03, -4.4559e-03],
  [-1.1432e-02, -1.1460e-02, -1.1452e-02,  …, -6.6646e-03,
   -5.6398e-03, -4.1324e-03],
  …,
  [-7.2097e-04, -1.1548e-03, -1.9542e-03,  …, -8.5659e-03,
   -1.0178e-02, -7.3830e-03],
  [-1.4636e-03, -2.0579e-03, -1.2460e-03,  …, -7.0964e-03,
   -8.0694e-03, -6.7985e-03],
  [-1.3251e-03, -1.2199e-03, -2.5166e-04,  …, -5.3541e-03,
   -4.6768e-03, -6.3938e-03]]],


[[[-8.4357e-03, -6.2008e-03, -5.3524e-03,  …, -6.0800e-03,
   -5.0559e-03, -4.6638e-03],
  [-7.0597e-03, -5.4656e-03, -3.1429e-03,  …, -3.2514e-03,
   -3.2598e-04,  7.5306e-04],
  [-6.7992e-03, -5.9121e-03, -4.3149e-03,  …,  2.1193e-03,
    1.0410e-03,  3.7161e-04],
  …,
  [ 3.8250e-03,  5.9898e-03,  5.1123e-03,  …,  1.3755e-03,
    7.5073e-05, -1.5698e-03],
```

```
          [-3.6402e-04,  4.3649e-04,  2.9724e-03,  …,  3.2420e-03,
           -3.5216e-04, -3.1524e-03],
          [-2.0100e-03, -1.4914e-03,  1.1221e-03,  …,  4.0872e-03,
            1.0445e-04, -3.7204e-03]],

         [[-7.2951e-03, -5.1234e-03, -4.2836e-03,  …, -6.4789e-03,
           -5.3358e-03, -4.9526e-03],
          [-4.8015e-03, -3.7202e-03, -1.8335e-03,  …, -2.5930e-03,
            7.6909e-06,  9.3889e-04],
          [-5.1584e-03, -4.6671e-03, -3.0318e-03,  …,  3.0771e-03,
            1.6873e-03,  4.4391e-04],
          …,
          [ 5.2527e-03,  6.8641e-03,  6.1848e-03,  …,  1.0127e-03,
           -4.4822e-04, -2.0694e-03],
          [ 1.0209e-03,  1.4040e-03,  3.6184e-03,  …,  3.0407e-03,
           -7.2960e-04, -3.4964e-03],
          [-5.8630e-04, -9.7428e-05,  2.0094e-03,  …,  4.2332e-03,
            1.9461e-04, -3.8850e-03]],

         [[-2.7560e-03, -1.0543e-03, -3.8633e-04,  …, -1.8456e-03,
           -7.1295e-04, -2.8012e-04],
          [-2.5056e-04,  2.1656e-04,  1.5857e-03,  …,  2.2090e-03,
            4.3140e-03,  4.7146e-03],
          [-6.6452e-04, -1.2257e-03,  8.5671e-05,  …,  6.4002e-03,
            5.3165e-03,  4.0602e-03],
          …,
          [ 7.5651e-03,  8.9069e-03,  8.2998e-03,  …,  2.8857e-03,
            2.1021e-03,  1.2114e-04],
          [ 4.0097e-03,  3.9546e-03,  5.6747e-03,  …,  4.4640e-03,
            1.6352e-03, -9.6658e-04],
          [ 3.3226e-03,  3.2819e-03,  4.4642e-03,  …,  5.9373e-03,
            2.5333e-03, -1.2098e-03]]])
tensor([ 3.2596e-09,  5.4017e-08,  1.2107e-08, -4.6566e-09,  4.4238e-08,
         2.5518e-07, -1.5832e-08,  1.1176e-07, -1.5832e-08, -6.4843e-08,
        -2.3283e-10,  0.0000e+00,  4.5984e-09, -4.8894e-09,  4.5984e-09,
        -1.0477e-09,  7.1013e-08,  3.1432e-09, -3.1665e-08,  6.7055e-08,
        -1.6298e-09,  6.7055e-08,  2.7940e-08,  1.0245e-08, -3.4575e-08,
         5.3842e-08, -2.9104e-10,  1.8161e-08,  5.9139e-08, -3.3528e-08,
         2.0955e-09,  1.7695e-08,  1.4435e-08, -2.7358e-09,  3.1781e-08,
        -6.3592e-09, -7.4506e-09,  4.9127e-08, -1.1642e-09,  7.2177e-09,
         1.0477e-08, -1.8626e-09, -1.9558e-08,  9.8953e-09, -2.6077e-08,
         1.9558e-08,  1.0245e-08, -4.6566e-10,  9.7789e-09,  9.3132e-10,
        -1.2515e-09,  5.5879e-09, -1.1409e-08,  2.7940e-08,  7.6834e-09,
        -3.4925e-08,  2.9802e-08,  4.6566e-09,  2.5611e-09,  2.5611e-09,
         9.3132e-10,  5.1223e-09, -7.4506e-09,  5.5879e-09, -5.4715e-09,
        -7.6834e-09, -1.1314e-09,  3.0268e-09,  6.9849e-10,  4.6566e-09,
        -2.9686e-09,  1.0245e-08, -4.1910e-09, -1.8626e-09, -2.3516e-08,
        -5.3260e-08, -2.7940e-09,  6.9849e-09, -2.1420e-08,  7.9162e-09,
```

```
        -1.8626e-07, -2.7940e-09, -1.3970e-09,  1.3039e-08, -3.7253e-09,
         2.7940e-09,  1.1350e-09, -9.0804e-09, -2.3283e-09,  3.9581e-08,
         9.8953e-10, -4.6566e-09, -1.9558e-08,  1.6065e-08,  4.3306e-08,
        -3.4925e-09])
tensor([ 0.0058,  0.0378, -0.0078,  0.0163,  0.0364, -0.0002, -0.0069,  0.0091,
        -0.0079, -0.0093, -0.0130,  0.0131,  0.0289,  0.0097,  0.0012,  0.0095,
        -0.0182, -0.0522,  0.0095, -0.0055,  0.0160, -0.0433,  0.0123, -0.0046,
        -0.0061, -0.0006,  0.0167, -0.0139, -0.0187, -0.0039,  0.0203,  0.0192,
        -0.0279,  0.0155,  0.0013,  0.0123, -0.0025,  0.0185,  0.0033,  0.0121,
         0.0038, -0.0181, -0.0082,  0.0156,  0.0243,  0.0085, -0.0078, -0.0054,
         0.0015, -0.0009,  0.0072, -0.0135, -0.0084,  0.0008, -0.0046, -0.0016,
        -0.0534,  0.0089,  0.0167, -0.0062, -0.0005,  0.0142, -0.0064,  0.0163,
         0.0087, -0.0063, -0.0015,  0.0093,  0.0183, -0.0117,  0.0182,  0.0023,
         0.0060,  0.0117, -0.0011, -0.0033,  0.0136,  0.0057, -0.0172, -0.0060,
         0.0128,  0.0130,  0.0149, -0.0051,  0.0513, -0.0013, -0.0067, -0.0221,
         0.0178, -0.0130, -0.0080, -0.0058, -0.0593, -0.0118, -0.0010, -0.0032])
tensor([ 2.9317e-03,  1.1918e-02, -6.4376e-03,  7.8289e-03,  8.5580e-03,
         1.1679e-02, -1.7571e-04,  1.3513e-02, -9.6631e-04,  1.1203e-02,
         5.2605e-03,  7.1078e-03,  6.8930e-03,  8.2488e-03,  4.2750e-03,
         5.3803e-03,  1.1137e-02, -1.3848e-02,  3.7950e-03, -6.6791e-03,
         1.0658e-02, -2.0615e-03,  1.8314e-02, -1.4032e-03,  1.9202e-02,
        -3.4892e-03,  8.7834e-03, -6.1498e-03,  4.7032e-03,  1.0079e-02,
         1.3228e-02,  1.5529e-02, -7.6999e-03,  1.1061e-02, -3.1095e-03,
         1.0848e-02, -7.0232e-05,  6.2386e-03, -6.5912e-03,  1.0767e-02,
        -5.6766e-03, -1.0892e-02, -4.4867e-03,  1.1968e-02,  1.9212e-02,
         5.9064e-03, -6.1610e-03, -1.9619e-03,  7.6767e-03, -1.0373e-03,
         3.8429e-03, -3.9146e-03, -2.7399e-03,  1.7295e-02, -7.8247e-03,
         2.3868e-03, -6.6386e-03,  3.3866e-03,  1.3163e-02, -2.2966e-03,
        -1.3764e-03,  1.6006e-03,  1.8998e-02,  8.6612e-03,  2.1743e-03,
        -1.3036e-03,  4.0585e-03,  9.0928e-03,  1.5037e-02, -4.1796e-03,
         1.1163e-02, -3.7811e-03,  1.0922e-02,  3.4683e-03, -9.7775e-03,
         2.0266e-02,  6.5568e-03,  9.3687e-03, -1.0982e-02, -1.7806e-04,
        -9.4166e-04, -3.4495e-03,  6.8957e-03,  5.4720e-04,  1.4777e-02,
        -9.0493e-05,  1.7912e-03, -1.9732e-02,  1.1318e-02, -8.4535e-03,
         9.1427e-03, -9.7031e-03, -1.0765e-02, -1.6060e-03,  1.7359e-02,
        -6.3084e-03])
tensor([[[[-4.5825e-04, -1.5196e-04, -1.4977e-04,  1.0763e-03,  1.2603e-03],
          [ 9.1055e-04,  9.0207e-04,  1.1467e-03,  1.0916e-03,  9.0043e-04],
          [ 7.9687e-04,  1.5736e-03,  1.5789e-03,  1.8119e-03,  1.5699e-03],
          [ 9.9928e-04,  1.1814e-03,  7.9820e-04,  1.1227e-03,  1.3264e-03],
          [-6.9100e-04, -5.7206e-04, -1.1326e-03, -1.1652e-03, -1.1350e-03]],

         [[ 3.4660e-03,  1.9136e-03,  2.4956e-05, -1.5633e-03, -1.6881e-03],
          [ 3.2663e-03,  1.2613e-03, -8.8371e-04, -1.7877e-03, -1.4260e-03],
          [ 2.0277e-03, -4.6614e-04, -1.4170e-03, -1.2742e-03, -8.4879e-04],
          [ 1.6067e-03, -8.2310e-04, -2.2573e-03, -9.9382e-04, -4.2926e-04],
          [ 1.4948e-03,  4.8639e-04, -9.8165e-04, -6.4519e-04,  8.8710e-06]],
```

```
 [[ 1.1603e-03,  9.9496e-04,  6.8119e-04,  3.4231e-04, -6.5209e-04],
  [ 1.1991e-03,  1.0039e-03,  7.7942e-04,  4.7118e-04, -2.9339e-04],
  [ 1.0327e-03,  7.6398e-04,  4.1412e-04,  3.5632e-04,  2.7731e-04],
  [ 6.7722e-04,  4.2018e-04,  1.5778e-04,  1.4747e-04, -2.9530e-05],
  [ 2.5444e-04,  1.0319e-04,  2.4408e-05,  1.8364e-04, -9.1232e-06]],

 …,

 [[ 7.5587e-04,  1.9576e-04,  3.4707e-04,  1.8174e-04, -6.4112e-04],
  [ 2.8674e-04,  2.5496e-04,  2.8977e-04,  3.0777e-04,  7.5908e-05],
  [ 1.5939e-04,  4.7149e-04,  1.4191e-04,  2.4517e-04,  2.9692e-04],
  [-1.4436e-04, -1.7368e-04, -5.3247e-04, -4.8579e-04, -3.5700e-04],
  [-9.6972e-05, -2.6924e-04, -5.0895e-04, -6.5807e-04, -8.0426e-04]],

 [[ 3.4992e-04, -3.0579e-04, -4.2981e-04, -5.2538e-04, -7.2426e-04],
  [-8.7404e-05,  2.4745e-05,  5.3816e-05,  7.5943e-05, -2.2071e-04],
  [-1.5376e-04,  3.6654e-04,  1.1851e-05,  1.2052e-04, -6.2802e-05],
  [-6.5134e-04, -5.5320e-04, -8.4519e-04, -5.6656e-04, -4.8303e-04],
  [-6.4319e-04, -3.3696e-04, -8.8463e-04, -8.5160e-04, -1.0127e-03]],

 [[ 1.1444e-03,  1.0161e-03,  1.1702e-03,  1.2351e-03,  7.1063e-04],
  [ 8.8364e-04,  9.1422e-04,  9.9569e-04,  1.2626e-03,  1.4836e-03],
  [ 4.2554e-04,  6.8465e-04,  6.4197e-04,  9.2008e-04,  1.3855e-03],
  [-3.0174e-04, -1.5377e-04, -1.3087e-04,  7.6771e-05,  7.5802e-05],
  [-2.7610e-04,  1.2582e-04,  1.4481e-04,  1.1704e-04, -1.8687e-04]]],


[[[ 4.8098e-04,  8.4815e-04,  7.1384e-04,  4.7158e-04,  7.8624e-04],
  [ 2.6579e-04,  7.1722e-04,  9.6299e-04,  1.0410e-03,  9.4651e-04],
  [-4.0118e-04, -1.0187e-04,  3.1108e-04,  4.4882e-04,  1.9577e-04],
  [-2.1986e-04, -1.5550e-04, -6.5642e-05,  2.7780e-04,  3.3990e-04],
  [ 6.0676e-04,  1.1319e-03,  1.0969e-03,  1.2116e-03,  1.0977e-03]],

 [[-4.1265e-04,  6.4118e-04,  7.4019e-04,  9.3907e-04,  9.8198e-04],
  [-8.8683e-04,  5.0590e-04,  7.1794e-04,  8.1776e-04,  3.7719e-04],
  [-1.1711e-03,  3.4835e-04,  1.8575e-04,  8.3444e-04,  8.9960e-04],
  [-1.0496e-03,  5.5390e-04,  4.0163e-05,  4.8577e-04,  1.0240e-03],
  [-1.0262e-03,  4.4012e-04,  2.7900e-04,  5.8580e-04,  1.7821e-03]],

 [[-7.5213e-04, -3.3464e-04, -1.3981e-04, -2.9180e-05, -6.2278e-05],
  [-9.9068e-04, -6.1608e-04, -3.0282e-04, -9.9753e-05, -8.0193e-06],
  [-1.0396e-03, -7.1220e-04, -4.3593e-04, -1.5823e-04,  1.9215e-05],
  [-1.1624e-03, -7.4521e-04, -4.5013e-04, -2.0167e-04, -2.5831e-06],
  [-9.9445e-04, -7.1270e-04, -2.8155e-04, -1.2032e-04, -2.4206e-05]],

 …,

 [[-9.2660e-05,  1.1576e-04,  3.7519e-04,  6.5605e-04,  7.0517e-04],
```

```
  [-2.5093e-04, -1.1378e-05,  2.1569e-04,  4.9977e-04,  5.5250e-04],
  [-4.0186e-04, -1.1256e-04,  1.3451e-04,  5.1083e-04,  7.0961e-04],
  [-4.3873e-04, -8.1759e-05,  2.3723e-04,  4.7977e-04,  6.2448e-04],
  [-3.3832e-04, -5.3988e-05,  2.2498e-04,  5.0687e-04,  6.3683e-04]],

 [[ 3.3561e-04,  4.6928e-04,  7.0900e-04,  6.7213e-04,  8.1126e-04],
  [ 1.5530e-04,  3.7738e-04,  6.1473e-04,  6.3312e-04,  6.6344e-04],
  [ 1.6184e-04,  2.1557e-04,  5.0722e-04,  6.2121e-04,  6.5087e-04],
  [ 1.5934e-04,  2.2519e-04,  4.7547e-04,  6.0897e-04,  6.4413e-04],
  [ 2.0122e-04,  2.6400e-04,  5.6075e-04,  5.7087e-04,  6.5409e-04]],

 [[ 1.1458e-04,  4.5675e-04,  8.7276e-04,  1.1086e-03,  1.0485e-03],
  [ 1.0798e-04,  3.2141e-04,  5.8546e-04,  8.0748e-04,  8.0187e-04],
  [-1.3252e-04,  2.3743e-04,  5.7007e-04,  6.3441e-04,  8.5153e-04],
  [ 8.4383e-05,  3.7728e-04,  8.0430e-04,  7.0024e-04,  9.1605e-04],
  [ 3.8668e-04,  5.5170e-04,  9.7474e-04,  1.0585e-03,  1.3620e-03]]],


[[[-1.8707e-03, -2.0241e-03, -5.8828e-04,  1.5520e-04,  6.8569e-04],
  [-6.0952e-04,  2.6346e-04,  8.4426e-04,  1.1863e-03,  5.0479e-04],
  [-1.2063e-03, -6.4381e-04, -8.8582e-04, -1.3022e-03, -1.9790e-03],
  [ 1.0443e-04, -1.6067e-05, -8.5970e-04, -1.0071e-03, -1.7974e-03],
  [-1.9647e-04,  2.5807e-04, -2.0889e-04, -3.1111e-04, -1.3492e-04]],

 [[-7.4529e-04, -1.0608e-03,  4.2828e-04,  7.2001e-04,  6.4524e-04],
  [-1.0386e-03, -6.0486e-04,  1.6992e-03,  8.7570e-04, -3.4796e-04],
  [-1.7851e-03,  6.0969e-04,  2.3935e-03,  8.7639e-04, -4.6519e-04],
  [-2.1239e-03,  1.3135e-03,  1.3790e-03,  5.0186e-05,  1.9303e-04],
  [-2.3660e-03,  1.8805e-03,  3.3145e-03,  3.5138e-04,  1.7484e-03]],

 [[-1.3033e-03, -4.1476e-04, -1.7482e-04, -3.4491e-04, -6.1795e-04],
  [-9.9767e-04, -8.0660e-04, -8.7365e-04, -7.1222e-04, -8.9395e-04],
  [-1.0196e-03, -1.1191e-03, -7.3846e-04, -3.1546e-04, -4.5664e-04],
  [-1.1943e-03, -1.0785e-03, -8.6485e-04, -2.7638e-04, -2.1601e-04],
  [-1.7100e-03, -1.5497e-03, -8.3214e-04,  4.4093e-05, -3.5698e-04]],

 …,

 [[-2.5670e-03, -2.6116e-03, -2.8544e-03, -2.5109e-03, -2.6291e-03],
  [-2.9126e-03, -2.9203e-03, -3.2397e-03, -2.9371e-03, -3.1044e-03],
  [-3.5925e-03, -3.2999e-03, -2.6939e-03, -2.4231e-03, -2.9472e-03],
  [-3.9026e-03, -3.7584e-03, -3.3994e-03, -2.8751e-03, -3.0300e-03],
  [-3.8823e-03, -3.5979e-03, -2.6168e-03, -1.8700e-03, -2.1893e-03]],

 [[-2.9274e-03, -3.0746e-03, -3.0976e-03, -2.6885e-03, -2.8553e-03],
  [-2.8960e-03, -3.1303e-03, -3.2404e-03, -3.1147e-03, -3.3134e-03],
  [-3.5159e-03, -3.5065e-03, -2.9672e-03, -3.0768e-03, -3.3954e-03],
  [-3.5226e-03, -3.4421e-03, -3.1716e-03, -3.1287e-03, -3.4884e-03],
```

```
  [-3.8292e-03, -3.4617e-03, -2.9641e-03, -2.8591e-03, -3.1843e-03]],

 [[-1.9961e-03, -2.1060e-03, -2.2775e-03, -1.9400e-03, -2.0317e-03],
  [-2.6375e-03, -2.5883e-03, -2.3019e-03, -2.2849e-03, -2.6193e-03],
  [-3.5475e-03, -3.2122e-03, -2.1226e-03, -2.1370e-03, -2.4287e-03],
  [-4.2504e-03, -3.3960e-03, -3.1705e-03, -2.9656e-03, -2.6275e-03],
  [-3.3871e-03, -2.5931e-03, -1.4821e-03, -1.3808e-03, -1.2383e-03]]],


…,


[[[-7.2535e-04,  1.1535e-04, -4.0075e-04, -8.1619e-04, -1.6751e-03],
  [-1.4932e-03, -9.9740e-04, -1.3203e-03, -1.0274e-03, -6.1567e-04],
  [-1.7900e-03, -1.1882e-03, -3.6050e-04,  2.0778e-04,  4.2693e-04],
  [-1.1107e-03, -1.0199e-03, -8.3548e-04, -6.6617e-04, -1.2480e-03],
  [-6.2436e-04, -1.5525e-03, -1.5779e-03, -1.6210e-03, -1.7864e-03]],

 [[ 3.5177e-04,  1.9875e-03,  3.5939e-03,  1.8737e-03,  2.4439e-03],
  [-1.7798e-04,  1.8076e-03,  3.3030e-03, -8.6559e-05,  1.0279e-03],
  [-4.6783e-04,  1.6606e-03,  1.9745e-03, -1.5089e-03, -9.8270e-04],
  [ 3.9099e-04,  3.1520e-04,  8.5138e-04, -1.6761e-03, -2.1376e-03],
  [ 6.3706e-05, -2.9326e-04,  8.8391e-04, -1.5136e-03, -1.0252e-03]],

 [[ 1.6302e-04, -3.9043e-04,  5.9555e-04,  9.4940e-04,  9.0797e-04],
  [ 9.4183e-05, -1.8420e-04,  9.7643e-04,  1.3814e-03,  1.0830e-03],
  [ 4.9747e-04,  1.9742e-04,  1.3430e-03,  1.4490e-03,  9.1095e-04],
  [ 1.0855e-03,  6.5784e-04,  1.3879e-03,  1.1057e-03,  7.3903e-04],
  [ 7.9104e-04,  4.4743e-04,  1.1329e-03,  1.0083e-03,  7.6534e-04]],

 …,

 [[-2.8747e-04,  8.8379e-05,  8.3583e-04,  1.0460e-03,  4.5170e-04],
  [-3.0533e-04,  4.7322e-04,  1.5240e-03,  1.1286e-03,  4.9535e-04],
  [ 5.4703e-06,  5.8677e-04,  1.5178e-03,  1.2986e-03,  9.5131e-04],
  [-1.3223e-04,  2.7459e-04,  1.0773e-03,  1.0770e-03,  7.6776e-04],
  [-4.7126e-04, -1.9675e-05,  5.3712e-04,  4.8504e-04,  4.4678e-04]],

 [[-9.7745e-04, -6.6669e-04, -4.2310e-04, -6.5660e-04, -1.1511e-03],
  [-1.0813e-03, -1.9647e-04,  1.8224e-05, -5.9640e-04, -1.1394e-03],
  [-5.9435e-04,  2.0822e-05,  2.7197e-04, -5.6174e-05, -4.9104e-04],
  [-1.0365e-03, -4.7657e-04, -5.6206e-05, -4.2089e-04, -6.8469e-04],
  [-1.6022e-03, -1.3098e-03, -9.0915e-04, -1.1058e-03, -9.9414e-04]],

 [[-1.0859e-03, -4.5336e-04,  6.9706e-04,  7.8121e-04,  3.0647e-04],
  [-9.3696e-04,  3.5625e-04,  1.3450e-03,  4.2324e-04,  2.0135e-04],
  [-3.0760e-04,  4.9465e-04,  1.1482e-03,  4.4945e-04,  3.1487e-04],
  [-5.6502e-04, -7.5838e-05,  1.9437e-04, -1.4007e-04, -8.3301e-05],
```

```
  [-6.1866e-04, -4.0220e-04, -4.8123e-05, -3.3043e-04, -2.7984e-04]]],


[[[-7.0813e-04, -9.1770e-04, -1.0759e-03, -9.6091e-04, -2.0569e-05],
  [ 8.7819e-04,  6.6553e-04, -3.5509e-04, -1.0914e-03, -1.3581e-03],
  [ 1.6045e-03,  1.3026e-03,  8.4129e-04, -8.5000e-04, -2.2579e-04],
  [ 8.8252e-04,  9.0608e-04,  1.2063e-04, -1.7434e-04, -8.8076e-06],
  [-8.7249e-04, -1.2518e-03, -5.8521e-04, -8.3377e-04, -1.0977e-03]],

 [[ 1.0104e-03, -2.1069e-03, -4.0609e-03, -4.3097e-03, -2.0928e-03],
  [ 1.1032e-03, -2.6041e-03, -5.0967e-03, -4.9854e-03, -1.9331e-03],
  [ 1.3072e-03, -2.0189e-03, -5.0104e-03, -3.8752e-03, -1.5821e-03],
  [-1.5292e-04, -1.4113e-03, -4.2535e-03, -3.8078e-03, -2.2486e-03],
  [ 3.2135e-05, -5.8233e-04, -3.5130e-03, -2.4538e-03, -2.7963e-03]],

 [[ 7.3687e-04,  8.6758e-04,  2.5977e-04, -1.0036e-04, -2.1599e-04],
  [ 2.6775e-04,  3.0918e-04,  1.8648e-04, -1.6342e-04, -3.9715e-04],
  [ 4.2907e-04,  6.0926e-04,  1.3799e-04, -3.0166e-04, -8.0693e-05],
  [ 1.4282e-04, -3.8340e-05, -1.6580e-04, -3.1062e-04, -1.6176e-04],
  [-2.4193e-04, -2.5792e-04, -4.0416e-04, -5.9554e-04, -4.4834e-04]],

 …,

 [[ 1.2133e-04, -2.3534e-04, -2.0073e-04, -2.0807e-04, -9.6379e-04],
  [ 1.7737e-04, -1.7553e-04, -1.8595e-04, -4.9949e-04, -8.6480e-04],
  [ 3.0699e-04,  6.8239e-04,  3.8648e-04, -5.7703e-05, -3.1252e-04],
  [ 2.8904e-04,  7.0228e-04,  5.7344e-04,  6.6891e-05,  5.1513e-05],
  [ 2.3810e-04,  4.2748e-04,  6.4202e-04,  2.4587e-04,  9.9520e-05]],

 [[ 2.8841e-05, -6.8888e-04, -4.9086e-04, -3.6702e-04, -1.0996e-03],
  [ 1.2575e-04, -4.6249e-04, -2.8509e-04, -5.7783e-04, -7.9157e-04],
  [ 3.7068e-04,  6.9510e-04,  3.0014e-04, -3.7576e-04, -3.3376e-04],
  [ 7.2898e-04,  1.2721e-03,  7.5400e-04,  1.4808e-04, -4.4602e-04],
  [ 3.4626e-04,  1.0527e-03,  9.4610e-04,  2.2569e-04, -4.5511e-04]],

 [[ 1.8131e-04, -3.6548e-04, -6.4820e-04, -8.8897e-04, -1.1293e-03],
  [ 3.3341e-04, -1.8960e-05, -1.1860e-04, -1.0496e-03, -7.4861e-04],
  [ 4.7580e-04,  6.8160e-04,  5.8804e-04, -3.0323e-04,  6.7885e-05],
  [ 3.0523e-04,  4.0014e-04,  5.0165e-04,  2.6690e-04,  1.1198e-04],
  [ 2.5939e-04,  6.7300e-05,  2.8632e-04,  2.3961e-04,  1.9237e-06]]],


[[[ 1.0817e-03,  5.3181e-04, -3.2835e-05, -8.9271e-05, -1.3556e-04],
  [ 7.3826e-05, -4.1490e-04, -1.0368e-03, -6.0094e-04, -4.4880e-04],
  [-2.5958e-04, -4.5371e-04, -3.9831e-04, -2.2512e-04, -3.5782e-04],
  [ 7.1092e-05, -2.7350e-04, -1.4522e-04,  4.6567e-05, -2.0670e-04],
  [ 8.2239e-04,  9.0231e-04,  7.4864e-04,  2.9467e-04,  2.4133e-04]],
```

```
          [[ 1.2238e-04, -1.2528e-04,  5.8565e-04, -3.3402e-04, -5.9068e-04],
           [-7.4974e-05, -1.3025e-04,  6.5415e-04,  5.0605e-04,  6.8993e-04],
           [ 5.6979e-04,  7.7845e-04,  1.1699e-03,  1.0313e-03,  7.2848e-04],
           [ 1.8105e-03,  1.7676e-03,  1.4529e-03,  1.5795e-03,  6.2580e-04],
           [ 1.3203e-03,  1.1046e-03,  1.6836e-03,  1.9716e-03,  8.8814e-04]],

          [[ 1.5080e-04, -3.4580e-05,  1.1283e-04,  2.1800e-04, -7.5057e-05],
           [ 1.6173e-04, -3.0047e-04,  2.2040e-04,  2.1999e-04, -3.3500e-05],
           [ 1.2811e-04,  2.2216e-05,  3.4201e-04,  4.5789e-04,  2.9732e-04],
           [-5.0484e-05,  3.0462e-04,  4.9270e-04,  3.8254e-04,  2.8881e-04],
           [ 1.6198e-04,  2.4127e-04,  4.2547e-04,  6.1584e-04,  3.6453e-04]],

          …,

          [[ 5.4932e-04,  4.0268e-04,  1.7512e-04,  3.9389e-05, -7.8929e-05],
           [ 2.4475e-04,  1.6190e-04,  1.1591e-04, -3.2704e-05,  9.5551e-06],
           [ 6.2687e-05,  1.1587e-04,  5.2466e-05, -2.4893e-05,  2.6634e-05],
           [ 1.8621e-04,  1.2760e-04,  1.8534e-04,  1.4653e-04,  4.5733e-05],
           [ 2.1959e-04,  1.2209e-04,  1.8845e-04,  2.1546e-04,  1.4912e-04]],

          [[ 8.4865e-05,  1.2542e-04,  4.5029e-05,  1.6985e-04,  2.6725e-04],
           [ 4.3555e-05,  1.4009e-04,  4.5342e-05, -1.3748e-04, -2.4173e-05],
           [-3.4979e-06,  1.2287e-04,  1.5106e-04,  1.1291e-05,  1.0076e-04],
           [ 9.0308e-05,  1.3299e-04,  1.0521e-04,  1.3892e-04,  2.0314e-04],
           [ 4.9304e-05,  1.3420e-04,  2.3886e-04,  3.4143e-04,  2.8199e-04]],

          [[ 4.2647e-04,  2.5379e-04, -1.4480e-04, -5.4389e-04, -6.5842e-04],
           [-1.3457e-04, -4.3256e-05,  7.6271e-05, -3.0340e-04, -2.1332e-04],
           [-3.7761e-04, -8.9518e-05, -4.6117e-05, -2.1868e-04, -2.4399e-04],
           [ 2.5604e-05,  1.5739e-04,  7.2420e-05,  2.8282e-04,  1.6070e-04],
           [ 3.7213e-04,  5.5401e-04,  3.1578e-04,  3.8590e-04,  2.7803e-04]]]])
tensor([ 2.3267e-09,  2.8014e-10,  3.7376e-09,  2.4480e-09,  7.3911e-10,
        -1.7021e-09, -1.4623e-08, -4.7119e-09, -1.1623e-09,  9.1723e-10,
        -1.4411e-09, -4.0563e-10,  7.5724e-09, -1.4588e-09,  1.2542e-09,
        -2.1246e-09,  1.0277e-10, -3.9004e-09,  6.2756e-09, -6.8166e-09,
        -1.3079e-09, -3.2696e-10, -7.1532e-10, -1.6185e-09,  3.2105e-10,
        -7.4476e-10, -3.4819e-09, -1.2554e-08,  8.0847e-09, -5.5026e-09,
        -5.6696e-10, -6.0350e-09, -1.7454e-09, -1.8700e-09, -3.2742e-11,
         3.0563e-09, -3.7351e-09, -1.0736e-09,  4.4360e-09, -5.5121e-10,
         5.6826e-10,  6.7638e-10,  3.7253e-09, -1.2551e-09,  1.2226e-08,
        -8.9506e-09, -8.0239e-10, -6.0500e-09, -2.1832e-09,  3.9081e-10,
        -1.6123e-09, -2.3374e-10,  4.3425e-09, -1.0326e-09, -2.4833e-09,
        -5.1138e-09,  2.8855e-09, -1.8646e-09,  5.2484e-10, -1.0953e-09,
         1.1026e-09,  6.5737e-10,  1.9127e-09,  1.6588e-10,  9.6134e-10,
        -6.3984e-10, -1.0746e-08, -4.3019e-09, -4.2223e-10, -1.2834e-09,
        -2.9627e-09, -3.2378e-10,  5.3915e-09, -2.3186e-09, -8.8806e-09,
         1.1869e-09,  5.0358e-10, -1.7799e-09,  1.8638e-10,  1.1079e-10,
         8.1855e-10,  4.3997e-10,  8.4095e-09, -8.4494e-09, -2.0067e-09,
```

```
         2.0075e-09,  1.6498e-09,  3.4670e-09,  2.8303e-09, -1.1962e-09,
         1.3550e-09, -3.2232e-09, -3.9449e-09, -3.7479e-09, -9.6031e-10,
        -6.6498e-10,  9.7361e-10, -1.2186e-08, -1.5441e-09,  3.8082e-09,
         2.3900e-09, -8.3264e-10, -1.0859e-09,  1.4152e-10,  2.0900e-09,
        -1.5643e-10, -5.9093e-09, -6.6166e-10,  2.5102e-10, -3.9845e-09,
         4.9840e-10,  1.4818e-09, -1.8149e-09, -5.3456e-10,  1.4068e-09,
        -7.2200e-10, -1.3384e-09, -4.1018e-10,  5.3860e-09,  4.1475e-09,
         5.6758e-09, -1.2915e-09,  1.2568e-09,  1.3260e-09,  8.5394e-10,
         1.2622e-09,  1.6199e-09, -3.0891e-09,  2.6298e-09,  4.1437e-09,
         1.5482e-09,  6.0032e-10, -2.3651e-09, -2.1638e-09, -2.2388e-09,
         6.1511e-09,  9.1653e-09, -4.2855e-09,  1.5280e-09,  4.0563e-10,
        -6.2835e-10, -4.8840e-10,  2.0865e-09, -1.0269e-09, -4.1770e-10,
         1.3276e-10, -3.1441e-08, -4.0706e-09,  2.4243e-09,  1.2215e-09,
         1.6958e-09, -5.3274e-09,  6.4912e-09, -3.3098e-09, -1.2452e-08,
        -3.5049e-09, -1.5061e-09,  6.6387e-10,  4.3292e-10,  4.4791e-09,
        -9.3106e-10, -2.7794e-09, -2.7285e-11, -2.0579e-09, -2.4716e-09,
         6.6029e-09,  1.1171e-09, -6.9667e-10,  4.3343e-11, -1.1221e-09,
        -1.1618e-08, -3.3014e-09, -1.4592e-09, -2.1650e-09, -1.0035e-08,
        -1.8107e-09, -1.4575e-09,  1.5254e-08, -1.0732e-10, -6.2687e-09,
        -4.0127e-09,  7.3072e-11, -3.9873e-09,  3.4620e-10, -2.5348e-09,
        -6.3847e-10,  2.5548e-08, -1.2584e-11,  1.3171e-09, -3.5606e-09,
         6.7212e-10,  2.6830e-09,  3.7668e-09, -2.5054e-10,  4.8152e-10,
         1.5261e-09, -3.4757e-10, -2.4531e-09,  9.1408e-10,  6.0422e-09,
         2.6176e-10, -3.3736e-09,  1.0658e-10,  5.8123e-10, -4.3810e-09,
        -9.5127e-10, -1.1301e-09,  1.7139e-09,  7.4579e-11,  2.3056e-09,
        -9.1140e-10,  8.8812e-10, -3.7118e-09, -4.9772e-10,  2.4735e-09,
        -2.7376e-10,  5.1405e-09,  3.6625e-09,  6.8425e-10,  2.1528e-09,
        -3.9595e-09,  8.7454e-10, -1.2907e-09, -1.3928e-08,  6.2943e-10,
        -1.7167e-09,  8.9727e-10, -2.1178e-09, -7.9564e-09, -6.0167e-10,
        -1.3497e-09,  1.9827e-09,  3.2716e-10,  9.1495e-10, -2.2701e-09,
         7.3296e-09,  2.9841e-09,  2.8197e-09,  6.1959e-10, -2.0755e-09,
         5.0068e-10, -2.6108e-10, -7.2309e-10, -1.1023e-09, -1.8888e-09,
         8.3496e-10, -5.4428e-10,  2.4665e-09, -1.1778e-09, -7.0829e-09,
        -5.1171e-10, -1.4643e-09,  7.7944e-10, -1.8458e-09,  1.0930e-08,
        -4.1764e-09])
tensor([ 1.2286e-02, -5.7108e-04,  5.1119e-03, -1.2576e-03, -2.4171e-03,
        -5.1987e-03, -7.8241e-03, -5.5843e-03,  2.5080e-02,  7.6845e-04,
         1.1026e-03, -3.1776e-02,  1.4758e-02, -8.3475e-03,  2.1006e-03,
         6.1168e-03, -2.6399e-03, -2.2579e-03, -2.4416e-03, -1.1255e-02,
         2.1552e-03, -4.7201e-02,  8.5738e-04, -6.2134e-03, -6.5827e-04,
         2.3665e-02, -8.1537e-03, -5.3253e-02,  1.4072e-02, -5.7264e-03,
         5.0897e-03,  9.8051e-03, -2.7988e-02, -1.4534e-02,  6.0871e-03,
        -1.4578e-03, -4.2180e-03, -1.7107e-03,  1.1617e-02,  1.5824e-03,
        -3.1109e-03, -3.2550e-03,  6.3581e-03,  1.1546e-03, -1.8918e-03,
        -1.6561e-02, -4.5972e-03,  3.3942e-03, -3.4348e-02, -1.3056e-03,
        -1.6408e-03,  5.6882e-03,  1.1641e-02,  2.0899e-03, -2.5352e-02,
         1.9733e-03,  3.1650e-04,  2.5798e-03,  2.0219e-02, -5.1385e-03,
         7.3495e-03, -8.1341e-03,  5.4332e-03,  1.9775e-03, -6.2754e-04,
```

```
        -4.5272e-04, -6.7619e-03,  5.5422e-03, -3.0857e-02,  2.0088e-03,
         2.3084e-02,  9.5132e-03,  4.3878e-03, -2.8047e-03,  1.1233e-03,
        -5.4468e-02, -3.4341e-04, -3.0037e-03, -1.3843e-03,  7.7654e-03,
         2.2884e-03,  4.8793e-03,  1.0040e-02,  5.0425e-03, -1.2772e-02,
         3.4805e-02, -2.7878e-03, -2.8552e-03,  2.9291e-02,  3.0849e-03,
         2.9263e-04, -3.4107e-03,  2.6476e-02, -7.0728e-03, -6.9076e-02,
         3.4994e-03, -1.0885e-03, -3.6416e-05, -3.7662e-04,  6.5601e-03,
         8.7909e-03, -2.9238e-03,  1.6431e-03, -1.7532e-03,  1.4277e-02,
         3.9494e-03,  3.3455e-03,  5.0701e-03,  1.2212e-02,  2.6402e-02,
        -3.4590e-03, -9.1113e-04,  5.6557e-03, -5.2390e-03, -1.5621e-03,
        -4.3929e-03, -3.9090e-03, -2.4685e-03, -9.0167e-03, -3.4823e-03,
         9.6659e-03,  1.4494e-02,  7.5447e-03,  3.1723e-03,  2.4226e-03,
         9.0633e-04,  1.0804e-02, -3.4484e-03, -4.1569e-04,  2.1123e-02,
        -1.9550e-03,  6.7129e-03,  4.9088e-03, -5.6817e-02, -6.2175e-03,
         4.3523e-02,  2.6949e-02, -2.2927e-02,  2.3549e-03, -2.2426e-02,
         1.2593e-02, -2.5049e-03,  5.1125e-03,  3.5765e-04,  5.4723e-03,
        -4.7835e-04, -2.6405e-03,  2.1817e-03,  2.0110e-02,  1.7698e-03,
         2.6931e-03,  5.7705e-03, -4.5746e-03, -2.0884e-02,  5.7472e-03,
        -7.2835e-03,  3.3520e-03, -1.0144e-03, -1.2997e-02, -9.3660e-03,
        -2.1367e-03, -3.1277e-03, -1.7297e-03,  8.0760e-04,  1.4812e-02,
         1.7802e-02, -8.0605e-03,  5.8596e-03, -8.4009e-03,  1.8375e-02,
         1.3166e-02, -7.7543e-04,  3.0671e-03,  8.2273e-03, -1.0165e-02,
         1.5468e-03,  8.9074e-04, -4.2360e-03,  8.5159e-03,  5.4866e-03,
        -4.5702e-03,  6.3941e-03, -1.6175e-03,  4.3307e-04, -3.5868e-03,
         1.2018e-03,  6.5795e-03,  3.9825e-03,  5.3004e-03,  1.5120e-03,
         4.0458e-03, -1.6205e-03, -1.7956e-02, -2.0515e-02,  9.3430e-04,
        -1.9195e-03, -5.4139e-03, -1.5228e-03,  2.3242e-03,  1.4763e-02,
         9.5567e-03,  1.7186e-03, -7.6934e-03,  1.2000e-03, -6.4023e-03,
         2.6783e-03, -2.9237e-03,  1.8704e-02,  1.6989e-03,  9.7377e-03,
         5.9012e-03,  4.3497e-03, -1.5749e-03, -7.9467e-03, -4.4534e-03,
        -1.0549e-03, -1.4411e-03,  3.6776e-02,  2.4609e-03,  1.9594e-03,
         3.5562e-03,  3.9548e-03, -1.6067e-02,  2.5598e-02, -1.0125e-01,
         7.3432e-05,  8.9234e-04,  1.2671e-02, -6.3468e-03, -6.0395e-03,
         1.1561e-02, -3.0974e-03,  4.6156e-03, -3.7860e-02,  2.1047e-03,
         8.7050e-03,  6.0101e-02, -1.4935e-02, -9.5850e-03,  3.4783e-03,
         4.8003e-03, -1.0839e-02,  1.6122e-02,  4.9870e-03, -1.3953e-02,
        -2.4060e-03, -3.6657e-02,  2.9220e-03,  3.4283e-03, -3.4095e-02,
         1.6312e-03,  2.0371e-03,  2.6491e-03,  6.6015e-03, -6.9433e-03,
         4.9920e-04])
tensor([ 9.9522e-03,  2.3019e-03, -5.0027e-04, -1.2150e-02, -1.6047e-03,
        -5.2986e-03, -6.9855e-03, -1.1909e-02,  2.4792e-02,  1.8191e-03,
         1.1133e-02, -3.0194e-02,  1.5170e-02, -7.2731e-03,  7.0435e-03,
         2.8554e-03, -6.5517e-05, -9.2742e-03, -4.8155e-03, -1.2878e-02,
         7.6541e-04, -2.9162e-02,  1.6397e-03, -1.1574e-03,  2.4244e-03,
         1.6106e-02, -7.5879e-03, -1.2137e-02,  1.5484e-02,  1.0387e-03,
         1.7808e-02,  1.0308e-02, -1.9819e-02, -2.6229e-02,  9.2071e-03,
        -4.7282e-03, -3.7128e-03, -1.4024e-03,  5.2301e-03, -2.9034e-04,
        -9.9732e-03, -6.1749e-03,  1.4443e-02,  1.0596e-03,  1.3369e-03,
```

```
        -1.9394e-02,  2.1594e-03,  6.6772e-03, -1.7424e-02, -4.2982e-03,
        -8.6604e-03,  7.6858e-03,  1.1526e-02,  9.4850e-03, -2.1471e-02,
        -4.7501e-03, -4.3556e-03,  4.3726e-03,  7.5242e-03, -7.9877e-03,
         5.1519e-03,  4.1337e-03, -3.2591e-03,  5.7870e-03, -3.0046e-04,
        -1.2317e-03, -4.3239e-03,  7.4965e-03, -3.3664e-02, -7.1391e-03,
         1.5815e-02,  1.5593e-02,  1.1602e-02, -3.5561e-03, -1.0453e-02,
        -2.9459e-02, -8.9393e-03, -5.5925e-04,  9.6928e-04,  3.0593e-03,
         3.8887e-03,  4.9918e-03,  7.5607e-03, -6.5530e-03, -4.2486e-03,
         2.6934e-02, -1.1413e-02, -1.0580e-02,  2.6433e-02, -1.2543e-03,
        -2.9063e-03, -5.0323e-03,  1.8659e-02, -9.4006e-03, -5.1595e-02,
         1.2423e-02, -1.0094e-04,  7.9733e-03, -5.0847e-04,  4.0310e-03,
         3.4631e-03, -4.0687e-03,  8.3220e-03, -5.4914e-05,  1.1482e-02,
        -1.0077e-04,  5.3039e-04,  1.7711e-03,  1.3805e-02,  2.0083e-02,
        -8.5405e-03, -3.5831e-03,  5.9742e-03, -7.0979e-03,  2.3658e-03,
         2.6450e-04, -1.0327e-03, -1.1677e-04, -1.2731e-02, -5.8231e-03,
         7.1320e-03,  2.6368e-02,  1.2874e-02,  5.8239e-03,  3.4023e-03,
        -1.6767e-03,  5.1810e-03, -2.4338e-03, -1.8808e-03,  2.6581e-02,
        -2.4328e-04,  7.7715e-03,  2.4007e-04, -2.9635e-02, -7.4700e-03,
         2.8394e-02,  1.2947e-02, -3.3641e-03, -6.4815e-05, -2.2311e-02,
         1.8028e-02, -4.3378e-03,  2.2322e-03,  2.5764e-03,  1.6256e-03,
         1.3280e-04,  6.9775e-03,  7.4104e-03,  1.5008e-02, -3.8329e-03,
        -9.5809e-04,  3.7380e-03, -1.0274e-02, -2.6659e-02,  1.9125e-03,
        -5.1141e-03,  1.0605e-03, -5.4213e-04, -1.0684e-02, -5.0643e-03,
        -4.1823e-03, -7.8470e-03,  3.1096e-03,  3.2362e-03,  1.3992e-02,
         1.1631e-02, -9.6670e-03,  9.9980e-03, -1.9649e-03,  1.5462e-02,
         9.8030e-03, -7.5624e-04,  4.2669e-04,  6.4076e-05, -9.6081e-03,
        -2.0222e-03, -4.0358e-03,  2.7748e-04,  1.4495e-02,  8.7448e-03,
        -7.8928e-03,  1.2939e-02, -6.7737e-03, -1.4080e-03, -3.1887e-03,
         2.5624e-04,  1.0850e-02,  6.7090e-04,  1.1050e-02, -2.5683e-03,
         5.6609e-03, -6.9102e-03, -1.3225e-02, -2.0967e-02,  3.3361e-03,
        -2.7717e-04, -3.6170e-03, -4.1038e-03,  5.8125e-03,  2.4762e-03,
         8.9842e-03,  9.6896e-03, -1.5134e-02, -3.2691e-03, -3.1664e-03,
         1.0591e-03, -2.2394e-04,  1.8339e-02,  5.4014e-03,  1.2973e-02,
         6.3848e-03,  8.0114e-03,  1.4475e-03,  5.6735e-03, -3.7093e-03,
         5.7067e-03, -6.3815e-03,  2.4773e-02, -1.6359e-03,  2.1553e-03,
        -7.2629e-05,  8.5603e-03, -1.0394e-02,  3.8432e-03, -5.8142e-02,
        -2.5930e-03,  6.3832e-03,  1.3853e-02,  4.9209e-04, -7.6377e-03,
         1.0177e-02, -2.0660e-03,  3.3885e-03, -2.8096e-02,  1.6346e-03,
        -8.3417e-03,  3.0380e-02, -4.4642e-03, -4.0837e-03, -1.7244e-03,
         6.2758e-03, -7.6949e-03,  8.8269e-03,  1.1930e-02, -4.0062e-03,
        -2.4058e-03, -1.7204e-02,  8.0785e-04,  2.9162e-03, -2.7915e-02,
        -1.5022e-03,  2.9167e-03, -8.4125e-04,  1.1723e-02, -8.8157e-03,
        -9.3153e-04])
tensor([[[[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
```

```
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],
```

```
[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


…,


[[[-0.0108, -0.0060, -0.0008],
  [-0.0071,  0.0081,  0.0087],
  [-0.0105,  0.0070,  0.0010]],

 [[ 0.0037,  0.0107,  0.0154],
  [ 0.0060,  0.0052,  0.0059],
  [ 0.0131,  0.0075,  0.0032]],

 [[-0.0023,  0.0052,  0.0026],
  [-0.0069, -0.0063, -0.0111],
  [-0.0040, -0.0057, -0.0104]],

 …,

 [[ 0.0026,  0.0040,  0.0072],
  [-0.0039, -0.0036,  0.0024],
  [-0.0098, -0.0068, -0.0053]],
```

```
 [[ 0.0199,  0.0160,  0.0131],
  [ 0.0131,  0.0096,  0.0072],
  [ 0.0121,  0.0137,  0.0098]],

 [[ 0.0045,  0.0086,  0.0098],
  [ 0.0039,  0.0044,  0.0059],
  [ 0.0075,  0.0063,  0.0068]]],


[[[-0.0500, -0.0446, -0.0269],
  [-0.0541, -0.0568, -0.0457],
  [-0.0466, -0.0522, -0.0508]],

 [[-0.0294, -0.0201, -0.0466],
  [-0.0165, -0.0119, -0.0301],
  [-0.0155, -0.0163, -0.0225]],

 [[-0.0286, -0.0285, -0.0444],
  [-0.0210, -0.0222, -0.0385],
  [-0.0143, -0.0166, -0.0282]],

 …,

 [[ 0.0109,  0.0206,  0.0221],
  [-0.0101,  0.0019,  0.0015],
  [-0.0169, -0.0042, -0.0060]],

 [[-0.0351, -0.0105, -0.0106],
  [-0.0313, -0.0125, -0.0082],
  [-0.0294, -0.0175, -0.0119]],

 [[-0.0242, -0.0099,  0.0018],
  [-0.0111, -0.0037,  0.0040],
  [-0.0025, -0.0032,  0.0011]]],


[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],
```

```
        …,

        [[ 0.0000,  0.0000,  0.0000],
         [ 0.0000,  0.0000,  0.0000],
         [ 0.0000,  0.0000,  0.0000]],

        [[ 0.0000,  0.0000,  0.0000],
         [ 0.0000,  0.0000,  0.0000],
         [ 0.0000,  0.0000,  0.0000]],

        [[ 0.0000,  0.0000,  0.0000],
         [ 0.0000,  0.0000,  0.0000],
         [ 0.0000,  0.0000,  0.0000]]]]])
tensor([ 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  9.1844e-03,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  1.9095e-02,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  3.4653e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -6.2631e-03,  0.0000e+00,  4.7640e-04,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  6.5681e-05,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -2.1018e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

```
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -3.3675e-06, -1.8812e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -6.3673e-03,  0.0000e+00,
         0.0000e+00,  5.5227e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         2.0520e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         3.4639e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  1.9750e-02,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -7.3860e-04,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  4.0753e-03,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  4.2494e-03,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -1.9783e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  3.3233e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  1.0852e-02, -6.2721e-02,  0.0000e+00])
tensor([[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[-2.8900e-02, -1.6708e-02,  3.5524e-03],
  [-7.6253e-04,  9.2040e-03,  9.9818e-04],
  [-3.5818e-02, -1.6863e-02, -6.1224e-03]],

 [[ 3.2232e-02, -2.0461e-02, -5.3317e-02],
  [ 2.3374e-02, -6.9490e-02, -6.4186e-02],
  [-7.6025e-02, -1.5416e-01, -1.2200e-01]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[-1.2717e-03,  9.6305e-04, -2.8084e-03],
  [-2.2459e-03, -2.4931e-03, -8.8902e-04],
  [ 1.6301e-03, -3.4514e-03, -5.5623e-04]],

 [[-2.7014e-03, -8.1252e-05,  5.0027e-04],
  [-9.7493e-04,  5.4748e-03,  9.8427e-03],
  [-1.3628e-02, -1.5087e-02, -1.0602e-02]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],
```

```
[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[-7.4781e-02, -5.2594e-02, -1.6594e-02],
  [-8.0055e-02, -5.7995e-02, -1.9606e-02],
  [-4.1765e-02, -3.4389e-02, -3.0778e-02]],
```

```
 [[-1.5734e-01, -1.7727e-01, -1.1266e-01],
  [-2.7185e-01, -3.1884e-01, -2.1481e-01],
  [-2.7271e-01, -2.9619e-01, -2.2274e-01]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
          …,

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]])
tensor([-5.6338e-02, -9.0150e-03,  0.0000e+00,  9.3909e-04,  8.5752e-03,
         1.7305e-02,  3.3061e-04,  1.4110e-02,  9.9216e-05,  0.0000e+00,
         3.3783e-06,  4.5621e-03,  8.4025e-03,  0.0000e+00, -2.5858e-03,
        -5.0647e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -1.9445e-05,  3.5511e-03,  0.0000e+00,
         1.0570e-03,  0.0000e+00,  1.4752e-02,  0.0000e+00,  3.4245e-04,
        -1.1266e-02,  8.3524e-03,  4.1548e-04,  0.0000e+00,  1.3340e-03,
        -3.2523e-03,  5.0329e-05,  3.6032e-02, -1.7651e-03,  2.5727e-03,
        -7.9922e-04,  2.4400e-03, -5.6327e-04,  5.4997e-03,  0.0000e+00,
        -2.8367e-02, -1.7441e-04,  0.0000e+00,  1.4868e-02,  0.0000e+00,
         0.0000e+00,  2.5571e-03, -5.9921e-03,  1.9877e-05, -9.0880e-03,
         0.0000e+00, -4.9972e-03,  0.0000e+00, -1.7223e-04,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  1.2356e-02,  0.0000e+00,  7.5334e-03,
         2.2585e-02, -5.5298e-05, -4.7096e-04,  1.6140e-02,  2.5816e-02,
         0.0000e+00,  0.0000e+00,  2.4362e-08,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  3.1392e-03, -7.1839e-04, -2.5434e-03, -2.4473e-03,
         0.0000e+00,  0.0000e+00,  9.4149e-05,  4.1258e-03, -8.4855e-04,
         0.0000e+00,  2.4320e-02,  3.7901e-03, -8.3338e-04,  0.0000e+00,
         4.7884e-02,  0.0000e+00,  9.6642e-04,  0.0000e+00,  2.4418e-04,
         0.0000e+00,  0.0000e+00,  3.6139e-02,  0.0000e+00, -1.1975e-05,
         7.3181e-02,  5.4117e-04,  3.2819e-02, -2.1007e-04,  0.0000e+00,
        -3.8350e-04, -3.7887e-02, -1.0341e-03, -6.2076e-04, -4.5357e-02,
        -1.0624e-02, -2.2407e-04,  0.0000e+00, -5.9381e-04,  1.7276e-03,
         8.1324e-04,  1.4786e-02,  8.6936e-05,  0.0000e+00, -3.8251e-03,
        -1.4399e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -5.4785e-04, -4.4306e-03,  0.0000e+00, -3.6164e-03,
         0.0000e+00,  0.0000e+00,  3.0238e-04, -1.0218e-02,  2.1667e-03,
         1.5804e-02,  0.0000e+00, -2.9234e-07,  0.0000e+00,  0.0000e+00,
         1.7316e-05, -4.2206e-02,  1.3033e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -8.1344e-02, -1.4746e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -2.8469e-02,  1.8053e-07,  0.0000e+00,
        -1.0958e-02,  0.0000e+00,  1.3474e-04, -4.6131e-08,  4.5558e-02,
        -5.2810e-03,  1.3484e-02,  0.0000e+00,  0.0000e+00, -4.1400e-02,
         0.0000e+00,  0.0000e+00, -2.7363e-04,  1.4444e-03,  1.1035e-02,
```

```
        -2.0432e-03,  0.0000e+00,  9.5167e-08,  0.0000e+00,  1.5480e-02,
        -1.1600e-02,  0.0000e+00,  7.9458e-04, -8.2308e-03,  2.5410e-03,
         1.8690e-07, -1.0316e-04, -3.3697e-04,  4.5195e-04, -7.1369e-05,
         0.0000e+00,  3.1550e-03,  5.7849e-03,  3.8853e-03, -1.0318e-02,
        -6.4737e-04,  0.0000e+00,  0.0000e+00, -8.9723e-02,  8.7797e-03,
        -4.0248e-03, -4.8877e-03,  3.3447e-02,  0.0000e+00, -1.3361e-09,
        -1.2771e-02, -1.6132e-02,  0.0000e+00,  1.8310e-02,  1.6758e-03,
         0.0000e+00,  3.7197e-02, -1.0644e-02,  0.0000e+00, -2.3382e-04,
         4.6489e-04,  3.5277e-03,  1.2993e-03, -8.4495e-05,  1.9488e-02,
        -1.1548e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  1.2223e-02,  0.0000e+00,  3.1486e-02,  1.3025e-04,
        -1.1740e-03,  1.4146e-02, -2.3575e-02,  6.4820e-03,  0.0000e+00,
        -5.9421e-03, -4.6381e-10,  0.0000e+00,  9.4749e-03, -7.0211e-05,
        -2.0409e-02,  0.0000e+00,  0.0000e+00, -2.5538e-02,  2.1326e-02,
         4.3827e-04, -3.3578e-02,  0.0000e+00, -1.4729e-02,  1.8731e-02,
         0.0000e+00,  1.4182e-02,  0.0000e+00, -1.7217e-06,  1.1788e-02,
        -3.2702e-07,  1.0623e-02,  9.4561e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -9.0064e-04,  5.1934e-02,  0.0000e+00,
         0.0000e+00,  2.1041e-02, -9.2811e-02, -1.5236e-03,  0.0000e+00,
        -9.5999e-04,  1.3650e-02,  2.0493e-02, -4.3103e-03,  1.3695e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  1.4229e-02,
        -6.6688e-03,  5.9800e-02,  2.0090e-04,  0.0000e+00, -2.4680e-04,
        -3.5157e-03,  2.3441e-04,  0.0000e+00,  1.6520e-02,  1.4440e-03,
         0.0000e+00,  5.7640e-03,  0.0000e+00,  1.0322e-04, -5.0005e-03,
         1.6980e-03, -1.8880e-04,  0.0000e+00,  3.6591e-08, -1.4222e-04,
         1.9879e-02,  1.2710e-07, -6.4248e-04,  0.0000e+00,  2.2655e-05,
         2.8428e-02, -4.8454e-03, -1.8383e-07,  5.4466e-02,  0.0000e+00,
         2.2818e-02,  1.9078e-03,  0.0000e+00,  7.0181e-03,  0.0000e+00,
         3.3189e-06, -1.4197e-03, -7.2676e-04, -1.1250e-02, -6.0865e-03,
        -2.7673e-05, -5.4406e-04,  4.2955e-06,  0.0000e+00, -3.2348e-05,
        -9.6245e-06,  2.7113e-07, -7.3974e-03,  4.9988e-02, -5.3285e-03,
         7.0826e-03,  0.0000e+00, -1.1678e-04, -7.4706e-02,  0.0000e+00,
         0.0000e+00, -4.4008e-02, -1.2563e-02,  0.0000e+00, -1.0975e-02,
         0.0000e+00,  0.0000e+00, -1.9885e-02, -2.3609e-02, -9.1990e-03,
         4.0070e-03,  4.3118e-03,  1.0602e-03,  6.1169e-08, -3.2833e-03,
         8.4479e-02,  8.2762e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  4.5159e-04, -8.3145e-04,  1.9859e-03, -1.5240e-03,
        -9.1024e-04,  1.2162e-03,  0.0000e+00, -5.6691e-03,  0.0000e+00,
         2.0257e-04, -1.0578e-04, -6.9744e-03,  6.5566e-08,  0.0000e+00,
        -2.5844e-02, -3.0695e-02,  5.0950e-08,  4.8289e-05,  3.3010e-02,
         0.0000e+00, -1.2594e-03,  0.0000e+00, -5.7632e-02,  4.4120e-03,
         6.0497e-05,  5.0059e-03, -1.9221e-03, -5.9545e-07,  8.5892e-04,
         0.0000e+00, -7.0032e-02,  0.0000e+00,  0.0000e+00])
tensor([[[[ 4.0419e-04, -7.8704e-03, -8.3116e-03],
          [ 4.0319e-03,  4.8139e-03, -1.5262e-03],
          [ 4.8042e-03,  7.7656e-03,  2.1408e-03]],

         [[ 0.0000e+00,  0.0000e+00,  1.7480e-03],
```

```
  [ 0.0000e+00,  0.0000e+00,  7.0987e-04],
  [ 9.7465e-08,  0.0000e+00,  3.9371e-04]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 3.7610e-04,  0.0000e+00,  1.4033e-03],
  [ 1.3512e-04,  0.0000e+00,  0.0000e+00],
  [ 1.4071e-03,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],
```

```
[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
          …,

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]])
tensor([ 2.9568e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  7.1710e-03,  0.0000e+00,
          0.0000e+00,  0.0000e+00, -3.5623e-03, -1.6357e-02, -3.4200e-01,
         -3.0414e-02,  1.8156e-04, -1.4277e-02,  0.0000e+00,  0.0000e+00,
         -1.6721e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          3.0371e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.0783e-01,
          7.5683e-02, -1.4127e-02, -5.6270e-03,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         -1.3320e-02,  0.0000e+00,  0.0000e+00, -1.5722e-01,  1.5969e-01,
          0.0000e+00, -3.2457e-05, -4.3113e-07,  0.0000e+00, -1.7757e-05,
          0.0000e+00,  0.0000e+00, -7.7458e-05,  0.0000e+00,  5.2803e-05,
          0.0000e+00,  0.0000e+00, -4.1786e-03, -1.4483e-01,  0.0000e+00,
          3.8039e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  2.0258e-01,
          0.0000e+00, -1.2778e-02,  0.0000e+00,  3.1749e-02,  9.1066e-06,
          0.0000e+00,  0.0000e+00, -3.3915e-02,  7.1959e-02,  0.0000e+00,
          0.0000e+00, -6.0687e-02, -2.0349e-02,  0.0000e+00,  0.0000e+00,
          4.2059e-03,  8.3853e-03,  2.2889e-03,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -6.3347e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  7.1341e-03,  0.0000e+00, -9.5800e-02, -1.2722e-02,
         -7.6463e-09,  0.0000e+00,  0.0000e+00,  0.0000e+00, -7.3177e-03,
          0.0000e+00,  0.0000e+00,  1.0266e-01,  0.0000e+00,  0.0000e+00,
          4.1367e-02,  0.0000e+00, -5.3607e-03,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  6.2677e-03,  1.2484e-01,
         -2.4693e-03,  0.0000e+00, -6.0137e-02,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  9.1018e-04,  0.0000e+00,
          1.8260e-02,  0.0000e+00,  2.2131e-03,  0.0000e+00, -3.2143e-03,
          0.0000e+00,  0.0000e+00, -1.3542e-01,  1.1997e-02,  1.2927e-01,
         -8.5665e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         -8.1878e-02,  0.0000e+00,  1.2670e-04,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  2.5048e-02,  0.0000e+00,  0.0000e+00, -1.7097e-02,
          0.0000e+00, -7.1386e-05,  3.3875e-02,  0.0000e+00,  0.0000e+00,
          1.0129e-01,  0.0000e+00,  0.0000e+00,  0.0000e+00,  1.8722e-04,
```

```
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -3.1407e-07, -4.3553e-01,  0.0000e+00, -2.8048e-02,  2.7147e-02,
        -1.6033e-02,  0.0000e+00, -6.9799e-04, -2.2119e-02,  7.1994e-03,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  1.9995e-03,  0.0000e+00,  7.8621e-03,  0.0000e+00,
         0.0000e+00,  1.7436e-03, -5.8271e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -4.7134e-02, -1.4977e-01,  0.0000e+00, -2.4651e-01,
        -3.7069e-04, -4.7270e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -1.0093e-01,  0.0000e+00,  0.0000e+00,  3.5509e-02,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  1.5846e-04,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  8.3400e-02,  0.0000e+00,
         0.0000e+00,  9.2112e-02, -7.2432e-02,  9.0048e-04,  0.0000e+00,
         2.7801e-02,  0.0000e+00,  1.5156e-01,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -5.6683e-03,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         5.1461e-02, -3.7993e-04,  0.0000e+00,  9.5046e-02,  0.0000e+00,
         0.0000e+00, -1.9344e-02, -8.7084e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00])
tensor([[ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        …,
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [-6.6649e-08,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00]])
tensor([-0.0018,  0.0000,  0.0000,  …,  0.0000, -0.0005,  0.0000])
tensor([[ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
         -3.9582e-08,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        …,
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00],
        [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
          0.0000e+00,  0.0000e+00]])
tensor([-5.5688e-07,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
         0.0000e+00,  0.0000e+00])
tensor([[1.3986e-07, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
```

```
         0.0000e+00],
        [3.0022e-10, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00],
        [3.7926e-11, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00],
        …,
        [5.9162e-16, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00],
        [5.8338e-16, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00],
        [5.8807e-16, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00]])
tensor([ 5.0464e-03,  1.1423e-02, -1.8006e-02,  8.2110e-03,  2.3420e-02,
         7.7866e-03,  2.9031e-02, -2.2212e-02,  4.8927e-02,  2.1843e-02,
        -4.1180e-02,  1.1076e-02, -1.0762e-01, -6.0137e-02,  4.1234e-03,
         6.0365e-02,  2.2849e-02,  2.0278e-02,  3.4257e-02, -6.8189e-02,
         2.0604e-04,  8.7297e-06,  8.6560e-06,  8.6705e-06,  8.7572e-06,
         8.6504e-06,  8.6855e-06,  8.6915e-06,  8.7104e-06,  8.7154e-06,
         8.6874e-06,  8.6480e-06,  8.6909e-06,  8.6648e-06,  8.6586e-06,
         8.6624e-06,  8.6553e-06,  8.7134e-06,  8.6552e-06,  8.7135e-06,
         8.6498e-06,  8.6421e-06,  8.7291e-06,  8.6513e-06,  8.7378e-06,
         8.7064e-06,  8.6645e-06,  8.6550e-06,  8.6795e-06,  8.6793e-06,
         8.7137e-06,  8.6579e-06,  8.7043e-06,  8.6746e-06,  8.6856e-06,
         8.6923e-06,  8.6578e-06,  8.7144e-06,  8.6466e-06,  8.6690e-06,
         8.7074e-06,  8.6808e-06,  8.7389e-06,  8.7323e-06,  8.7373e-06,
         8.6824e-06,  8.7422e-06,  8.6898e-06,  8.7392e-06,  8.7419e-06,
         8.6725e-06,  8.7170e-06,  8.6598e-06,  8.6619e-06,  8.6851e-06,
         8.7601e-06,  8.7138e-06,  8.6547e-06,  8.6473e-06,  8.7299e-06,
         8.6546e-06,  8.6553e-06,  8.7408e-06,  8.7025e-06,  8.6617e-06,
         8.7377e-06,  8.6843e-06,  8.7188e-06,  8.7334e-06,  8.6752e-06,
         8.6491e-06,  8.6566e-06,  8.7152e-06,  8.6604e-06,  8.6543e-06,
         8.7079e-06,  8.7076e-06,  8.6473e-06,  8.7441e-06,  8.6922e-06,
         8.6947e-06,  8.6589e-06,  8.7227e-06,  8.6625e-06,  8.6561e-06,
         8.7350e-06,  8.6680e-06,  8.6536e-06,  8.6493e-06,  8.7379e-06,
         8.6681e-06,  8.7210e-06,  8.6772e-06,  8.7696e-06,  8.6735e-06,
         8.6493e-06,  8.6765e-06,  8.6516e-06,  8.7375e-06,  8.6642e-06,
         8.7336e-06,  8.6455e-06,  8.7409e-06,  8.7387e-06,  8.6506e-06,
         8.6505e-06,  8.6956e-06,  8.6514e-06,  8.7009e-06,  8.6602e-06,
         8.6521e-06,  8.7499e-06,  8.6473e-06,  8.7032e-06,  8.6683e-06,
         8.6760e-06,  8.6760e-06,  8.6431e-06,  8.7219e-06,  8.6399e-06,
         8.7205e-06,  8.7354e-06,  8.6882e-06,  8.6485e-06,  8.6853e-06,
         8.6951e-06,  8.6754e-06,  8.7258e-06,  8.6429e-06,  8.7203e-06,
         8.7069e-06,  8.7166e-06,  8.6884e-06,  8.7469e-06,  8.6647e-06,
         8.6493e-06,  8.6588e-06,  8.6759e-06,  8.7314e-06,  8.6508e-06,
         8.7106e-06,  8.7130e-06,  8.6389e-06,  8.6728e-06,  8.7335e-06,
         8.7266e-06,  8.6674e-06,  8.6468e-06,  8.7354e-06,  8.7049e-06,
         8.7124e-06,  8.7127e-06,  8.7326e-06,  8.7419e-06,  8.6704e-06,
         8.6521e-06,  8.6724e-06,  8.6439e-06,  8.7154e-06,  8.7177e-06,
```

```
8.7104e-06,  8.6698e-06,  8.6502e-06,  8.6748e-06,  8.7100e-06,
8.7271e-06,  8.6511e-06,  8.7365e-06,  8.7012e-06,  8.7200e-06,
8.6497e-06,  8.6709e-06,  8.7035e-06,  8.6789e-06,  8.6466e-06,
8.6634e-06,  8.6543e-06,  8.6892e-06,  8.6562e-06,  8.6525e-06,
8.6621e-06,  8.7331e-06,  8.6694e-06,  8.7305e-06,  8.7280e-06,
8.7211e-06,  8.6796e-06,  8.7136e-06,  8.6491e-06,  8.7023e-06,
8.6471e-06,  8.7085e-06,  8.6856e-06,  8.6584e-06,  8.7403e-06,
8.6426e-06,  8.7162e-06,  8.7187e-06,  8.6855e-06,  8.7437e-06,
8.7403e-06,  8.6402e-06,  8.6403e-06,  8.6542e-06,  8.7356e-06,
8.7163e-06,  8.6579e-06,  8.6604e-06,  8.7068e-06,  8.6416e-06,
8.6792e-06,  8.7270e-06,  8.6649e-06,  8.7015e-06,  8.6739e-06,
8.6983e-06,  8.6605e-06,  8.7203e-06,  8.6499e-06,  8.6765e-06,
8.6451e-06,  8.6372e-06,  8.6684e-06,  8.6737e-06,  8.6701e-06,
8.6786e-06,  8.6428e-06,  8.6616e-06,  8.6528e-06,  8.7044e-06,
8.7123e-06,  8.6883e-06,  8.7189e-06,  8.7059e-06,  8.6599e-06,
8.6922e-06,  8.6489e-06,  8.6675e-06,  8.6630e-06,  8.6461e-06,
8.6664e-06,  8.6616e-06,  8.6677e-06,  8.6752e-06,  8.6681e-06,
8.7386e-06,  8.6882e-06,  8.6720e-06,  8.6542e-06,  8.6968e-06,
8.6430e-06,  8.6628e-06,  8.6864e-06,  8.6364e-06,  8.6749e-06,
8.7308e-06,  8.6574e-06,  8.7163e-06,  8.6470e-06,  8.6879e-06,
8.6560e-06,  8.6928e-06,  8.6489e-06,  8.7194e-06,  8.7013e-06,
8.8293e-06,  8.6491e-06,  8.7058e-06,  8.6458e-06,  8.7058e-06,
8.8275e-06,  8.7378e-06,  8.6426e-06,  8.6575e-06,  8.7120e-06,
8.6513e-06,  8.6983e-06,  8.6406e-06,  8.6552e-06,  8.7365e-06,
8.6656e-06,  8.7128e-06,  8.7383e-06,  8.7296e-06,  8.7362e-06,
8.7397e-06,  8.6913e-06,  8.6594e-06,  8.6917e-06,  8.6638e-06,
8.6516e-06,  8.7305e-06,  8.7427e-06,  8.6456e-06,  8.7320e-06,
8.6659e-06,  8.6670e-06,  8.6673e-06,  8.6438e-06,  8.6630e-06,
8.6531e-06,  8.6851e-06,  8.7104e-06,  8.6500e-06,  8.6782e-06,
8.6751e-06,  8.7180e-06,  8.6856e-06,  8.6735e-06,  8.7206e-06,
8.7335e-06,  8.7291e-06,  8.7389e-06,  8.6869e-06,  8.6821e-06,
8.6890e-06,  8.7358e-06,  8.6445e-06,  8.6622e-06,  8.7362e-06,
8.6472e-06,  8.6897e-06,  8.7079e-06,  8.7275e-06,  8.7992e-06,
8.6567e-06,  8.7195e-06,  8.6520e-06,  8.6935e-06,  8.6527e-06,
8.7450e-06,  8.6773e-06,  8.7396e-06,  8.6922e-06,  8.7013e-06,
8.7312e-06,  8.7246e-06,  8.7189e-06,  8.6658e-06,  8.6478e-06,
8.6703e-06,  8.6576e-06,  8.7746e-06,  8.6697e-06,  8.6786e-06,
8.6640e-06,  8.7103e-06,  8.7463e-06,  8.6479e-06,  8.6378e-06,
8.7404e-06,  8.6660e-06,  8.7257e-06,  8.7393e-06,  8.7053e-06,
8.7399e-06,  8.6899e-06,  8.8203e-06,  8.6700e-06,  8.6706e-06,
8.6952e-06,  8.6587e-06,  8.7205e-06,  8.7226e-06,  8.6506e-06,
8.6840e-06,  8.6510e-06,  8.7167e-06,  8.7038e-06,  8.6832e-06,
8.6455e-06,  8.7120e-06,  8.6860e-06,  8.6870e-06,  8.6736e-06,
8.6514e-06,  8.6557e-06,  8.6465e-06,  8.6599e-06,  8.6484e-06,
8.6612e-06,  8.7041e-06,  8.7200e-06,  8.6671e-06,  8.6611e-06,
8.6929e-06,  8.6481e-06,  8.7422e-06,  8.7447e-06,  8.6570e-06,
8.7078e-06,  8.7255e-06,  8.6899e-06,  8.7157e-06,  8.6498e-06,
8.6782e-06,  8.6526e-06,  8.6635e-06,  8.6890e-06,  8.7015e-06,
```

```
8.7335e-06,  8.6683e-06,  8.6922e-06,  8.6485e-06,  8.6660e-06,
8.6465e-06,  8.6578e-06,  8.7403e-06,  8.6414e-06,  8.6726e-06,
8.6510e-06,  8.6441e-06,  8.6703e-06,  8.6504e-06,  8.6462e-06,
8.6344e-06,  8.7077e-06,  8.7315e-06,  8.6648e-06,  8.6414e-06,
8.7278e-06,  8.7163e-06,  8.6557e-06,  8.7113e-06,  8.6489e-06,
8.6663e-06,  8.6598e-06,  8.7160e-06,  8.7207e-06,  8.6949e-06,
8.6637e-06,  8.6705e-06,  8.6997e-06,  8.6526e-06,  8.6472e-06,
8.6714e-06,  8.6430e-06,  8.6572e-06,  8.7399e-06,  8.7354e-06,
8.6740e-06,  8.6430e-06,  8.6500e-06,  8.7257e-06,  8.7377e-06,
8.6798e-06,  8.7051e-06,  8.7157e-06,  8.6551e-06,  8.6882e-06,
8.7205e-06,  8.6618e-06,  8.6644e-06,  8.7323e-06,  8.7242e-06,
8.6499e-06,  8.6873e-06,  8.6648e-06,  8.6442e-06,  8.6425e-06,
8.6583e-06,  8.6744e-06,  8.6739e-06,  8.6567e-06,  8.6976e-06,
8.6515e-06,  8.7196e-06,  8.6477e-06,  8.6738e-06,  8.7060e-06,
8.8619e-06,  8.6559e-06,  8.6696e-06,  8.6473e-06,  8.6582e-06,
8.6801e-06,  8.7360e-06,  8.7028e-06,  8.7241e-06,  8.6567e-06,
8.6587e-06,  8.7026e-06,  8.6455e-06,  8.6484e-06,  8.6439e-06,
8.7117e-06,  8.6874e-06,  8.6697e-06,  8.6522e-06,  8.6533e-06,
8.6508e-06,  8.6550e-06,  8.7266e-06,  8.7033e-06,  8.6517e-06,
8.6488e-06,  8.6486e-06,  8.6606e-06,  8.6497e-06,  8.7071e-06,
8.7110e-06,  8.7149e-06,  8.6998e-06,  8.6639e-06,  8.7155e-06,
8.7000e-06,  8.6536e-06,  8.8052e-06,  8.6375e-06,  8.6890e-06,
8.6681e-06,  8.6417e-06,  8.7255e-06,  8.6927e-06,  8.7416e-06,
8.6458e-06,  8.6540e-06,  8.6401e-06,  8.6518e-06,  8.6740e-06,
8.6884e-06,  8.7270e-06,  8.7243e-06,  8.6416e-06,  8.6677e-06,
8.6924e-06,  8.6417e-06,  8.6705e-06,  8.6457e-06,  8.6783e-06,
8.6654e-06,  8.6624e-06,  8.6540e-06,  8.7127e-06,  8.7541e-06,
8.7060e-06,  8.6499e-06,  8.6663e-06,  8.7015e-06,  8.6881e-06,
8.7233e-06,  8.7093e-06,  8.6638e-06,  8.6518e-06,  8.6490e-06,
8.7299e-06,  8.6870e-06,  8.6502e-06,  8.6691e-06,  8.6526e-06,
8.6447e-06,  8.6449e-06,  8.6555e-06,  8.6443e-06,  8.6873e-06,
8.7299e-06,  8.7401e-06,  8.6536e-06,  8.7243e-06,  8.6495e-06,
8.6698e-06,  8.7068e-06,  8.7440e-06,  8.6771e-06,  8.6440e-06,
8.6898e-06,  8.7397e-06,  8.6450e-06,  8.6622e-06,  8.6679e-06,
8.6554e-06,  8.6800e-06,  8.7178e-06,  8.7262e-06,  8.6541e-06,
8.6339e-06,  8.7444e-06,  8.6580e-06,  8.6619e-06,  8.6428e-06,
8.7178e-06,  8.6395e-06,  8.6444e-06,  8.7430e-06,  8.6686e-06,
8.7405e-06,  8.6506e-06,  8.7085e-06,  8.6554e-06,  8.7201e-06,
8.6812e-06,  8.6867e-06,  8.6595e-06,  8.6649e-06,  8.6795e-06,
8.6485e-06,  8.6482e-06,  8.7161e-06,  8.7261e-06,  8.6681e-06,
8.6735e-06,  8.6519e-06,  8.6729e-06,  8.6719e-06,  8.6815e-06,
8.6866e-06,  8.6570e-06,  8.6609e-06,  8.6500e-06,  8.6526e-06,
8.6426e-06,  8.7142e-06,  8.6790e-06,  8.7283e-06,  8.6842e-06,
8.6689e-06,  8.6817e-06,  8.6797e-06,  8.6526e-06,  8.6453e-06,
8.7201e-06,  8.6435e-06,  8.6586e-06,  8.6564e-06,  8.6925e-06,
8.6583e-06,  8.6423e-06,  8.7168e-06,  8.7387e-06,  8.7053e-06,
8.7323e-06,  8.7424e-06,  8.6513e-06,  8.7186e-06,  8.6419e-06,
8.7432e-06,  8.6670e-06,  8.6593e-06,  8.6474e-06,  8.7182e-06,
```

```
8.7145e-06,  8.6613e-06,  8.6478e-06,  8.6437e-06,  8.6380e-06,
8.7077e-06,  8.6756e-06,  8.6875e-06,  8.7358e-06,  8.6698e-06,
8.6577e-06,  8.7330e-06,  8.6701e-06,  8.6631e-06,  8.6536e-06,
8.7142e-06,  8.6585e-06,  8.7309e-06,  8.7175e-06,  8.6448e-06,
8.6442e-06,  8.6704e-06,  8.6792e-06,  8.6544e-06,  8.6513e-06,
8.6478e-06,  8.6607e-06,  8.6783e-06,  8.6482e-06,  8.6745e-06,
8.7229e-06,  8.6693e-06,  8.7397e-06,  8.6644e-06,  8.6422e-06,
8.6677e-06,  8.7049e-06,  8.6457e-06,  8.7397e-06,  8.6626e-06,
8.6999e-06,  8.6890e-06,  8.6935e-06,  8.6635e-06,  8.6717e-06,
8.7442e-06,  8.6781e-06,  8.7122e-06,  8.6498e-06,  8.6761e-06,
8.7261e-06,  8.7198e-06,  8.6949e-06,  8.6886e-06,  8.7347e-06,
8.6479e-06,  8.6486e-06,  8.7238e-06,  8.6616e-06,  8.7152e-06,
8.7089e-06,  8.6642e-06,  8.6435e-06,  8.6576e-06,  8.6838e-06,
8.6428e-06,  8.6603e-06,  8.6442e-06,  8.6516e-06,  8.6541e-06,
8.6524e-06,  8.6957e-06,  8.6567e-06,  8.8018e-06,  8.6776e-06,
8.6696e-06,  8.6721e-06,  8.7352e-06,  8.6679e-06,  8.7279e-06,
8.7294e-06,  8.6486e-06,  8.6578e-06,  8.6850e-06,  8.6796e-06,
8.7189e-06,  8.7267e-06,  8.6613e-06,  8.6496e-06,  8.6605e-06,
8.6995e-06,  8.7385e-06,  8.6922e-06,  8.7125e-06,  8.7191e-06,
8.7056e-06,  8.6482e-06,  8.7260e-06,  8.6951e-06,  8.7376e-06,
8.7085e-06,  8.6545e-06,  8.6962e-06,  8.6606e-06,  8.7129e-06,
8.6699e-06,  8.6959e-06,  8.6693e-06,  8.7180e-06,  8.6798e-06,
8.6516e-06,  8.7174e-06,  8.7292e-06,  8.6913e-06,  8.8034e-06,
8.6844e-06,  8.6544e-06,  8.6496e-06,  8.7438e-06,  8.7040e-06,
8.6767e-06,  8.6426e-06,  8.6472e-06,  8.7175e-06,  8.6564e-06,
8.7294e-06,  8.7344e-06,  8.6411e-06,  8.6764e-06,  8.6983e-06,
8.7095e-06,  8.7360e-06,  8.6459e-06,  8.7369e-06,  8.6814e-06,
8.6954e-06,  8.6719e-06,  8.6749e-06,  8.6597e-06,  8.6589e-06,
8.7122e-06,  8.6802e-06,  8.7370e-06,  8.6459e-06,  8.6757e-06,
8.6453e-06,  8.6654e-06,  8.6542e-06,  8.6545e-06,  8.6497e-06,
8.7299e-06,  8.6495e-06,  8.6570e-06,  8.7373e-06,  8.7377e-06,
8.6535e-06,  8.6432e-06,  8.7736e-06,  8.6568e-06,  8.6419e-06,
8.7178e-06,  8.6525e-06,  8.6599e-06,  8.6667e-06,  8.6611e-06,
8.6809e-06,  8.6868e-06,  8.6453e-06,  8.7426e-06,  8.6453e-06,
8.6414e-06,  8.7135e-06,  8.7306e-06,  8.7279e-06,  8.6814e-06,
8.6606e-06,  8.7026e-06,  8.6436e-06,  8.6579e-06,  8.7194e-06,
8.6662e-06,  8.7082e-06,  8.6438e-06,  8.6642e-06,  8.7375e-06,
8.7055e-06,  8.7277e-06,  8.6670e-06,  8.7027e-06,  8.6411e-06,
8.7210e-06,  8.6609e-06,  8.6468e-06,  8.8157e-06,  8.7718e-06,
8.6471e-06,  8.7407e-06,  8.6435e-06,  8.7423e-06,  8.6804e-06,
8.7361e-06,  8.7312e-06,  8.6744e-06,  8.7356e-06,  8.6537e-06,
8.6651e-06,  8.7550e-06,  8.7262e-06,  8.7223e-06,  8.7252e-06,
8.6730e-06,  8.7060e-06,  8.6632e-06,  8.6671e-06,  8.7104e-06,
8.6687e-06,  8.6530e-06,  8.6625e-06,  8.6612e-06,  8.7327e-06,
8.6734e-06,  8.6462e-06,  8.6585e-06,  8.6983e-06,  8.6903e-06,
8.6505e-06,  8.6877e-06,  8.7402e-06,  8.6547e-06,  8.7407e-06,
8.7457e-06,  8.6543e-06,  8.6424e-06,  8.6471e-06,  8.6809e-06,
8.6472e-06,  8.6514e-06,  8.7183e-06,  8.7074e-06,  8.6924e-06,
```

```
         8.6790e-06,  8.7111e-06,  8.6531e-06,  8.7195e-06,  8.7386e-06,
         8.6869e-06,  8.6505e-06,  8.6739e-06,  8.7201e-06,  8.7147e-06,
         8.7181e-06,  8.6840e-06,  8.7339e-06,  8.6797e-06,  8.6593e-06,
         8.7291e-06,  8.7261e-06,  8.7278e-06,  8.7194e-06,  8.6510e-06,
         8.6850e-06,  8.7347e-06,  8.6489e-06,  8.6446e-06,  8.6526e-06,
         8.6928e-06,  8.7143e-06,  8.7183e-06,  8.7395e-06,  8.7340e-06,
         8.7314e-06,  8.7093e-06,  8.7301e-06,  8.7110e-06,  8.7275e-06,
         8.7156e-06,  8.6539e-06,  8.7815e-06,  8.6492e-06,  8.6546e-06,
         8.7177e-06,  8.7391e-06,  8.6652e-06,  8.6715e-06,  8.6488e-06,
         8.6590e-06,  8.6589e-06,  8.6478e-06,  8.7888e-06,  8.7079e-06,
         8.7368e-06,  8.6992e-06,  8.7039e-06,  8.7169e-06,  8.7416e-06,
         8.6836e-06,  8.6407e-06,  8.6790e-06,  8.7143e-06,  8.6738e-06,
         8.6641e-06,  8.7153e-06,  8.6515e-06,  8.7278e-06,  8.7177e-06,
         8.7165e-06,  8.7356e-06,  8.6492e-06,  8.6580e-06,  8.6493e-06,
         8.6637e-06,  8.6601e-06,  8.6548e-06,  8.6763e-06,  8.6527e-06,
         8.7055e-06,  8.6926e-06,  8.7272e-06,  8.7445e-06,  8.6586e-06,
         8.6511e-06,  8.6667e-06,  8.6486e-06,  8.6783e-06,  8.7279e-06,
         8.6681e-06,  8.6773e-06,  8.7411e-06,  8.6423e-06,  8.6800e-06,
         8.7227e-06,  8.6486e-06,  8.6590e-06,  8.6661e-06,  8.7078e-06,
         8.6668e-06,  8.7113e-06,  8.7103e-06,  8.6673e-06,  8.7324e-06])
end of p.grad

False
Epoch 8 finished
Epoch [8/10], Loss: 1.1430
************************************************************

Learning rate: 0.001

another way to print learning rate:
tensor([[[[-1.5292e-03, -9.7089e-04, -1.0638e-03,  …, -2.6307e-03,
           -2.7360e-03, -2.1407e-03],
          [-1.3223e-03, -7.6802e-04, -6.1750e-04,  …, -2.5085e-03,
           -2.7198e-03, -2.7608e-03],
          [-1.0300e-03, -9.0660e-04, -5.6357e-04,  …, -1.0394e-03,
           -1.7533e-03, -2.5209e-03],
          …,
          [-4.0183e-03, -4.3149e-03, -4.3882e-03,  …, -3.6403e-03,
           -3.7786e-03, -3.5852e-03],
          [-4.3041e-03, -4.4066e-03, -4.7193e-03,  …, -3.4056e-03,
           -4.0268e-03, -3.7253e-03],
          [-3.7332e-03, -4.0296e-03, -4.8343e-03,  …, -4.1741e-03,
           -3.9575e-03, -3.6904e-03]],

         [[ 2.8837e-03,  3.7153e-03,  3.8532e-03,  …,  7.9224e-04,
            7.1954e-04,  1.2225e-03],
          [ 3.4469e-03,  4.0741e-03,  4.4786e-03,  …,  5.7386e-04,
            3.8875e-04,  3.3617e-04],
```

```
  [ 3.3650e-03,  3.5089e-03,  3.9911e-03,  …,  1.4317e-03,
    9.4200e-04,  6.8973e-04],
  …,
  [ 2.6340e-04,  1.0886e-04, -2.2031e-04,  …,  4.1801e-04,
    4.3746e-04,  7.4368e-04],
  [-3.6111e-05, -1.8460e-04, -5.6531e-04,  …,  1.1409e-03,
    4.9517e-04,  4.1714e-04],
  [-2.8813e-04, -4.8097e-04, -8.3821e-04,  …, -5.1342e-05,
    1.0476e-04, -4.0716e-05]],

 [[ 4.5598e-03,  5.1278e-03,  5.3507e-03,  …,  1.3817e-03,
    1.1267e-03,  1.3697e-03],
  [ 4.8554e-03,  5.4096e-03,  5.8252e-03,  …,  1.1904e-03,
    6.4530e-04,  1.7191e-04],
  [ 4.9842e-03,  4.8153e-03,  5.2245e-03,  …,  1.9280e-03,
    9.7523e-04,  4.0836e-04],
  …,
  [ 1.1302e-03,  8.4113e-04,  5.8419e-04,  …,  5.7030e-04,
    4.2485e-04,  4.8936e-04],
  [ 8.2951e-04,  5.5216e-04,  9.9272e-05,  …,  1.2929e-03,
    6.5072e-04,  2.6173e-04],
  [ 3.6786e-05, -7.6553e-05, -3.4915e-04,  …,  1.4828e-04,
    1.1541e-04, -2.2463e-04]]],


[[[ 2.0158e-02,  2.1238e-02,  2.6518e-02,  …,  1.9513e-02,
    2.0866e-02,  2.4378e-02],
  [ 2.4095e-02,  2.2410e-02,  2.4330e-02,  …,  2.1139e-02,
    2.0482e-02,  2.2696e-02],
  [ 2.2601e-02,  2.2503e-02,  2.6327e-02,  …,  1.7016e-02,
    1.8175e-02,  1.9311e-02],
  …,
  [ 2.2387e-02,  2.3196e-02,  2.7262e-02,  …,  1.9748e-02,
    1.9486e-02,  1.9456e-02],
  [ 2.2285e-02,  2.1780e-02,  2.3011e-02,  …,  2.2135e-02,
    2.1008e-02,  2.0431e-02],
  [ 2.0297e-02,  2.1796e-02,  2.3966e-02,  …,  2.3537e-02,
    1.9595e-02,  2.2279e-02]],

 [[ 3.0724e-02,  3.1567e-02,  3.8409e-02,  …,  3.1583e-02,
    3.3301e-02,  3.7286e-02],
  [ 3.5680e-02,  3.3134e-02,  3.4615e-02,  …,  3.3892e-02,
    3.3558e-02,  3.4760e-02],
  [ 3.4697e-02,  3.4219e-02,  3.7257e-02,  …,  3.0409e-02,
    3.2845e-02,  3.3582e-02],
  …,
  [ 3.3849e-02,  3.4470e-02,  4.1740e-02,  …,  3.5318e-02,
    3.4569e-02,  3.3550e-02],
```

```
  [ 3.6703e-02,  3.4045e-02,  3.6700e-02,  …,  3.8222e-02,
    3.6017e-02,  3.4846e-02],
  [ 3.5070e-02,  3.5084e-02,  3.8559e-02,  …,  4.1155e-02,
    3.6711e-02,  3.8311e-02]],

 [[ 3.2965e-02,  3.3506e-02,  4.0219e-02,  …,  3.4788e-02,
    3.6314e-02,  3.8807e-02],
  [ 3.7072e-02,  3.3878e-02,  3.6186e-02,  …,  3.6736e-02,
    3.5509e-02,  3.5923e-02],
  [ 3.6329e-02,  3.4566e-02,  3.8159e-02,  …,  3.4207e-02,
    3.5051e-02,  3.4111e-02],
  …,
  [ 3.3366e-02,  3.4963e-02,  4.1166e-02,  …,  3.4818e-02,
    3.2820e-02,  3.1387e-02],
  [ 3.6564e-02,  3.4806e-02,  3.6435e-02,  …,  3.6677e-02,
    3.3936e-02,  3.2311e-02],
  [ 3.5711e-02,  3.5669e-02,  3.9000e-02,  …,  3.7787e-02,
    3.3545e-02,  3.3899e-02]]],


[[[-7.1214e-03, -7.4443e-03, -7.4045e-03,  …, -4.3968e-03,
   -3.6520e-03, -3.3845e-03],
  [-6.9154e-03, -7.0242e-03, -7.2614e-03,  …, -4.6447e-03,
   -4.4852e-03, -3.4310e-03],
  [-7.2437e-03, -6.9195e-03, -6.3548e-03,  …, -5.1432e-03,
   -3.8746e-03, -3.9234e-03],
  …,
  [-3.3963e-03, -3.5310e-03, -3.3700e-03,  …,  1.3285e-03,
    3.0371e-03,  2.6277e-03],
  [-3.4096e-03, -3.1932e-03, -2.7344e-03,  …,  2.7056e-03,
    3.6258e-03,  3.6222e-03],
  [-2.9067e-03, -2.6296e-03, -1.9049e-03,  …,  3.9113e-03,
    5.0196e-03,  4.6542e-03]],

 [[-3.1385e-03, -3.5278e-03, -3.2578e-03,  …, -5.8666e-04,
    3.8098e-04,  7.4493e-04],
  [-2.6374e-03, -2.6633e-03, -2.5923e-03,  …, -8.0260e-04,
   -2.8042e-05,  8.4668e-04],
  [-2.3769e-03, -2.3834e-03, -1.8582e-03,  …, -9.4573e-04,
    7.1662e-04,  4.0015e-04],
  …,
  [ 5.1295e-04,  4.9660e-04,  2.1249e-04,  …,  4.1205e-03,
    5.3462e-03,  5.3903e-03],
  [-1.9965e-04,  6.7761e-04,  1.0738e-03,  …,  5.2436e-03,
    6.1195e-03,  6.2972e-03],
  [ 6.5093e-04,  1.1774e-03,  1.7417e-03,  …,  6.3331e-03,
    7.6286e-03,  7.1702e-03]],
```

```
 [[-1.5889e-03, -1.7973e-03, -1.2896e-03,  …,  7.2510e-04,
    1.0486e-03,  1.7538e-03],
  [-7.8882e-04, -7.6301e-04, -8.4776e-04,  …,  1.2509e-04,
    8.3555e-04,  1.8456e-03],
  [-5.5304e-04, -4.8375e-04, -8.5182e-05,  …, -2.9556e-04,
    1.3438e-03,  1.3713e-03],
  …,
  [ 2.9801e-03,  2.4718e-03,  2.0248e-03,  …,  4.1427e-03,
    4.8742e-03,  4.6263e-03],
  [ 1.8301e-03,  2.1004e-03,  2.0545e-03,  …,  4.9080e-03,
    5.5353e-03,  5.4700e-03],
  [ 2.2479e-03,  2.1740e-03,  2.4252e-03,  …,  5.9023e-03,
    6.6853e-03,  6.2573e-03]]],


…,


[[[ 5.5411e-04,  3.7044e-04, -3.8301e-06,  …,  1.5732e-03,
    2.8633e-03,  2.9228e-03],
  [-9.8759e-04, -8.6551e-04, -4.6570e-04,  …,  1.0406e-03,
    2.1118e-03,  2.7794e-03],
  [ 4.2041e-04, -6.4397e-04, -1.9154e-04,  …,  2.9391e-03,
    3.0564e-03,  3.2442e-03],
  …,
  [ 4.1651e-04, -6.7598e-05,  2.2772e-04,  …, -2.3336e-03,
   -2.8581e-03, -1.6523e-03],
  [ 2.2247e-03,  1.8109e-03,  1.0768e-03,  …, -3.4657e-03,
   -2.6767e-03, -1.5886e-03],
  [ 2.3028e-03,  2.6281e-03,  1.2728e-03,  …, -3.3442e-03,
   -2.4273e-03, -1.8788e-03]],

 [[-2.1770e-03, -2.0972e-03, -2.2560e-03,  …,  7.9104e-04,
    2.6429e-03,  2.7812e-03],
  [-3.1742e-03, -2.7632e-03, -1.6051e-03,  …,  2.9186e-04,
    1.4923e-03,  2.3324e-03],
  [-1.7292e-03, -1.8837e-03, -1.1304e-03,  …,  2.1144e-03,
    1.9805e-03,  1.8511e-03],
  …,
  [-5.1059e-04, -1.2999e-03, -1.4796e-03,  …, -5.2792e-03,
   -5.9299e-03, -5.0254e-03],
  [ 8.0545e-04, -8.2254e-05, -1.1764e-03,  …, -7.1141e-03,
   -6.4456e-03, -5.5258e-03],
  [ 2.9873e-04,  8.7639e-05, -1.1616e-03,  …, -7.6247e-03,
   -6.7997e-03, -6.0520e-03]],

 [[-7.5896e-04, -7.0748e-04, -7.2195e-04,  …,  2.8600e-03,
    4.6154e-03,  4.7026e-03],
```

```
  [-1.1782e-03, -8.5943e-04,  1.3474e-04,  …,  2.2293e-03,
    3.4425e-03,  4.2477e-03],
  [ 7.3053e-04,  4.7412e-04,  6.2315e-04,  …,  3.7936e-03,
    3.8493e-03,  3.5007e-03],
  …,
  [ 2.1640e-03,  5.5540e-04, -2.1268e-04,  …, -5.0738e-03,
   -5.9062e-03, -5.4301e-03],
  [ 3.1573e-03,  1.6561e-03,  8.1706e-05,  …, -7.2130e-03,
   -6.7021e-03, -6.2006e-03],
  [ 2.4051e-03,  1.4706e-03,  3.6237e-05,  …, -7.7884e-03,
   -7.3064e-03, -6.8783e-03]]],


[[[ 2.0120e-03,  2.5099e-03,  2.7438e-03,  …,  1.4528e-03,
    2.1782e-03,  2.7027e-03],
  [-1.3661e-04,  8.4969e-04,  1.7768e-03,  …,  1.0371e-03,
    1.5769e-03,  2.4847e-03],
  [-9.4853e-04, -4.1741e-04,  7.5684e-04,  …,  9.0178e-04,
    1.4050e-03,  1.3618e-03],
  …,
  [ 2.2953e-03,  2.6785e-03,  1.7122e-03,  …, -2.1435e-03,
   -2.0904e-03, -1.4019e-03],
  [ 2.1633e-03,  2.4655e-03,  2.1665e-03,  …, -2.3648e-03,
   -2.4258e-03, -1.6952e-03],
  [ 1.0190e-03,  1.9923e-03,  2.2053e-03,  …, -2.9580e-03,
   -1.8526e-03, -1.9152e-03]],

 [[-5.4102e-03, -4.2091e-03, -3.5522e-03,  …, -1.6353e-03,
   -3.6377e-05,  5.9333e-04],
  [-6.8185e-03, -5.4028e-03, -4.0080e-03,  …, -1.4335e-03,
   -2.8858e-04,  5.7512e-04],
  [-6.4737e-03, -5.3658e-03, -3.8990e-03,  …, -1.5398e-03,
   -8.9173e-04, -7.2317e-04],
  …,
  [ 7.2537e-04,  1.2341e-03, -5.9471e-04,  …, -6.2703e-03,
   -6.1194e-03, -6.2122e-03],
  [ 4.6639e-04,  9.5825e-04,  2.4888e-04,  …, -7.0600e-03,
   -7.0374e-03, -6.6969e-03],
  [-1.6367e-03, -6.5205e-04, -5.0040e-04,  …, -6.9055e-03,
   -5.6824e-03, -6.2659e-03]],

 [[-4.7478e-03, -3.7566e-03, -2.5884e-03,  …,  2.0665e-03,
    4.0604e-03,  5.2125e-03],
  [-4.9933e-03, -3.5415e-03, -1.7652e-03,  …,  3.2306e-03,
    4.3247e-03,  5.3653e-03],
  [-3.5090e-03, -2.3175e-03, -1.1573e-03,  …,  3.0220e-03,
    3.6353e-03,  4.1133e-03],
  …,
```

```
  [ 6.4428e-03,  6.5858e-03,  4.6003e-03,  …, -3.0695e-03,
   -3.4469e-03, -4.0632e-03],
  [ 6.4954e-03,  6.2554e-03,  4.7068e-03,  …, -4.2941e-03,
   -4.6972e-03, -4.7350e-03],
  [ 4.0288e-03,  3.8883e-03,  3.6677e-03,  …, -4.8001e-03,
   -4.0768e-03, -5.0284e-03]]],


[[[-1.4219e-03, -2.6904e-03, -1.6330e-03,  …, -1.6692e-04,
   -2.9259e-04,  1.2071e-03],
  [-3.8967e-03, -4.7200e-03, -3.4687e-03,  …, -3.1856e-03,
   -1.4885e-03,  8.5436e-04],
  [ 9.4218e-04, -2.7979e-03, -3.7185e-03,  …, -9.3078e-04,
    1.0225e-03,  2.3377e-03],
  …,
  [-2.4027e-03, -2.6602e-03, -1.4014e-03,  …,  2.2847e-03,
    3.7499e-03,  2.2537e-03],
  [ 1.0522e-03,  1.0228e-03,  3.1764e-03,  …, -1.1110e-03,
   -3.1921e-04, -2.0386e-03],
  [ 2.1104e-03,  2.0063e-03,  1.8175e-03,  …, -3.9506e-04,
   -3.0785e-03, -7.2947e-03]],

 [[ 3.6221e-03,  2.7027e-03,  3.6272e-03,  …,  7.1151e-03,
    8.2563e-03,  1.0069e-02],
  [ 1.7510e-03,  1.8768e-03,  2.8100e-03,  …,  4.7033e-03,
    7.6317e-03,  1.0849e-02],
  [ 7.4079e-03,  3.9746e-03,  2.6709e-03,  …,  9.1546e-03,
    1.0984e-02,  1.2973e-02],
  …,
  [ 6.3665e-03,  7.0184e-03,  8.0585e-03,  …,  1.1015e-02,
    1.2809e-02,  9.9382e-03],
  [ 9.5430e-03,  1.0625e-02,  1.3310e-02,  …,  6.1242e-03,
    6.3321e-03,  3.5122e-03],
  [ 1.1733e-02,  1.1100e-02,  1.1023e-02,  …,  4.7376e-03,
    1.6979e-03, -3.5645e-03]],

 [[-1.4937e-03, -2.8298e-03, -1.5901e-03,  …,  3.3489e-03,
    5.1632e-03,  7.3834e-03],
  [-2.6783e-03, -3.0017e-03, -1.4263e-03,  …,  2.3189e-03,
    5.7966e-03,  1.0125e-02],
  [ 3.4714e-03,  3.5436e-04, -1.3142e-03,  …,  8.3839e-03,
    1.0599e-02,  1.3326e-02],
  …,
  [ 8.9317e-03,  1.0157e-02,  1.1468e-02,  …,  1.5493e-02,
    1.7213e-02,  1.5923e-02],
  [ 1.2217e-02,  1.3440e-02,  1.5997e-02,  …,  1.0433e-02,
    1.0609e-02,  9.7783e-03],
  [ 1.4904e-02,  1.3891e-02,  1.3904e-02,  …,  7.8676e-03,
```

```
           4.3349e-03,  2.3078e-03]]]])
tensor([ 4.6566e-10,  2.4214e-08, -2.3283e-10, -2.0373e-08,  4.0047e-08,
         5.3085e-08,  2.7940e-08, -7.2643e-08,  1.3039e-08, -1.3970e-08,
         7.4506e-09,  4.3656e-09,  2.6077e-08, -2.9802e-08, -6.7521e-09,
         6.9849e-10, -7.6834e-09,  1.4901e-08, -2.0023e-08, -1.9558e-08,
         9.3714e-09, -1.5774e-08, -1.1921e-07,  4.7497e-08,  1.1642e-09,
        -3.3062e-08,  7.6834e-09,  1.8626e-09, -3.3062e-08, -5.4948e-08,
        -2.5262e-08, -1.4435e-08, -1.3970e-09, -1.3970e-09,  2.6193e-08,
        -7.8580e-09, -2.3283e-09, -3.5390e-08, -1.6764e-08, -1.6851e-08,
        -4.6566e-10,  1.5803e-08, -6.9849e-09, -3.5740e-08,  1.2899e-07,
         3.4925e-10, -9.0804e-09, -3.2596e-09, -5.0291e-08,  1.2573e-08,
        -7.2177e-09,  4.1910e-09, -2.0023e-08,  4.7963e-08,  2.5611e-09,
         2.7940e-09,  6.8452e-08, -4.2492e-09,  1.3155e-08,  1.3970e-09,
        -2.4680e-08,  1.1452e-08,  3.6554e-08, -5.2969e-09,  3.8417e-09,
        -1.3970e-09,  6.9849e-09,  1.3039e-08, -6.5193e-09,  1.2107e-08,
         7.2177e-09,  7.6834e-09, -2.7474e-08,  4.6566e-08, -1.3039e-08,
         2.8871e-08,  1.1176e-08,  1.3039e-08, -4.5402e-09,  1.1059e-08,
         1.7870e-07, -3.2596e-08,  2.3632e-08,  1.3039e-08, -1.6973e-07,
         3.4925e-09, -6.0536e-09,  1.0710e-08, -5.3551e-09,  1.8626e-09,
         3.1199e-08, -6.5193e-08,  3.3528e-08,  7.2177e-09,  1.8161e-08,
        -4.6566e-10])
tensor([-3.2990e-02,  2.0713e-02, -4.3274e-03,  7.5254e-03,  3.4543e-02,
        -3.0038e-02,  4.5634e-02, -5.4455e-02,  2.8331e-02,  1.2127e-02,
         6.8413e-03,  1.1741e-02, -6.3810e-02,  1.1572e-02,  3.7331e-02,
         6.0355e-03,  9.8433e-03, -8.2048e-02, -3.0553e-02,  1.0808e-03,
         2.9550e-02,  5.9926e-02,  3.5458e-02,  2.8855e-02,  7.6752e-03,
         8.9153e-03, -3.5585e-03,  1.7646e-02, -1.4269e-02, -3.3022e-02,
         1.5569e-02, -4.9827e-02,  2.2756e-02,  7.6946e-03,  3.3254e-02,
         2.4941e-02, -1.6486e-02, -2.0753e-03,  2.9291e-02,  6.3663e-03,
         4.7913e-03,  4.4178e-03, -6.1750e-03,  1.9215e-02,  6.5574e-02,
        -2.1934e-02,  2.1328e-03,  2.3865e-02,  6.5778e-03, -1.7272e-02,
        -1.8102e-02,  1.2162e-02,  1.6262e-02, -5.2907e-03,  5.6304e-03,
         3.3588e-02, -9.3130e-02,  2.4848e-03,  2.1536e-02,  1.7699e-02,
        -9.1010e-04,  2.5836e-05, -5.4595e-03, -3.2231e-03, -2.3201e-02,
         3.0735e-02,  3.4655e-02,  7.9140e-02,  1.8507e-02,  3.4495e-02,
        -1.6301e-03, -7.0050e-03,  4.4020e-03, -1.0184e-02, -2.0478e-02,
         4.5243e-03,  1.2520e-03,  1.4802e-02,  4.0623e-02,  1.9299e-02,
        -2.4991e-02, -3.6735e-02,  6.7775e-03,  2.5997e-02, -3.7054e-03,
        -1.8118e-02,  3.3947e-02, -2.4342e-02,  3.4516e-03, -1.6844e-02,
        -4.4537e-03, -2.8883e-02, -1.7299e-01,  2.1440e-02, -4.4705e-03,
         1.2915e-03])
tensor([-0.0111, -0.0187, -0.0119,  0.0085,  0.0144,  0.0146,  0.0131,  0.0091,
         0.0187,  0.0089,  0.0072,  0.0124, -0.0410,  0.0262,  0.0268,  0.0117,
         0.0082, -0.0176,  0.0006, -0.0066,  0.0223,  0.0209, -0.0114, -0.0010,
         0.0028, -0.0140,  0.0054, -0.0012,  0.0048,  0.0033,  0.0074, -0.0306,
         0.0019,  0.0073, -0.0031,  0.0191, -0.0060, -0.0167,  0.0130, -0.0072,
        -0.0128, -0.0107, -0.0133,  0.0134, -0.0147, -0.0173, -0.0125,  0.0069,
         0.0259, -0.0063, -0.0031, -0.0032,  0.0044,  0.0169, -0.0079,  0.0002,
```

```
         0.0074,  0.0070,  0.0130,  0.0025,  0.0021,  0.0030,  0.0007,  0.0070,
        -0.0131,  0.0163,  0.0253,  0.0351,  0.0074,  0.0109,  0.0017, -0.0154,
         0.0253, -0.0416,  0.0039,  0.0015,  0.0050,  0.0266, -0.0031,  0.0124,
        -0.0093, -0.0519,  0.0099,  0.0236, -0.0326, -0.0112,  0.0269,  0.0009,
         0.0083, -0.0154,  0.0142,  0.0145, -0.0132,  0.0226,  0.0100, -0.0057])
tensor([[[[ 7.7685e-04,  5.1423e-04,  1.6254e-04, -1.0393e-04,  6.6996e-04],
          [ 1.1897e-03,  4.1601e-04,  5.8038e-04,  8.3245e-04,  1.0992e-03],
          [ 6.6996e-04,  6.8401e-04,  8.3786e-04,  9.6985e-04,  1.0630e-03],
          [ 6.3910e-04,  1.0221e-03,  9.4181e-04,  7.8991e-04,  6.4604e-04],
          [ 8.1507e-04,  5.6866e-04,  4.5404e-04, -5.5384e-05,  8.0476e-05]],

         [[-2.3044e-03, -2.4502e-03, -4.4159e-04,  7.5271e-04, -1.5688e-03],
          [-2.4711e-03, -2.5543e-03,  2.0692e-04,  2.0337e-04, -4.8012e-04],
          [-2.1897e-03, -1.2567e-03, -4.4250e-04,  3.2760e-04,  9.5674e-04],
          [-1.4053e-03,  3.1422e-04,  1.0895e-03,  1.6831e-03,  1.9154e-03],
          [-1.0950e-03,  4.1248e-04,  6.8442e-04,  1.7616e-03,  1.9383e-03]],

         [[-2.7610e-05, -7.4378e-04, -5.0604e-04, -2.4388e-04,  1.4408e-04],
          [-7.2498e-05, -5.9501e-04, -5.5738e-04, -7.3822e-05, -2.4483e-04],
          [-5.7438e-04, -5.2187e-04, -1.3746e-04, -1.1791e-04, -3.3004e-04],
          [-7.7723e-04, -6.2817e-04, -3.9416e-04, -4.5280e-05, -5.4570e-05],
          [-6.5084e-04, -5.3239e-04, -3.4486e-04, -3.6101e-04, -2.0222e-04]],

         …,

         [[ 9.4896e-04,  5.4439e-04,  5.7991e-04,  8.0727e-04,  9.4660e-04],
          [ 9.6432e-04,  7.7256e-04,  6.7533e-04,  7.2276e-04,  6.0385e-04],
          [ 8.6874e-04,  8.9267e-04,  9.2202e-04,  7.2916e-04,  5.3939e-04],
          [ 4.4311e-04,  5.1141e-04,  5.1998e-04,  4.1558e-04,  2.4989e-04],
          [ 2.3433e-04,  9.4506e-05,  1.4702e-04, -8.4100e-05, -1.8116e-04]],

         [[ 1.6332e-03,  1.2625e-03,  1.1824e-03,  1.2787e-03,  1.4656e-03],
          [ 1.3306e-03,  1.1162e-03,  1.0131e-03,  1.0938e-03,  1.0586e-03],
          [ 1.2350e-03,  1.2562e-03,  1.2465e-03,  1.1585e-03,  9.6233e-04],
          [ 9.1497e-04,  9.2895e-04,  9.2935e-04,  7.5463e-04,  6.1354e-04],
          [ 4.4607e-04,  3.5817e-04,  3.9837e-04,  3.5123e-04,  2.3840e-05]],

         [[ 4.4727e-04,  3.3111e-05, -3.2791e-05,  7.8507e-05,  4.5462e-06],
          [ 4.4124e-04,  4.4331e-04,  2.7512e-04, -8.6833e-05, -5.0305e-04],
          [ 1.9938e-04,  1.5785e-04,  5.5846e-04,  1.4367e-04, -4.9842e-04],
          [-3.2106e-04, -2.9498e-04, -2.0033e-04, -7.8608e-05, -2.8517e-04],
          [-2.4904e-04, -3.9067e-04, -5.7104e-04, -7.6172e-04, -5.5777e-04]]],


        [[[-2.1574e-03, -2.0936e-03, -1.7818e-03, -1.6080e-03, -1.8095e-03],
          [-7.2721e-04, -8.2537e-04, -1.0111e-03, -1.3634e-03, -1.4688e-03],
          [-7.7918e-04, -6.5519e-04, -1.0187e-03, -3.7506e-04,  1.1459e-04],
          [-2.1212e-03, -1.7165e-03, -8.1569e-04, -1.2790e-05,  8.7043e-04],
```

```
  [-1.9433e-03, -1.0947e-03, -6.7850e-06,  6.5381e-04,  1.0063e-03]],

 [[ 1.7155e-03,  6.6059e-04, -3.6181e-04, -9.0374e-04, -9.8015e-05],
  [ 2.4451e-03,  8.2488e-04,  5.4765e-04, -1.2068e-04,  1.3765e-04],
  [ 2.8877e-03,  1.7629e-03,  2.1564e-03,  9.8312e-04,  5.2422e-04],
  [ 4.9782e-03,  2.7257e-03,  2.6520e-03,  2.5515e-04,  1.1240e-03],
  [ 3.0982e-03,  4.3584e-05,  4.1364e-04, -4.0944e-04, -2.1874e-04]],

 [[ 2.6636e-04,  5.6049e-04,  6.9306e-04,  3.3617e-04, -1.0190e-04],
  [ 4.7783e-04,  6.5771e-04,  5.8305e-04,  2.6883e-04, -9.8150e-06],
  [ 3.5236e-04,  2.9421e-04,  3.7075e-04,  2.1005e-04,  2.7953e-04],
  [ 4.7925e-04,  5.2932e-04,  3.4976e-04,  2.2111e-04,  9.1296e-05],
  [ 6.3432e-04,  7.1517e-04,  6.0429e-04,  1.7323e-04, -2.5552e-05]],

 …,

 [[-1.5261e-04, -1.6470e-04,  1.7568e-04,  2.8279e-04,  3.6868e-04],
  [-4.4464e-04, -3.6951e-04, -3.2926e-04, -2.7897e-04, -2.5915e-04],
  [-6.7019e-04, -7.0494e-04, -5.4181e-04, -4.9420e-04, -3.3268e-04],
  [-5.3223e-04, -6.3562e-04, -5.3526e-04, -5.9964e-04, -5.1944e-04],
  [ 9.9534e-05, -1.6098e-04, -2.5831e-05, -3.7345e-04, -3.2211e-04]],

 [[-5.6077e-04, -4.4182e-04, -4.3470e-05, -1.0393e-05,  2.6304e-04],
  [-1.8407e-04, -1.1615e-04, -1.4786e-04, -2.9388e-04, -3.2031e-04],
  [-6.6665e-04, -7.0700e-04, -5.6852e-04, -5.3090e-04, -5.4732e-04],
  [-9.3725e-04, -9.9075e-04, -7.1510e-04, -3.7874e-04, -2.6325e-04],
  [-5.9836e-04, -8.1615e-04, -5.2210e-04, -5.3574e-04, -3.4530e-04]],

 [[-4.0199e-04, -1.8932e-04,  9.9999e-05,  1.2291e-04, -3.2672e-04],
  [-4.8041e-04, -4.1399e-04, -5.2216e-04, -5.5849e-04, -7.0857e-04],
  [-5.2846e-04, -6.7118e-04, -4.3061e-04, -1.0339e-04,  2.7614e-05],
  [-6.4056e-05, -1.6557e-04, -1.7636e-04,  1.3875e-04, -1.6165e-04],
  [ 1.1611e-03,  6.9671e-04,  4.2662e-04,  5.7714e-04, -4.9844e-05]]],


[[[ 9.2492e-04,  4.2970e-04, -2.3966e-04, -4.0060e-04,  9.9494e-05],
  [ 4.8452e-04,  6.2581e-04,  4.1536e-04,  1.6650e-04, -5.2353e-04],
  [ 7.3888e-04,  3.8540e-04,  2.5880e-04, -3.5360e-04, -1.4072e-03],
  [-4.2687e-06, -5.7779e-04, -1.1594e-03, -1.5332e-03, -1.8964e-03],
  [-3.6314e-04, -7.3082e-04, -1.2415e-03, -9.9212e-04, -3.2811e-04]],

 [[ 3.8253e-04,  5.3543e-04,  6.1095e-04, -4.8920e-04,  8.7700e-04],
  [-1.2293e-04,  8.9606e-04,  1.1280e-03,  7.8648e-04,  1.5830e-04],
  [-3.9156e-04, -6.3343e-04,  3.9178e-04,  9.9117e-04, -7.0015e-04],
  [ 1.3000e-03,  2.8480e-04,  9.0636e-04,  4.3102e-04, -1.9229e-03],
  [ 2.4398e-03,  1.2410e-03,  2.6801e-04, -1.2905e-03, -2.7290e-03]],

 [[ 1.9203e-04,  1.7285e-04,  2.7215e-04,  3.7623e-04,  8.1478e-04],
```

```
  [ 4.3962e-04,  5.2840e-04,  4.9038e-04,  5.3943e-04,  6.7638e-04],
  [ 5.8337e-04,  5.6467e-04,  6.5321e-04,  6.2072e-04,  7.9711e-04],
  [ 2.5953e-04,  7.0854e-04,  8.5550e-04,  8.6915e-04,  1.0116e-03],
  [ 2.8743e-04,  4.7454e-04,  6.5055e-04,  6.3487e-04,  6.1299e-04]],

 …,

 [[ 2.8626e-04,  1.2917e-04,  2.3585e-04,  7.2237e-05,  2.4895e-04],
  [ 5.0017e-04,  2.8854e-04,  1.0074e-04, -1.8579e-05,  1.6068e-04],
  [ 3.6048e-04,  3.7817e-04,  3.2526e-04, -7.1836e-06,  4.6880e-05],
  [ 3.6919e-04,  5.1087e-04,  6.6203e-04,  3.9916e-04,  2.0947e-04],
  [-2.8447e-05,  2.6843e-05,  4.2090e-04, -3.6242e-05, -2.2399e-04]],

 [[-8.4809e-05, -1.8950e-04, -1.9766e-04, -4.4713e-04, -1.7846e-04],
  [-1.0295e-04,  2.8321e-05,  2.3608e-05, -1.0182e-04,  3.0091e-05],
  [-3.1059e-05, -1.7144e-05,  1.4858e-05, -3.9631e-04, -4.8678e-04],
  [ 2.0635e-04,  1.2390e-04,  9.0938e-05, -2.8650e-04, -5.8368e-04],
  [-3.2674e-04, -1.5016e-04, -6.0413e-05, -7.3145e-04, -9.2464e-04]],

 [[-1.6965e-04,  2.7579e-04,  3.0713e-04,  3.0111e-04,  6.7885e-04],
  [ 6.1741e-04,  3.4438e-04,  1.6159e-04,  3.3692e-04,  1.5317e-04],
  [ 6.5705e-04,  3.5525e-04,  3.8229e-04,  4.3544e-05,  1.4104e-04],
  [ 3.5545e-04,  4.4829e-04,  6.7938e-04,  5.8213e-04,  2.2699e-04],
  [-4.3709e-04, -5.9948e-05, -5.7946e-05,  1.9127e-05, -5.0282e-04]]],


…,


[[[ 5.4423e-04,  1.1775e-03,  1.3951e-03,  5.2657e-04, -1.8076e-04],
  [ 1.5947e-03,  1.9088e-03,  1.2575e-03,  3.0825e-04, -5.6326e-04],
  [ 1.6335e-03,  1.7150e-03,  1.2568e-03,  3.7479e-04, -2.4106e-04],
  [ 7.9570e-04,  6.5562e-04,  1.2333e-04, -1.9479e-04, -7.7288e-04],
  [-7.0535e-04, -1.1319e-03, -9.3363e-04, -1.0290e-03, -7.9490e-04]],

 [[ 2.2750e-03,  3.9510e-03,  2.3730e-03,  1.4344e-03, -2.1659e-03],
  [ 1.5029e-03,  1.3121e-03,  2.5018e-04,  4.7658e-04, -2.2642e-03],
  [ 4.7217e-04, -5.0356e-04,  3.9853e-04, -5.3301e-05, -2.8026e-04],
  [ 8.6679e-04, -6.1795e-05, -2.4740e-04,  2.7389e-04, -4.2171e-04],
  [ 1.1421e-03, -5.5454e-04, -2.2475e-03, -8.0971e-04, -6.3919e-04]],

 [[-2.3186e-05,  1.2806e-04,  6.0395e-04,  1.6399e-04,  6.1484e-04],
  [-2.4477e-04, -5.1658e-04, -2.7962e-04, -2.2895e-04,  2.3543e-04],
  [-5.0665e-04, -9.8598e-04, -7.1501e-04, -4.7643e-04,  7.3444e-05],
  [-9.3961e-04, -1.2510e-03, -1.3356e-03, -6.4060e-04, -2.9272e-04],
  [-1.4008e-03, -1.4298e-03, -1.1869e-03, -7.5894e-04, -3.5680e-04]],

 …,
```

```
 [[-6.8474e-04, -2.7132e-04, -1.2485e-05, -4.1098e-06,  5.1181e-05],
  [-7.7409e-04, -4.9918e-04, -1.5649e-04,  3.4089e-05,  2.6164e-04],
  [-5.7655e-04, -6.1848e-04, -3.6508e-04, -3.6879e-05,  4.9344e-05],
  [-6.8250e-04, -7.1718e-04, -5.8335e-04, -2.6520e-04, -2.8944e-04],
  [-6.4959e-04, -5.7275e-04, -3.2852e-04, -3.2400e-04, -2.2686e-04]],

 [[-1.1476e-03, -9.5903e-04, -6.6208e-04, -4.3003e-04, -4.7820e-04],
  [-1.0732e-03, -6.6983e-04, -4.9977e-04, -2.4022e-04, -2.0330e-04],
  [-7.7615e-04, -6.2155e-04, -4.4565e-04, -3.1875e-04, -3.4834e-04],
  [-5.4396e-04, -6.6002e-04, -6.0434e-04, -4.4202e-04, -3.9908e-04],
  [-4.6659e-04, -5.4517e-04, -5.1551e-04, -5.4356e-04, -3.0587e-04]],

 [[-1.3128e-04,  5.6935e-04,  5.2257e-04,  1.2430e-03,  1.1853e-03],
  [-2.0278e-05,  3.5782e-04,  5.0136e-04,  9.2178e-04,  1.4484e-03],
  [-1.6371e-04, -7.1067e-04, -2.2028e-04,  5.4537e-04,  7.6030e-04],
  [-4.1183e-04, -6.6310e-04, -4.2608e-04,  2.8736e-04,  5.4858e-04],
  [-8.1126e-04, -3.9587e-04,  5.5988e-04,  6.2625e-04,  9.1072e-04]]],


[[[ 4.7788e-04,  1.0130e-04,  3.0517e-04,  1.2359e-04, -4.7321e-04],
  [ 1.0884e-03,  4.8466e-04,  1.2120e-03,  1.1251e-03, -5.7292e-04],
  [ 2.4067e-04, -6.8552e-04,  7.2258e-04,  8.1893e-04,  6.0152e-04],
  [ 6.8884e-05,  1.4558e-04,  8.5749e-04,  1.8980e-03,  2.8576e-03],
  [ 4.3696e-04,  1.1883e-03,  2.8128e-03,  2.4237e-03,  1.8743e-03]],

 [[-1.0832e-03, -4.8054e-05,  1.6413e-03,  4.9846e-04,  9.9866e-05],
  [-1.4842e-04, -9.5240e-04,  4.8201e-04,  1.0435e-03,  2.2794e-03],
  [-1.1764e-03, -8.9827e-04, -7.4083e-04,  7.5318e-04,  1.4132e-03],
  [-2.6199e-03, -2.9713e-03,  4.7529e-04,  1.6302e-03,  1.8364e-03],
  [-3.6607e-03, -1.5927e-03,  1.7996e-03,  2.3633e-03,  1.1486e-03]],

 [[ 6.6836e-04,  7.6601e-04,  9.5613e-04,  1.1214e-03,  8.9781e-04],
  [ 2.5583e-04,  2.8993e-04,  7.5819e-04,  1.0087e-03,  7.9942e-04],
  [-3.1993e-04, -2.1063e-04,  2.1295e-04,  3.3081e-04,  1.2054e-03],
  [-7.8592e-04, -5.9418e-04,  1.4504e-04,  5.7653e-04,  7.4424e-04],
  [-1.2393e-03, -8.3554e-04,  8.6080e-05,  5.5276e-04,  5.9430e-04]],

 …,

 [[ 1.6302e-04, -1.0900e-05, -1.2564e-04, -2.5259e-04, -5.9561e-04],
  [-1.6504e-04, -3.0990e-04, -4.7075e-05, -8.0490e-05, -4.6589e-04],
  [-2.5406e-04, -3.1469e-04,  3.1696e-05,  3.0538e-05,  2.0772e-04],
  [ 1.6062e-04,  1.1634e-04,  1.7634e-04,  5.9153e-04,  9.1014e-04],
  [-3.0614e-04, -7.6532e-05,  5.3878e-04,  1.4039e-03,  1.3867e-03]],

 [[-2.7679e-04, -3.6806e-04, -2.8394e-04, -5.2668e-04, -5.1246e-04],
  [-3.8004e-04, -4.4715e-04, -3.8508e-04, -5.0781e-04, -7.8877e-04],
```

```
  [-6.2052e-04, -6.9194e-04, -5.3590e-04, -4.9942e-04, -2.4536e-04],
  [ 7.6874e-05, -8.0051e-05, -3.6152e-05,  1.2368e-04,  8.7255e-04],
  [ 2.0250e-04,  2.6892e-04,  8.9378e-04,  1.7265e-03,  1.8562e-03]],

 [[ 2.5236e-04,  6.7184e-05,  6.1491e-04,  1.1329e-03,  1.6687e-04],
  [-3.0837e-04,  2.9689e-05,  6.7560e-04,  9.2192e-04,  2.9350e-04],
  [-9.1244e-04, -7.6065e-04,  1.5871e-04,  6.2767e-04,  1.0357e-03],
  [-1.1114e-03, -6.0946e-04,  1.1303e-03,  1.9965e-03,  2.5651e-03],
  [-1.1927e-03, -5.7779e-04,  1.9649e-03,  2.6627e-03,  2.4168e-03]]],


[[[ 2.2733e-03,  1.3024e-03,  1.6633e-03,  2.1948e-03,  2.4413e-03],
  [ 1.2676e-03,  1.0933e-03,  6.1865e-04,  1.0387e-03,  1.5593e-03],
  [ 9.5507e-04,  9.8172e-04,  9.8469e-04,  1.4216e-03,  2.1652e-03],
  [ 2.9801e-04,  2.8060e-04,  5.4169e-04,  1.0417e-03,  1.1404e-03],
  [ 4.1484e-04,  4.0240e-04,  1.2537e-04,  2.2454e-04, -3.5829e-05]],

 [[ 6.4610e-03,  3.9139e-03,  1.9755e-03,  5.4750e-03,  7.7206e-03],
  [ 6.8615e-03,  3.7862e-03,  1.6869e-03,  3.3728e-03,  4.6573e-03],
  [ 6.5242e-03,  3.6511e-03,  1.6534e-03, -3.5086e-04,  1.8690e-03],
  [ 5.6614e-03,  3.4684e-03, -1.0030e-04, -2.3934e-03,  9.1557e-05],
  [ 4.2930e-03,  2.6673e-03, -6.1034e-04, -2.3009e-03, -1.4267e-03]],

 [[ 2.6646e-04, -4.1283e-06, -4.8863e-04, -8.3291e-04,  4.0076e-04],
  [-1.7559e-05,  2.5328e-04, -5.3179e-04, -7.8749e-04,  7.5119e-05],
  [-1.5834e-04,  8.9201e-06, -1.0190e-04, -2.7916e-04, -7.0562e-04],
  [ 4.5114e-04,  6.9069e-04,  5.0331e-04, -2.7350e-04, -5.3716e-04],
  [ 4.1379e-04,  6.5521e-04,  4.1476e-04,  6.4602e-05, -1.5867e-04]],

 …,

 [[ 4.8605e-05,  6.1903e-05, -3.8494e-04, -5.3019e-04,  1.7733e-04],
  [-2.9268e-04, -2.3454e-04, -5.9937e-04, -5.6295e-04, -1.8812e-04],
  [-5.9042e-04, -5.4912e-04, -5.0223e-04, -3.9384e-04, -1.7897e-04],
  [-4.1923e-04, -3.5265e-04, -3.2275e-04, -2.4178e-04, -3.4469e-05],
  [ 5.2674e-06,  3.4088e-05, -5.7940e-05, -2.6254e-05,  5.6863e-05]],

 [[ 9.6888e-05, -7.1405e-05, -4.1375e-04, -3.4475e-04,  3.5478e-05],
  [-3.1902e-04, -1.5861e-04, -4.8871e-04, -5.5710e-04, -1.5458e-04],
  [-5.2067e-04, -4.5935e-04, -4.8172e-04, -4.7468e-04, -1.6109e-04],
  [-3.4496e-04, -3.4251e-04, -3.5731e-04, -2.9859e-04, -3.8778e-05],
  [-4.0975e-05, -2.5745e-04, -3.0572e-04, -1.8075e-04, -2.3937e-04]],

 [[ 4.9141e-04,  1.1202e-04, -4.9717e-04,  9.8257e-05,  2.0586e-03],
  [ 1.0919e-05,  1.8372e-04, -5.5817e-04, -2.5440e-04,  1.5473e-03],
  [-1.9990e-04, -9.3684e-05, -2.2023e-04,  1.6412e-04,  1.1428e-03],
  [ 6.5604e-04,  7.5461e-04,  3.7129e-04,  4.6425e-04,  1.1420e-03],
  [ 1.3160e-03,  1.1709e-03,  5.3823e-04,  6.5989e-04,  9.0697e-04]]]])
```

```
tensor([ 1.6438e-09, -3.9095e-09, -7.0617e-10,  8.7548e-09,  1.9735e-09,
         3.6653e-10, -6.5063e-10,  4.0664e-09,  1.0323e-10,  1.3642e-11,
         4.5686e-09, -3.9593e-09, -1.6432e-09, -1.1535e-09, -4.1157e-09,
         2.4064e-09,  3.1064e-09,  3.0037e-09,  3.9917e-09, -1.3701e-09,
         1.0368e-09, -2.1093e-09,  2.4875e-10, -6.1823e-10,  4.7694e-10,
         7.4221e-09, -2.8658e-09,  3.7192e-08,  7.4515e-09, -5.3440e-09,
         2.1396e-09, -4.9988e-10,  5.3562e-10,  4.6385e-09,  2.1789e-09,
        -3.6439e-09, -2.5641e-09, -4.1291e-09,  2.2563e-09, -4.6285e-10,
         1.3223e-09, -7.6352e-09,  1.0669e-08,  1.3574e-09, -5.4555e-11,
        -7.3092e-09,  2.2485e-09,  1.6094e-08, -8.4142e-09, -3.0830e-09,
         4.1382e-09,  5.1257e-09,  3.6397e-09,  4.7558e-09, -6.5788e-10,
        -8.6836e-09,  4.1459e-09,  4.6782e-10, -1.1158e-08, -2.2034e-09,
        -7.5579e-10,  6.4301e-10, -1.5307e-09,  2.3625e-09,  1.6066e-09,
        -9.0719e-10,  1.3229e-08,  8.5697e-09, -4.5225e-10,  4.4088e-09,
         5.0324e-09,  3.1519e-09, -8.9045e-10, -4.4031e-09,  1.6904e-08,
        -2.9753e-09, -5.2257e-09, -5.5667e-10,  2.8908e-09, -7.1733e-10,
        -6.5498e-10,  3.2025e-10, -2.7961e-09, -1.1004e-08,  2.1434e-09,
        -1.7126e-09, -1.0510e-09, -2.6679e-09, -7.6898e-10, -1.5930e-11,
         9.9946e-09,  6.0274e-10, -1.6548e-09,  2.1618e-08,  3.6615e-09,
         3.4352e-09,  2.6247e-10, -2.2950e-09, -5.8496e-09,  2.3050e-09,
         1.4477e-09,  9.3337e-10,  2.1441e-09,  9.4133e-10,  1.0262e-09,
        -2.9559e-10,  1.5007e-10, -1.2396e-09, -1.1312e-10, -2.7594e-09,
        -2.9354e-09,  1.5513e-09,  2.6239e-10, -1.4120e-10,  1.5666e-09,
        -6.9082e-10, -3.6753e-09,  2.1264e-09,  1.2419e-09, -6.7737e-09,
         2.3099e-08,  8.3320e-09,  4.8112e-10,  1.0351e-09,  1.6371e-11,
         4.0131e-10, -1.1572e-08, -3.3773e-09,  7.6472e-11,  3.6080e-09,
         5.5911e-10, -8.7310e-09, -3.5113e-09, -4.2023e-09,  1.7465e-10,
        -2.4829e-10,  1.3042e-09, -5.2430e-10,  5.0418e-09, -9.1821e-09,
        -6.7497e-10, -2.6415e-10, -5.5093e-09, -1.3849e-09,  2.3183e-09,
        -1.4115e-09, -1.6755e-08, -3.6624e-09, -6.2505e-10, -1.3281e-09,
        -6.8323e-10,  1.2171e-09, -1.5369e-09, -5.8797e-09, -8.6909e-09,
        -1.7994e-09,  4.4350e-09,  6.9576e-10, -2.9338e-09, -1.8493e-09,
         1.1841e-08, -1.0468e-09, -1.2183e-09, -4.0400e-09, -6.7874e-10,
         1.0963e-09, -6.3596e-10,  8.8540e-10, -1.8021e-09, -1.8987e-09,
        -2.6607e-09,  8.7321e-09,  3.6084e-09, -7.9746e-09,  3.9634e-09,
        -5.2041e-09, -4.8383e-09, -2.7012e-09, -3.4900e-09,  1.2128e-10,
        -1.2527e-09,  1.4126e-09, -5.1257e-09,  6.4883e-10,  7.0668e-09,
        -8.8250e-10,  4.8996e-09, -3.7794e-09, -8.2225e-09,  3.5199e-09,
        -6.3586e-10, -6.4734e-09, -5.8330e-09,  3.8214e-10, -2.8558e-10,
        -2.1775e-10, -1.4895e-09, -3.3979e-09,  2.3069e-09,  1.3826e-08,
         3.0604e-09,  3.3219e-10,  4.7635e-10, -1.3826e-09, -2.2219e-09,
        -2.5511e-10, -2.8297e-10, -3.6302e-09,  2.4661e-09, -1.4394e-09,
         2.6375e-10, -1.4583e-09, -2.6595e-09,  3.0868e-09,  1.5186e-09,
        -1.3197e-09,  3.1846e-09,  2.5602e-09, -1.2401e-09, -9.9840e-10,
         6.2153e-10, -9.7060e-10,  2.3259e-08, -2.5487e-08,  5.7418e-10,
         5.9208e-10, -2.3214e-09, -3.6375e-10, -2.4278e-09,  2.6066e-09,
        -1.2310e-09, -3.5405e-09, -1.7947e-09, -3.5263e-09, -2.2001e-09,
        -5.9836e-09, -5.4149e-09, -1.0805e-09,  1.0357e-08, -7.3305e-10,
```

```
        -3.9982e-10, -4.1190e-09, -1.4081e-09, -2.9869e-09,  1.0002e-08,
        -8.8403e-10, -3.6139e-09,  3.1531e-09,  1.3440e-09,  7.3625e-09,
        -4.2678e-10,  3.0174e-09,  4.7100e-10, -1.3370e-09,  6.9208e-10,
         3.9378e-09])
tensor([ 4.0768e-03,  4.6482e-03, -3.0908e-03,  1.0727e-02, -6.8669e-04,
         1.9199e-03,  3.5215e-04, -3.1002e-03,  6.2135e-03,  9.2312e-03,
        -1.0771e-03, -5.2392e-02, -1.7576e-03,  3.9606e-02,  8.0602e-03,
        -3.3239e-03,  6.0022e-04, -5.3836e-02,  4.4384e-02,  4.2871e-02,
        -4.0527e-03, -2.0458e-02,  3.3540e-03,  3.3129e-02,  9.5753e-03,
        -2.9116e-02,  1.2971e-02,  2.2902e-01,  2.6373e-02, -2.9826e-03,
         1.2154e-02, -6.0866e-02,  4.7703e-02, -4.9955e-02,  2.2354e-03,
         1.2261e-03,  1.1625e-02,  5.0168e-03,  1.5463e-02, -4.6165e-03,
         6.6616e-03,  2.5700e-03,  1.6346e-05, -5.0512e-03,  1.5806e-04,
        -5.6931e-02,  1.8266e-02,  8.7471e-03, -7.0507e-03,  1.7842e-02,
        -8.8299e-03,  1.1863e-02,  1.2608e-02,  1.0034e-02, -9.2607e-03,
         3.2314e-02, -1.9797e-03,  1.4670e-03,  4.0007e-02, -1.0047e-02,
         6.3896e-03, -7.0230e-03, -3.1188e-02,  8.0102e-03, -1.0158e-03,
         8.1600e-03, -1.1322e-02, -1.7891e-03, -1.4075e-01, -1.1084e-02,
         5.6694e-03, -1.6359e-04,  1.8199e-02, -4.1193e-04,  7.5175e-03,
         1.1683e-02,  5.8121e-03,  3.9747e-03,  5.2372e-03,  9.7834e-04,
        -5.4393e-03, -2.6940e-03,  8.9702e-03,  1.9850e-03, -1.5282e-02,
         4.3701e-03, -1.0915e-02, -5.9350e-03, -4.7529e-04, -5.8206e-04,
        -8.6332e-03, -1.0277e-02, -1.4719e-03, -2.2098e-02, -6.4553e-02,
         3.0202e-04,  2.7037e-03,  3.6320e-03, -7.9159e-04, -8.7447e-04,
         2.9435e-02, -2.6420e-03,  8.9904e-03,  2.1366e-03,  5.5052e-03,
        -2.2704e-03,  2.8748e-02, -5.0371e-03,  6.0048e-03,  1.3698e-02,
         8.2596e-03, -2.9540e-04,  6.7638e-03,  8.1027e-03,  4.7610e-04,
        -7.6238e-03,  8.9529e-03,  3.5770e-03, -9.8506e-03,  4.1882e-02,
        -3.7553e-03,  1.1001e-03,  2.6851e-03, -3.7184e-03, -2.7734e-03,
         1.7113e-02,  1.2261e-03,  9.0222e-03,  3.4125e-04,  7.7335e-03,
        -7.1612e-03,  2.5545e-02,  5.0204e-03, -1.6226e-02, -1.4087e-02,
         3.0619e-03, -7.0894e-02,  1.4763e-02, -4.3282e-03, -7.6460e-03,
         1.1356e-04, -2.8241e-03, -2.5924e-02, -2.4803e-03,  3.4630e-03,
         2.8814e-03, -2.0070e-02, -8.8319e-03,  2.0942e-02, -2.8999e-02,
        -2.3029e-03,  1.6606e-03,  4.2747e-02,  1.6711e-02, -1.3553e-01,
         2.1452e-03,  1.7066e-03,  6.0946e-03, -1.2358e-02,  1.9996e-02,
        -1.0386e-02,  4.0221e-03,  5.0538e-03, -2.7762e-03,  4.1404e-03,
         8.5159e-03, -1.8130e-02, -3.5422e-04,  2.8079e-02,  3.4981e-03,
         1.0684e-02, -2.5297e-03,  4.1816e-03, -5.8102e-02, -3.6709e-03,
        -5.3132e-03, -1.9397e-03, -1.8783e-03,  2.4813e-03,  6.8535e-03,
        -1.4588e-04,  1.8344e-03, -9.6827e-03, -7.5208e-04, -4.5722e-03,
        -4.9322e-03, -6.2949e-02,  3.0955e-03,  6.1849e-03,  1.0936e-02,
        -3.1708e-03, -5.9977e-03,  1.9939e-02, -8.8832e-03, -2.4500e-03,
        -3.5370e-03,  5.4114e-03, -1.6380e-02, -2.6274e-03,  2.8672e-02,
         4.7562e-03,  1.0661e-02,  1.4022e-02, -8.1831e-03,  1.3004e-02,
        -3.5397e-03, -1.6019e-02,  9.7977e-03, -2.2494e-03,  7.8593e-03,
        -7.8541e-04, -1.9319e-02, -1.4239e-03,  9.4747e-03, -5.0362e-03,
         8.7974e-03,  3.4186e-04,  1.0775e-02,  1.4931e-04,  3.2466e-03,
```

```
         1.1351e-02,  1.3178e-02, -2.1573e-02, -5.8410e-02, -6.6791e-02,
         6.6548e-03, -1.8539e-03, -2.7700e-03, -4.2696e-03, -2.2026e-02,
         8.4282e-02,  5.9778e-03,  1.2256e-02, -5.8319e-02, -2.6154e-03,
        -2.7286e-02,  3.4403e-02, -1.4387e-02, -2.9549e-02, -2.5522e-03,
         2.7108e-03, -3.3801e-03,  1.4983e-01, -8.7044e-04, -7.5595e-03,
        -2.3168e-03,  1.2565e-01,  9.5233e-03, -1.2242e-02,  5.8856e-03,
         1.0880e-02, -3.4230e-03, -4.5235e-03, -2.0706e-03,  1.4116e-03,
         8.5681e-03])
tensor([-6.4043e-04,  4.3035e-03, -5.7558e-03, -5.8288e-03,  4.8408e-03,
         2.8846e-03,  4.0324e-03,  1.3545e-02,  1.4554e-02,  1.6135e-02,
        -1.1204e-02, -2.5582e-02, -1.7881e-02,  4.5461e-02,  6.0727e-03,
        -1.2669e-02, -2.1301e-04, -4.0677e-02,  2.2690e-03,  1.3446e-02,
        -6.0979e-03,  1.2405e-02,  4.0320e-03,  1.9189e-02,  8.9538e-03,
        -5.3335e-02,  8.1569e-03,  7.0884e-02,  3.0883e-02,  1.9677e-03,
         9.2808e-03, -4.0901e-02,  1.2547e-02, -3.5514e-02,  6.2157e-03,
         4.4407e-03, -4.3008e-03,  4.0827e-03,  1.0701e-02, -6.2244e-04,
         8.7419e-03,  1.0889e-02, -8.6946e-04, -5.1258e-03, -2.0711e-03,
        -5.0712e-02,  1.2351e-02,  6.7438e-03, -2.3804e-02,  2.5025e-02,
        -4.8326e-03,  2.2336e-02,  1.5806e-02,  2.3617e-02,  1.2529e-04,
         8.2794e-03, -3.5330e-03,  1.1711e-03,  2.9632e-02, -1.0668e-02,
         1.3697e-02, -8.8610e-03, -2.8811e-02,  1.5545e-03, -2.7583e-03,
         1.3127e-02, -1.5746e-02, -4.4114e-03, -6.7454e-02, -9.6840e-03,
         7.2352e-03, -1.5388e-02,  1.8390e-02, -4.1591e-03,  1.6563e-02,
         3.0011e-02,  1.9752e-02,  1.3216e-02,  6.3235e-03, -2.0720e-04,
        -5.5907e-03,  1.0820e-03,  5.0294e-03, -1.1004e-02, -6.5970e-03,
         5.2629e-03,  4.3525e-03, -6.4488e-03,  5.1525e-03, -2.7001e-03,
        -3.6273e-03, -1.6496e-02,  5.8194e-03,  4.3739e-03, -1.3511e-02,
         1.6041e-02,  4.0747e-03,  8.5431e-04, -3.9646e-04, -6.3598e-03,
         2.4504e-02,  1.6434e-03,  1.0776e-02,  6.0369e-03,  7.0922e-03,
         3.4794e-03,  2.9728e-02, -1.1011e-02,  2.8849e-02,  1.0042e-02,
         1.2138e-02,  1.9273e-03,  1.9752e-02,  1.3356e-02,  1.1716e-04,
        -8.3409e-03,  1.1916e-02,  1.0129e-02, -2.0396e-02,  4.9966e-02,
        -1.9799e-02, -3.4090e-03,  7.0486e-03, -8.5078e-03,  1.9286e-03,
         3.0250e-02, -1.6860e-03,  7.6582e-03,  4.2271e-04,  1.4631e-02,
        -1.0228e-02,  2.4979e-02,  1.2868e-02, -2.5713e-02, -1.1087e-02,
         1.7545e-02, -3.5027e-02, -5.9124e-03, -6.3025e-03,  3.2046e-04,
         4.0554e-03, -1.2666e-03, -1.3798e-02,  6.2062e-03,  4.3028e-03,
         3.2487e-03, -3.1336e-03, -1.8145e-02,  3.0065e-02, -2.5023e-02,
         3.6564e-03,  2.7464e-05,  1.1044e-02,  3.4557e-02, -6.7390e-02,
         6.0509e-03,  1.6777e-02,  1.1249e-02, -1.4206e-02,  1.1888e-02,
        -3.1224e-03,  5.7940e-03, -1.9165e-03,  3.9403e-03,  9.1012e-03,
         1.5606e-02, -1.7860e-02,  2.9605e-03,  1.4533e-02,  6.2620e-03,
         2.2195e-03, -8.2523e-04,  4.1658e-03, -4.0763e-02, -4.8609e-03,
        -2.2195e-03,  1.7842e-04, -8.4528e-03, -2.5193e-03,  6.3279e-03,
         3.4990e-03, -4.0801e-03, -4.3370e-03,  5.4080e-04, -3.8281e-03,
         3.7344e-03, -1.2450e-02,  1.8577e-02,  1.3303e-02,  2.0378e-02,
        -2.0793e-03, -1.1637e-02,  1.8081e-02, -6.1786e-03, -1.0230e-03,
        -8.7143e-04,  8.2642e-03, -1.3517e-02, -6.1384e-03,  1.1348e-02,
```

```
         4.8665e-03,  1.9644e-02,  3.7877e-02, -9.6000e-03,  2.3775e-02,
        -8.9394e-03, -2.1650e-02,  1.2365e-02, -7.6206e-04,  6.5019e-04,
        -1.5452e-04, -6.2758e-03,  9.6529e-03,  1.1798e-03, -1.5752e-02,
         5.4191e-03,  4.4593e-03,  3.0009e-02,  7.6876e-03,  1.5748e-02,
         1.7596e-02,  1.3488e-02, -3.1898e-02, -4.5709e-02, -3.1123e-02,
         1.5290e-02, -4.2589e-03,  1.2452e-03, -8.5695e-03, -1.6996e-02,
         5.4815e-02,  7.2219e-03, -1.6786e-03, -6.4784e-03,  3.2342e-03,
         3.4736e-03,  3.6348e-02, -3.4067e-02, -1.5582e-02,  1.7814e-04,
         8.2064e-03,  3.1732e-03,  5.2718e-02, -2.2406e-03,  5.9935e-03,
        -5.4746e-03,  2.9092e-02,  3.7303e-03, -9.5015e-03,  1.0998e-02,
         2.0897e-02, -2.2396e-03, -1.5104e-03,  2.8327e-03,  7.0598e-04,
         1.8769e-02])
tensor([[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,
```

```
[[[-8.9119e-03, -1.3375e-02, -5.4698e-03],
  [-1.1567e-02, -2.3099e-02, -1.2891e-02],
  [-4.1107e-03, -1.9535e-02, -1.9226e-02]],

 [[ 1.1081e-02, -5.7313e-05, -9.6381e-03],
  [ 6.8586e-03, -8.1367e-03, -2.5368e-02],
  [ 9.3080e-03, -1.1911e-02, -2.2293e-02]],

 [[-6.1069e-03, -2.7050e-03, -1.7198e-02],
  [-2.3102e-02, -1.6215e-02, -2.8689e-02],
  [-2.6713e-02, -3.1748e-02, -3.4969e-02]],

 …,

 [[-3.3413e-02, -3.3653e-02, -1.9726e-02],
  [-4.8846e-02, -3.9405e-02, -2.2165e-02],
  [-5.0684e-02, -3.6494e-02, -2.8628e-02]],

 [[-1.9439e-02, -1.0860e-02,  9.3326e-03],
  [-5.6056e-02, -3.0172e-02, -9.8621e-03],
  [-4.6089e-02, -1.1989e-02,  5.4356e-03]],

 [[ 1.3700e-02,  4.8066e-03, -1.8804e-05],
  [-3.8842e-03, -9.0877e-03, -1.1709e-02],
  [-1.5504e-02, -1.9688e-02, -1.7092e-02]]],


[[[ 2.3482e-03, -6.2377e-03, -7.8793e-03],
  [ 5.4411e-03, -1.9673e-03, -2.4044e-03],
  [ 1.7301e-03, -1.1222e-02, -9.8536e-03]],

 [[ 3.0645e-03, -1.1595e-02,  1.4732e-03],
  [-2.5259e-02, -3.4598e-02, -2.3261e-02],
  [-1.1472e-02, -9.3049e-03,  1.5899e-03]],

 [[ 2.6531e-02,  2.0232e-02,  2.5319e-02],
  [-5.4740e-03, -6.7698e-03,  6.0643e-03],
  [ 4.7199e-03, -4.6051e-05,  8.0722e-03]],

 …,

 [[ 2.4850e-02,  2.0613e-02,  2.9125e-02],
  [ 1.4565e-02,  1.2803e-02,  9.1194e-03],
  [-4.2453e-03, -1.2596e-02, -1.6166e-02]],

 [[ 4.6838e-02,  2.0691e-02,  1.7216e-02],
  [-4.9293e-03, -2.6215e-02, -1.9247e-02],
  [-2.9859e-03, -1.8604e-02, -9.3489e-03]],
```

```
          [[ 3.2498e-02,  2.5490e-02,  3.1026e-02],
           [-6.0993e-03, -6.2669e-03,  1.0924e-02],
           [-1.2054e-02, -1.0345e-02,  4.4465e-03]]],


         [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([ 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  7.1649e-03,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -2.4600e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  1.3327e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	-2.2718e-02,	0.0000e+00,	-6.0244e-03,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	-5.6444e-04,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	1.5316e-01,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	-5.9227e-08,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	4.8252e-07,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	1.8982e-02,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	-7.4432e-02,	0.0000e+00,
0.0000e+00,	-1.1148e-02,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	-6.9848e-03,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
-1.2378e-02,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
9.0317e-03,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	5.8072e-02,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,	0.0000e+00,
0.0000e+00,	0.0000e+00,	0.0000e+00,	-2.8794e-04,	0.0000e+00,

```
          0.0000e+00,  0.0000e+00,  3.5912e-04,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00, -1.9328e-04,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -1.8402e-03, -9.4683e-05,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  6.3642e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -8.6028e-02, -6.6755e-03,  0.0000e+00])
tensor([[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[ 4.3886e-02,  7.9370e-03, -1.8935e-02],
           [-5.5253e-03, -1.6901e-02, -2.7706e-02],
           [-9.2463e-02, -5.4414e-02, -3.6953e-02]],

          [[-3.4633e-02,  4.9792e-03,  1.0212e-02],
           [-5.7632e-03,  4.8419e-02,  3.5340e-02],
           [-6.1161e-02,  9.9676e-03,  2.0821e-02]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


         [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 2.3225e-02,  6.3664e-04, -2.3159e-02],
  [-2.4011e-02, -2.6030e-02, -3.5148e-04],
  [-1.5974e-02, -1.0553e-02,  1.7476e-02]],

 [[ 5.0040e-03, -2.1094e-03, -4.1300e-03],
  [ 2.0263e-02, -2.8983e-02, -4.2108e-02],
  [ 1.0988e-02, -4.9443e-02, -3.9851e-02]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,
```

```
[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[-3.7226e-01, -2.7421e-01, -1.3562e-01],
  [-3.7631e-01, -2.4080e-01, -1.0726e-01],
  [-3.1667e-01, -1.8900e-01, -8.6765e-02]],

 [[-3.6818e-01, -4.1408e-01, -2.8099e-01],
  [-4.8669e-01, -5.3330e-01, -3.2798e-01],
  [-4.0849e-01, -4.5344e-01, -2.9042e-01]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[-2.0707e-05,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


         [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([ 4.9599e-02, -9.8315e-03,  0.0000e+00, -1.2284e-03,  1.4348e-01,
         6.3397e-02, -2.4136e-03, -6.5481e-04,  4.5998e-02,  0.0000e+00,
         1.8859e-04, -1.3483e-03,  4.4193e-02, -2.8302e-05,  0.0000e+00,
         4.2238e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00, -2.0923e-03,
        -1.0108e-04,  9.8828e-06, -9.0062e-09, -2.5384e-03,  0.0000e+00,
        -2.9093e-02,  0.0000e+00, -2.6490e-03,  0.0000e+00, -1.0872e-02,
         3.0250e-02, -1.4388e-02,  3.1340e-03,  0.0000e+00, -1.3478e-03,
         7.6211e-05, -1.5842e-02,  9.8957e-02,  1.1617e-02, -4.4254e-03,
         7.1331e-04,  3.5094e-03,  3.5561e-04, -1.0417e-02,  0.0000e+00,
         1.5991e-02,  2.6220e-03,  0.0000e+00, -5.0443e-03, -3.2154e-04,
         0.0000e+00,  1.3173e-02, -7.2563e-03,  6.4720e-03, -6.6924e-03,
        -2.9330e-03,  1.1429e-02,  0.0000e+00,  7.0508e-04,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  2.5451e-03,  1.6721e-03,  2.0295e-02,
         5.6233e-02, -3.0302e-02,  4.5007e-02, -2.6775e-02,  4.4688e-02,
         1.3512e-07,  0.0000e+00,  2.2136e-03,  0.0000e+00, -6.3780e-03,
         6.9259e-05, -2.2030e-02, -1.5173e-03, -5.4830e-03, -5.3689e-03,
         0.0000e+00,  0.0000e+00, -5.5805e-04, -1.7699e-02, -5.7409e-03,
```

```
 3.7536e-03, -1.2917e-02,  1.3105e-02,  1.4588e-02, -2.8751e-03,
 6.0421e-03,  4.9761e-04,  0.0000e+00, -5.2442e-06, -4.2377e-04,
 0.0000e+00,  1.2430e-05,  2.5920e-03,  0.0000e+00,  6.3830e-04,
-4.5791e-03, -1.9377e-02, -1.7031e-02, -5.2204e-03,  2.7392e-04,
-1.0871e-03,  2.3569e-04,  2.0969e-02,  4.3876e-03,  7.3478e-02,
-3.3749e-02, -5.3203e-04, -8.1317e-05, -4.5627e-03, -9.8018e-02,
-4.5185e-03,  1.1915e-02, -1.7568e-03,  0.0000e+00,  1.0538e-02,
-3.6466e-02, -3.6676e-02,  0.0000e+00,  0.0000e+00, -4.5126e-03,
 1.7277e-04, -4.3141e-03, -6.9548e-04,  2.7312e-05,  1.4365e-02,
 2.4588e-05,  0.0000e+00, -5.9429e-04, -3.7142e-02,  5.1878e-03,
 5.7982e-02, -1.2573e-05,  0.0000e+00,  0.0000e+00,  1.3125e-04,
 1.0426e-03, -1.6010e-02,  1.2426e-01,  6.0034e-05,  0.0000e+00,
 0.0000e+00,  0.0000e+00, -7.5745e-03, -1.1041e-02, -1.3682e-04,
 0.0000e+00,  0.0000e+00,  3.2482e-02,  5.9888e-04,  0.0000e+00,
 2.4196e-02,  0.0000e+00,  5.9985e-04,  2.8041e-03,  2.8739e-02,
 4.5359e-04,  3.9050e-02,  0.0000e+00,  0.0000e+00,  1.0163e-01,
 6.6557e-06, -8.1890e-04,  6.3957e-03,  4.6776e-04,  6.2661e-03,
 2.6847e-03, -5.0710e-05, -2.5011e-04,  0.0000e+00,  1.5541e-01,
 7.4432e-03,  0.0000e+00,  1.9172e-03,  2.1739e-02,  3.3327e-04,
-3.0809e-04, -2.5006e-02,  4.1057e-03,  1.4993e-03,  1.8659e-02,
 0.0000e+00, -1.4299e-03, -1.4716e-02,  5.0083e-03,  6.1570e-02,
-1.2661e-03, -2.1152e-04,  3.3760e-04,  1.1887e-01, -3.3025e-02,
-1.5394e-03,  3.3251e-02,  3.3842e-02,  0.0000e+00,  7.5330e-06,
 4.5966e-02, -6.1327e-02,  0.0000e+00,  2.8282e-03,  1.3507e-02,
-8.3417e-04,  1.2259e-02,  5.9982e-04, -2.9907e-03, -3.1169e-04,
-1.0329e-01, -2.2041e-04, -2.3192e-03,  1.2183e-02,  5.8426e-02,
 2.4488e-02,  0.0000e+00,  0.0000e+00,  7.8652e-05,  0.0000e+00,
 0.0000e+00,  0.0000e+00, -7.0616e-04, -3.4642e-02, -8.7881e-06,
 5.6912e-05,  5.0856e-02,  1.0789e-01, -2.2416e-02,  0.0000e+00,
-1.7597e-02,  0.0000e+00,  0.0000e+00, -5.7437e-03,  2.6340e-05,
-5.2106e-04,  0.0000e+00,  3.6286e-03, -3.9666e-02, -1.1780e-01,
 0.0000e+00,  1.0703e-02,  0.0000e+00, -4.1367e-03,  2.9867e-03,
 0.0000e+00, -1.0819e-01,  0.0000e+00,  1.6834e-04,  8.3768e-03,
 0.0000e+00, -1.3073e-01,  6.8818e-03,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  3.6491e-04, -1.2409e-03,  0.0000e+00,
-1.5530e-04, -9.9740e-02, -4.3211e-02, -1.7113e-03,  0.0000e+00,
 3.2943e-02, -2.9400e-02, -2.6064e-03,  7.3598e-03,  1.1823e-01,
 0.0000e+00,  3.4336e-05, -6.5890e-04,  0.0000e+00,  1.4080e-02,
 1.4127e-02,  4.0371e-03,  2.7338e-03, -4.5741e-04,  1.9403e-03,
 1.3723e-03,  1.2655e-03,  0.0000e+00,  2.4412e-02, -1.2392e-02,
 2.2499e-04, -5.9053e-02,  1.0302e-04, -1.2400e-03, -2.5366e-03,
-3.1567e-04,  4.4864e-03,  0.0000e+00,  1.9306e-05,  8.9587e-05,
 7.3379e-03, -2.0510e-04, -3.4454e-04,  0.0000e+00, -2.6318e-04,
-6.8214e-03,  5.1377e-02, -8.0698e-05, -1.3005e-02,  0.0000e+00,
-2.1507e-02,  4.8551e-03,  0.0000e+00, -1.5810e-02,  0.0000e+00,
 5.3659e-04, -1.8490e-02,  1.6454e-02,  1.8124e-02, -5.7346e-03,
-5.1035e-04,  3.0134e-04,  0.0000e+00, -3.7401e-04,  2.8993e-03,
 2.0165e-03, -1.0793e-03, -1.5651e-03,  9.3845e-02,  2.7072e-03,
```

```
         3.0773e-02,  1.8458e-03,  0.0000e+00,  8.7283e-02,  0.0000e+00,
         1.4574e-03,  3.0939e-03,  3.9415e-02,  0.0000e+00,  4.1816e-02,
         0.0000e+00,  0.0000e+00,  2.3174e-02,  3.2992e-02, -2.9073e-02,
        -7.7448e-04, -2.8492e-02, -1.1122e-02,  6.0029e-04,  6.1286e-02,
         1.5712e-01, -5.4086e-02,  0.0000e+00,  0.0000e+00, -3.5057e-04,
        -4.5227e-05,  2.7256e-04, -4.6670e-03,  7.7039e-03, -1.2096e-03,
         2.0082e-05,  2.8483e-03,  0.0000e+00,  1.2407e-02, -6.7886e-06,
        -4.6088e-05,  4.3034e-04, -1.9872e-02,  0.0000e+00,  0.0000e+00,
        -2.8163e-02,  2.5286e-02, -1.4144e-04,  7.7388e-04,  3.0350e-02,
        -4.3799e-04,  7.6797e-04,  0.0000e+00, -5.7810e-02, -2.9085e-03,
         3.4409e-06,  1.7718e-02,  6.5968e-03,  0.0000e+00, -3.8343e-03,
         0.0000e+00, -1.0056e-01, -1.1165e-03,  0.0000e+00])
tensor([[[[ 2.2039e-04,  1.5606e-02,  1.3762e-04],
          [-3.6693e-03,  1.6380e-02,  8.3339e-03],
          [ 1.2164e-04,  9.0997e-03, -6.1355e-04]],

         [[ 0.0000e+00,  0.0000e+00,  1.2961e-02],
          [ 7.5841e-05,  6.1456e-05,  9.1848e-03],
          [ 1.1767e-05,  1.8579e-04,  7.9332e-03]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  2.6496e-03,  7.8807e-05],
          [ 1.3447e-06,  2.4817e-03,  1.4588e-03],
          [ 8.6601e-08,  1.1822e-03,  0.0000e+00]],

         [[ 0.0000e+00, -2.4983e-05, -7.4793e-05],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,
```

```
[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


         [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([ 6.0101e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00, -1.4062e-04,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  4.8387e-01,  5.0297e-02,
         6.4828e-02,  0.0000e+00, -1.7041e-02,  0.0000e+00,  0.0000e+00,
         4.3270e-02,  0.0000e+00,  0.0000e+00, -1.8988e-05,  0.0000e+00,
        -6.9511e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00, -8.6989e-02,
        -1.9806e-01,  8.3726e-03,  3.1227e-02,  0.0000e+00, -5.3647e-04,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -2.8645e-02, -2.0867e-03,  0.0000e+00, -6.0676e-02,  4.5394e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  8.2895e-03,
         0.0000e+00,  0.0000e+00, -6.7480e-02,  5.1612e-01,  0.0000e+00,
         1.2841e-01,  0.0000e+00,  0.0000e+00,  0.0000e+00,  3.2808e-01,
         0.0000e+00,  1.2695e-02,  0.0000e+00, -2.1228e-02,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  7.8568e-02,  7.9144e-03,  1.1424e-04,
        -1.2172e-02,  5.0621e-02,  9.9506e-02, -3.6981e-05,  0.0000e+00,
         0.0000e+00, -7.0658e-03,  0.0000e+00,  1.3942e-06,  0.0000e+00,
```

```
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -9.0296e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -3.5800e-02,  0.0000e+00, -4.0968e-02,  4.6763e-02,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -7.0738e-02,
          0.0000e+00,  2.0082e-06, -1.3818e-02, -6.8412e-02,  0.0000e+00,
         -2.4829e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  2.5236e-03,
          0.0000e+00,  0.0000e+00,  0.0000e+00, -1.1372e-02, -8.4525e-02,
          0.0000e+00,  0.0000e+00, -1.8443e-01,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00, -6.1364e-06,  0.0000e+00,
         -7.4515e-02,  0.0000e+00,  1.4472e-03, -3.4988e-05, -1.4000e-02,
          0.0000e+00,  0.0000e+00,  2.8244e-02, -6.2592e-02, -4.4142e-02,
         -7.3246e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  4.8015e-03,  0.0000e+00,  0.0000e+00,
          1.3368e-01,  0.0000e+00,  0.0000e+00, -1.7432e-03,  0.0000e+00,
          0.0000e+00, -1.3771e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00, -6.4030e-02,  0.0000e+00, -5.2267e-05,
         -9.1404e-02,  3.9246e-06,  0.0000e+00,  0.0000e+00,  2.5641e-04,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          1.1838e-04,  1.4027e-01,  0.0000e+00,  1.5028e-01, -2.4170e-02,
         -9.4118e-02,  0.0000e+00,  0.0000e+00, -2.6025e-02,  8.2311e-02,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  2.5074e-03,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  5.1243e-02,  0.0000e+00,  0.0000e+00,
          0.0000e+00, -4.7442e-02, -1.0860e-01,  0.0000e+00, -4.0598e-03,
         -2.0161e-04,  1.0736e-01,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          1.4012e-01,  0.0000e+00,  0.0000e+00, -3.8492e-01,  0.0000e+00,
          0.0000e+00,  0.0000e+00, -1.8470e-03,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  1.9218e-02,  0.0000e+00,
          0.0000e+00,  1.5534e-01, -7.2347e-02, -3.6753e-05,  0.0000e+00,
         -5.6121e-02,  0.0000e+00,  1.2860e-01,  5.4082e-02, -6.9098e-07,
         -1.8221e-03,  0.0000e+00,  0.0000e+00, -2.1570e-05,  4.3678e-02,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -1.0021e-03,
         -1.6647e-03,  0.0000e+00,  0.0000e+00, -1.2604e-01,  0.0000e+00,
          0.0000e+00, -3.9835e-02, -1.0544e-01,  0.0000e+00,  0.0000e+00,
          0.0000e+00])
tensor([[[-7.8434e-05,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
            0.0000e+00,  0.0000e+00],
         [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
            0.0000e+00,  0.0000e+00],
         [-1.6707e-04,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
            0.0000e+00,  0.0000e+00],
         …,
         [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
            0.0000e+00,  0.0000e+00],
         [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
            0.0000e+00,  0.0000e+00],
         [ 0.0000e+00,  0.0000e+00,  0.0000e+00,  …,  0.0000e+00,
            0.0000e+00,  0.0000e+00]])
```

```
tensor([-1.1233e-04,  0.0000e+00, -4.0196e-03,  …,  0.0000e+00,
        -1.6401e-05,  0.0000e+00])
tensor([[0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        …,
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.]])
tensor([0., 0., 0.,  …, 0., 0., 0.])
tensor([[0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        …,
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.]])
tensor([ 4.0033e-02, -1.8221e-02, -2.8827e-02,  3.9573e-02,  3.2860e-02,
        -3.3329e-02, -1.2412e-02, -1.1789e-02, -1.3465e-01,  1.6942e-02,
         3.0577e-02, -6.5244e-02,  2.4688e-02,  1.8936e-02,  4.2574e-03,
         1.6542e-02,  2.4166e-02,  4.1301e-02,  1.0683e-01, -9.7942e-02,
         9.8010e-05,  6.4653e-06,  5.9928e-06,  5.5417e-06,  6.6799e-06,
         5.6123e-06,  5.5321e-06,  6.3025e-06,  5.5742e-06,  5.5472e-06,
         5.5189e-06,  5.8988e-06,  5.5308e-06,  5.5378e-06,  5.8537e-06,
         6.0597e-06,  5.5650e-06,  5.5610e-06,  5.7812e-06,  5.5488e-06,
         5.5590e-06,  5.7655e-06,  5.8575e-06,  5.6430e-06,  5.7514e-06,
         5.5387e-06,  5.5403e-06,  5.6077e-06,  5.5177e-06,  6.1433e-06,
         5.5447e-06,  5.5780e-06,  5.5431e-06,  6.1245e-06,  5.5291e-06,
         5.5313e-06,  5.5426e-06,  5.5631e-06,  5.7984e-06,  5.5287e-06,
         5.5517e-06,  5.5312e-06,  5.7146e-06,  5.7164e-06,  5.6359e-06,
         5.5359e-06,  5.7319e-06,  5.5394e-06,  6.5611e-06,  5.8061e-06,
         5.5339e-06,  5.5527e-06,  6.0271e-06,  5.5821e-06,  5.5056e-06,
         6.5936e-06,  5.5493e-06,  5.6128e-06,  5.6044e-06,  5.9392e-06,
         5.5948e-06,  5.5979e-06,  5.7844e-06,  5.5304e-06,  5.5470e-06,
         5.9234e-06,  5.5288e-06,  5.5811e-06,  5.6155e-06,  6.1083e-06,
         5.6866e-06,  5.5884e-06,  5.5846e-06,  6.0113e-06,  5.7536e-06,
         5.5491e-06,  5.5391e-06,  5.6260e-06,  5.8043e-06,  5.5244e-06,
         5.5162e-06,  5.9175e-06,  5.5630e-06,  5.5815e-06,  5.5487e-06,
         5.6555e-06,  5.5208e-06,  5.5873e-06,  5.8631e-06,  5.7873e-06,
         5.5363e-06,  5.6005e-06,  5.5240e-06,  6.6837e-06,  5.5200e-06,
         5.7789e-06,  5.5254e-06,  5.6278e-06,  5.6963e-06,  5.9705e-06,
         6.4912e-06,  5.6723e-06,  6.5924e-06,  5.7232e-06,  5.5810e-06,
         5.7660e-06,  5.5308e-06,  5.5766e-06,  5.5252e-06,  5.5592e-06,
         5.9384e-06,  5.8355e-06,  5.8623e-06,  5.5389e-06,  5.5478e-06,
         6.0590e-06,  5.5246e-06,  5.7303e-06,  6.5027e-06,  5.7682e-06,
         5.5923e-06,  5.7851e-06,  5.5154e-06,  5.9460e-06,  6.1967e-06,
         5.5072e-06,  6.0426e-06,  5.5799e-06,  5.6654e-06,  5.5836e-06,
         5.5721e-06,  5.5503e-06,  5.5194e-06,  5.8524e-06,  6.0686e-06,
```

```
5.9091e-06,  6.0330e-06,  5.5129e-06,  5.6526e-06,  5.5686e-06,
5.5353e-06,  6.3634e-06,  5.6589e-06,  5.5098e-06,  5.6548e-06,
6.4832e-06,  5.5311e-06,  5.7985e-06,  5.6807e-06,  5.5313e-06,
5.5726e-06,  5.5606e-06,  5.6108e-06,  5.7893e-06,  5.5260e-06,
5.5490e-06,  5.5144e-06,  5.6197e-06,  5.5575e-06,  5.5768e-06,
5.5469e-06,  6.0006e-06,  5.7295e-06,  5.5352e-06,  6.3666e-06,
5.6189e-06,  5.7896e-06,  5.7747e-06,  5.5420e-06,  5.5979e-06,
5.9025e-06,  5.5524e-06,  5.5371e-06,  5.5139e-06,  5.7703e-06,
5.9974e-06,  5.9096e-06,  5.5309e-06,  5.6011e-06,  5.7906e-06,
5.8984e-06,  5.6341e-06,  6.1157e-06,  5.6224e-06,  5.6533e-06,
5.5599e-06,  5.5291e-06,  5.5482e-06,  5.7654e-06,  5.5320e-06,
5.9507e-06,  5.5351e-06,  6.2518e-06,  5.5727e-06,  5.7224e-06,
5.6628e-06,  5.5655e-06,  5.5996e-06,  5.5270e-06,  5.7788e-06,
5.7241e-06,  5.7274e-06,  5.7417e-06,  5.8885e-06,  6.5153e-06,
5.5708e-06,  5.9389e-06,  5.5719e-06,  6.3077e-06,  5.8000e-06,
5.5182e-06,  5.6427e-06,  5.5458e-06,  5.5446e-06,  5.5270e-06,
5.5291e-06,  5.5473e-06,  5.5801e-06,  5.6793e-06,  5.5316e-06,
5.8673e-06,  5.8369e-06,  5.5226e-06,  5.5126e-06,  5.5401e-06,
5.5088e-06,  5.7011e-06,  5.5629e-06,  5.6249e-06,  5.5330e-06,
5.5664e-06,  5.5346e-06,  6.3609e-06,  6.3704e-06,  6.0002e-06,
5.5318e-06,  5.6692e-06,  5.5635e-06,  5.5573e-06,  5.6373e-06,
6.0652e-06,  5.5595e-06,  6.0576e-06,  5.5082e-06,  5.5255e-06,
5.6627e-06,  5.5290e-06,  5.5032e-06,  5.6865e-06,  5.5021e-06,
5.6423e-06,  5.9346e-06,  5.5114e-06,  5.7110e-06,  6.1108e-06,
5.6138e-06,  5.5449e-06,  6.4282e-06,  5.7765e-06,  5.5175e-06,
5.6069e-06,  5.5201e-06,  5.5832e-06,  5.6254e-06,  5.5198e-06,
6.9827e-06,  5.8146e-06,  6.2689e-06,  5.8134e-06,  5.5459e-06,
6.8627e-06,  5.6208e-06,  5.7001e-06,  5.6573e-06,  5.5604e-06,
5.6272e-06,  5.5324e-06,  5.6871e-06,  5.5567e-06,  5.6493e-06,
5.5508e-06,  5.5641e-06,  5.7786e-06,  5.6173e-06,  5.7220e-06,
5.6650e-06,  5.5482e-06,  5.9883e-06,  5.5266e-06,  5.9799e-06,
5.8466e-06,  5.6250e-06,  5.7403e-06,  5.8312e-06,  5.6264e-06,
5.5258e-06,  5.5367e-06,  5.9632e-06,  5.6819e-06,  5.5649e-06,
5.5547e-06,  5.5137e-06,  6.4621e-06,  5.8243e-06,  5.5269e-06,
5.5030e-06,  5.5659e-06,  6.1331e-06,  5.5126e-06,  5.6016e-06,
5.6974e-06,  5.6112e-06,  5.7680e-06,  5.5084e-06,  6.2276e-06,
5.5068e-06,  5.6493e-06,  5.6848e-06,  5.5418e-06,  5.6641e-06,
5.7052e-06,  6.3057e-06,  5.5608e-06,  5.6066e-06,  6.8045e-06,
6.0185e-06,  5.5693e-06,  5.7450e-06,  5.5430e-06,  5.6547e-06,
5.8178e-06,  5.5276e-06,  5.7215e-06,  5.5155e-06,  5.4979e-06,
5.6830e-06,  5.6336e-06,  5.5837e-06,  5.9927e-06,  5.6419e-06,
5.5359e-06,  5.5798e-06,  6.7584e-06,  6.0467e-06,  5.5213e-06,
5.5295e-06,  6.5205e-06,  5.7710e-06,  5.6553e-06,  5.8113e-06,
5.6793e-06,  5.5646e-06,  5.5934e-06,  5.8856e-06,  5.5952e-06,
5.7709e-06,  5.5050e-06,  6.9720e-06,  5.5361e-06,  6.0440e-06,
5.5095e-06,  5.8964e-06,  5.6069e-06,  5.6366e-06,  5.7900e-06,
5.5203e-06,  5.6100e-06,  5.5908e-06,  5.5267e-06,  6.2031e-06,
5.7114e-06,  5.5369e-06,  5.5074e-06,  5.5104e-06,  5.5231e-06,
```

5.6805e-06, 5.5802e-06, 5.7775e-06, 5.5804e-06, 5.6392e-06,
5.9139e-06, 5.5453e-06, 5.5725e-06, 6.0549e-06, 6.0778e-06,
5.5048e-06, 5.5793e-06, 5.7672e-06, 5.7282e-06, 5.8968e-06,
5.5333e-06, 5.6061e-06, 5.4987e-06, 5.5661e-06, 5.5800e-06,
5.5139e-06, 5.8080e-06, 5.5705e-06, 5.5182e-06, 5.5502e-06,
5.6682e-06, 5.5295e-06, 5.5409e-06, 5.7255e-06, 5.5271e-06,
5.8126e-06, 5.8776e-06, 6.4750e-06, 5.6418e-06, 5.5465e-06,
5.8767e-06, 5.7601e-06, 6.0233e-06, 5.8916e-06, 5.8090e-06,
5.6832e-06, 5.5791e-06, 5.6411e-06, 5.5266e-06, 5.7307e-06,
5.6125e-06, 5.5814e-06, 5.6099e-06, 5.5795e-06, 5.6226e-06,
5.5458e-06, 5.5775e-06, 6.2911e-06, 5.5756e-06, 5.5574e-06,
5.5267e-06, 5.5262e-06, 6.2950e-06, 5.5658e-06, 5.6873e-06,
5.5380e-06, 5.6062e-06, 5.5719e-06, 5.7495e-06, 5.6486e-06,
6.1044e-06, 5.7182e-06, 5.5846e-06, 6.3499e-06, 5.7630e-06,
5.5021e-06, 6.3115e-06, 5.5590e-06, 5.7082e-06, 5.5216e-06,
5.5640e-06, 5.5209e-06, 5.9792e-06, 5.6700e-06, 5.5820e-06,
5.6497e-06, 5.5216e-06, 5.5398e-06, 5.7177e-06, 5.6982e-06,
5.5857e-06, 5.5204e-06, 5.5333e-06, 5.5362e-06, 6.3007e-06,
6.0010e-06, 5.6176e-06, 5.6241e-06, 6.0582e-06, 5.5231e-06,
7.1390e-06, 5.9369e-06, 6.1774e-06, 5.6675e-06, 5.5612e-06,
6.0850e-06, 5.6606e-06, 5.5225e-06, 5.5956e-06, 5.5935e-06,
5.5350e-06, 5.5261e-06, 5.6783e-06, 5.6394e-06, 5.6462e-06,
6.3966e-06, 5.5098e-06, 5.5419e-06, 5.5893e-06, 5.5716e-06,
5.5779e-06, 5.6444e-06, 6.4382e-06, 5.5333e-06, 5.9700e-06,
5.9961e-06, 5.5622e-06, 5.5855e-06, 5.5996e-06, 5.5547e-06,
5.5644e-06, 5.6031e-06, 6.1727e-06, 6.0231e-06, 5.5516e-06,
5.5344e-06, 5.6503e-06, 6.8219e-06, 5.8738e-06, 5.5072e-06,
5.5121e-06, 5.7390e-06, 6.5104e-06, 5.5069e-06, 5.8039e-06,
5.8406e-06, 6.0443e-06, 5.6871e-06, 5.5571e-06, 6.1647e-06,
5.5072e-06, 6.4607e-06, 5.5994e-06, 5.7131e-06, 5.5502e-06,
5.5174e-06, 5.8047e-06, 5.5313e-06, 5.8400e-06, 5.5323e-06,
5.5338e-06, 5.5644e-06, 5.8973e-06, 5.5548e-06, 6.5842e-06,
5.5194e-06, 6.0316e-06, 5.5557e-06, 5.5484e-06, 5.5152e-06,
5.5889e-06, 5.5473e-06, 5.9016e-06, 6.0484e-06, 5.5860e-06,
6.4944e-06, 5.5144e-06, 5.8653e-06, 5.5233e-06, 5.6787e-06,
5.6946e-06, 5.6838e-06, 5.9329e-06, 5.7984e-06, 5.5018e-06,
5.6566e-06, 5.8491e-06, 5.6173e-06, 6.4515e-06, 5.5899e-06,
6.0530e-06, 5.5427e-06, 5.7765e-06, 5.5320e-06, 5.6693e-06,
6.2034e-06, 6.4910e-06, 5.6466e-06, 5.5126e-06, 5.9148e-06,
5.5804e-06, 6.1580e-06, 5.6201e-06, 5.5908e-06, 5.6013e-06,
5.7186e-06, 5.7792e-06, 5.5778e-06, 6.0163e-06, 5.6528e-06,
5.5822e-06, 5.6311e-06, 5.6885e-06, 5.7789e-06, 6.0995e-06,
5.7203e-06, 5.8423e-06, 5.5299e-06, 5.5794e-06, 5.6155e-06,
5.4969e-06, 6.2497e-06, 5.9841e-06, 5.5209e-06, 5.5126e-06,
5.5811e-06, 5.7739e-06, 5.5401e-06, 5.6081e-06, 5.5321e-06,
5.5243e-06, 5.6062e-06, 5.5194e-06, 5.5283e-06, 5.5030e-06,
6.0807e-06, 5.5642e-06, 5.5627e-06, 5.6459e-06, 5.9133e-06,
5.7034e-06, 5.5531e-06, 5.5077e-06, 6.3844e-06, 5.5165e-06,

5.5323e-06, 5.5151e-06, 6.0526e-06, 5.5704e-06, 5.7446e-06,
5.5850e-06, 5.7329e-06, 5.9630e-06, 5.7544e-06, 5.5247e-06,
5.6299e-06, 5.7033e-06, 5.5570e-06, 6.5482e-06, 5.5156e-06,
5.9726e-06, 5.7903e-06, 5.7548e-06, 5.5491e-06, 5.6876e-06,
5.7998e-06, 5.5378e-06, 5.5976e-06, 5.7898e-06, 5.5704e-06,
5.5655e-06, 5.6151e-06, 5.9476e-06, 5.7254e-06, 5.6747e-06,
5.5569e-06, 5.5081e-06, 5.5227e-06, 5.8846e-06, 5.5250e-06,
5.5705e-06, 5.6615e-06, 5.5250e-06, 5.9264e-06, 5.9189e-06,
5.5530e-06, 5.9504e-06, 5.6872e-06, 5.5628e-06, 5.8419e-06,
5.6586e-06, 5.5068e-06, 5.5156e-06, 5.5376e-06, 5.6170e-06,
5.8079e-06, 5.5543e-06, 5.5269e-06, 5.6754e-06, 5.5280e-06,
5.6208e-06, 6.0209e-06, 5.6751e-06, 6.1306e-06, 5.6834e-06,
6.0487e-06, 5.5417e-06, 5.6465e-06, 5.8005e-06, 5.5289e-06,
6.1971e-06, 5.5157e-06, 6.1775e-06, 6.0981e-06, 6.0478e-06,
5.7816e-06, 5.5190e-06, 5.5764e-06, 5.8158e-06, 5.5172e-06,
5.6242e-06, 5.5534e-06, 5.5204e-06, 5.5226e-06, 6.5172e-06,
5.7337e-06, 5.7320e-06, 5.6351e-06, 5.5568e-06, 5.5735e-06,
5.5540e-06, 5.8936e-06, 5.9004e-06, 5.9745e-06, 5.5091e-06,
5.9131e-06, 5.5490e-06, 5.6209e-06, 5.5973e-06, 5.5736e-06,
5.8226e-06, 5.5397e-06, 5.6850e-06, 6.8690e-06, 6.1123e-06,
5.5157e-06, 5.5254e-06, 5.6363e-06, 5.5167e-06, 5.6584e-06,
5.6382e-06, 5.7081e-06, 5.5677e-06, 5.5040e-06, 5.5148e-06,
5.5803e-06, 5.5978e-06, 6.0557e-06, 5.8435e-06, 5.9897e-06,
5.5194e-06, 5.7965e-06, 5.5353e-06, 6.3956e-06, 5.5629e-06,
6.3191e-06, 5.6301e-06, 5.6099e-06, 5.5421e-06, 5.7366e-06,
5.5354e-06, 5.5498e-06, 5.5385e-06, 5.5753e-06, 5.5423e-06,
5.5293e-06, 6.2629e-06, 5.5222e-06, 5.5803e-06, 5.5231e-06,
5.5865e-06, 5.5611e-06, 5.7133e-06, 6.3177e-06, 6.9363e-06,
5.5111e-06, 5.5449e-06, 5.8245e-06, 5.8370e-06, 5.5626e-06,
5.5049e-06, 5.8925e-06, 5.5935e-06, 5.5589e-06, 5.5971e-06,
5.9617e-06, 5.7069e-06, 5.6317e-06, 5.5247e-06, 6.2775e-06,
6.3309e-06, 5.6344e-06, 5.6792e-06, 6.5301e-06, 5.5179e-06,
6.2883e-06, 6.0888e-06, 5.5207e-06, 5.5831e-06, 5.7714e-06,
6.3423e-06, 5.5078e-06, 5.7119e-06, 5.7746e-06, 5.5270e-06,
5.7009e-06, 5.5492e-06, 6.0805e-06, 5.5954e-06, 5.7687e-06,
5.6584e-06, 5.6388e-06, 5.5886e-06, 5.7178e-06, 5.6798e-06,
5.9856e-06, 5.6631e-06, 6.8495e-06, 5.5661e-06, 5.7710e-06,
5.5989e-06, 5.7950e-06, 5.9843e-06, 6.0828e-06, 5.5542e-06,
5.5150e-06, 5.5079e-06, 5.7590e-06, 5.8267e-06, 5.7013e-06,
5.8030e-06, 6.4103e-06, 5.6390e-06, 5.6474e-06, 5.5273e-06,
5.5802e-06, 5.5273e-06, 5.7934e-06, 5.5385e-06, 5.6052e-06,
5.9184e-06, 6.4277e-06, 5.6678e-06, 5.5482e-06, 5.8308e-06,
5.5509e-06, 5.5961e-06, 5.5210e-06, 5.5223e-06, 5.6520e-06,
5.5908e-06, 5.5791e-06, 5.8056e-06, 6.7625e-06, 6.7016e-06,
5.7622e-06, 5.6874e-06, 5.6706e-06, 5.7199e-06, 5.5305e-06,
5.8281e-06, 5.6304e-06, 6.1126e-06, 5.8653e-06, 5.6088e-06,
5.5369e-06, 6.5060e-06, 5.6126e-06, 5.6080e-06, 6.4970e-06,
6.1509e-06, 5.5393e-06, 5.5424e-06, 5.9853e-06, 5.5645e-06,

```
          5.5578e-06,  5.5836e-06,  5.5319e-06,  5.5835e-06,  5.6361e-06,
          5.5318e-06,  5.6494e-06,  5.5318e-06,  5.5212e-06,  6.1990e-06,
          5.6866e-06,  5.5096e-06,  5.7456e-06,  5.5515e-06,  5.6839e-06,
          5.7872e-06,  5.5945e-06,  5.7373e-06,  5.7950e-06,  6.0469e-06,
          5.6494e-06,  5.5783e-06,  5.5653e-06,  5.5341e-06,  5.5176e-06,
          5.5067e-06,  5.5465e-06,  6.0291e-06,  5.5654e-06,  5.8787e-06,
          5.5129e-06,  5.5814e-06,  5.5126e-06,  5.5762e-06,  5.5563e-06,
          5.5793e-06,  5.5133e-06,  5.7065e-06,  6.0496e-06,  6.0080e-06,
          5.6941e-06,  5.6294e-06,  5.6252e-06,  5.5767e-06,  5.6833e-06,
          5.5155e-06,  5.8535e-06,  5.5588e-06,  5.8529e-06,  5.5868e-06,
          6.2756e-06,  5.5440e-06,  6.3937e-06,  5.6973e-06,  5.6455e-06,
          5.6155e-06,  5.5702e-06,  5.6093e-06,  5.5346e-06,  5.6183e-06,
          5.5771e-06,  5.6085e-06,  6.7415e-06,  5.6705e-06,  5.5559e-06,
          5.5637e-06,  5.7896e-06,  5.5262e-06,  5.5211e-06,  5.6396e-06,
          5.5771e-06,  5.8095e-06,  5.8810e-06,  6.7473e-06,  6.2502e-06,
          5.7395e-06,  5.5337e-06,  5.5538e-06,  5.5743e-06,  5.8095e-06,
          5.5228e-06,  5.6504e-06,  5.5217e-06,  5.5417e-06,  5.5231e-06,
          5.5493e-06,  5.5738e-06,  5.6331e-06,  5.6255e-06,  5.5671e-06,
          5.5699e-06,  5.7418e-06,  5.8305e-06,  5.9237e-06,  5.6168e-06,
          5.9805e-06,  5.5542e-06,  5.5697e-06,  5.5177e-06,  5.5836e-06,
          5.5217e-06,  5.5180e-06,  5.6320e-06,  5.7291e-06,  5.6116e-06,
          5.7166e-06,  5.5421e-06,  5.7672e-06,  6.0496e-06,  5.6654e-06,
          5.9923e-06,  6.1189e-06,  5.8172e-06,  5.7653e-06,  5.5303e-06,
          5.6162e-06,  5.7373e-06,  5.5660e-06,  5.5368e-06,  6.2907e-06,
          6.1437e-06,  5.5659e-06,  5.5615e-06,  5.5203e-06,  5.9637e-06])
end of p.grad

False
Epoch 9 finished
Epoch [9/10], Loss: 1.5284
************************************************************

Learning rate: 0.001

another way to print learning rate:
tensor([[[[-2.7791e-03, -3.2911e-03, -3.6152e-03,  …, -3.4367e-03,
           -3.3921e-03, -2.8269e-03],
          [-3.6450e-03, -3.5271e-03, -3.7016e-03,  …, -3.2424e-03,
           -3.0840e-03, -2.3695e-03],
          [-3.9159e-03, -3.9212e-03, -3.7512e-03,  …, -2.8176e-03,
           -2.3642e-03, -1.9664e-03],
          …,
          [-1.4273e-03, -8.9690e-04, -3.2974e-04,  …, -1.5187e-04,
           -8.9974e-05, -2.8279e-04],
          [-7.2926e-04, -4.6218e-04,  1.1811e-04,  …, -8.4309e-04,
           -1.0183e-03, -8.2489e-04],
          [-8.0407e-04, -3.8712e-04, -3.0190e-04,  …, -1.5042e-03,
           -1.7290e-03, -1.5139e-03]],
```

```
 [[-2.0808e-03, -2.3577e-03, -2.7531e-03,  …, -3.1156e-03,
   -2.8677e-03, -2.1531e-03],
  [-3.0269e-03, -2.8147e-03, -2.9789e-03,  …, -3.0292e-03,
   -2.6544e-03, -1.8403e-03],
  [-3.4705e-03, -3.4651e-03, -3.2929e-03,  …, -2.9432e-03,
   -2.3200e-03, -1.7040e-03],
  …,
  [-1.6195e-03, -1.0417e-03, -5.3638e-04,  …, -7.7178e-04,
   -8.8006e-04, -1.0265e-03],
  [-1.0629e-03, -5.6988e-04, -1.3906e-04,  …, -1.4335e-03,
   -1.7466e-03, -1.5582e-03],
  [-1.0748e-03, -4.3956e-04, -4.8131e-04,  …, -1.9314e-03,
   -2.2168e-03, -2.0395e-03]],

 [[-3.9669e-03, -4.1116e-03, -4.2846e-03,  …, -3.5570e-03,
   -3.5753e-03, -3.2052e-03],
  [-4.7978e-03, -4.3925e-03, -4.3450e-03,  …, -3.5449e-03,
   -3.3523e-03, -2.8171e-03],
  [-5.2424e-03, -5.0390e-03, -4.6229e-03,  …, -3.6595e-03,
   -3.1070e-03, -2.7791e-03],
  …,
  [-3.8124e-03, -3.4175e-03, -2.9286e-03,  …, -2.1109e-03,
   -2.2678e-03, -2.7038e-03],
  [-3.2772e-03, -3.1409e-03, -2.6163e-03,  …, -2.5238e-03,
   -3.0096e-03, -3.1503e-03],
  [-2.9485e-03, -2.5282e-03, -2.5470e-03,  …, -2.5852e-03,
   -3.0990e-03, -3.1835e-03]]],


[[[ 4.0442e-02,  4.3517e-02,  4.1012e-02,  …,  4.8327e-02,
    4.7794e-02,  4.4984e-02],
  [ 3.5576e-02,  3.8134e-02,  3.9714e-02,  …,  5.0500e-02,
    4.9839e-02,  4.7621e-02],
  [ 2.8312e-02,  3.3452e-02,  3.7420e-02,  …,  5.0878e-02,
    4.9519e-02,  4.8906e-02],
  …,
  [ 3.2093e-02,  2.8249e-02,  2.4796e-02,  …,  3.5774e-02,
    3.6137e-02,  3.2399e-02],
  [ 3.1833e-02,  2.9704e-02,  2.7104e-02,  …,  3.7911e-02,
    3.3540e-02,  3.0900e-02],
  [ 3.2427e-02,  3.0410e-02,  2.4990e-02,  …,  3.6892e-02,
    3.3352e-02,  2.8579e-02]],

 [[ 3.8876e-03,  6.4575e-03,  3.4586e-03,  …,  4.4127e-03,
    3.9980e-03,  3.2270e-03],
  [-2.5678e-03,  8.3997e-04,  2.6514e-03,  …,  6.4887e-03,
    5.3713e-03,  5.0515e-03],
```

```
  [-9.7481e-03, -3.7965e-03,  3.1179e-04,  …,  6.3300e-03,
    4.2663e-03,  5.7075e-03],
  …,
  [-6.2301e-03, -8.5950e-03, -1.2590e-02,  …, -6.4965e-03,
   -5.3769e-03, -6.7170e-03],
  [-6.3820e-03, -7.4778e-03, -9.9925e-03,  …, -3.6728e-03,
   -7.6311e-03, -8.4367e-03],
  [-4.9077e-03, -6.8415e-03, -1.2797e-02,  …, -3.4555e-03,
   -7.4599e-03, -1.1039e-02]],

 [[-9.8057e-04,  1.8635e-03, -1.6289e-03,  …, -5.2370e-03,
   -6.0529e-03, -6.3292e-03],
  [-6.2935e-03, -4.0386e-03, -3.7327e-03,  …, -5.3285e-03,
   -6.7632e-03, -6.5438e-03],
  [-1.3932e-02, -9.4514e-03, -6.9303e-03,  …, -7.0036e-03,
   -8.8815e-03, -6.1682e-03],
  …,
  [-6.6529e-03, -9.4159e-03, -1.4570e-02,  …, -1.5730e-02,
   -1.5089e-02, -1.5945e-02],
  [-6.0917e-03, -8.0594e-03, -1.2256e-02,  …, -1.0823e-02,
   -1.5523e-02, -1.6363e-02],
  [-5.0418e-03, -7.9349e-03, -1.4141e-02,  …, -9.0322e-03,
   -1.3430e-02, -1.7460e-02]]],


[[[ 2.6860e-03,  1.5835e-03,  1.4249e-03,  …,  2.4538e-03,
    2.2379e-03,  1.7144e-03],
  [ 2.8701e-03,  2.8228e-03,  2.9413e-03,  …,  3.5405e-03,
    3.2881e-03,  2.3936e-03],
  [ 2.8612e-03,  2.9496e-03,  3.0366e-03,  …,  3.9997e-03,
    3.6163e-03,  2.7027e-03],
  …,
  [ 1.9655e-03,  1.9293e-03,  3.0449e-03,  …,  2.9747e-03,
    3.7031e-03,  3.8730e-03],
  [ 1.6280e-03,  1.2388e-03,  2.2373e-03,  …,  1.3191e-03,
    1.4677e-03,  2.1559e-03],
  [ 3.3387e-03,  2.5023e-03,  2.1479e-03,  …,  1.6866e-03,
    8.7497e-04,  8.7985e-04]],

 [[ 3.6222e-05, -1.1663e-03, -1.4897e-03,  …,  3.4136e-04,
    3.9986e-04,  1.6992e-05],
  [ 4.3201e-04,  2.6133e-04,  2.5447e-04,  …,  1.0399e-03,
    8.7820e-04,  2.4379e-04],
  [ 5.5525e-04,  6.0279e-04,  6.7054e-04,  …,  1.1814e-03,
    8.7012e-04,  1.7533e-04],
  …,
  [ 8.5955e-04,  9.2011e-04,  1.9966e-03,  …,  1.4534e-04,
    9.0493e-04,  9.9740e-04],
```

```
  [ 3.7655e-04,  1.6147e-04,  1.0039e-03,  …, -1.9197e-03,
   -1.7815e-03, -1.0747e-03],
  [ 1.5242e-03,  9.2739e-04,  4.9462e-04,  …, -1.6584e-03,
   -2.1685e-03, -2.3916e-03]],

 [[-1.7098e-03, -2.3960e-03, -2.1935e-03,  …, -1.3812e-03,
   -1.0543e-03, -9.7350e-04],
  [-1.2569e-03, -9.1891e-04, -5.9896e-04,  …, -8.6516e-04,
   -6.8000e-04, -8.6638e-04],
  [-1.0494e-03, -6.2479e-04, -1.6745e-04,  …, -7.1728e-04,
   -1.0782e-03, -1.4533e-03],
  …,
  [-3.5365e-04, -2.9778e-04,  5.1247e-04,  …, -1.6808e-03,
   -9.7308e-04, -8.3828e-04],
  [-1.1122e-03, -1.3573e-03, -3.3026e-04,  …, -3.5775e-03,
   -3.4374e-03, -2.8404e-03],
  [ 1.0825e-04, -4.6579e-04, -6.5309e-04,  …, -3.3756e-03,
   -3.8776e-03, -3.9767e-03]]],


…,


[[[ 4.7837e-04,  1.4615e-03,  2.1655e-03,  …,  4.6694e-03,
    5.4724e-03,  4.1887e-03],
  [ 4.7072e-04, -6.8468e-04, -9.0317e-04,  …,  1.6946e-03,
    2.8004e-03,  2.9049e-03],
  [-1.7878e-03, -1.7783e-03, -2.2019e-03,  …,  2.5375e-04,
    6.5940e-04,  1.7380e-03],
  …,
  [-9.4276e-04, -5.4474e-04, -1.0068e-03,  …, -7.8403e-04,
   -1.2735e-03, -1.4813e-03],
  [ 7.0821e-04,  5.8439e-04,  1.1780e-04,  …,  1.6170e-03,
    8.7575e-04, -3.0514e-04],
  [ 2.1802e-03,  2.5714e-03,  2.4628e-03,  …,  9.4091e-04,
    1.1106e-03,  1.4326e-03]],

 [[-1.2457e-03, -6.7865e-04, -1.4310e-05,  …,  3.1465e-03,
    3.7346e-03,  2.7692e-03],
  [-9.5305e-04, -2.2229e-03, -2.5523e-03,  …,  7.2032e-04,
    1.9517e-03,  2.4015e-03],
  [-3.0225e-03, -2.8837e-03, -3.2224e-03,  …, -2.3011e-04,
    2.6387e-04,  1.7625e-03],
  …,
  [-8.1902e-04, -3.6833e-04, -1.0632e-03,  …, -1.5532e-03,
   -1.9701e-03, -1.5653e-03],
  [ 6.7254e-04,  6.7780e-04, -2.1316e-05,  …,  7.4769e-04,
    4.6967e-04,  2.0583e-04],
```

```
  [ 1.7351e-03,  2.2258e-03,  2.0535e-03,  …,  3.9448e-05,
    7.0552e-04,  1.7910e-03]],

 [[-4.5968e-03, -3.9219e-03, -2.8440e-03,  …, -3.5577e-04,
   -1.1507e-04, -1.2474e-03],
  [-4.3795e-03, -5.4576e-03, -5.4672e-03,  …, -2.6427e-03,
   -1.8553e-03, -1.6562e-03],
  [-6.2548e-03, -5.8972e-03, -6.4037e-03,  …, -3.5493e-03,
   -3.0895e-03, -1.8703e-03],
  …,
  [-5.1496e-03, -4.2164e-03, -4.1850e-03,  …, -4.6948e-03,
   -4.6940e-03, -3.8372e-03],
  [-3.7482e-03, -3.2880e-03, -3.4192e-03,  …, -2.5094e-03,
   -2.4491e-03, -2.3221e-03],
  [-3.4621e-03, -2.5560e-03, -2.4284e-03,  …, -3.3501e-03,
   -2.6140e-03, -1.2857e-03]]],


[[[-2.7132e-02, -2.9027e-02, -2.9317e-02,  …, -2.5457e-02,
   -2.3944e-02, -2.3896e-02],
  [-3.2897e-02, -3.4056e-02, -3.2927e-02,  …, -2.6193e-02,
   -2.5314e-02, -2.5440e-02],
  [-3.5314e-02, -3.6346e-02, -3.5156e-02,  …, -2.8325e-02,
   -2.7507e-02, -2.7549e-02],
  …,
  [-2.3860e-02, -2.5084e-02, -2.4939e-02,  …, -2.6184e-02,
   -2.7569e-02, -2.6302e-02],
  [-2.1940e-02, -2.1860e-02, -1.9181e-02,  …, -1.9174e-02,
   -2.1648e-02, -2.3125e-02],
  [-1.8767e-02, -1.7081e-02, -1.3803e-02,  …, -1.3509e-02,
   -1.5813e-02, -1.8994e-02]],

 [[-2.7048e-02, -2.9530e-02, -2.9485e-02,  …, -2.4591e-02,
   -2.3686e-02, -2.1974e-02],
  [-3.1836e-02, -3.2990e-02, -3.1829e-02,  …, -2.4984e-02,
   -2.4512e-02, -2.3273e-02],
  [-3.3749e-02, -3.4724e-02, -3.3327e-02,  …, -2.6334e-02,
   -2.6075e-02, -2.5286e-02],
  …,
  [-2.2126e-02, -2.3908e-02, -2.4614e-02,  …, -2.6761e-02,
   -2.7561e-02, -2.5765e-02],
  [-2.0460e-02, -2.0659e-02, -1.8690e-02,  …, -2.1009e-02,
   -2.2854e-02, -2.3210e-02],
  [-1.6354e-02, -1.6098e-02, -1.3496e-02,  …, -1.5635e-02,
   -1.7821e-02, -1.9547e-02]],

 [[-3.2408e-02, -3.3445e-02, -3.3007e-02,  …, -2.9015e-02,
   -2.8478e-02, -2.7217e-02],
```

```
  [-3.5695e-02, -3.5944e-02, -3.5055e-02,  …, -2.9221e-02,
   -2.8742e-02, -2.7825e-02],
  [-3.6720e-02, -3.6971e-02, -3.6390e-02,  …, -2.9982e-02,
   -2.9677e-02, -2.9305e-02],
  …,
  [-2.7578e-02, -2.8377e-02, -2.8230e-02,  …, -2.9183e-02,
   -2.9936e-02, -2.8470e-02],
  [-2.6102e-02, -2.5784e-02, -2.3552e-02,  …, -2.4711e-02,
   -2.6685e-02, -2.7198e-02],
  [-2.3181e-02, -2.2125e-02, -1.9367e-02,  …, -2.0744e-02,
   -2.3076e-02, -2.5094e-02]]],


[[[ 5.1390e-03,  3.8933e-03,  3.5646e-03,  …,  1.0583e-02,
    1.0822e-02,  9.1667e-03],
  [ 3.8397e-03,  3.2694e-03,  5.4251e-03,  …,  1.1113e-02,
    1.3003e-02,  1.1063e-02],
  [ 5.7886e-03,  8.6669e-03,  1.1575e-02,  …,  9.7668e-03,
    1.1378e-02,  1.0339e-02],
  …,
  [ 2.1328e-03,  2.0173e-03,  5.9720e-05,  …,  1.0404e-03,
    8.2747e-04, -2.9873e-03],
  [ 9.8151e-04,  1.3591e-04,  1.1537e-03,  …,  1.9527e-03,
    1.5131e-03, -1.6895e-03],
  [ 2.6678e-03,  2.8754e-03,  2.7769e-03,  …,  8.1430e-04,
    1.5535e-03, -6.3936e-04]],

 [[ 1.1191e-02,  9.7837e-03,  9.6491e-03,  …,  1.8014e-02,
    1.8729e-02,  1.8202e-02],
  [ 1.0961e-02,  1.0046e-02,  1.2443e-02,  …,  1.8028e-02,
    1.9837e-02,  1.9179e-02],
  [ 1.4504e-02,  1.6547e-02,  1.9200e-02,  …,  1.6135e-02,
    1.6346e-02,  1.6318e-02],
  …,
  [ 1.4843e-02,  1.4330e-02,  1.3056e-02,  …,  1.2613e-02,
    1.1916e-02,  1.0514e-02],
  [ 1.4226e-02,  1.3526e-02,  1.5141e-02,  …,  1.5081e-02,
    1.4201e-02,  1.3357e-02],
  [ 1.6051e-02,  1.6180e-02,  1.6845e-02,  …,  1.4666e-02,
    1.5106e-02,  1.5134e-02]],

 [[ 8.0953e-03,  7.3331e-03,  8.6289e-03,  …,  1.8757e-02,
    1.8104e-02,  1.7962e-02],
  [ 8.1057e-03,  8.5628e-03,  1.2142e-02,  …,  1.8455e-02,
    1.9760e-02,  1.9321e-02],
  [ 1.1161e-02,  1.4693e-02,  1.8647e-02,  …,  1.5740e-02,
    1.6459e-02,  1.6966e-02],
  …,
```

```
          [ 1.2981e-02,  1.3414e-02,  1.3117e-02,  …,  1.4225e-02,
            1.3707e-02,  1.3412e-02],
          [ 1.3978e-02,  1.3861e-02,  1.6164e-02,  …,  1.6500e-02,
            1.5326e-02,  1.5472e-02],
          [ 1.5958e-02,  1.7349e-02,  1.8869e-02,  …,  1.5860e-02,
            1.6462e-02,  1.7309e-02]]])
tensor([ 4.6566e-10,  2.0340e-06, -2.4913e-08, -1.0245e-08,  4.1211e-08,
        -3.3993e-08, -3.7253e-08, -3.0734e-08, -2.7940e-09,  3.7835e-09,
         9.6043e-09, -1.2806e-08,  3.2596e-09,  3.7020e-08, -6.9849e-10,
        -5.4715e-09, -8.6729e-09,  1.3970e-07,  4.6566e-10,  1.6531e-08,
         3.4925e-09, -2.0489e-08,  4.5635e-08,  1.3271e-08, -8.9640e-09,
         3.1316e-08,  2.2119e-09,  1.8626e-08, -6.2864e-09,  8.2888e-08,
        -1.0012e-08, -3.0268e-08, -1.8626e-09, -2.6776e-09, -4.3830e-08,
         3.0268e-09,  1.9791e-09,  1.3970e-09,  1.5832e-08,  2.9831e-09,
         2.2352e-08, -1.2791e-08, -2.6776e-09,  6.6357e-09, -4.2841e-08,
         1.7695e-08, -5.2387e-09,  4.5635e-08,  7.0781e-08,  1.0798e-08,
        -3.3318e-09,  1.6764e-08, -1.8626e-09,  1.6950e-07, -1.4072e-08,
         4.5984e-09,  4.6566e-08,  5.5879e-09,  1.4552e-09, -4.4238e-08,
        -4.8354e-06, -7.7998e-09,  1.3737e-08,  1.3388e-08,  1.1816e-08,
        -9.0804e-09, -1.0477e-08, -2.5332e-07, -3.0268e-09,  1.3970e-09,
        -4.0163e-09, -8.1491e-09,  6.9849e-09, -6.5193e-09,  2.9802e-08,
        -1.8859e-08,  4.9331e-09, -9.7789e-09,  5.0291e-08, -8.8476e-09,
         2.6613e-07, -4.1677e-08, -1.2107e-08, -3.2596e-09,  2.1607e-07,
         4.5402e-09, -2.2352e-08, -4.1910e-09,  2.4447e-09,  1.6112e-07,
         1.6298e-08,  6.0536e-09,  4.1444e-08, -1.0012e-08, -1.7090e-07,
         3.6089e-08])
tensor([-1.2954e-02, -1.9991e-03,  8.7189e-03, -3.7730e-03,  5.2190e-02,
        -3.3662e-02,  1.4285e-02, -2.6534e-02,  1.2008e-02, -3.1084e-02,
        -2.6752e-02,  7.1120e-04, -2.6955e-02,  2.2224e-02,  1.8218e-02,
        -7.7045e-03, -4.2334e-02, -4.8553e-02, -3.1639e-02,  1.5513e-02,
         3.0338e-03, -7.4059e-03,  9.5939e-02, -3.4214e-03, -3.1466e-02,
         5.8463e-03, -8.9862e-03,  8.6061e-03, -3.8643e-02, -2.0825e-02,
        -1.5204e-03, -2.4782e-03,  1.2423e-02, -2.3520e-03,  3.7465e-03,
         1.1968e-03,  6.4777e-03,  2.8043e-02,  1.5619e-02,  3.4389e-02,
         2.1142e-02,  1.4742e-02,  1.2531e-02, -2.8178e-03,  2.8486e-03,
        -6.1569e-03,  9.6602e-03,  2.0452e-02,  2.4486e-02,  2.0061e-03,
        -1.1220e-02,  8.3285e-03,  1.3621e-02,  7.6865e-03,  1.9529e-02,
         2.0503e-02, -2.3912e-02, -5.0950e-03,  8.6563e-04,  1.4697e-02,
        -1.5911e-06, -1.7783e-03, -4.6853e-02, -6.0819e-03, -1.5178e-02,
         1.2421e-02,  2.1414e-02,  1.1677e-01,  8.8527e-04,  4.8837e-04,
        -8.0565e-03,  6.6074e-03,  1.9651e-02,  3.3208e-02,  4.8224e-02,
        -3.4758e-02, -6.0898e-03,  2.3827e-02, -1.1493e-02,  1.2313e-02,
        -1.1030e-01,  2.0068e-02,  5.3751e-04,  1.8164e-02, -9.7230e-02,
         1.6794e-02,  2.7048e-02, -2.0081e-02, -5.0262e-03,  3.0534e-02,
         2.7395e-02, -2.0194e-02, -8.0356e-02,  2.0890e-02,  8.9560e-03,
         2.4229e-02])
tensor([-3.3142e-03, -3.3683e-02,  3.2929e-03,  6.1669e-04,  2.1411e-03,
        -5.3690e-03,  2.1521e-02, -9.9940e-03,  1.5884e-02, -1.5830e-02,
```

```
        -9.7523e-03,  1.3584e-03, -1.4303e-02,  2.6571e-02,  1.1419e-02,
        -7.0748e-03, -1.9752e-02, -1.5584e-03, -2.4926e-02,  9.5567e-03,
         2.6686e-03, -5.6399e-03, -9.0506e-03,  5.3140e-03, -1.7756e-02,
         2.6063e-03, -6.6703e-03,  1.2056e-02, -2.0718e-02,  4.3583e-03,
        -4.3480e-03, -2.2520e-03,  1.2380e-02, -5.5095e-04, -1.1128e-04,
         2.0577e-03,  6.4533e-03,  9.9893e-03,  2.5282e-02,  1.3255e-02,
         1.1780e-02,  9.6246e-03,  3.5360e-03, -4.0473e-03,  2.6493e-03,
        -1.2591e-02,  6.2756e-03,  2.3288e-02,  4.1791e-02, -5.4351e-03,
        -1.2165e-02,  1.8194e-02,  1.4116e-02,  5.0285e-02,  2.0483e-02,
         7.6968e-03,  6.5963e-03, -4.8578e-03,  2.4639e-04,  1.9404e-02,
         1.0428e-02, -5.3223e-06, -1.3534e-02, -7.1856e-03, -1.2733e-02,
         1.2302e-02,  1.2644e-02, -2.8298e-02, -6.3735e-05,  6.0339e-03,
        -3.4916e-03,  1.4119e-03,  3.8981e-02,  1.4763e-02, -3.3933e-03,
        -2.2563e-02, -3.2026e-03,  3.0831e-02, -9.6059e-03,  8.1839e-03,
        -1.3409e-02,  7.7790e-02, -9.5528e-04,  1.3250e-02, -3.6329e-02,
         1.1102e-02,  1.8281e-02, -5.2142e-03, -1.3725e-03, -7.2787e-04,
         2.0492e-02,  2.2185e-02,  9.4602e-03,  1.7602e-02,  4.2766e-02,
         5.4256e-03])
tensor([[[[ 1.1458e-04,  4.3659e-04,  8.6179e-04,  4.2869e-04,  4.8510e-04],
          [-1.2666e-03, -7.1257e-04,  3.3586e-04,  1.2986e-03,  5.4141e-04],
          [-9.9675e-04, -8.1050e-04, -5.2512e-04, -2.5561e-04, -3.3086e-04],
          [-7.2403e-04, -5.9153e-04, -4.6877e-04, -1.0630e-03, -1.5582e-03],
          [ 1.0007e-03,  1.3076e-03,  8.2667e-04, -2.0461e-04, -1.1304e-03]],

         [[ 8.1959e-04,  2.8010e-03,  1.8368e-03,  5.9312e-03,  8.0957e-04],
          [ 4.8585e-04,  4.8004e-03,  2.2244e-03,  5.1133e-03,  1.2836e-03],
          [ 4.2241e-04,  2.4739e-03,  1.0692e-03, -9.8010e-04, -7.8354e-04],
          [ 1.5803e-03,  1.1087e-03,  8.1663e-04, -2.6143e-04, -1.2368e-03],
          [-9.5988e-04, -2.0983e-05,  4.8972e-04,  1.0972e-03, -1.0271e-03]],

         [[ 1.1338e-04,  3.3524e-04,  8.2204e-04,  9.7787e-04,  1.0064e-03],
          [ 3.1184e-04,  6.2762e-04,  8.2190e-04,  5.0996e-04,  1.9421e-04],
          [ 2.8990e-04,  5.8108e-04,  5.9460e-04,  2.2545e-04, -1.0585e-04],
          [ 1.2541e-04,  4.8940e-04,  3.6650e-04,  3.4178e-04,  2.4518e-04],
          [ 1.9178e-05,  2.1940e-04,  4.9589e-04,  7.9553e-04,  7.8259e-04]],

         …,

         [[-1.5101e-03, -2.9500e-04,  5.1774e-04,  1.5744e-03,  1.1307e-03],
          [-6.7099e-04,  2.1201e-04,  7.4533e-04,  9.0412e-04,  9.6661e-04],
          [-7.6466e-04,  2.8425e-05,  5.4224e-04,  6.4543e-04,  9.4008e-04],
          [-1.3837e-03, -1.0134e-03,  2.8619e-04,  9.9138e-04,  2.2086e-03],
          [-1.4433e-03, -8.4293e-04,  4.6630e-04,  1.5257e-03,  2.5827e-03]],

         [[-8.7353e-04, -5.5760e-04,  9.0211e-05,  4.3051e-04,  3.0118e-04],
          [-7.9607e-04, -4.3195e-04,  2.8070e-04,  3.2745e-04,  5.7199e-04],
          [-1.1606e-03, -1.0148e-03,  2.3945e-05,  2.2079e-04,  5.0823e-04],
          [-2.0766e-03, -2.1775e-03, -6.7529e-04,  2.2661e-06,  6.5362e-04],
```

```
  [-1.6023e-03, -1.3434e-03,  3.0159e-05,  3.6470e-04,  8.8583e-04]],

 [[-3.2582e-04,  1.3180e-03,  1.1562e-03,  1.4451e-03,  7.6149e-04],
  [ 6.6226e-05,  1.5655e-03,  1.4250e-03,  7.8963e-04, -2.1274e-04],
  [-8.1684e-04,  4.0384e-04,  2.8275e-04,  2.9949e-04, -2.4208e-04],
  [-1.5673e-03, -1.6726e-03, -1.4775e-03, -4.7047e-04,  5.7728e-06],
  [-1.8176e-03, -1.6371e-03, -8.9808e-04, -6.0568e-04,  6.0749e-04]]],


[[[-6.9524e-04, -6.2567e-04, -9.6263e-04, -1.5020e-03, -1.6620e-03],
  [-1.3124e-03, -1.2834e-03, -1.0961e-03, -1.8024e-03, -1.9394e-03],
  [-9.5492e-04, -1.5449e-03, -1.5556e-03, -1.5771e-03, -1.1981e-03],
  [ 6.1156e-04, -4.0775e-04, -4.9426e-04, -5.7760e-04, -8.7288e-04],
  [ 1.3397e-03,  4.6190e-04, -3.1037e-04, -7.7364e-04, -5.1584e-04]],

 [[ 2.3781e-03,  3.9216e-04,  1.0193e-03,  1.8951e-04,  4.0542e-04],
  [ 3.1316e-03,  4.9566e-04,  6.6557e-04,  2.6185e-04,  4.2882e-04],
  [ 4.2083e-03,  1.6135e-05,  5.3608e-04, -4.0714e-04, -5.6506e-05],
  [ 4.8212e-03,  5.8960e-04,  1.0664e-03, -3.7695e-04, -4.3181e-04],
  [ 4.4640e-03,  1.6344e-03,  9.3349e-04, -7.0057e-04,  6.6290e-05]],

 [[ 1.1310e-03,  1.3552e-03,  1.5721e-03,  1.8487e-03,  1.6969e-03],
  [ 7.8908e-04,  1.1815e-03,  1.5592e-03,  1.7537e-03,  1.7059e-03],
  [ 5.7841e-04,  9.6929e-04,  1.1060e-03,  1.3082e-03,  1.3987e-03],
  [ 4.0872e-04,  8.0735e-04,  8.1236e-04,  9.6885e-04,  1.0980e-03],
  [ 3.5237e-04,  4.4511e-04,  4.0640e-04,  5.0873e-04,  4.8568e-04]],

 …,

 [[ 8.2705e-04,  1.4059e-03,  1.8261e-03,  1.9516e-03,  2.4226e-03],
  [ 7.1900e-04,  8.1795e-04,  1.0442e-03,  1.1641e-03,  1.2368e-03],
  [ 7.3474e-05,  4.7645e-04,  3.8079e-04,  3.4451e-04,  5.8341e-04],
  [-5.7480e-04, -3.5902e-05,  2.1289e-05, -1.1222e-04,  3.3286e-05],
  [-9.2224e-04, -8.9017e-04, -8.8479e-04, -1.1163e-03, -9.5823e-04]],

 [[-6.8796e-04, -1.0070e-03, -8.5554e-04, -6.9357e-04, -3.3939e-04],
  [-6.5973e-04, -1.0232e-03, -1.3801e-03, -1.6492e-03, -1.5795e-03],
  [-1.8438e-03, -1.9860e-03, -2.0799e-03, -2.3949e-03, -2.3307e-03],
  [-1.6392e-03, -1.7890e-03, -2.1554e-03, -2.3786e-03, -2.1943e-03],
  [-1.5156e-03, -1.3881e-03, -1.4987e-03, -1.8802e-03, -1.9911e-03]],

 [[-4.3939e-06,  2.2752e-04,  7.2158e-04,  6.6414e-04,  9.3836e-04],
  [ 3.7441e-04,  3.1595e-04,  4.0234e-04,  4.3115e-04,  6.5434e-04],
  [-3.8678e-04, -1.4936e-04, -3.3545e-04, -2.9873e-04, -3.1775e-04],
  [-5.2061e-04, -3.3241e-04, -1.0262e-04, -6.4819e-05, -2.1482e-04],
  [-3.0533e-04, -7.4475e-04, -8.6073e-04, -1.0221e-03, -9.3033e-04]]],
```

```
[[[ 1.2834e-03,  1.6291e-03,  1.5602e-03,  1.4771e-03,  2.1681e-03],
  [ 1.1951e-03,  1.6189e-03,  1.9006e-03,  2.1032e-03,  2.3196e-03],
  [ 9.4895e-04,  8.7530e-04,  6.0301e-04,  9.2369e-04,  1.5050e-03],
  [ 5.1247e-04,  5.0793e-04,  8.2843e-04,  7.8553e-04,  7.5139e-04],
  [ 5.9715e-04,  1.5580e-03,  1.7366e-03,  1.8282e-03,  1.6518e-03]],

 [[ 2.0932e-03,  2.4209e-03,  4.3724e-05,  7.3024e-04, -4.5819e-04],
  [ 1.7223e-03,  2.2968e-03, -6.8440e-04,  5.5539e-04, -6.3460e-04],
  [-4.7914e-04,  9.2154e-04,  3.3366e-04, -6.7383e-04, -5.3692e-04],
  [-8.6819e-04,  8.2115e-04,  2.4407e-04,  1.0876e-04,  9.3934e-04],
  [-5.3030e-04,  4.4199e-04,  2.5296e-04,  9.4098e-04,  1.4000e-03]],

 [[-6.4411e-04, -3.9638e-04, -5.1472e-04, -6.5219e-04, -5.5183e-04],
  [-6.0224e-04, -1.2533e-04, -2.5088e-04, -3.7705e-04, -4.6313e-04],
  [-7.4056e-04, -5.1168e-04, -5.2021e-04, -4.6094e-04, -5.7471e-04],
  [-4.4766e-04, -2.6779e-04, -2.9105e-04, -4.0883e-04, -4.6661e-04],
  [-3.8620e-04, -2.5073e-04, -1.8996e-04, -3.9790e-04, -4.1481e-04]],

 …,

 [[ 3.1759e-04,  5.8839e-04, -1.5068e-04, -4.1681e-04, -9.3495e-04],
  [ 3.9159e-04,  5.1954e-04, -3.0020e-04, -3.2291e-04, -1.2754e-03],
  [-3.1932e-04,  1.9276e-04, -1.0256e-05, -2.1429e-04, -7.1304e-04],
  [ 2.9407e-04,  7.5045e-04,  5.2492e-04,  1.5213e-04, -2.0347e-04],
  [ 3.9422e-04,  5.2227e-04,  6.1500e-04,  5.5764e-04,  3.5542e-04]],

 [[ 3.5657e-03,  3.7650e-03,  3.1291e-03,  2.3769e-03,  2.1134e-03],
  [ 3.4356e-03,  3.4859e-03,  2.8845e-03,  2.2384e-03,  1.9231e-03],
  [ 2.9637e-03,  3.1161e-03,  2.8639e-03,  1.8060e-03,  1.7805e-03],
  [ 2.5995e-03,  3.0414e-03,  2.7011e-03,  2.1017e-03,  2.1010e-03],
  [ 2.4040e-03,  2.9157e-03,  2.7565e-03,  2.6173e-03,  2.5910e-03]],

 [[-1.3890e-03, -8.3001e-04, -9.6205e-04, -1.3094e-03, -1.3255e-03],
  [-1.1396e-03, -2.0879e-03, -2.1336e-03, -1.7936e-03, -1.7311e-03],
  [-2.3781e-03, -1.8598e-03, -1.5673e-03, -1.2646e-03, -1.8311e-03],
  [-1.1058e-03, -3.7292e-04, -5.3371e-04, -6.3967e-04, -9.6017e-04],
  [-6.3796e-04, -4.4268e-04, -3.9946e-04, -7.1648e-04, -7.4098e-04]]],


…,


[[[-1.2721e-03, -7.6641e-04, -1.0598e-03, -5.3514e-04,  3.5600e-05],
  [-1.3880e-03, -8.8949e-04, -2.6264e-04,  1.4185e-04, -2.7052e-04],
  [-8.2342e-04, -8.4274e-04, -5.0046e-04, -4.6089e-04, -6.9368e-04],
  [-5.0707e-04,  9.4778e-05,  1.2940e-04,  1.3324e-04, -5.4354e-04],
  [-5.2035e-04, -4.1714e-05, -1.9643e-05,  1.5406e-04,  2.5884e-04]],
```

```
 [[-5.4899e-03, -6.9913e-03, -6.1093e-03, -3.0395e-03, -3.9294e-03],
  [-5.5857e-03, -4.9522e-03, -4.1608e-03, -2.9109e-03, -1.4624e-03],
  [-3.6969e-03, -3.4970e-03, -2.1532e-03, -1.5627e-03, -1.6954e-04],
  [-2.4906e-03, -1.0763e-03, -8.5195e-04, -1.5395e-03, -2.6974e-04],
  [-1.7318e-03, -1.8520e-03, -1.8618e-03, -9.4734e-04,  1.3876e-04]],

 [[-7.2803e-05, -3.6595e-04, -8.3239e-04, -8.9759e-04, -7.6190e-04],
  [-3.5368e-04, -5.7601e-04, -9.4068e-04, -1.1709e-03, -1.1530e-03],
  [-6.1142e-04, -7.5393e-04, -8.6547e-04, -1.0056e-03, -9.7659e-04],
  [-1.0243e-03, -7.7491e-04, -9.4701e-04, -9.4890e-04, -7.4892e-04],
  [-8.2765e-04, -8.1640e-04, -1.0805e-03, -1.2199e-03, -9.5136e-04]],

 …,

 [[-1.2168e-04, -9.5136e-04, -1.6054e-03, -1.7570e-03, -1.5845e-03],
  [-1.7763e-03, -1.9430e-03, -2.1617e-03, -2.1810e-03, -1.5190e-03],
  [-1.3705e-03, -1.6924e-03, -1.4057e-03, -1.3218e-03, -1.0161e-03],
  [-1.2505e-03, -1.1562e-03, -1.4502e-03, -1.1609e-03, -1.1814e-03],
  [-1.1428e-03, -1.0018e-03, -1.2276e-03, -1.1225e-03, -8.2049e-04]],

 [[ 2.9569e-04,  3.7705e-04,  2.6839e-04,  1.5517e-04,  2.2320e-04],
  [-6.5953e-04, -4.3968e-04,  8.4747e-05,  1.1266e-04,  2.9654e-04],
  [-5.0781e-04, -5.2096e-04, -2.1991e-04, -7.3108e-05,  2.0689e-04],
  [-1.1457e-04, -2.9457e-04, -5.4629e-05,  1.0709e-04,  2.7474e-05],
  [-1.4572e-04, -2.4140e-04, -1.0157e-04,  1.1470e-04,  9.4341e-05]],

 [[-1.2780e-03, -2.1775e-03, -3.9295e-03, -3.8505e-03, -3.2257e-03],
  [-3.1152e-03, -3.7183e-03, -3.7442e-03, -4.0432e-03, -2.6156e-03],
  [-3.4323e-03, -3.6009e-03, -3.3417e-03, -2.2497e-03, -2.4023e-03],
  [-3.8945e-03, -3.4478e-03, -2.9639e-03, -2.9167e-03, -2.2930e-03],
  [-2.6209e-03, -2.4036e-03, -2.9030e-03, -2.4412e-03, -2.3998e-03]]],


[[[ 2.7650e-03,  3.3505e-03,  2.1386e-03, -1.0937e-04, -2.0453e-04],
  [ 2.0714e-03,  2.3562e-03,  2.4777e-03,  1.4718e-03,  4.8254e-04],
  [ 5.7865e-04,  8.8288e-04,  1.5906e-03,  5.5317e-04, -2.9568e-04],
  [-1.6996e-04,  2.4543e-04,  4.3613e-04,  3.0946e-04, -3.3226e-04],
  [-8.2211e-04, -9.0373e-04, -2.2170e-03, -2.2498e-03, -1.4230e-03]],

 [[ 7.5973e-04,  2.1343e-03,  5.0107e-03,  1.4806e-03,  7.9461e-04],
  [-8.3408e-04,  2.7642e-03,  3.9271e-03,  6.2810e-06, -1.1487e-03],
  [-1.4244e-03,  1.6730e-03,  2.2813e-03,  3.3309e-04, -4.0651e-04],
  [-1.1010e-03,  3.8059e-04,  2.0844e-03,  2.0814e-03,  9.6684e-05],
  [-8.6415e-04,  6.3677e-04,  2.8199e-03,  1.8448e-03,  1.5275e-03]],

 [[-1.3937e-04, -5.1896e-04, -4.6822e-04, -4.2614e-04, -4.5535e-04],
  [-2.3066e-04, -8.8473e-04, -6.5806e-04, -4.4793e-04, -5.9371e-04],
  [ 1.5736e-04, -4.9719e-04, -7.0860e-04, -7.3052e-04, -9.3830e-04],
```

```
  [ 7.0489e-04,  2.6400e-04, -2.3348e-04, -5.0805e-04, -5.9827e-04],
  [ 9.2159e-04,  2.8519e-04, -3.5094e-05, -4.0295e-04, -2.1851e-04]],

 …,

 [[-1.2894e-04,  4.5102e-07,  8.5436e-04,  6.6653e-04, -1.8284e-04],
  [ 3.0564e-04, -3.7437e-04,  8.9325e-05,  2.8204e-05, -4.5719e-04],
  [ 7.5453e-04,  7.6307e-04,  3.4298e-04, -2.5979e-04, -3.5177e-04],
  [ 1.9709e-03,  1.7599e-03,  9.8303e-04,  4.7451e-04, -4.5727e-04],
  [ 1.9322e-03,  1.6582e-03,  1.3865e-03,  1.1804e-03,  5.1227e-04]],

 [[ 9.0868e-04,  1.1323e-03,  9.4645e-04, -2.1273e-04, -6.8686e-04],
  [ 1.5585e-03,  1.4764e-03,  9.2857e-04,  2.6838e-04,  3.4658e-05],
  [ 1.7876e-03,  2.1033e-03,  1.5015e-03,  6.2461e-04,  4.4319e-04],
  [ 1.6667e-03,  1.8810e-03,  1.6337e-03,  4.3169e-04, -3.4453e-04],
  [ 1.5251e-03,  1.0969e-03,  7.6868e-04,  5.4406e-06, -4.5486e-04]],

 [[ 3.7187e-04,  1.1509e-04,  1.5683e-04, -2.3102e-04, -1.1793e-03],
  [-3.8139e-04, -4.5460e-04, -5.2748e-04, -1.4463e-03, -1.5295e-03],
  [ 8.1184e-05,  4.5438e-04,  3.4879e-04, -1.4130e-03, -1.6315e-03],
  [ 1.0404e-03,  1.1706e-03, -1.0757e-04, -9.7577e-04, -1.3836e-03],
  [ 3.5142e-04,  9.1902e-04, -1.8757e-05, -5.7010e-04, -6.6123e-04]]],


[[[ 4.6596e-04, -4.7479e-04, -1.8292e-03, -2.0496e-03, -1.0402e-03],
  [ 9.9867e-04,  5.3722e-04,  4.4921e-04,  4.7089e-04, -4.7219e-04],
  [-1.6923e-03, -1.7057e-03, -1.3017e-03, -1.2331e-03, -1.0352e-03],
  [-2.4456e-03, -1.8540e-03, -1.0490e-03, -1.1411e-03, -1.1466e-03],
  [-1.1738e-03, -9.1444e-04, -4.9852e-04, -4.0719e-04, -3.7124e-04]],

 [[-1.5247e-03, -2.7624e-03, -2.6776e-03, -6.2764e-04, -1.6723e-03],
  [-5.3130e-04, -1.8849e-03, -3.1887e-03, -2.2693e-03, -8.0244e-04],
  [-3.3581e-04, -1.5371e-03, -1.5933e-03, -2.0482e-03, -1.1067e-04],
  [-1.4147e-03, -2.5312e-03,  1.3094e-05, -1.0280e-03, -6.3679e-04],
  [-3.3141e-03, -2.2181e-03,  6.3580e-04,  5.4248e-05,  9.8928e-06]],

 [[-5.9194e-04, -3.5774e-04, -5.4309e-04, -4.9541e-04, -1.6313e-04],
  [-2.9421e-04, -3.2886e-04, -2.1185e-04, -2.2628e-04, -1.3848e-04],
  [-1.5892e-05, -5.5467e-05, -1.7375e-04,  5.2419e-05, -3.2878e-04],
  [ 3.8914e-04,  3.4743e-04,  3.3233e-04,  9.4257e-05, -5.5295e-05],
  [ 3.9712e-04,  4.4109e-04,  3.7240e-04,  8.9276e-05, -1.6969e-04]],

 …,

 [[-1.1938e-03, -1.0888e-03, -1.5720e-03, -2.4676e-03, -2.3072e-03],
  [-9.5324e-04, -1.1353e-03, -1.6171e-03, -2.2061e-03, -2.1151e-03],
  [-1.4178e-03, -1.7721e-03, -2.0388e-03, -2.2559e-03, -2.2203e-03],
  [-1.3322e-03, -1.6113e-03, -2.0180e-03, -2.3794e-03, -2.5741e-03],
```

```
          [-1.5203e-03, -2.0147e-03, -2.3043e-03, -2.3231e-03, -2.5224e-03]],

         [[-1.3982e-03, -1.2637e-03, -1.3645e-03, -1.6792e-03, -1.9576e-03],
          [-1.2016e-03, -1.4568e-03, -1.5203e-03, -1.8110e-03, -1.8186e-03],
          [-2.0967e-03, -2.2137e-03, -2.2859e-03, -2.3617e-03, -2.6584e-03],
          [-2.5309e-03, -2.5999e-03, -2.8832e-03, -2.9371e-03, -2.7657e-03],
          [-2.6184e-03, -2.8467e-03, -2.9654e-03, -2.8899e-03, -3.0187e-03]],

         [[ 7.7045e-04,  6.5253e-04,  4.0847e-04, -8.0006e-04, -8.8385e-04],
          [ 1.0100e-03,  1.0608e-04, -6.0827e-04, -8.9068e-04, -7.0916e-04],
          [ 9.4735e-04,  3.4553e-05, -2.1121e-04, -1.3713e-04, -4.6691e-04],
          [ 1.4516e-04, -6.5806e-04, -4.8005e-04, -6.4464e-04, -3.0876e-04],
          [-1.8884e-04, -5.6998e-04, -1.0175e-03, -7.6874e-04, -5.4554e-04]]]])
tensor([ 2.0397e-09,  3.2524e-09, -1.0547e-08,  7.5715e-10,  1.8117e-09,
        -4.3178e-09,  9.3377e-09, -9.8186e-09, -1.8936e-09,  8.2024e-09,
        -9.7584e-09, -6.8701e-09, -1.0462e-09,  2.5169e-09, -3.5934e-09,
         1.7740e-08, -9.5497e-12,  4.2149e-09,  6.8194e-09,  6.9871e-09,
        -2.4239e-09, -8.0416e-09, -1.2201e-09,  3.1491e-11,  2.1782e-09,
        -2.3028e-09,  6.9772e-10,  2.9493e-09,  6.8030e-10, -6.8675e-10,
        -4.1055e-09, -6.2130e-09,  1.5692e-08,  3.4490e-08,  1.0952e-08,
         2.2291e-09,  8.7508e-10, -4.4558e-09, -1.0870e-09,  4.9616e-10,
        -2.6193e-09, -1.5202e-09,  1.8388e-09, -4.2986e-10,  1.3069e-09,
         2.3883e-09,  1.2159e-09, -3.0770e-09, -1.4904e-08,  2.8992e-09,
        -2.5934e-09, -5.9762e-09, -2.8176e-09,  2.3996e-09,  1.4504e-09,
         2.5300e-09,  4.8178e-09,  3.7425e-09,  2.4106e-09,  7.2762e-10,
         3.5170e-09,  4.4290e-09,  4.2853e-09,  3.3343e-09, -9.4087e-10,
        -6.1607e-10,  5.4059e-09, -4.9087e-09,  1.0454e-08,  3.7000e-09,
         2.8194e-10, -1.4751e-08, -1.1297e-08,  9.8680e-09,  1.6021e-09,
         6.4058e-09, -4.2246e-10, -4.8169e-10,  3.5033e-10,  6.3955e-10,
        -5.0309e-09, -1.5561e-08, -1.7670e-08,  2.1414e-09, -3.8636e-10,
        -5.6621e-09,  2.2349e-09, -2.3842e-09,  4.5618e-09,  1.9330e-10,
        -1.7946e-09,  2.1553e-10, -3.8372e-09, -4.0315e-09, -2.0800e-08,
        -8.5674e-10,  1.6780e-09,  2.4420e-09,  3.8817e-09, -2.7858e-09,
         1.0364e-08, -1.7503e-09,  3.2638e-09, -1.9158e-09,  9.7072e-10,
        -1.7822e-09,  3.2014e-10,  2.5721e-09,  9.4508e-10, -2.7794e-09,
         5.1873e-09, -2.4088e-09,  3.1482e-09,  4.5385e-09, -1.0814e-09,
         3.1593e-09,  2.0682e-09, -4.0291e-09, -2.1025e-09, -4.3490e-09,
        -3.3151e-09,  4.8572e-09,  8.9813e-10,  1.1831e-09,  9.1234e-12,
        -3.2742e-10,  5.4171e-09, -2.2237e-10, -5.4249e-10,  1.8943e-08,
         7.9465e-09, -9.4097e-09, -5.4573e-10, -3.0668e-09,  2.4434e-09,
         3.9381e-09,  2.6459e-09, -6.8629e-09, -5.7034e-09,  2.8962e-09,
         4.9851e-09,  2.5528e-10, -3.3097e-09, -1.3599e-09,  1.0776e-09,
        -4.5732e-09,  2.5115e-09, -2.5629e-08, -4.4955e-09,  4.6202e-10,
        -1.0757e-08,  5.6352e-09,  2.6007e-09,  2.9295e-09, -6.1017e-09,
        -6.3820e-09,  7.0020e-09,  3.4540e-09, -1.4110e-08,  1.5938e-08,
        -7.2814e-09,  5.8208e-11, -4.3894e-10,  3.7846e-09, -2.0550e-09,
        -4.3295e-09, -7.0759e-10,  4.0845e-09, -5.4166e-09,  5.9364e-09,
        -9.0649e-09, -5.5056e-09,  9.4951e-10, -1.9088e-10, -1.0464e-08,
```

```
         4.9719e-09, -5.2760e-09, -3.5343e-09,  4.1364e-09,  3.9418e-09,
        -7.1237e-09, -8.0755e-09, -2.1064e-09,  2.0966e-09, -8.7402e-09,
         7.1108e-10, -4.6250e-09,  3.2560e-10, -3.2642e-09,  8.9577e-09,
         7.7989e-11, -1.4643e-10,  4.5657e-10,  2.0612e-09,  5.3884e-09,
         4.4020e-10,  2.9877e-09, -6.9306e-10, -7.9459e-10, -3.9286e-09,
         2.0361e-10, -2.3259e-08, -2.5295e-09,  9.6176e-10,  1.2789e-09,
         8.4071e-10, -9.2223e-09, -2.3286e-09, -1.3281e-09,  2.9844e-09,
        -1.6032e-09, -5.8574e-10, -2.0000e-09, -3.7389e-09, -5.1930e-09,
         1.3533e-09, -9.4883e-10,  1.3658e-09,  2.5932e-09,  3.9768e-10,
         1.9795e-09,  8.4757e-09,  4.5996e-09, -2.9985e-11, -1.0038e-09,
         3.3532e-09,  6.1441e-09, -5.5006e-09, -6.4773e-10,  7.5818e-09,
         8.6328e-10, -5.9208e-10,  2.7380e-09,  4.2269e-09,  8.4012e-10,
         1.6019e-09,  1.0082e-08, -4.6712e-09, -2.8971e-09, -2.7017e-09,
        -7.6038e-09, -7.9291e-09, -3.8669e-09, -7.7329e-09,  7.6327e-09,
        -7.2807e-09, -1.1824e-08, -1.6020e-09,  1.4193e-09,  3.2193e-08,
         7.7075e-09,  6.7794e-09,  1.0206e-09,  2.3559e-09,  1.1173e-09,
         9.4385e-09])
tensor([ 0.0097,  0.0057, -0.0027, -0.0337,  0.0025, -0.0008,  0.0278,  0.0400,
        -0.0012, -0.0049, -0.0185, -0.0694, -0.0052, -0.0154, -0.0036,  0.0156,
         0.0004, -0.0079, -0.0218, -0.0299,  0.0027, -0.0148,  0.0086,  0.0167,
         0.0121,  0.0372,  0.0171, -0.0209, -0.0138, -0.0148,  0.0136, -0.0288,
        -0.0134, -0.0356,  0.0222,  0.0060,  0.0020, -0.0133,  0.0317,  0.0154,
        -0.0042,  0.0074,  0.0083,  0.0021,  0.0119,  0.0811,  0.0026,  0.0470,
        -0.0278, -0.0074, -0.0102,  0.0136,  0.0271,  0.0185,  0.0140,  0.0067,
         0.0061,  0.0318, -0.0217, -0.0116, -0.0049,  0.0115,  0.0051, -0.0020,
         0.0018, -0.0060,  0.0039, -0.0107, -0.0763, -0.0013, -0.0036,  0.0388,
         0.0133,  0.0128, -0.0285, -0.0260, -0.0085, -0.0039,  0.0070,  0.0114,
         0.0290,  0.0359, -0.0013, -0.0216,  0.0009,  0.0242, -0.0002, -0.0006,
         0.0081, -0.0083, -0.0117, -0.0126, -0.0013, -0.0049, -0.0594, -0.0087,
         0.0080,  0.0162, -0.0106, -0.0044,  0.0146,  0.0036, -0.0025, -0.0006,
        -0.0087,  0.0286, -0.0062,  0.0167, -0.0100,  0.0339, -0.0052, -0.0070,
        -0.0156, -0.0126,  0.0004,  0.0172, -0.0010,  0.0122, -0.0120,  0.0093,
        -0.0224,  0.0186,  0.0138,  0.0180, -0.0112, -0.0091, -0.0082,  0.0167,
        -0.0052,  0.0426,  0.0062,  0.0174, -0.0039,  0.0018, -0.0209,  0.0135,
        -0.0113,  0.0526,  0.0028,  0.0391,  0.0065, -0.0015, -0.0043,  0.0029,
        -0.0105, -0.0148, -0.0179, -0.0100,  0.0042, -0.0421,  0.0151, -0.0017,
        -0.0148, -0.0008, -0.0367, -0.0026, -0.0055,  0.0041,  0.0293, -0.0177,
         0.0617,  0.0112,  0.0059,  0.0142,  0.0068,  0.0193, -0.0378,  0.0169,
         0.0214, -0.0009,  0.0428,  0.0050,  0.0001,  0.0648, -0.0096, -0.0047,
         0.0032,  0.0265, -0.0040,  0.0061, -0.0099,  0.0060, -0.0354, -0.0055,
         0.0019, -0.0062, -0.0398, -0.0081,  0.0126, -0.0116, -0.0062, -0.0034,
        -0.0170, -0.0069, -0.0042, -0.0009, -0.0059,  0.0141,  0.0044,  0.0089,
        -0.0068, -0.0088,  0.0213, -0.0063, -0.0027,  0.0003, -0.0059,  0.0033,
         0.0077,  0.0214,  0.0107, -0.0004,  0.0177, -0.0163, -0.0088,  0.0152,
         0.0027, -0.0057, -0.0103,  0.0029, -0.0049,  0.0191, -0.0145, -0.0063,
        -0.0147, -0.0043,  0.0047,  0.0033,  0.0041, -0.0625, -0.0223, -0.0051,
         0.0185, -0.0070,  0.0044, -0.0176, -0.0115, -0.0167,  0.0132,  0.0108,
         0.0129,  0.0401, -0.0297,  0.0153, -0.0209, -0.0047, -0.0340,  0.0482,
```

```
         0.0062, -0.0382, -0.0102,  0.0076, -0.0023, -0.0138,  0.0034, -0.0145])
tensor([ 9.2455e-03,  1.0497e-02, -1.3495e-02, -9.0083e-03,  7.6809e-05,
        -1.8821e-03,  2.1892e-02,  3.9079e-02, -9.1959e-04, -1.2332e-02,
        -1.9587e-02, -8.4155e-02, -2.2924e-03, -3.2655e-02, -6.1325e-03,
         1.4393e-02, -1.1477e-03, -1.2342e-02, -3.0272e-02, -3.9347e-02,
         8.0780e-03, -2.1444e-02,  8.4543e-03,  8.9597e-03,  1.0897e-02,
        -1.5387e-02, -6.0043e-04, -4.7444e-02, -8.4041e-03, -8.0687e-03,
         1.5029e-02, -3.5130e-02, -3.9963e-03, -4.1121e-02,  1.2143e-02,
         4.3932e-03,  2.9104e-04, -7.3783e-03,  2.1401e-02,  1.5153e-02,
        -5.4226e-03,  1.2647e-02,  1.0174e-02, -3.2063e-03,  1.4185e-02,
         4.5266e-02,  5.4934e-03,  4.0964e-02, -3.1397e-02, -5.1104e-03,
        -2.1999e-02,  1.5294e-02,  2.7159e-02,  2.5794e-02,  1.9495e-02,
         1.6185e-04,  2.4544e-03,  3.4514e-02, -2.5249e-02, -5.6322e-03,
        -1.0708e-02,  1.7493e-02,  4.6540e-03,  1.0581e-03,  7.1694e-03,
        -8.6459e-03, -8.8434e-05, -1.7900e-02, -5.3071e-02, -4.4658e-03,
         1.0446e-03,  3.7008e-02,  1.9156e-02,  1.2609e-02, -2.2122e-02,
        -3.0901e-02, -1.4352e-02, -1.1603e-02,  8.2809e-03,  8.5584e-03,
         2.6041e-02,  3.2081e-02, -3.3009e-03, -7.5740e-03, -5.4325e-03,
         2.0693e-02, -2.4497e-04,  1.6271e-03,  6.5728e-03, -4.1938e-03,
        -2.2772e-02, -5.3366e-03,  4.4620e-04, -1.0905e-02, -5.9096e-02,
        -1.1992e-02,  8.6054e-03,  1.8624e-02, -1.1787e-02,  1.0147e-02,
         3.6235e-03,  2.0774e-04, -4.0620e-03,  1.6288e-03, -2.5113e-02,
         2.1906e-02, -1.2033e-02,  1.0130e-02, -1.1662e-02,  3.0257e-02,
        -1.2907e-03, -1.2385e-02, -2.1016e-02, -1.2142e-02,  1.4956e-03,
         8.6220e-03, -2.8394e-03,  9.6722e-03, -1.7464e-02,  3.1862e-03,
         3.0362e-03,  1.4221e-02,  1.2592e-02,  1.1094e-02, -9.3498e-03,
        -1.3809e-02, -1.1708e-02,  2.2390e-02, -4.4625e-04,  4.5408e-02,
         2.7113e-03,  9.3438e-03, -5.8926e-03, -9.7662e-04, -2.8443e-02,
         1.8844e-02, -9.7140e-03,  5.0974e-02, -4.4024e-04,  3.1987e-02,
         2.9806e-03, -8.8747e-03, -1.2556e-02,  9.2023e-03, -1.2500e-02,
        -1.9724e-02, -2.1519e-02, -1.9463e-02, -2.6445e-03, -3.2163e-02,
         7.5544e-03, -5.6654e-03, -1.4877e-02, -9.9489e-03, -3.3502e-02,
         7.6455e-03, -5.7510e-03,  3.2320e-03,  2.8148e-02, -3.1199e-02,
         3.2231e-02,  1.4639e-02,  3.0833e-03,  1.0940e-02,  6.4542e-03,
         1.7187e-02, -3.3876e-02,  1.6632e-02,  1.3871e-02,  5.4734e-03,
         4.2237e-02, -2.4096e-03, -1.0073e-02,  1.1877e-02,  1.6007e-02,
        -7.9104e-03, -2.4122e-04,  1.9234e-02, -4.1464e-04,  6.3528e-03,
        -1.6839e-02,  1.3432e-02, -3.0890e-02, -7.6916e-03, -1.1437e-03,
        -8.4535e-03, -2.1379e-02, -1.0554e-02,  1.1423e-02, -2.0404e-02,
        -9.0134e-03, -1.2024e-03, -1.0089e-02, -9.3310e-03, -7.5073e-03,
        -3.6927e-03, -1.1367e-02,  3.6104e-03,  1.1803e-03,  9.4127e-03,
        -1.3378e-02, -5.3719e-03,  2.2938e-02, -1.2542e-02,  3.8797e-03,
        -2.2959e-03, -1.5238e-03,  9.2871e-03,  5.7084e-03,  2.3539e-02,
         6.5282e-03, -4.8670e-03,  1.0736e-02, -2.0107e-02, -1.1869e-02,
         1.7571e-02, -1.3486e-03, -2.1000e-03, -1.8563e-02,  1.6275e-03,
        -7.5599e-03,  3.1586e-02, -1.1298e-02, -1.0222e-02, -1.3141e-02,
        -3.6440e-03,  5.1057e-03,  3.2879e-03,  6.3445e-03, -1.9973e-02,
        -2.6736e-02, -4.0726e-03,  6.2524e-04, -1.1678e-02,  1.0317e-03,
```

```
        -1.0793e-02,  2.5869e-03, -1.0249e-02, -2.4336e-03,  8.9229e-03,
         1.4520e-02,  2.8179e-02, -1.1400e-02,  8.9839e-03, -2.3782e-02,
        -1.0788e-02, -4.0061e-02,  3.5887e-02,  2.0623e-04, -3.7707e-02,
        -1.1797e-02,  6.3424e-03, -5.6170e-03, -1.3539e-02,  4.7062e-03,
        -1.6052e-02])
tensor([[[[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         …,

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]]],


        [[[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         …,

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
```

```
   [ 0.0000,  0.0000,  0.0000]],

  [[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]],

  [[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]],

  [[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]],

  [[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]],

  …,

  [[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]],

  [[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]],

  [[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]]],


…,


[[[ 0.0064,  0.0196,  0.0087],
   [-0.0069,  0.0009, -0.0035],
   [-0.0024,  0.0014,  0.0015]],

  [[-0.0060, -0.0053, -0.0054],
   [-0.0089, -0.0042, -0.0054],
   [-0.0119, -0.0065, -0.0043]],
```

```
 [[ 0.0008, -0.0035, -0.0085],
  [ 0.0060, -0.0016, -0.0080],
  [ 0.0080,  0.0050, -0.0015]],

 …,

 [[ 0.0187,  0.0315,  0.0312],
  [ 0.0214,  0.0317,  0.0287],
  [ 0.0214,  0.0308,  0.0308]],

 [[ 0.0183,  0.0118, -0.0035],
  [ 0.0115,  0.0099, -0.0062],
  [ 0.0014,  0.0006, -0.0101]],

 [[ 0.0172,  0.0129,  0.0055],
  [ 0.0125,  0.0100,  0.0007],
  [ 0.0133,  0.0162,  0.0075]]],


[[[-0.0030, -0.0011, -0.0015],
  [-0.0054, -0.0206, -0.0287],
  [-0.0034, -0.0244, -0.0269]],

 [[ 0.0004,  0.0066,  0.0104],
  [ 0.0006,  0.0050,  0.0080],
  [ 0.0076,  0.0033,  0.0048]],

 [[ 0.0085,  0.0018,  0.0021],
  [ 0.0051, -0.0010,  0.0025],
  [ 0.0148,  0.0143,  0.0219]],

 …,

 [[ 0.0058,  0.0044,  0.0090],
  [ 0.0028, -0.0015,  0.0049],
  [ 0.0068,  0.0014,  0.0066]],

 [[ 0.0056,  0.0152,  0.0307],
  [-0.0173,  0.0025,  0.0160],
  [-0.0183, -0.0049,  0.0081]],

 [[ 0.0100,  0.0090,  0.0154],
  [ 0.0018, -0.0008,  0.0037],
  [ 0.0119,  0.0093,  0.0073]]],


[[[ 0.0000,  0.0000,  0.0000],
```

```
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         …,

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]]]]])
tensor([ 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  7.5460e-02,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  3.2193e-03,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  4.5950e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  1.1082e-02,  0.0000e+00,  1.0567e-02,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -3.0807e-05,
```

```
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  7.8812e-03,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00, -6.8775e-03,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00, -3.2407e-02,  0.0000e+00,
 0.0000e+00, -2.8827e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 6.5965e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 2.2163e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00, -4.7356e-02,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  2.6206e-03,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  1.8984e-03,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  1.1607e-03,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  4.8796e-03, -4.3756e-05,  0.0000e+00,  0.0000e+00,
 0.0000e+00, -4.9888e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
```

```
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
          0.0000e+00,  1.6196e-02,  6.1637e-03,  0.0000e+00])
tensor([[[[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          …,

          [[ 0.0266,  0.0008, -0.0256],
           [ 0.0673,  0.0177, -0.0153],
           [-0.0034, -0.0371, -0.0497]],

          [[ 0.1482,  0.1318,  0.1155],
           [ 0.2861,  0.3155,  0.2532],
           [ 0.3217,  0.3837,  0.3004]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]]],


         [[[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          [[ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000],
           [ 0.0000,  0.0000,  0.0000]],

          …,

          [[-0.0375, -0.0646, -0.0595],
           [-0.0298, -0.0461, -0.0489],
```

```
   [-0.0066, -0.0054, -0.0165]],

  [[ 0.0058,  0.0034,  0.0086],
   [-0.0044, -0.0007,  0.0047],
   [-0.0118, -0.0025, -0.0012]],

  [[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]],

  [[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]],

  [[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]],

  …,

  [[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]],

  [[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]],

  [[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]]],


…,


[[[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]],

  [[ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000],
   [ 0.0000,  0.0000,  0.0000]],
```

```
 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[-0.0063,  0.0027,  0.0035],
  [ 0.0070,  0.0127,  0.0049],
  [ 0.0207,  0.0060, -0.0055]],

 [[-0.0275, -0.0132,  0.0053],
  [ 0.0113,  0.0279,  0.0479],
  [ 0.0324,  0.0366,  0.0509]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000],
```

```
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         …,

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000]]]])
tensor([ 9.3537e-02, -7.1817e-03,  0.0000e+00,  2.1625e-02, -4.4708e-03,
         5.8657e-02,  2.2341e-03,  2.3354e-02,  1.4853e-02,  0.0000e+00,
         8.1803e-04, -8.1705e-03, -4.0348e-02,  0.0000e+00,  2.8413e-05,
         9.0298e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -7.7668e-07,  0.0000e+00,  3.9316e-03,  0.0000e+00,
         1.3252e-02,  0.0000e+00,  2.4164e-03,  0.0000e+00, -3.1232e-04,
        -7.7189e-04,  3.0652e-02, -1.0304e-03,  0.0000e+00, -3.6151e-04,
         8.9321e-03,  2.9462e-03, -6.4432e-03, -2.0077e-03,  2.0103e-02,
         2.1647e-02, -2.3591e-03, -4.8563e-04, -3.0486e-03,  0.0000e+00,
        -3.3616e-03,  0.0000e+00,  0.0000e+00,  3.3021e-03,  1.2483e-04,
         0.0000e+00, -5.5628e-03, -3.8503e-04, -6.1539e-09, -2.9349e-03,
         0.0000e+00,  1.8157e-02,  3.1414e-03,  8.7689e-05,  0.0000e+00,
         7.0916e-04,  0.0000e+00, -2.4464e-03,  0.0000e+00, -2.9570e-03,
        -2.2951e-02, -1.7011e-02, -5.7398e-03, -8.8367e-02,  3.5836e-02,
         6.1155e-08,  0.0000e+00, -1.4140e-04,  0.0000e+00, -1.0042e-06,
         0.0000e+00,  2.1742e-03,  0.0000e+00,  3.7009e-04, -2.2611e-03,
         0.0000e+00,  0.0000e+00,  1.4642e-03, -1.1061e-02,  8.0286e-03,
         0.0000e+00,  7.2784e-02, -2.4659e-02,  9.8433e-05,  0.0000e+00,
         6.5479e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  6.6053e-05,
         0.0000e+00,  0.0000e+00,  5.3534e-02,  0.0000e+00,  1.6177e-03,
         8.6671e-03, -9.5407e-04,  1.0394e-01,  5.6307e-03,  4.5106e-06,
        -1.2635e-02, -1.2626e-02,  6.3418e-03, -2.1956e-03,  2.7460e-02,
        -1.4457e-02,  0.0000e+00,  0.0000e+00, -3.5598e-03,  2.8059e-02,
        -1.1110e-02, -7.4233e-04,  6.2000e-07,  0.0000e+00, -3.0072e-02,
```

```
 1.8871e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,  3.4419e-04,
 0.0000e+00, -5.9769e-03,  1.0420e-04,  0.0000e+00, -7.3074e-04,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  5.6063e-02,  2.2158e-02,
 3.8096e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  3.5599e-07,
-2.0436e-03, -1.6570e-02, -1.3565e-02,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  8.4994e-03, -1.6873e-03,  0.0000e+00,
 0.0000e+00,  0.0000e+00, -1.4115e-02,  1.4297e-06, -2.4374e-09,
-7.4551e-03,  0.0000e+00,  0.0000e+00,  4.2762e-03,  3.9411e-02,
-1.3508e-02,  1.0362e-04,  0.0000e+00,  0.0000e+00, -3.2065e-03,
 0.0000e+00,  0.0000e+00,  5.2952e-04,  1.7869e-03,  3.2768e-02,
 1.6639e-07,  0.0000e+00,  0.0000e+00,  0.0000e+00, -4.1049e-02,
-4.1681e-02,  0.0000e+00, -1.2266e-05, -1.6565e-02,  5.9758e-04,
 0.0000e+00,  4.0780e-04, -2.7793e-04, -8.8635e-05,  2.7430e-02,
 0.0000e+00,  1.3456e-03,  1.4939e-02,  2.9336e-03, -1.5301e-02,
 5.3830e-04,  0.0000e+00,  0.0000e+00, -1.5432e-02,  3.7174e-02,
 0.0000e+00,  2.4336e-03, -1.0562e-02,  0.0000e+00,  0.0000e+00,
-3.7254e-02, -1.6796e-02, -1.6747e-06, -1.2022e-02,  0.0000e+00,
 0.0000e+00, -3.3465e-03,  0.0000e+00,  0.0000e+00, -6.0456e-05,
 1.3935e-02,  2.1754e-04, -1.8316e-04, -1.5130e-03,  4.8348e-03,
 8.0239e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00,  0.0000e+00,  1.2578e-04,  7.8514e-03,  0.0000e+00,
 0.0000e+00, -6.1299e-03,  1.8404e-02, -3.7625e-04,  0.0000e+00,
 8.4381e-03,  0.0000e+00,  0.0000e+00,  5.3361e-06,  1.9125e-03,
 4.5010e-03,  0.0000e+00,  2.7615e-03, -2.0920e-02, -1.9117e-02,
 0.0000e+00,  1.6190e-02,  0.0000e+00, -8.8620e-04,  2.7551e-02,
 0.0000e+00,  1.5845e-02,  0.0000e+00,  0.0000e+00, -1.8786e-03,
 0.0000e+00,  5.2091e-02, -1.8388e-02,  0.0000e+00,  0.0000e+00,
 3.4181e-05,  0.0000e+00, -1.8015e-04,  2.7288e-02, -1.6887e-06,
 0.0000e+00,  1.6028e-02,  2.0996e-02, -1.3934e-05,  0.0000e+00,
-1.1243e-03,  1.7735e-02,  5.0323e-03,  1.5140e-03, -3.2024e-02,
 0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  1.0501e-02,
 8.2368e-03,  1.3616e-02, -8.8668e-03,  0.0000e+00, -1.2732e-02,
-1.5446e-02,  0.0000e+00,  0.0000e+00,  4.6751e-02, -6.5709e-02,
 0.0000e+00,  5.0756e-02,  0.0000e+00,  1.5752e-03, -2.8912e-03,
 1.7210e-03,  0.0000e+00,  3.8399e-04,  1.7661e-04, -1.0021e-06,
-5.4869e-03,  1.6324e-04, -3.1185e-04,  0.0000e+00, -2.7929e-07,
-3.5151e-02, -8.3619e-02, -5.3566e-05,  5.4346e-02,  0.0000e+00,
 5.5176e-03, -6.5556e-04,  0.0000e+00, -1.6404e-02,  0.0000e+00,
 0.0000e+00,  7.1426e-03,  1.5349e-04,  3.5202e-03,  5.6531e-03,
 2.1304e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
-4.2496e-06,  2.0403e-04, -7.9568e-04, -2.8345e-02, -3.2955e-06,
 1.7463e-02,  0.0000e+00,  0.0000e+00, -1.8479e-01,  0.0000e+00,
-1.0152e-03,  2.2093e-02, -2.0326e-02,  6.5120e-05, -2.6629e-02,
 0.0000e+00,  0.0000e+00, -5.0218e-04, -3.6438e-02,  2.5756e-02,
 1.8001e-02,  1.0738e-02,  1.2148e-03,  0.0000e+00, -3.3266e-03,
-1.2842e-03,  6.8613e-04,  0.0000e+00,  0.0000e+00,  0.0000e+00,
 0.0000e+00, -8.7631e-04, -2.9041e-04,  8.7109e-04,  0.0000e+00,
 1.7531e-03,  1.6558e-04,  0.0000e+00,  1.5170e-02,  0.0000e+00,
```

```
         0.0000e+00,  0.0000e+00, -1.0530e-02,  0.0000e+00,  0.0000e+00,
        -7.7461e-02, -1.6570e-01, -4.4840e-09,  0.0000e+00,  9.7600e-03,
         0.0000e+00, -3.7597e-04,  0.0000e+00,  2.6848e-02,  8.2226e-04,
         1.8141e-04,  5.4150e-03,  9.7469e-03,  0.0000e+00, -1.5751e-02,
         0.0000e+00,  1.7567e-02,  0.0000e+00,  0.0000e+00])
tensor([[[[ 5.1998e-03, -1.6215e-02, -8.7862e-03],
          [ 1.4840e-03, -1.7040e-02, -1.4020e-02],
          [-1.7451e-03, -7.3971e-03, -1.1087e-02]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  1.6613e-08]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 2.7930e-03,  2.4855e-03,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


        [[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

         …,

         [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
          [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  …,

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


…,


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

  [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
   [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],
```

```
 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 …,

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

 [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
  [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]],


[[[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
```

```
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          …,

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]],

          [[ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00],
           [ 0.0000e+00,  0.0000e+00,  0.0000e+00]]]])
tensor([-2.5959e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  1.1487e-02, -1.4269e-01, -5.7299e-02,
        -4.2137e-02,  8.9275e-03, -8.5105e-03,  0.0000e+00,  0.0000e+00,
         2.1846e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         2.4321e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  3.2082e-02,
         1.3331e-02, -4.5263e-03,  1.0958e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         3.1929e-02,  0.0000e+00,  0.0000e+00, -1.2727e-02,  1.0165e-01,
         0.0000e+00,  0.0000e+00, -1.5251e-04,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -2.8527e-02,  0.0000e+00, -1.1489e-03,
         0.0000e+00,  0.0000e+00, -1.9618e-03, -2.7357e-02,  0.0000e+00,
         5.7696e-02, -5.4149e-05,  0.0000e+00,  0.0000e+00, -4.0188e-02,
         3.9489e-04,  1.9334e-02,  0.0000e+00, -1.0499e-02,  1.9628e-03,
         0.0000e+00,  0.0000e+00,  2.0127e-03,  1.9814e-02,  0.0000e+00,
         0.0000e+00, -1.4251e-01, -1.0840e-01, -7.6402e-05,  0.0000e+00,
         0.0000e+00,  4.6232e-02, -2.5379e-03,  5.7278e-04,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00, -2.4280e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -1.0122e-05, -8.0962e-03,  0.0000e+00, -4.4747e-02, -2.6698e-02,
         2.8200e-06,  0.0000e+00,  0.0000e+00,  0.0000e+00,  6.3113e-03,
         0.0000e+00,  0.0000e+00,  1.9563e-01,  0.0000e+00,  0.0000e+00,
        -1.5067e-02,  0.0000e+00,  4.4507e-06,  0.0000e+00,  4.1838e-04,
         0.0000e+00,  0.0000e+00, -4.7253e-03,  2.5536e-02,  3.1859e-01,
```

```
         0.0000e+00,  0.0000e+00, -5.8975e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -1.0831e-03,  0.0000e+00, -2.3385e-03,  1.3343e-06, -2.1673e-05,
         0.0000e+00,  0.0000e+00,  4.6947e-02, -5.0626e-02,  7.1037e-02,
         2.2283e-02,  0.0000e+00, -8.6305e-08,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -1.0651e-05,  0.0000e+00,  0.0000e+00,
        -1.0985e-02,  0.0000e+00,  0.0000e+00, -1.3478e-07,  0.0000e+00,
         0.0000e+00,  8.2863e-02,  0.0000e+00,  0.0000e+00, -7.0253e-04,
         0.0000e+00,  0.0000e+00, -2.1412e-01,  0.0000e+00,  0.0000e+00,
        -3.9091e-03,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -9.5590e-05,
         0.0000e+00, -1.4719e-01,  0.0000e+00,  1.5034e-02,  7.2665e-03,
        -1.5112e-02, -1.1665e-05,  0.0000e+00,  8.0785e-03, -5.0886e-02,
         0.0000e+00,  0.0000e+00,  1.6044e-03, -1.1184e-07,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  5.5637e-03,  0.0000e+00,
         2.1575e-06,  0.0000e+00, -2.3854e-03,  1.0461e-02,  0.0000e+00,
         0.0000e+00,  8.6650e-02, -9.2136e-03,  0.0000e+00,  8.9078e-02,
         3.9884e-03, -1.4294e-02,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -1.8853e-01,  0.0000e+00,  0.0000e+00,  2.1282e-01,  0.0000e+00,
         0.0000e+00,  0.0000e+00, -5.9827e-03,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  1.5567e-01, -6.8637e-04,
         0.0000e+00, -4.5415e-02, -7.4855e-02, -5.3960e-03,  0.0000e+00,
         4.0541e-02,  0.0000e+00,  6.8631e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00, -5.2680e-04,
         0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,  0.0000e+00,
        -1.6427e-02,  0.0000e+00,  0.0000e+00, -1.0548e-01,  2.2112e-04,
         0.0000e+00,  3.0437e-02, -3.4113e-02,  0.0000e+00,  0.0000e+00,
         0.0000e+00])
tensor([[0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        …,
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.]])
tensor([-9.4150e-03,  0.0000e+00, -4.5146e-05,  …,  0.0000e+00,
        -5.7085e-04,  0.0000e+00])
tensor([[0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        …,
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.],
        [0., 0., 0.,  …, 0., 0., 0.]])
tensor([-0.0001,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000])
tensor([[3.8472e-07, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00],
        [1.3334e-08, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
```

```
         0.0000e+00],
        [6.8501e-08, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00],
        …,
        [1.0784e-12, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00],
        [1.0512e-12, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00],
        [9.2442e-13, 0.0000e+00, 0.0000e+00,  …, 0.0000e+00, 0.0000e+00,
         0.0000e+00]])
tensor([ 1.7923e-02,  1.6702e-02, -1.9121e-02,  3.2198e-02,  4.9198e-02,
        -3.4794e-02,  1.2391e-02,  3.6692e-02, -1.8096e-02, -6.5308e-02,
        -7.0390e-02,  2.7345e-03, -4.0005e-02,  2.9927e-02, -8.8724e-04,
         3.4420e-03,  4.5999e-03,  2.3141e-02,  8.0945e-03,  3.6020e-03,
         1.1174e-04,  7.8653e-06,  7.8942e-06,  8.0496e-06,  7.8931e-06,
         7.9952e-06,  8.1078e-06,  7.8855e-06,  8.1284e-06,  8.1376e-06,
         8.1079e-06,  7.9136e-06,  8.1157e-06,  8.0565e-06,  7.9230e-06,
         7.8924e-06,  8.0327e-06,  8.1291e-06,  7.9340e-06,  8.1406e-06,
         8.0375e-06,  7.9584e-06,  7.7528e-06,  7.9895e-06,  7.9928e-06,
         8.1263e-06,  8.0681e-06,  8.0133e-06,  8.0853e-06,  7.8765e-06,
         8.1370e-06,  8.0354e-06,  8.1230e-06,  7.8825e-06,  8.1045e-06,
         8.1100e-06,  8.0486e-06,  8.1344e-06,  7.9178e-06,  8.0735e-06,
         8.1240e-06,  8.0976e-06,  8.0337e-06,  8.0329e-06,  8.1253e-06,
         8.1069e-06,  8.0280e-06,  8.1225e-06,  7.8815e-06,  7.8872e-06,
         8.0561e-06,  8.1327e-06,  7.8926e-06,  8.0135e-06,  8.1010e-06,
         7.8824e-06,  8.1318e-06,  7.9945e-06,  8.0031e-06,  7.5868e-06,
         7.9931e-06,  8.0132e-06,  7.9342e-06,  8.1339e-06,  8.0465e-06,
         7.6227e-06,  8.1001e-06,  8.1284e-06,  8.1237e-06,  7.8862e-06,
         7.9673e-06,  8.0384e-06,  8.1303e-06,  7.8962e-06,  7.9336e-06,
         8.1326e-06,  8.1332e-06,  8.0078e-06,  7.9095e-06,  8.1253e-06,
         8.1084e-06,  7.9074e-06,  8.1405e-06,  8.0241e-06,  8.0417e-06,
         8.1071e-06,  8.0587e-06,  8.0095e-06,  7.9191e-06,  7.9239e-06,
         8.0771e-06,  8.1261e-06,  8.0862e-06,  7.8889e-06,  8.0807e-06,
         7.9335e-06,  8.0944e-06,  7.9810e-06,  8.0676e-06,  7.8929e-06,
         7.8556e-06,  7.9739e-06,  7.8795e-06,  8.0326e-06,  8.0186e-06,
         7.9202e-06,  8.1225e-06,  8.0233e-06,  8.1238e-06,  8.0395e-06,
         7.9114e-06,  7.8366e-06,  7.9234e-06,  8.1337e-06,  8.0715e-06,
         7.8838e-06,  8.0724e-06,  7.9587e-06,  7.8796e-06,  7.9459e-06,
         8.1206e-06,  7.9420e-06,  8.1012e-06,  7.9215e-06,  7.8820e-06,
         8.1233e-06,  7.8811e-06,  8.1386e-06,  7.9919e-06,  8.1241e-06,
         8.1281e-06,  8.1383e-06,  8.1099e-06,  7.7919e-06,  7.8872e-06,
         7.9187e-06,  7.9083e-06,  8.0806e-06,  8.0980e-06,  8.0315e-06,
         8.1329e-06,  7.8617e-06,  7.9728e-06,  8.0437e-06,  8.1043e-06,
         7.8536e-06,  8.0432e-06,  7.9374e-06,  8.0844e-06,  8.1226e-06,
         8.1361e-06,  8.1318e-06,  8.1239e-06,  7.9169e-06,  8.0723e-06,
         8.0336e-06,  8.0624e-06,  7.9986e-06,  8.1417e-06,  8.1274e-06,
         8.1370e-06,  7.8852e-06,  7.9376e-06,  8.0978e-06,  7.8746e-06,
         8.1186e-06,  7.9345e-06,  7.9556e-06,  8.1211e-06,  8.1218e-06,
```

```
7.9113e-06,  8.0719e-06,  8.1326e-06,  8.0921e-06,  7.9448e-06,
7.8871e-06,  7.9176e-06,  8.1160e-06,  7.9870e-06,  7.9144e-06,
7.8958e-06,  8.1148e-06,  7.8881e-06,  8.1196e-06,  8.0876e-06,
8.1302e-06,  8.0771e-06,  8.1231e-06,  7.9341e-06,  8.1265e-06,
7.9104e-06,  8.1297e-06,  7.8759e-06,  8.0214e-06,  8.0221e-06,
7.9735e-06,  8.1272e-06,  8.1317e-06,  8.0956e-06,  7.9304e-06,
8.0275e-06,  7.9481e-06,  7.9540e-06,  7.9023e-06,  7.8635e-06,
8.1224e-06,  7.9074e-06,  8.0277e-06,  7.8678e-06,  7.9337e-06,
8.0819e-06,  8.0943e-06,  8.0366e-06,  8.1291e-06,  8.0783e-06,
8.1308e-06,  8.0259e-06,  8.1312e-06,  7.9563e-06,  8.0817e-06,
7.9369e-06,  7.9221e-06,  8.0637e-06,  8.0578e-06,  8.0687e-06,
8.0872e-06,  7.9671e-06,  8.0224e-06,  8.0020e-06,  8.1145e-06,
8.1393e-06,  8.1101e-06,  7.8530e-06,  7.8833e-06,  7.8871e-06,
8.1202e-06,  7.9585e-06,  8.0154e-06,  8.0329e-06,  8.0011e-06,
7.8918e-06,  8.0256e-06,  7.8863e-06,  8.0794e-06,  8.0658e-06,
8.0925e-06,  8.1017e-06,  8.0796e-06,  7.9591e-06,  8.1194e-06,
8.0006e-06,  7.8907e-06,  8.1017e-06,  7.9726e-06,  7.8742e-06,
8.1209e-06,  8.0469e-06,  7.8630e-06,  7.9452e-06,  8.1004e-06,
7.9885e-06,  8.1113e-06,  7.9917e-06,  8.1152e-06,  8.1295e-06,
7.8975e-06,  7.9383e-06,  7.8628e-06,  7.9309e-06,  8.1250e-06,
7.8756e-06,  8.1259e-06,  7.9527e-06,  7.9749e-06,  8.1350e-06,
7.9888e-06,  8.1279e-06,  7.9638e-06,  8.0399e-06,  8.1063e-06,
8.0649e-06,  8.1327e-06,  7.9343e-06,  8.1202e-06,  8.0229e-06,
8.0900e-06,  8.1132e-06,  7.8981e-06,  8.1218e-06,  7.9018e-06,
7.9178e-06,  8.1183e-06,  8.0147e-06,  7.9295e-06,  8.1216e-06,
8.0464e-06,  8.0819e-06,  7.8903e-06,  7.9789e-06,  8.0476e-06,
8.0375e-06,  8.0991e-06,  7.8856e-06,  7.9181e-06,  8.0920e-06,
8.0850e-06,  8.1380e-06,  7.8620e-06,  8.0819e-06,  8.1188e-06,
8.0586e-06,  8.1241e-06,  7.9389e-06,  8.1022e-06,  7.8751e-06,
8.1066e-06,  8.1149e-06,  7.9706e-06,  8.0609e-06,  8.0996e-06,
7.9656e-06,  7.8779e-06,  8.1197e-06,  8.1340e-06,  7.8689e-06,
7.9029e-06,  8.1391e-06,  7.9382e-06,  8.1210e-06,  7.9525e-06,
7.8859e-06,  8.0914e-06,  8.0332e-06,  8.1070e-06,  8.1234e-06,
8.0619e-06,  8.1096e-06,  8.1299e-06,  7.8792e-06,  7.9901e-06,
8.0702e-06,  8.0081e-06,  7.8925e-06,  7.8985e-06,  8.0709e-06,
8.0673e-06,  7.8924e-06,  7.9826e-06,  7.9788e-06,  7.9312e-06,
8.0829e-06,  8.0424e-06,  8.1260e-06,  7.7191e-06,  8.1163e-06,
7.9653e-06,  8.1114e-06,  7.9085e-06,  8.0546e-06,  7.8828e-06,
8.1150e-06,  7.9078e-06,  8.1167e-06,  8.1064e-06,  7.9353e-06,
8.0711e-06,  8.0054e-06,  8.1236e-06,  8.1204e-06,  7.8806e-06,
7.9596e-06,  8.1336e-06,  8.0932e-06,  8.0933e-06,  8.0617e-06,
7.9733e-06,  8.0306e-06,  7.9347e-06,  7.9908e-06,  7.9855e-06,
7.9070e-06,  8.1261e-06,  8.1400e-06,  7.8890e-06,  7.9003e-06,
8.1114e-06,  8.0329e-06,  7.9825e-06,  8.0402e-06,  7.9158e-06,
8.1324e-06,  8.1311e-06,  8.0985e-06,  8.1289e-06,  8.0066e-06,
8.0929e-06,  7.9255e-06,  8.0215e-06,  8.1168e-06,  8.1310e-06,
8.0766e-06,  8.0522e-06,  8.1109e-06,  7.9522e-06,  8.0474e-06,
7.9242e-06,  7.9095e-06,  7.8563e-06,  8.0020e-06,  8.0732e-06,
```

```
7.9119e-06,  7.9501e-06,  7.8710e-06,  7.9100e-06,  7.9352e-06,
7.9842e-06,  8.1199e-06,  8.1087e-06,  8.0506e-06,  7.9583e-06,
8.1206e-06,  8.1228e-06,  7.9945e-06,  8.1308e-06,  7.9905e-06,
8.0712e-06,  8.0256e-06,  7.8561e-06,  8.1365e-06,  8.1042e-06,
8.0546e-06,  8.0630e-06,  7.8558e-06,  8.0254e-06,  7.9657e-06,
8.0830e-06,  8.0057e-06,  8.0113e-06,  7.9954e-06,  8.1042e-06,
7.8772e-06,  7.9642e-06,  8.0204e-06,  7.8524e-06,  7.9784e-06,
8.0870e-06,  7.8676e-06,  8.1408e-06,  7.9680e-06,  8.1054e-06,
8.1415e-06,  8.0637e-06,  7.8850e-06,  8.0940e-06,  8.1308e-06,
7.9867e-06,  8.1172e-06,  8.0780e-06,  7.9744e-06,  7.9696e-06,
8.0242e-06,  8.0750e-06,  8.0731e-06,  8.0429e-06,  7.8748e-06,
7.8963e-06,  8.1143e-06,  7.9755e-06,  7.8754e-06,  8.1240e-06,
7.9308e-06,  7.8955e-06,  7.8770e-06,  7.9640e-06,  8.0489e-06,
7.8464e-06,  8.1002e-06,  8.1270e-06,  8.1238e-06,  8.0158e-06,
8.0695e-06,  8.1161e-06,  7.9817e-06,  8.0008e-06,  7.9895e-06,
7.8695e-06,  8.1117e-06,  8.0770e-06,  8.0124e-06,  8.0208e-06,
8.0244e-06,  7.9922e-06,  7.8756e-06,  8.1295e-06,  7.8964e-06,
7.9161e-06,  8.0241e-06,  8.0312e-06,  8.0156e-06,  8.1273e-06,
8.1332e-06,  8.1209e-06,  7.8445e-06,  7.8747e-06,  8.1390e-06,
8.1325e-06,  7.9825e-06,  7.8948e-06,  7.9291e-06,  8.0967e-06,
8.0547e-06,  7.9534e-06,  7.8671e-06,  8.1102e-06,  7.8907e-06,
7.9256e-06,  7.9081e-06,  7.9635e-06,  8.0440e-06,  7.8787e-06,
8.1171e-06,  7.8720e-06,  8.1225e-06,  7.9611e-06,  8.0686e-06,
8.1066e-06,  7.9420e-06,  8.0700e-06,  7.9312e-06,  8.0722e-06,
8.0252e-06,  8.0496e-06,  7.9026e-06,  8.1399e-06,  7.8673e-06,
8.1282e-06,  7.9002e-06,  8.0426e-06,  8.1199e-06,  8.1064e-06,
8.1265e-06,  8.1318e-06,  7.9203e-06,  7.9048e-06,  8.0098e-06,
7.8713e-06,  8.1139e-06,  7.9228e-06,  8.0615e-06,  7.9700e-06,
7.9764e-06,  7.9753e-06,  7.9075e-06,  7.9408e-06,  8.1016e-06,
8.0861e-06,  7.8110e-06,  7.9902e-06,  7.8648e-06,  8.0119e-06,
7.8830e-06,  8.1299e-06,  7.9485e-06,  8.0828e-06,  7.9864e-06,
7.8767e-06,  7.8496e-06,  7.9773e-06,  8.0647e-06,  7.8861e-06,
8.0383e-06,  7.8833e-06,  8.1105e-06,  8.1350e-06,  8.0130e-06,
7.9543e-06,  7.9408e-06,  8.0158e-06,  7.8938e-06,  7.9917e-06,
8.1334e-06,  7.9920e-06,  7.9745e-06,  7.9407e-06,  7.8706e-06,
8.0419e-06,  7.9301e-06,  8.1295e-06,  8.0264e-06,  8.1133e-06,
8.0867e-06,  7.8648e-06,  7.8971e-06,  8.0739e-06,  8.0760e-06,
8.0102e-06,  7.9402e-06,  8.1358e-06,  8.1217e-06,  8.0451e-06,
8.0883e-06,  8.0132e-06,  8.0570e-06,  8.0535e-06,  8.0662e-06,
7.8692e-06,  8.0287e-06,  8.0050e-06,  7.9801e-06,  7.9159e-06,
7.9736e-06,  8.1361e-06,  8.0853e-06,  7.8546e-06,  8.0976e-06,
8.0635e-06,  8.0988e-06,  7.8773e-06,  8.0156e-06,  7.9431e-06,
8.1350e-06,  7.9506e-06,  7.9100e-06,  7.9294e-06,  8.1082e-06,
8.0053e-06,  7.9575e-06,  8.1392e-06,  7.8668e-06,  8.1284e-06,
7.4894e-06,  7.9312e-06,  7.9376e-06,  8.1373e-06,  7.9743e-06,
7.9169e-06,  8.0339e-06,  8.0303e-06,  7.9410e-06,  8.1271e-06,
8.1241e-06,  8.0158e-06,  7.9137e-06,  7.9588e-06,  7.9864e-06,
8.1283e-06,  8.0553e-06,  8.1186e-06,  7.7174e-06,  8.0604e-06,
```

```
8.0300e-06,  8.0839e-06,  8.0654e-06,  7.8967e-06,  7.9114e-06,
8.1424e-06,  7.9077e-06,  8.0608e-06,  8.1331e-06,  7.9156e-06,
7.9757e-06,  8.0636e-06,  8.0705e-06,  8.0495e-06,  8.0173e-06,
7.9376e-06,  8.0491e-06,  8.0891e-06,  7.9514e-06,  8.0849e-06,
8.1099e-06,  7.8816e-06,  8.0784e-06,  7.8971e-06,  7.9820e-06,
7.8793e-06,  8.1314e-06,  7.9799e-06,  7.9108e-06,  8.0579e-06,
7.8535e-06,  8.1055e-06,  7.8575e-06,  7.8876e-06,  7.8916e-06,
7.9303e-06,  8.0888e-06,  8.1206e-06,  7.9411e-06,  8.0778e-06,
8.1113e-06,  8.1447e-06,  8.1293e-06,  8.1153e-06,  7.8709e-06,
7.9677e-06,  7.9491e-06,  8.0979e-06,  8.0473e-06,  8.1341e-06,
8.1288e-06,  7.8936e-06,  7.9180e-06,  7.8889e-06,  8.0868e-06,
7.9218e-06,  8.0421e-06,  8.0007e-06,  8.0205e-06,  8.0165e-06,
7.9209e-06,  8.1229e-06,  7.9594e-06,  7.8828e-06,  7.8752e-06,
8.0755e-06,  8.0744e-06,  8.1220e-06,  8.0688e-06,  8.0929e-06,
8.1042e-06,  7.9747e-06,  8.0287e-06,  8.1102e-06,  8.0842e-06,
8.1275e-06,  8.1289e-06,  7.8930e-06,  7.9242e-06,  7.8821e-06,
8.1167e-06,  7.9117e-06,  8.1049e-06,  7.8672e-06,  8.1236e-06,
7.8588e-06,  7.9994e-06,  8.1193e-06,  8.1123e-06,  8.0110e-06,
8.1337e-06,  8.0560e-06,  8.1189e-06,  8.0140e-06,  8.1348e-06,
8.0798e-06,  7.8706e-06,  8.0927e-06,  8.1340e-06,  8.0941e-06,
8.0139e-06,  8.1386e-06,  8.0334e-06,  7.8822e-06,  7.9142e-06,
8.0926e-06,  8.0460e-06,  7.9325e-06,  7.8322e-06,  8.1264e-06,
8.0743e-06,  7.9238e-06,  8.0118e-06,  8.1322e-06,  8.0112e-06,
7.5404e-06,  8.0278e-06,  7.9810e-06,  8.0941e-06,  7.8763e-06,
7.8517e-06,  8.1169e-06,  7.9685e-06,  7.8802e-06,  8.0910e-06,
7.8732e-06,  7.8799e-06,  8.0857e-06,  8.0008e-06,  7.9309e-06,
7.8680e-06,  8.0948e-06,  8.0411e-06,  7.9432e-06,  8.0705e-06,
7.9543e-06,  8.0397e-06,  7.8933e-06,  8.0096e-06,  7.9445e-06,
8.0850e-06,  7.9834e-06,  8.0062e-06,  8.0383e-06,  8.0971e-06,
7.8947e-06,  7.9738e-06,  7.9143e-06,  8.0214e-06,  7.9564e-06,
8.1215e-06,  7.9393e-06,  7.8889e-06,  7.8823e-06,  8.0404e-06,
8.0651e-06,  8.1062e-06,  7.9353e-06,  7.8614e-06,  7.9564e-06,
7.9366e-06,  7.8774e-06,  8.1107e-06,  8.1040e-06,  8.1018e-06,
7.9944e-06,  8.1247e-06,  7.9422e-06,  8.0415e-06,  8.1101e-06,
7.8944e-06,  7.8825e-06,  7.9785e-06,  8.0643e-06,  7.8643e-06,
8.1273e-06,  8.1257e-06,  8.0592e-06,  8.1192e-06,  7.9943e-06,
8.1235e-06,  8.0077e-06,  7.9232e-06,  7.8539e-06,  7.8782e-06,
7.9416e-06,  8.0903e-06,  7.9804e-06,  8.0346e-06,  8.0870e-06,
7.8371e-06,  8.1146e-06,  7.8848e-06,  7.7610e-06,  7.9988e-06,
8.0534e-06,  7.8504e-06,  8.1214e-06,  8.1238e-06,  7.8745e-06,
7.8897e-06,  8.1356e-06,  8.0696e-06,  7.8863e-06,  8.1369e-06,
8.0568e-06,  8.0393e-06,  8.0610e-06,  7.9939e-06,  8.1160e-06,
8.0663e-06,  7.9765e-06,  8.0534e-06,  8.1076e-06,  7.8735e-06,
7.9590e-06,  8.1013e-06,  8.0079e-06,  8.0309e-06,  8.0790e-06,
7.9528e-06,  8.0239e-06,  7.9498e-06,  7.9157e-06,  7.8791e-06,
7.9705e-06,  8.0190e-06,  8.1402e-06,  8.1296e-06,  8.1204e-06,
8.0848e-06,  8.1284e-06,  7.9003e-06,  8.1379e-06,  7.7370e-06,
8.1164e-06,  8.0163e-06,  8.0898e-06,  8.1389e-06,  8.1381e-06,
```

```
        8.1225e-06,  8.1074e-06,  8.0505e-06,  7.8861e-06,  7.8879e-06,
        8.0500e-06,  8.1090e-06,  8.1179e-06,  8.1298e-06,  7.9792e-06,
        8.0985e-06,  7.8035e-06,  8.0361e-06,  7.9264e-06,  8.0284e-06,
        7.8776e-06,  8.1393e-06,  7.8659e-06,  8.0598e-06,  8.1015e-06,
        8.1272e-06,  8.1223e-06,  8.1220e-06,  8.1310e-06,  8.1291e-06,
        8.1220e-06,  8.0199e-06,  7.8871e-06,  7.9853e-06,  8.0306e-06,
        8.1328e-06,  7.9019e-06,  8.0724e-06,  8.0603e-06,  7.9878e-06,
        8.0326e-06,  7.8998e-06,  7.9176e-06,  7.8736e-06,  7.8592e-06,
        7.9978e-06,  8.1213e-06,  8.1368e-06,  8.1332e-06,  7.8994e-06,
        8.1087e-06,  7.9938e-06,  8.0741e-06,  8.1374e-06,  8.0748e-06,
        8.0590e-06,  8.1260e-06,  7.9768e-06,  8.1061e-06,  8.1302e-06,
        8.1216e-06,  7.9952e-06,  7.9226e-06,  7.9048e-06,  8.0141e-06,
        7.8930e-06,  8.0073e-06,  8.0220e-06,  8.0597e-06,  8.0211e-06,
        8.1293e-06,  8.1154e-06,  8.1119e-06,  8.0281e-06,  7.9994e-06,
        7.9390e-06,  8.0403e-06,  7.9500e-06,  7.8874e-06,  8.0879e-06,
        7.8875e-06,  7.8879e-06,  7.8651e-06,  7.9479e-06,  8.0875e-06,
        8.1176e-06,  7.9400e-06,  8.0258e-06,  8.0576e-06,  7.8471e-06,
        7.8905e-06,  8.1380e-06,  8.1312e-06,  8.0699e-06,  7.5308e-06])
end of p.grad

False
Epoch 10 finished
Epoch [10/10], Loss: 0.9569
************************************************************
```

[7]:
```
# read test_label location
labels_path = '/Users/Limit/imagenet_annot/validation_set_labels.csv'
labels_df = pd.read_csv(labels_path)
# find the labels of only the first 20 classes
labels_df_leq_20 = labels_df[labels_df['label'] <= 20]
labels_validation_images = labels_df_leq_20['label'].tolist()
```

[8]:
```
model.eval()
# find the test images
image_path = "/Users/Limit/imagenet-object-localization-challenge_validation/
 ↪val"
filenames_image_path = []
for root, _, filenames in os.walk(image_path):
    for i in filenames:
        if (i.split('.')[0] in labels_df_leq_20['ImageId'].tolist()):
            filenames_image_path.append(i)
true_label = 0
counter = 0
correct_labels = 0
start_time = time.time()
grab_980_max_val = []
```

```
for i in range(len(filenames_image_path)):
    counter +=1
    image_name = image_path + '/' + filenames_image_path[i]
    input_image = Image.open(image_name)

    input_tensor = preprocess(input_image)
    input_batch = input_tensor.unsqueeze(0) # create a mini-batch as expected␣
 ↪by the model

    # move the input and model to GPU for speed if available
    if torch.cuda.is_available():
        input_batch = input_batch.to('cuda')
        model.to('cuda')

    if (counter%100 == 0):
        print("currently at", counter, 'current time is', time.time() -␣
 ↪start_time)
    with torch.no_grad():
        output = model(input_batch)

    # print the prediction results in this format
    ␣
 ↪print('************************************************************************************')
    print('Predicting Test Sample', counter, ':   Prediction is Correct?')
    if (torch.argmax(output[0]).item() == labels_validation_images[i]):
        print('Yes')
        correct_labels += 1
    else:
        print('No')
    prob_softmax = torch.softmax(output[0], dim = 0)
    print("first 20 classes probability:", prob_softmax[:20])
    print()
    print('max probability is', torch.max(prob_softmax, dim = 0))
    print()
    print('the max probability of the rest of 980 dim is')
    print(torch.topk(prob_softmax[20:], k=4))
    ␣
 ↪print('End**************************************************************************************')
    print()

    grab_980_max_val.append(torch.topk(prob_softmax[20:], k=4)[1][1:])
    # The output has unnormalized scores. To get probabilities, can run a␣
 ↪softmax on it.
    probabilities = torch.nn.functional.softmax(output[0], dim=0)

print('the overall testing error is')
print(correct_labels/counter)
```

```
**********************************************************************************
Predicting Test Sample 1 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.4681e-03, 3.3850e-02, 9.2109e-03,
8.0761e-04, 6.6302e-03, 2.0496e-03,
        4.1118e-03, 3.4562e-02, 5.2602e-02, 3.1094e-03, 1.4737e-02, 4.7815e-03,
        5.8257e-02, 3.2129e-02, 3.7541e-04, 2.4207e-02, 6.4940e-01, 2.5755e-03,
        1.7851e-02, 3.7593e-02])

max probability is torch.return_types.max(
values=tensor(0.6494),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5425e-03, 5.4084e-06, 5.4082e-06, 5.4070e-06]),
indices=tensor([  0, 197, 910, 323]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 2 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.8318e-04, 1.1328e-03, 2.2354e-03,
2.5609e-04, 1.3723e-03, 2.6751e-04,
        3.0774e-04, 5.2287e-03, 7.4244e-03, 1.0242e-03, 1.2036e-02, 1.7859e-03,
        8.0503e-03, 3.5215e-02, 3.8192e-05, 1.0650e-02, 7.3591e-01, 1.6052e-03,
        3.3591e-02, 1.3774e-01])

max probability is torch.return_types.max(
values=tensor(0.7359),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6329e-03, 7.4947e-07, 7.4946e-07, 7.4943e-07]),
indices=tensor([  0, 655, 230, 691]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 3 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0052, 0.0572, 0.0047, 0.0035, 0.0120,
0.0199, 0.0106, 0.0204, 0.0472,
        0.0083, 0.2475, 0.0039, 0.0390, 0.0504, 0.0010, 0.1477, 0.0630, 0.0084,
        0.0596, 0.1460])

max probability is torch.return_types.max(
values=tensor(0.2475),
```

```
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.7108e-03, 4.1991e-05, 4.1979e-05, 4.1955e-05]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 4 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.8684e-05, 1.2647e-06, 8.8203e-03,
3.3907e-03, 5.4236e-03, 7.3242e-05,
        1.6255e-04, 8.6174e-05, 1.9246e-04, 2.4334e-03, 2.2137e-04, 4.8783e-07,
        3.6609e-04, 9.0015e-01, 6.4755e-06, 9.8064e-05, 2.0178e-02, 1.6260e-03,
        5.2592e-03, 5.1404e-02])

max probability is torch.return_types.max(
values=tensor(0.9001),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6331e-05, 2.3337e-09, 2.3326e-09, 2.3325e-09]),
indices=tensor([  0, 150, 158, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 5 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.5914e-02, 6.9837e-01, 5.6032e-03,
5.0239e-03, 1.2996e-02, 3.2303e-02,
        2.0515e-02, 2.6267e-02, 2.5567e-02, 4.6308e-03, 2.4632e-02, 4.1968e-03,
        3.2962e-02, 5.6085e-03, 1.7418e-03, 4.5918e-02, 9.8293e-03, 6.4497e-04,
        5.8355e-03, 3.5585e-03])

max probability is torch.return_types.max(
values=tensor(0.6984),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2986e-04, 1.8715e-05, 1.8709e-05, 1.8693e-05]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 6 :   Prediction is Correct?
```

```
No
first 20 classes probability: tensor([0.0243, 0.0010, 0.0328, 0.0174, 0.0645,
0.0817, 0.0587, 0.0018, 0.0119,
        0.0240, 0.0264, 0.0025, 0.0709, 0.4532, 0.0044, 0.0098, 0.0303, 0.0250,
        0.0207, 0.0358])

max probability is torch.return_types.max(
values=tensor(0.4532),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7326e-04, 3.2402e-06, 3.2397e-06, 3.2393e-06]),
indices=tensor([  0, 771, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 7 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.6053e-04, 9.9554e-01, 3.6821e-05,
1.6936e-05, 8.0506e-05, 2.1338e-04,
        1.7783e-04, 1.4325e-03, 6.3195e-04, 8.8396e-06, 9.8841e-05, 6.3170e-06,
        4.9837e-04, 5.7698e-06, 6.6221e-06, 9.5614e-04, 1.6290e-05, 5.5130e-07,
        7.3143e-06, 2.0092e-06])

max probability is torch.return_types.max(
values=tensor(0.9955),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1000e-08, 2.6843e-10, 2.6820e-10, 2.6754e-10]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 8 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0901, 0.0878, 0.0086, 0.0015, 0.0083,
0.0084, 0.0070, 0.3360, 0.2882,
        0.0062, 0.0201, 0.0114, 0.0369, 0.0073, 0.0006, 0.0210, 0.0306, 0.0017,
        0.0136, 0.0052])

max probability is torch.return_types.max(
values=tensor(0.3360),
indices=tensor(7))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([2.2620e-04, 9.8031e-06, 9.8010e-06, 9.7997e-06]),
indices=tensor([  0, 910, 316, 323]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 9 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.7626e-04, 4.3055e-05, 8.4676e-05,
2.3007e-04, 4.9532e-04, 4.2890e-04,
        2.2550e-04, 5.4901e-04, 3.3500e-04, 1.8162e-03, 7.1462e-04, 1.0198e-02,
        3.5827e-03, 3.2930e-03, 8.2587e-01, 2.1449e-03, 2.6450e-03, 3.2227e-02,
        1.1164e-01, 2.4348e-03])

max probability is torch.return_types.max(
values=tensor(0.8259),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4437e-05, 7.9472e-07, 7.9462e-07, 7.9455e-07]),
indices=tensor([  0, 255, 337, 263]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 10 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.3788e-04, 1.5439e-04, 1.3802e-04,
1.4334e-04, 6.5680e-04, 2.6127e-04,
        2.4483e-04, 8.4883e-03, 1.3558e-02, 2.7124e-03, 1.0070e-02, 7.4550e-05,
        7.2591e-04, 2.0135e-02, 1.1093e-06, 4.3988e-03, 1.2916e-01, 1.8533e-04,
        6.8072e-02, 7.3749e-01])

max probability is torch.return_types.max(
values=tensor(0.7375),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.1657e-03, 2.9045e-08, 2.9036e-08, 2.9025e-08]),
indices=tensor([  0, 914, 771, 337]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 11 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.2346e-03, 1.6883e-04, 1.5158e-03,
1.2444e-03, 3.9892e-03, 2.2231e-03,
```

```
        1.1043e-03, 3.9786e-02, 8.9671e-02, 1.0996e-01, 1.0351e-02, 3.0352e-04,
        2.5843e-03, 2.6388e-02, 1.8739e-04, 4.2230e-03, 2.3755e-02, 5.9222e-03,
        6.0028e-01, 7.2780e-02])

max probability is torch.return_types.max(
values=tensor(0.6003),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.9382e-04, 7.7116e-07, 7.7083e-07, 7.7053e-07]),
indices=tensor([  0, 914, 771,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 12 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.5267e-05, 4.3957e-05, 4.5273e-06,
3.1155e-06, 8.3089e-06, 2.3017e-05,
        1.2400e-05, 3.8047e-04, 6.3251e-05, 3.5326e-05, 1.3409e-03, 9.8917e-01,
        8.6055e-04, 1.3320e-04, 2.4565e-04, 6.0974e-05, 4.2611e-03, 1.7609e-04,
        2.6655e-03, 4.3501e-04])

max probability is torch.return_types.max(
values=tensor(0.9892),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8436e-06, 7.8753e-09, 7.8734e-09, 7.8729e-09]),
indices=tensor([  0, 691, 382, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 13 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.5568e-03, 9.3667e-01, 1.1401e-03,
2.6315e-04, 1.6269e-03, 1.2068e-03,
        1.1036e-03, 1.2031e-02, 7.0939e-03, 2.7100e-04, 3.3991e-03, 4.6066e-04,
        5.7933e-03, 1.9850e-03, 1.0702e-04, 5.2427e-03, 1.5093e-02, 1.5463e-04,
        2.8203e-03, 1.7489e-03])

max probability is torch.return_types.max(
values=tensor(0.9367),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([4.0305e-05, 1.9847e-07, 1.9843e-07, 1.9840e-07]),
indices=tensor([  0, 910, 323, 726]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 14 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0062, 0.0056, 0.0047, 0.0037, 0.0091,
0.0815, 0.0242, 0.0035, 0.0268,
        0.0148, 0.4427, 0.0110, 0.2316, 0.0420, 0.0010, 0.0512, 0.0217, 0.0015,
        0.0057, 0.0069])

max probability is torch.return_types.max(
values=tensor(0.4427),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.9264e-05, 4.8816e-06, 4.8803e-06, 4.8755e-06]),
indices=tensor([  0, 319, 771,  44]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 15 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0030, 0.0009, 0.0139, 0.0139, 0.0206,
0.0079, 0.0109, 0.0120, 0.0296,
        0.0293, 0.0149, 0.0011, 0.0322, 0.5110, 0.0006, 0.0051, 0.1344, 0.0052,
        0.0234, 0.1145])

max probability is torch.return_types.max(
values=tensor(0.5110),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6337e-03, 1.3685e-05, 1.3678e-05, 1.3678e-05]),
indices=tensor([  0, 914, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 16 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3359e-06, 4.7627e-06, 1.5361e-07,
1.0188e-07, 6.0542e-07, 1.3963e-06,
        6.1379e-07, 1.6446e-07, 2.2307e-06, 1.2345e-06, 9.9532e-01, 7.2756e-06,
        2.0069e-04, 3.7004e-04, 4.5971e-09, 4.0028e-03, 6.1806e-05, 5.2694e-07,
        4.5888e-06, 1.9961e-05])
```

```
max probability is torch.return_types.max(
values=tensor(0.9953),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6472e-09, 2.5340e-12, 2.5301e-12, 2.5221e-12]),
indices=tensor([  0, 319, 771, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 17 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3615e-13, 7.1319e-13, 1.0585e-11,
5.8284e-11, 4.3291e-10, 9.4721e-13,
        8.6746e-13, 1.4047e-10, 9.0519e-11, 4.1354e-11, 4.1040e-09, 2.1591e-15,
        4.6858e-12, 3.0044e-07, 7.3369e-17, 6.7846e-12, 3.4885e-06, 1.4344e-10,
        5.1038e-05, 9.9993e-01])

max probability is torch.return_types.max(
values=tensor(0.9999),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6313e-05, 5.1828e-21, 5.1807e-21, 5.1679e-21]),
indices=tensor([  0, 337, 655, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 18 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0557, 0.0019, 0.0158, 0.0105, 0.0205,
0.0312, 0.0100, 0.0385, 0.1089,
        0.2936, 0.0818, 0.0009, 0.0413, 0.0631, 0.0006, 0.0259, 0.0232, 0.0043,
        0.0794, 0.0826])

max probability is torch.return_types.max(
values=tensor(0.2936),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.1619e-04, 1.0215e-05, 1.0206e-05, 1.0205e-05]),
indices=tensor([  0, 771, 316, 896]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 19 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.1109e-04, 7.8613e-06, 6.5504e-06,
1.2550e-05, 4.1361e-05, 3.2328e-05,
        2.9691e-05, 5.3109e-05, 2.5674e-05, 5.4906e-05, 5.9536e-05, 2.3919e-03,
        3.8063e-04, 2.3095e-04, 9.5809e-01, 3.1149e-05, 1.6572e-04, 3.1749e-02,
        6.3083e-03, 1.9492e-04])

max probability is torch.return_types.max(
values=tensor(0.9581),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.3332e-07, 2.8799e-08, 2.8796e-08, 2.8783e-08]),
indices=tensor([  0, 748, 337, 255]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 20 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.1302e-02, 7.7023e-04, 6.6204e-03,
5.3741e-04, 2.0312e-03, 5.4771e-04,
        3.8346e-04, 4.5750e-01, 2.8610e-01, 1.5205e-01, 1.2286e-03, 4.4881e-05,
        1.5483e-02, 6.0444e-04, 5.1475e-05, 1.9516e-03, 4.6201e-03, 1.1220e-04,
        7.0193e-03, 1.0161e-03])

max probability is torch.return_types.max(
values=tensor(0.4575),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7853e-06, 2.5414e-08, 2.5407e-08, 2.5404e-08]),
indices=tensor([  0, 316,  85, 896]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 21 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.1247, 0.3266, 0.0194, 0.0207, 0.0338,
0.2382, 0.0830, 0.0152, 0.0367,
        0.0079, 0.0169, 0.0025, 0.0142, 0.0160, 0.0006, 0.0069, 0.0042, 0.0013,
        0.0042, 0.0103])

max probability is torch.return_types.max(
values=tensor(0.3266),
```

```
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.0735e-04, 1.7194e-05, 1.7182e-05, 1.7179e-05]),
indices=tensor([  0, 319, 316,  44]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 22 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0116, 0.0021, 0.1136, 0.0444, 0.0665,
0.0948, 0.0379, 0.0118, 0.0554,
        0.0963, 0.0553, 0.0003, 0.0479, 0.2145, 0.0004, 0.0138, 0.0217, 0.0031,
        0.0216, 0.0775])

max probability is torch.return_types.max(
values=tensor(0.2145),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.0271e-04, 9.3128e-06, 9.3059e-06, 9.3028e-06]),
indices=tensor([  0, 771, 914, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 23 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.0766e-08, 1.2069e-08, 2.2857e-07,
3.2773e-07, 7.2235e-06, 1.6819e-07,
        3.7194e-08, 1.2116e-04, 3.4623e-05, 3.7306e-05, 3.5273e-06, 7.1482e-08,
        4.1589e-07, 4.3489e-05, 1.1145e-06, 1.4298e-06, 6.1229e-06, 4.5098e-05,
        9.9823e-01, 1.4631e-03])

max probability is torch.return_types.max(
values=tensor(0.9982),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9691e-07, 3.5119e-13, 3.5110e-13, 3.5109e-13]),
indices=tensor([  0, 337, 197, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 24 :   Prediction is Correct?
Yes
```

```
first 20 classes probability: tensor([1.4193e-09, 1.0000e+00, 3.1937e-10,
1.9756e-11, 1.2081e-09, 3.9526e-09,
        1.5443e-09, 1.5458e-07, 3.0714e-08, 5.9696e-13, 3.5042e-09, 7.9418e-11,
        2.4770e-07, 5.7999e-11, 1.2254e-12, 5.0606e-09, 8.2108e-09, 3.6482e-14,
        5.3656e-11, 1.6841e-11])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5969e-15, 1.0995e-20, 1.0991e-20, 1.0967e-20]),
indices=tensor([  0, 316, 319, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 25 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0302, 0.0047, 0.0119, 0.0025, 0.0129,
0.0046, 0.0057, 0.0234, 0.0463,
        0.0277, 0.0541, 0.0095, 0.0574, 0.1396, 0.0212, 0.0667, 0.0675, 0.1319,
        0.1979, 0.0442])

max probability is torch.return_types.max(
values=tensor(0.1979),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.1531e-04, 4.1080e-05, 4.1064e-05, 4.1060e-05]),
indices=tensor([  0, 323, 771, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 26 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0074, 0.0024, 0.0071, 0.0053, 0.0125,
0.0105, 0.0072, 0.0647, 0.0688,
        0.0302, 0.0179, 0.0213, 0.0310, 0.0802, 0.0072, 0.0095, 0.0986, 0.0458,
        0.2307, 0.1999])

max probability is torch.return_types.max(
values=tensor(0.2307),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([5.7996e-03, 3.7486e-05, 3.7464e-05, 3.7462e-05]),
indices=tensor([  0, 914, 158, 127]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 27 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.9914e-05, 2.3468e-05, 3.6050e-04,
1.5491e-05, 2.7943e-04, 1.8368e-06,
        1.5883e-06, 9.7302e-04, 3.6325e-04, 1.9216e-05, 1.8793e-04, 1.1174e-05,
        1.7389e-04, 3.1075e-03, 2.6256e-06, 1.1575e-04, 1.6204e-01, 2.2943e-03,
        1.6651e-01, 6.6148e-01])

max probability is torch.return_types.max(
values=tensor(0.6615),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9745e-03, 9.1341e-10, 9.1337e-10, 9.1325e-10]),
indices=tensor([  0, 230, 691, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 28 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3709e-02, 3.9349e-03, 1.4193e-03,
1.5304e-04, 7.4823e-04, 2.4771e-03,
        2.4338e-03, 2.0258e-01, 7.3720e-01, 3.5785e-03, 1.3918e-03, 3.6943e-05,
        2.7821e-02, 7.5711e-04, 2.5334e-05, 1.3115e-03, 1.6705e-04, 3.5828e-05,
        9.3725e-05, 8.0242e-05])

max probability is torch.return_types.max(
values=tensor(0.7372),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.7404e-07, 5.0032e-08, 5.0024e-08, 4.9993e-08]),
indices=tensor([  0, 319, 836, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 29 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.2699e-05, 3.8304e-06, 1.1752e-06,
4.9869e-07, 1.7585e-06, 2.7167e-06,
        3.0667e-06, 5.3689e-05, 5.9643e-06, 5.6810e-06, 5.0439e-05, 9.8816e-01,
```

```
        4.8307e-04, 5.8272e-05, 2.8095e-03, 1.6206e-06, 7.2114e-03, 2.7936e-04,
        7.0149e-04, 1.0984e-04])

max probability is torch.return_types.max(
values=tensor(0.9882),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9753e-07, 3.3692e-10, 3.3647e-10, 3.3639e-10]),
indices=tensor([  0, 691,  82, 230]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 30 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.0488e-03, 4.4426e-03, 1.0932e-03,
3.3094e-04, 1.6115e-03, 1.1898e-03,
        2.9054e-03, 5.3766e-01, 4.2235e-01, 2.0960e-03, 1.7034e-03, 1.9541e-04,
        1.8437e-02, 5.6623e-04, 2.2791e-04, 1.0092e-03, 1.5977e-04, 1.8742e-04,
        4.3989e-04, 1.2209e-04])

max probability is torch.return_types.max(
values=tensor(0.5377),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6337e-06, 2.3956e-07, 2.3953e-07, 2.3950e-07]),
indices=tensor([  0, 836, 316, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 31 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7968e-02, 7.8192e-04, 1.0005e-01,
6.6594e-02, 4.2232e-02, 3.5462e-02,
        1.1753e-01, 5.8984e-03, 1.5046e-02, 4.9569e-01, 6.2414e-03, 1.9756e-04,
        2.6155e-02, 5.6776e-02, 9.2835e-04, 6.0102e-04, 3.7643e-03, 2.5622e-03,
        1.0695e-03, 2.5506e-03])

max probability is torch.return_types.max(
values=tensor(0.4957),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5052e-05, 1.9800e-06, 1.9790e-06, 1.9788e-06]),
```

```
indices=tensor([  0, 771, 323, 572]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 32 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.1717e-04, 8.1872e-06, 1.5985e-05,
3.5657e-05, 1.1387e-04, 4.2402e-05,
        5.1853e-05, 7.2115e-04, 3.1006e-04, 8.6629e-04, 3.4216e-05, 1.1715e-03,
        8.0451e-04, 3.3059e-04, 9.4303e-01, 4.2024e-05, 2.2694e-04, 2.3983e-02,
        2.6989e-02, 3.8192e-04])

max probability is torch.return_types.max(
values=tensor(0.9430),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.4622e-07, 2.7727e-08, 2.7722e-08, 2.7718e-08]),
indices=tensor([  0, 748, 323, 255]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 33 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.9962e-05, 1.3694e-06, 2.4882e-04,
1.1300e-04, 5.4908e-04, 2.5202e-05,
        1.6349e-05, 7.9866e-05, 3.5247e-04, 8.3644e-04, 4.1009e-04, 2.3154e-06,
        3.7084e-04, 8.5938e-01, 2.7199e-06, 4.8870e-04, 6.0921e-02, 9.3201e-04,
        1.3002e-02, 6.2126e-02])

max probability is torch.return_types.max(
values=tensor(0.8594),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.8459e-05, 2.0143e-09, 2.0132e-09, 2.0131e-09]),
indices=tensor([  0, 914, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 34 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.0947e-06, 3.8163e-04, 2.8794e-05,
3.4008e-06, 1.6318e-05, 5.2959e-06,
        8.5149e-06, 3.3243e-04, 5.2130e-04, 1.6831e-06, 1.0084e-02, 3.2545e-05,
        9.8271e-01, 1.8479e-04, 2.4052e-05, 5.6139e-03, 3.2717e-05, 2.2878e-06,
```

```
        4.9210e-06, 5.9578e-06])

max probability is torch.return_types.max(
values=tensor(0.9827),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.0610e-09, 3.1013e-10, 3.0966e-10, 3.0949e-10]),
indices=tensor([  0, 319, 836, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 35 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.6560e-03, 5.2241e-03, 3.1331e-04,
1.2357e-04, 1.1716e-03, 4.1915e-03,
        1.5314e-03, 2.6815e-03, 1.5089e-02, 4.6441e-03, 8.5462e-02, 2.0704e-03,
        3.1977e-02, 1.3628e-02, 9.5927e-04, 7.5739e-01, 5.8420e-02, 1.0726e-03,
        5.5341e-03, 3.8580e-03])

max probability is torch.return_types.max(
values=tensor(0.7574),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3908e-05, 1.0488e-06, 1.0471e-06, 1.0469e-06]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 36 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0265, 0.0146, 0.0366, 0.0179, 0.0434,
0.2227, 0.2694, 0.0078, 0.0536,
        0.1192, 0.0501, 0.0012, 0.0596, 0.0377, 0.0007, 0.0167, 0.0122, 0.0012,
        0.0024, 0.0026])

max probability is torch.return_types.max(
values=tensor(0.2694),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.2670e-05, 3.9804e-06, 3.9786e-06, 3.9781e-06]),
indices=tensor([  0, 319, 771, 896]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 37 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0176, 0.0018, 0.0064, 0.0037, 0.0164,
0.0682, 0.0229, 0.0041, 0.0324,
        0.0253, 0.0982, 0.0086, 0.0398, 0.1955, 0.0217, 0.0283, 0.0313, 0.2009,
        0.0991, 0.0689])

max probability is torch.return_types.max(
values=tensor(0.2009),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.0743e-04, 9.1068e-06, 9.1039e-06, 9.0995e-06]),
indices=tensor([  0, 771, 914, 323]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 38 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.4819e-04, 3.5919e-04, 2.2431e-04,
7.2407e-05, 6.1393e-04, 2.1274e-04,
        1.6357e-04, 1.2892e-02, 1.5149e-02, 2.1601e-03, 4.6806e-03, 7.8578e-03,
        4.6573e-03, 1.1450e-02, 1.7129e-03, 1.1474e-02, 2.5581e-01, 1.4917e-02,
        5.4908e-01, 1.0362e-01])

max probability is torch.return_types.max(
values=tensor(0.5491),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4709e-03, 9.1063e-07, 9.1059e-07, 9.1033e-07]),
indices=tensor([  0, 197, 230, 914]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 39 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.9908e-02, 3.8113e-03, 1.9625e-03,
3.6395e-04, 2.0440e-03, 3.3423e-02,
        1.7442e-02, 3.5190e-02, 7.7939e-01, 1.2067e-02, 3.9224e-03, 2.2415e-04,
        7.0619e-02, 2.5301e-03, 1.2036e-04, 5.1932e-03, 7.1812e-04, 1.6714e-04,
        2.1747e-04, 2.2116e-04])

max probability is torch.return_types.max(
```

```
values=tensor(0.7794),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9163e-06, 4.9065e-07, 4.9036e-07, 4.9014e-07]),
indices=tensor([  0, 319, 836, 794]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 40 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7040e-03, 6.2026e-04, 7.2370e-03,
4.4394e-03, 1.0496e-02, 7.0038e-03,
        6.8077e-03, 3.1012e-03, 2.1001e-02, 1.2349e-02, 9.7049e-02, 3.6204e-04,
        1.1831e-01, 6.3257e-01, 8.3408e-05, 2.1110e-02, 3.9530e-02, 7.0574e-04,
        2.8321e-03, 1.2035e-02])

max probability is torch.return_types.max(
values=tensor(0.6326),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.7593e-05, 6.3130e-07, 6.3109e-07, 6.3095e-07]),
indices=tensor([  0, 771, 910, 323]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 41 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0564, 0.0121, 0.0134, 0.0058, 0.0284,
0.2654, 0.1472, 0.0082, 0.1065,
        0.0550, 0.0577, 0.0014, 0.0450, 0.0895, 0.0027, 0.0513, 0.0113, 0.0113,
        0.0075, 0.0118])

max probability is torch.return_types.max(
values=tensor(0.2654),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4174e-04, 1.2529e-05, 1.2523e-05, 1.2518e-05]),
indices=tensor([  0, 319, 771, 896]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 42 :   Prediction is Correct?
```

```
Yes
first 20 classes probability: tensor([2.8234e-05, 4.5157e-06, 2.1548e-07,
2.3124e-07, 1.0869e-06, 1.1536e-05,
        1.0077e-06, 5.6571e-07, 3.7222e-06, 8.2204e-06, 9.9413e-01, 4.2965e-04,
        1.0027e-03, 8.7827e-05, 2.9567e-07, 3.9971e-03, 1.5475e-04, 3.3312e-06,
        8.4430e-05, 5.1402e-05])

max probability is torch.return_types.max(
values=tensor(0.9941),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0760e-08, 1.9323e-11, 1.9299e-11, 1.9258e-11]),
indices=tensor([  0, 319, 771, 476]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 43 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.4326e-06, 2.6087e-05, 2.2849e-05,
1.2785e-06, 3.0038e-05, 1.1186e-06,
        3.5905e-06, 2.8271e-03, 1.2499e-03, 1.8948e-05, 1.2707e-04, 9.1004e-04,
        1.9271e-03, 2.5488e-03, 4.5365e-06, 3.3478e-04, 9.4729e-01, 1.2556e-04,
        7.2727e-03, 3.5135e-02])

max probability is torch.return_types.max(
values=tensor(0.9473),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3194e-04, 6.8828e-10, 6.8827e-10, 6.8811e-10]),
indices=tensor([  0, 230, 691, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 44 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.2310e-02, 6.0704e-05, 1.3625e-03,
8.8527e-04, 5.1292e-03, 1.0928e-03,
        7.2276e-04, 2.3582e-04, 3.0049e-04, 2.4559e-03, 3.2750e-04, 1.0485e-03,
        4.7339e-04, 6.7947e-04, 2.7725e-01, 1.4385e-04, 3.5516e-04, 5.9871e-01,
        7.3898e-02, 2.3838e-03])

max probability is torch.return_types.max(
values=tensor(0.5987),
indices=tensor(17))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.9978e-06, 1.6906e-07, 1.6902e-07, 1.6898e-07]),
indices=tensor([  0, 337, 748, 255]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 45 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9790e-01, 7.3598e-06, 2.4555e-04,
6.4947e-05, 2.4881e-04, 1.1931e-04,
        9.5066e-05, 1.0380e-05, 2.8703e-05, 1.6621e-04, 1.0377e-05, 1.0959e-05,
        2.0575e-05, 9.0357e-05, 2.7697e-06, 8.3466e-07, 1.0797e-05, 3.5066e-04,
        1.2661e-04, 4.8507e-04])

max probability is torch.return_types.max(
values=tensor(0.9979),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7838e-07, 1.8750e-10, 1.8745e-10, 1.8745e-10]),
indices=tensor([  0, 323, 914, 454]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 46 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.6148e-04, 1.9220e-03, 3.5777e-04,
1.2384e-04, 6.1688e-04, 5.4930e-03,
        1.8112e-03, 2.9814e-04, 6.6530e-03, 5.0464e-04, 8.3173e-01, 1.9258e-04,
        7.7135e-02, 1.2541e-02, 1.0938e-05, 5.8303e-02, 1.0737e-03, 5.3807e-05,
        1.7085e-04, 4.1132e-04])

max probability is torch.return_types.max(
values=tensor(0.8317),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1142e-06, 3.7879e-08, 3.7792e-08, 3.7763e-08]),
indices=tensor([  0, 319, 771, 869]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 47 :   Prediction is Correct?
Yes
```

```
first 20 classes probability: tensor([9.2457e-03, 4.6536e-03, 3.1755e-03,
2.3374e-03, 1.3357e-02, 3.8861e-01,
        1.8646e-01, 1.4107e-02, 2.7752e-01, 3.5704e-02, 1.0468e-02, 1.0391e-03,
        2.4798e-02, 9.4468e-03, 2.9254e-04, 8.2861e-03, 3.8591e-03, 7.2144e-04,
        2.5172e-03, 1.7743e-03])

max probability is torch.return_types.max(
values=tensor(0.3886),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9201e-05, 1.7107e-06, 1.7096e-06, 1.7093e-06]),
indices=tensor([  0, 319, 896, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 48 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.7323e-04, 9.5982e-01, 2.5756e-04,
1.0130e-04, 5.4322e-04, 1.8947e-03,
        1.0653e-03, 1.2902e-02, 1.0279e-02, 8.4139e-05, 1.4407e-03, 2.0738e-04,
        7.3226e-03, 3.9250e-04, 3.0720e-05, 1.0798e-03, 1.3197e-03, 1.2354e-05,
        3.7654e-04, 2.4525e-04])

max probability is torch.return_types.max(
values=tensor(0.9598),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.3898e-06, 5.5805e-08, 5.5782e-08, 5.5741e-08]),
indices=tensor([  0, 316, 319,  44]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 49 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.4307e-03, 4.6561e-03, 6.2271e-03,
2.7323e-02, 1.0482e-01, 5.7518e-01,
        2.4500e-01, 5.9668e-04, 3.4852e-03, 4.6787e-03, 1.7559e-03, 1.6615e-04,
        2.4901e-03, 3.9852e-03, 1.6948e-03, 4.3198e-03, 3.2505e-04, 4.5689e-04,
        3.8594e-03, 5.7569e-04])

max probability is torch.return_types.max(
values=tensor(0.5752),
indices=tensor(5))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4918e-05, 2.0831e-06, 2.0819e-06, 2.0805e-06]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 50 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.8252e-04, 7.6545e-04, 1.9434e-01,
2.2276e-01, 5.1558e-01, 5.0674e-02,
        1.4985e-02, 1.7672e-05, 1.2434e-05, 2.8958e-05, 3.0741e-05, 7.5359e-07,
        4.0472e-05, 5.9916e-05, 3.9142e-05, 4.7894e-06, 4.0403e-06, 3.8441e-05,
        2.9005e-04, 4.1704e-05])

max probability is torch.return_types.max(
values=tensor(0.5156),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9093e-06, 1.0402e-07, 1.0398e-07, 1.0392e-07]),
indices=tensor([  0, 319, 316, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 51 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7553e-03, 7.1994e-03, 3.3767e-04,
8.2779e-05, 5.5844e-04, 2.6383e-03,
        2.2360e-03, 2.9764e-02, 1.1090e-01, 2.7906e-04, 3.7297e-02, 3.8746e-03,
        7.9109e-01, 4.1798e-03, 1.0307e-04, 4.9296e-03, 1.7619e-03, 1.0396e-04,
        2.2891e-04, 3.9017e-04])

max probability is torch.return_types.max(
values=tensor(0.7911),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.4191e-06, 3.0882e-07, 3.0875e-07, 3.0865e-07]),
indices=tensor([  0, 836, 319,  44]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 52 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.1849e-03, 4.4961e-03, 7.3825e-04,
```

```
3.3976e-04, 3.1267e-03, 6.1516e-03,
        2.3639e-03, 1.9212e-01, 4.1990e-01, 2.4554e-02, 5.2851e-02, 2.3915e-04,
        2.1177e-02, 6.6737e-03, 1.0797e-04, 1.7208e-01, 6.8825e-03, 9.5865e-04,
        5.8263e-02, 2.3991e-02])

max probability is torch.return_types.max(
values=tensor(0.4199),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2488e-04, 7.2755e-07, 7.2717e-07, 7.2662e-07]),
indices=tensor([  0, 319, 316, 896]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 53 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.5065e-07, 7.7883e-07, 3.5674e-09,
6.3853e-09, 4.0707e-08, 1.3981e-06,
        1.0617e-07, 1.5225e-08, 4.5609e-07, 4.8161e-07, 9.9941e-01, 4.4267e-07,
        1.4058e-05, 1.7648e-05, 8.2589e-11, 5.4486e-04, 3.9808e-06, 1.0901e-08,
        1.0601e-06, 3.0344e-06])

max probability is torch.return_types.max(
values=tensor(0.9994),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1278e-10, 2.5473e-14, 2.5412e-14, 2.5340e-14]),
indices=tensor([  0, 319, 771, 869]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 54 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.4533e-03, 3.4603e-04, 1.5495e-03,
7.4242e-04, 4.5092e-03, 5.1942e-04,
        2.7862e-04, 2.3090e-02, 7.7958e-03, 4.2229e-03, 2.8564e-03, 7.3748e-03,
        1.0352e-02, 1.2386e-02, 1.4024e-02, 2.6353e-03, 1.9970e-02, 2.4188e-02,
        8.2862e-01, 3.1825e-02])

max probability is torch.return_types.max(
values=tensor(0.8286),
indices=tensor(18))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([2.8621e-04, 1.0297e-06, 1.0293e-06, 1.0293e-06]),
indices=tensor([  0, 197, 337, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 55 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.0897e-04, 3.5080e-04, 2.5370e-02,
3.8975e-02, 1.6942e-01, 2.5094e-01,
        4.5053e-01, 1.9152e-04, 4.5878e-04, 3.3887e-03, 1.7663e-04, 6.3079e-05,
        5.2005e-04, 5.7034e-03, 3.9626e-02, 3.6285e-04, 1.7606e-04, 6.4402e-03,
        5.9452e-03, 5.4254e-04])

max probability is torch.return_types.max(
values=tensor(0.4505),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5958e-06, 2.2372e-07, 2.2363e-07, 2.2363e-07]),
indices=tensor([  0, 910, 263, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 56 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.7373e-03, 1.8020e-04, 6.7045e-04,
2.5787e-04, 2.1414e-03, 4.3472e-04,
        2.5374e-04, 2.8348e-02, 3.8854e-02, 1.2143e-02, 6.2966e-03, 2.6481e-03,
        1.0122e-02, 2.5423e-02, 1.6686e-03, 1.4274e-02, 6.0973e-02, 1.5196e-02,
        7.4218e-01, 3.3970e-02])

max probability is torch.return_types.max(
values=tensor(0.7422),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7654e-04, 1.0042e-06, 1.0040e-06, 1.0040e-06]),
indices=tensor([  0, 914, 197, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 57 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.6100e-04, 9.6115e-01, 6.7722e-04,
2.1351e-04, 2.0191e-03, 6.9807e-04,
```

```
        8.2795e-04, 9.8455e-03, 4.0186e-03, 6.5627e-05, 2.4255e-03, 2.6134e-04,
        2.9332e-03, 6.2044e-04, 2.7398e-04, 1.0784e-03, 2.0720e-03, 7.6180e-04,
        6.9619e-03, 2.1732e-03])

max probability is torch.return_types.max(
values=tensor(0.9611),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5529e-05, 4.4212e-08, 4.4204e-08, 4.4200e-08]),
indices=tensor([  0, 323, 888, 910]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 58 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.0815e-05, 1.1852e-06, 1.0565e-07,
2.7796e-08, 8.7451e-08, 2.3946e-07,
        1.3334e-07, 9.5875e-06, 9.8985e-07, 6.2435e-07, 5.7571e-04, 9.9873e-01,
        1.4131e-04, 2.8367e-06, 4.4157e-06, 2.6212e-06, 4.3355e-04, 4.6288e-06,
        6.4607e-05, 8.8801e-06])

max probability is torch.return_types.max(
values=tensor(0.9987),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.2633e-09, 5.2982e-12, 5.2958e-12, 5.2941e-12]),
indices=tensor([  0, 914, 323, 691]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 59 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.4222e-05, 1.1925e-05, 3.4016e-05,
7.2778e-05, 2.1139e-04, 4.6416e-05,
        6.0996e-05, 1.2827e-04, 1.5701e-04, 4.8044e-04, 6.9596e-04, 1.2319e-06,
        2.3368e-05, 2.3770e-03, 2.6142e-07, 1.6862e-05, 2.7009e-03, 2.9783e-04,
        6.9355e-03, 9.8367e-01])

max probability is torch.return_types.max(
values=tensor(0.9837),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([2.0167e-03, 2.0455e-09, 2.0444e-09, 2.0440e-09]),
indices=tensor([  0, 655, 337, 914]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 60 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.4195e-05, 2.0810e-02, 1.9170e-05,
8.4859e-06, 3.2350e-05, 2.0761e-04,
        5.5214e-05, 5.1314e-06, 2.0514e-05, 3.7285e-06, 9.6044e-01, 1.1605e-04,
        9.3695e-03, 1.7210e-04, 3.6099e-07, 8.5801e-03, 1.0075e-04, 6.0996e-07,
        2.0499e-05, 8.8043e-06])

max probability is torch.return_types.max(
values=tensor(0.9604),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.1314e-09, 5.1220e-11, 5.1050e-11, 5.0947e-11]),
indices=tensor([  0, 319, 771, 316]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 61 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.4463e-05, 5.4809e-05, 3.7132e-01,
4.6072e-01, 1.5167e-01, 1.3300e-02,
        2.8590e-03, 3.7751e-07, 1.7616e-07, 4.0491e-06, 4.3010e-07, 5.0766e-09,
        1.2573e-06, 3.7079e-06, 1.0875e-06, 1.3676e-08, 6.6717e-08, 1.2471e-06,
        2.5809e-06, 2.2307e-06])

max probability is torch.return_types.max(
values=tensor(0.4607),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8333e-08, 2.3425e-10, 2.3420e-10, 2.3419e-10]),
indices=tensor([  0, 319, 771, 316]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 62 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0037, 0.0342, 0.0192, 0.0085, 0.0230,
0.0133, 0.0203, 0.2914, 0.2723,
        0.0488, 0.0387, 0.0010, 0.0491, 0.0199, 0.0011, 0.0236, 0.0207, 0.0027,
```

```
        0.0336, 0.0193])

max probability is torch.return_types.max(
values=tensor(0.2914),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4851e-03, 5.6961e-05, 5.6960e-05, 5.6952e-05]),
indices=tensor([  0, 316, 896,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 63 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.0766e-03, 3.7510e-01, 9.8259e-04,
2.7680e-04, 1.8120e-03, 8.9572e-04,
        1.3745e-03, 3.8448e-01, 1.0786e-01, 2.2435e-04, 6.6965e-03, 3.8327e-03,
        1.0526e-01, 8.8777e-04, 2.1822e-04, 2.5212e-03, 4.0085e-03, 6.0767e-05,
        1.5215e-03, 5.9153e-04])

max probability is torch.return_types.max(
values=tensor(0.3845),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2612e-05, 3.0623e-07, 3.0609e-07, 3.0604e-07]),
indices=tensor([  0, 316, 319,  44]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 64 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0106, 0.0028, 0.0048, 0.0052, 0.0174,
0.0971, 0.0311, 0.0416, 0.3507,
        0.2020, 0.0497, 0.0005, 0.0376, 0.0588, 0.0004, 0.0493, 0.0134, 0.0013,
        0.0113, 0.0092])

max probability is torch.return_types.max(
values=tensor(0.3507),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0347e-04, 5.0426e-06, 5.0396e-06, 5.0384e-06]),
indices=tensor([  0, 319, 896, 771]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 65 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.2057e-04, 4.7840e-05, 3.8198e-01,
3.8660e-01, 2.1128e-01, 1.3732e-02,
        5.7311e-03, 1.0296e-05, 9.1605e-06, 1.3311e-04, 6.5725e-06, 1.3976e-08,
        2.0145e-05, 1.4140e-04, 2.3834e-06, 7.2879e-07, 1.2346e-06, 8.9036e-06,
        2.2870e-05, 4.7052e-05])

max probability is torch.return_types.max(
values=tensor(0.3866),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.9518e-07, 4.2665e-09, 4.2659e-09, 4.2635e-09]),
indices=tensor([  0, 319, 771, 910]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 66 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.5166e-03, 3.3876e-05, 1.7619e-04,
6.7712e-05, 6.5573e-04, 1.3906e-04,
        1.5924e-04, 3.4895e-04, 5.3915e-04, 7.4060e-04, 3.1777e-04, 8.3168e-04,
        9.2290e-04, 3.2035e-03, 1.1425e-01, 2.4224e-04, 6.8013e-04, 8.1919e-01,
        4.7174e-02, 2.7484e-03])

max probability is torch.return_types.max(
values=tensor(0.8192),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.7272e-06, 5.1421e-08, 5.1399e-08, 5.1385e-08]),
indices=tensor([  0, 748, 428, 337]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 67 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.0666e-04, 4.5262e-04, 7.1984e-02,
2.1425e-01, 4.4571e-01, 2.0798e-01,
        5.7896e-02, 1.5752e-05, 2.3362e-05, 8.5434e-05, 4.6221e-05, 1.8125e-06,
        4.2911e-05, 1.8516e-04, 1.0103e-04, 1.6888e-05, 4.4819e-06, 6.5266e-05,
        5.1252e-04, 7.7422e-05])
```

```
max probability is torch.return_types.max(
values=tensor(0.4457),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8272e-06, 1.4440e-07, 1.4432e-07, 1.4422e-07]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 68 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.4693e-01, 6.7665e-03, 6.7262e-03,
3.2031e-04, 8.6349e-04, 2.1353e-03,
        2.2310e-03, 1.4090e-01, 4.7995e-01, 2.7202e-03, 6.7041e-05, 1.0229e-05,
        1.0164e-02, 5.4641e-05, 2.1217e-05, 7.1112e-05, 2.6139e-05, 1.4841e-05,
        1.9084e-05, 1.1422e-05])

max probability is torch.return_types.max(
values=tensor(0.4799),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.0019e-08, 5.2399e-09, 5.2367e-09, 5.2357e-09]),
indices=tensor([  0, 319, 836, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 69 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0518e-04, 4.9129e-05, 1.5084e-04,
7.6036e-05, 3.8416e-04, 5.5192e-05,
        3.2700e-05, 1.8974e-03, 1.8761e-03, 1.0557e-03, 1.4730e-03, 3.1188e-04,
        5.5492e-04, 1.7438e-02, 3.2904e-05, 1.8675e-03, 2.5519e-01, 1.7323e-03,
        3.5755e-01, 3.5448e-01])

max probability is torch.return_types.max(
values=tensor(0.3575),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6142e-03, 8.0921e-08, 8.0915e-08, 8.0900e-08]),
indices=tensor([  0, 655, 337, 914]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 70 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.5222e-03, 6.3608e-06, 1.3427e-03,
7.6178e-04, 2.5340e-03, 1.6789e-04,
        2.3280e-04, 9.4283e-04, 2.9037e-03, 2.8812e-02, 3.6271e-03, 2.9448e-05,
        2.8196e-03, 7.8919e-01, 6.4708e-05, 1.7541e-03, 5.8419e-02, 9.3124e-03,
        3.1110e-02, 6.3391e-02])

max probability is torch.return_types.max(
values=tensor(0.7892),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.1323e-05, 2.1881e-08, 2.1878e-08, 2.1873e-08]),
indices=tensor([  0, 323, 771, 914]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 71 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.6070e-03, 7.5457e-01, 3.7271e-04,
1.9262e-04, 3.4849e-04, 2.7497e-03,
        2.3990e-03, 1.0669e-02, 2.9674e-03, 8.2755e-05, 3.8924e-03, 1.9815e-01,
        1.2890e-02, 1.1446e-04, 1.7960e-04, 2.3756e-04, 1.2684e-03, 2.5537e-05,
        1.9666e-04, 6.3635e-05])

max probability is torch.return_types.max(
values=tensor(0.7546),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.5644e-07, 2.0249e-08, 2.0245e-08, 2.0240e-08]),
indices=tensor([  0, 323, 382, 910]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 72 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.0662e-04, 3.8216e-04, 4.6637e-01,
2.8668e-01, 2.2994e-01, 1.0350e-02,
        5.5714e-03, 1.8025e-05, 1.1190e-05, 9.4774e-05, 3.5003e-05, 3.0284e-07,
        9.8366e-05, 8.1730e-05, 1.6803e-05, 3.3611e-06, 6.0085e-06, 2.3652e-05,
        5.0979e-05, 2.8514e-05])

max probability is torch.return_types.max(
```

```
values=tensor(0.4664),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0619e-06, 3.1943e-08, 3.1938e-08, 3.1932e-08]),
indices=tensor([  0, 319, 910, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 73 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.8866e-04, 1.3209e-04, 4.1517e-01,
3.1049e-01, 2.3776e-01, 2.5150e-02,
        1.0118e-02, 2.7325e-05, 3.1405e-05, 3.3423e-04, 1.8220e-05, 1.0046e-07,
        5.1376e-05, 1.2279e-04, 9.4423e-06, 2.9721e-06, 3.9393e-06, 2.3220e-05,
        6.7435e-05, 6.4818e-05])

max probability is torch.return_types.max(
values=tensor(0.4152),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4563e-06, 2.8890e-08, 2.8877e-08, 2.8873e-08]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 74 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.5229e-06, 1.5436e-06, 5.0046e-08,
1.7307e-07, 4.6279e-07, 4.3603e-07,
        3.8808e-07, 2.9106e-05, 1.9449e-06, 3.0886e-06, 2.4158e-05, 1.1630e-01,
        2.9720e-04, 1.2473e-05, 8.7529e-01, 6.3731e-07, 6.8649e-05, 4.7546e-03,
        3.1680e-03, 3.8757e-05])

max probability is torch.return_types.max(
values=tensor(0.8753),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2594e-08, 5.0833e-11, 5.0825e-11, 5.0790e-11]),
indices=tensor([  0, 691, 230, 428]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 75 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.5963e-01, 8.8165e-04, 7.6216e-02,
1.6615e-02, 3.4400e-02, 9.8996e-02,
        4.5757e-02, 1.0887e-02, 6.4139e-02, 4.5512e-01, 3.4610e-03, 5.9975e-05,
        1.0762e-02, 8.1625e-03, 8.2918e-04, 3.2748e-03, 7.5608e-04, 3.1411e-03,
        4.1806e-03, 2.0517e-03])

max probability is torch.return_types.max(
values=tensor(0.4551),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.9863e-06, 7.1220e-07, 7.1206e-07, 7.1181e-07]),
indices=tensor([  0, 771, 896, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 76 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.2454e-04, 4.9223e-05, 3.3353e-05,
4.1478e-05, 1.4168e-04, 2.8059e-05,
        2.0893e-05, 1.5955e-04, 1.8336e-04, 1.0105e-04, 3.8876e-03, 3.4696e-04,
        1.7889e-04, 2.0361e-02, 9.2628e-05, 3.7502e-04, 2.5750e-02, 1.8839e-02,
        1.1675e-01, 8.0802e-01])

max probability is torch.return_types.max(
values=tensor(0.8080),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.4548e-03, 5.2834e-08, 5.2823e-08, 5.2815e-08]),
indices=tensor([  0, 655, 337, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 77 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.9280e-03, 3.8809e-04, 6.7957e-02,
2.4272e-02, 8.8431e-02, 3.6239e-01,
        4.2632e-01, 3.9944e-04, 2.7452e-03, 1.1533e-02, 5.7036e-04, 5.2090e-05,
        1.1526e-03, 5.8079e-03, 5.4186e-04, 4.8114e-04, 3.6529e-04, 1.2659e-03,
        2.3686e-03, 8.4732e-04])

max probability is torch.return_types.max(
values=tensor(0.4263),
```

```
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.8070e-06, 1.9197e-07, 1.9196e-07, 1.9195e-07]),
indices=tensor([  0, 910, 914, 896]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 78 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0029, 0.0074, 0.0020, 0.0008, 0.0017,
0.0018, 0.0019, 0.0111, 0.0066,
        0.0018, 0.0837, 0.6598, 0.0710, 0.0132, 0.0036, 0.0209, 0.0585, 0.0046,
        0.0195, 0.0101])

max probability is torch.return_types.max(
values=tensor(0.6598),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.3874e-04, 1.7567e-05, 1.7567e-05, 1.7566e-05]),
indices=tensor([  0, 323, 910, 914]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 79 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.0722e-09, 3.2364e-11, 2.9927e-11,
6.6533e-10, 2.6372e-09, 1.9598e-09,
        1.5373e-09, 2.5963e-09, 1.2806e-10, 9.5631e-09, 8.5531e-11, 1.2574e-06,
        2.9000e-08, 2.6778e-09, 9.9997e-01, 8.6595e-11, 1.5088e-09, 2.6463e-05,
        6.1401e-06, 2.2076e-09])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8324e-14, 2.4127e-16, 2.4123e-16, 2.4123e-16]),
indices=tensor([  0, 255, 337, 748]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 80 :   Prediction is Correct?
Yes
```

```
first 20 classes probability: tensor([1.4353e-04, 5.8595e-04, 1.9087e-01,
2.1707e-01, 5.2621e-01, 5.0166e-02,
        1.4391e-02, 1.2572e-05, 9.2214e-06, 2.5185e-05, 2.2357e-05, 4.8577e-07,
        2.6654e-05, 4.8235e-05, 3.0339e-05, 3.7910e-06, 3.0452e-06, 3.2982e-05,
        2.5626e-04, 3.4749e-05])

max probability is torch.return_types.max(
values=tensor(0.5262),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0932e-06, 6.4287e-08, 6.4269e-08, 6.4225e-08]),
indices=tensor([  0, 319, 316, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 81 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.6482e-04, 8.8284e-03, 7.3885e-06,
8.0747e-07, 1.1073e-05, 1.1044e-05,
        1.3428e-05, 8.5498e-01, 1.3467e-01, 1.0545e-05, 2.6917e-05, 5.9599e-06,
        1.2140e-03, 1.7562e-06, 1.6386e-06, 3.7417e-05, 5.0104e-06, 1.6149e-07,
        5.6140e-06, 5.6157e-07])

max probability is torch.return_types.max(
values=tensor(0.8550),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.0843e-09, 8.5896e-11, 8.5865e-11, 8.5831e-11]),
indices=tensor([  0, 836, 316, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 82 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.2544e-03, 3.6956e-03, 1.4852e-03,
1.8546e-04, 6.9059e-04, 1.7437e-03,
        2.7036e-03, 2.6679e-01, 6.9792e-01, 1.6511e-03, 4.7816e-04, 1.6238e-05,
        1.5984e-02, 1.9547e-04, 1.5169e-05, 1.0017e-04, 2.8463e-05, 1.6219e-05,
        2.6150e-05, 1.2417e-05])

max probability is torch.return_types.max(
values=tensor(0.6979),
indices=tensor(8))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2814e-07, 9.9307e-09, 9.9296e-09, 9.9249e-09]),
indices=tensor([  0, 319, 836, 794]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 83 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.7849e-06, 2.4146e-06, 4.9703e-06,
5.0036e-06, 4.3058e-05, 2.3840e-06,
        1.5735e-06, 5.5386e-05, 4.1097e-05, 2.2667e-05, 1.6457e-04, 5.2602e-07,
        1.7753e-06, 5.1146e-04, 8.3823e-07, 1.3327e-05, 1.0462e-03, 1.3097e-03,
        1.9275e-01, 8.0238e-01])

max probability is torch.return_types.max(
values=tensor(0.8024),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6423e-03, 2.1107e-10, 2.1095e-10, 2.1089e-10]),
indices=tensor([  0, 337, 655, 158]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 84 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.7656e-03, 8.2795e-01, 3.2127e-03,
8.2076e-04, 2.6266e-03, 8.9898e-03,
        1.1398e-02, 2.6373e-02, 4.3044e-02, 2.2980e-04, 3.5618e-03, 5.7473e-04,
        6.4283e-02, 7.1150e-04, 2.2006e-04, 1.5447e-03, 7.8110e-04, 4.7322e-05,
        1.5111e-04, 1.0245e-04])

max probability is torch.return_types.max(
values=tensor(0.8279),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.9187e-06, 6.4831e-07, 6.4752e-07, 6.4729e-07]),
indices=tensor([  0, 319, 316,  44]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 85 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.6397e-04, 2.4078e-04, 1.3239e-03,
```

```
1.0811e-03, 5.0758e-03, 5.2871e-04,
        3.3545e-04, 1.7604e-02, 1.3860e-02, 8.8624e-03, 3.7592e-03, 6.4875e-05,
        1.7307e-03, 1.5204e-02, 2.7219e-05, 1.3372e-03, 5.0891e-02, 9.1385e-04,
        3.5095e-01, 5.1350e-01])

max probability is torch.return_types.max(
values=tensor(0.5135),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1480e-02, 7.9813e-07, 7.9793e-07, 7.9788e-07]),
indices=tensor([  0, 914, 337,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 86 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.9038e-04, 1.9816e-04, 3.0479e-05,
9.4504e-06, 2.0580e-05, 3.9727e-05,
        3.5203e-05, 1.7295e-04, 4.6215e-05, 3.0057e-05, 4.8640e-03, 9.7774e-01,
        4.0280e-03, 8.8308e-04, 2.0076e-04, 6.5796e-05, 9.8941e-03, 1.9915e-04,
        5.6076e-04, 4.8203e-04])

max probability is torch.return_types.max(
values=tensor(0.9777),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7151e-06, 1.1581e-08, 1.1578e-08, 1.1573e-08]),
indices=tensor([  0, 691,  82, 382]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 87 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.6178e-05, 4.6425e-07, 6.5943e-06,
9.0786e-06, 7.2160e-05, 5.1270e-06,
        5.4668e-06, 5.6610e-05, 2.9867e-05, 2.2240e-04, 8.1151e-05, 9.3390e-06,
        1.0271e-06, 4.5371e-04, 1.3858e-05, 3.9783e-06, 5.9903e-04, 1.3521e-02,
        7.6705e-01, 2.1770e-01])

max probability is torch.return_types.max(
values=tensor(0.7671),
indices=tensor(18))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([1.4192e-04, 8.6335e-11, 8.6289e-11, 8.6254e-11]),
indices=tensor([  0, 158, 655, 337]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 88 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.9191e-06, 1.9892e-05, 7.5388e-07,
5.3637e-07, 2.8278e-06, 7.1724e-06,
        1.5634e-06, 1.4616e-06, 1.9399e-05, 4.5865e-06, 9.8837e-01, 3.7057e-05,
        3.2036e-03, 1.7782e-03, 3.6852e-08, 5.7964e-03, 6.4887e-04, 1.2757e-06,
        2.1776e-05, 8.0960e-05])

max probability is torch.return_types.max(
values=tensor(0.9884),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7003e-08, 3.0879e-11, 3.0847e-11, 3.0763e-11]),
indices=tensor([  0, 319, 771, 316]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 89 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.0346e-01, 2.2894e-04, 4.1557e-03,
1.9432e-03, 1.0353e-02, 5.4686e-03,
        5.3916e-03, 1.4765e-03, 1.2430e-02, 4.5922e-02, 9.1693e-03, 9.7312e-04,
        1.7186e-02, 4.8177e-01, 3.7806e-04, 4.6493e-03, 4.8213e-02, 1.1517e-02,
        1.0225e-02, 2.4353e-02])

max probability is torch.return_types.max(
values=tensor(0.4818),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.3454e-05, 7.1331e-07, 7.1320e-07, 7.1309e-07]),
indices=tensor([  0, 323, 771, 910]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 90 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.4905e-03, 8.4678e-05, 3.6810e-02,
1.7013e-02, 6.3344e-02, 4.4558e-03,
```

```
        1.4483e-03, 1.1290e-02, 8.2628e-03, 6.4187e-02, 1.9793e-03, 4.0214e-05,
        1.9762e-03, 2.0762e-02, 6.9968e-05, 8.5839e-04, 4.7497e-03, 1.6834e-03,
        6.1297e-01, 1.4412e-01])

max probability is torch.return_types.max(
values=tensor(0.6130),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.3097e-04, 8.3166e-08, 8.3165e-08, 8.3131e-08]),
indices=tensor([  0, 914,  91, 197]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 91 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.3440e-03, 1.0955e-03, 4.7987e-02,
4.2351e-02, 7.1413e-02, 9.6113e-02,
        7.1455e-02, 3.8964e-02, 9.0973e-02, 3.8899e-01, 2.0207e-02, 3.3818e-04,
        9.3861e-03, 3.2244e-02, 2.8788e-04, 4.2111e-03, 7.3462e-03, 2.4266e-03,
        3.2596e-02, 2.6689e-02])

max probability is torch.return_types.max(
values=tensor(0.3890),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.4791e-04, 6.4710e-06, 6.4690e-06, 6.4681e-06]),
indices=tensor([  0, 771, 914, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 92 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.7045e-04, 8.7031e-04, 2.4623e-04,
1.7789e-05, 1.4790e-04, 2.2489e-05,
        1.3427e-05, 1.0024e-02, 7.0144e-03, 2.5883e-04, 4.8329e-03, 4.9865e-03,
        1.7369e-02, 9.7790e-03, 8.2253e-05, 1.6557e-02, 8.5907e-01, 8.3282e-04,
        2.2299e-02, 4.4196e-02])

max probability is torch.return_types.max(
values=tensor(0.8591),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([3.6772e-04, 1.4804e-07, 1.4802e-07, 1.4799e-07]),
indices=tensor([  0, 910, 197, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 93 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0157, 0.1208, 0.0026, 0.0009, 0.0034,
0.0440, 0.0118, 0.0561, 0.2614,
        0.0067, 0.0838, 0.0066, 0.2592, 0.0262, 0.0004, 0.0573, 0.0240, 0.0004,
        0.0029, 0.0042])

max probability is torch.return_types.max(
values=tensor(0.2614),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6929e-04, 1.2082e-05, 1.2069e-05, 1.2068e-05]),
indices=tensor([  0, 319, 316, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 94 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.8052e-05, 5.7189e-06, 3.4815e-06,
1.9326e-06, 4.1235e-06, 6.3401e-06,
        3.3102e-06, 6.8482e-05, 1.4239e-05, 1.8350e-05, 2.1311e-03, 9.8764e-01,
        1.2284e-03, 2.4022e-04, 1.0819e-04, 3.7551e-05, 6.4287e-03, 1.7781e-04,
        1.3860e-03, 4.5324e-04])

max probability is torch.return_types.max(
values=tensor(0.9876),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3823e-06, 1.9534e-09, 1.9528e-09, 1.9519e-09]),
indices=tensor([  0, 914, 691,  82]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 95 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4800e-03, 1.2098e-04, 4.1188e-03,
5.1725e-03, 9.9229e-03, 6.2650e-03,
        4.0729e-03, 1.5438e-03, 5.4853e-03, 3.4364e-02, 3.1720e-02, 2.2229e-03,
        2.6071e-02, 6.4031e-01, 5.5317e-04, 6.8478e-03, 1.1380e-01, 5.6725e-03,
```

```
        3.6065e-02, 6.3306e-02])

max probability is torch.return_types.max(
values=tensor(0.6403),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1046e-04, 7.0525e-07, 7.0496e-07, 7.0486e-07]),
indices=tensor([  0, 914, 771, 323]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 96 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.0828e-02, 2.5361e-01, 6.5944e-03,
4.6987e-04, 1.2403e-03, 4.0974e-03,
        6.3946e-03, 2.1234e-01, 4.5136e-01, 9.0053e-04, 5.0875e-04, 9.9382e-05,
        4.0965e-02, 1.3481e-04, 1.3955e-05, 1.8569e-04, 2.0094e-04, 4.6328e-06,
        1.8159e-05, 1.1735e-05])

max probability is torch.return_types.max(
values=tensor(0.4514),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8364e-07, 1.6624e-08, 1.6603e-08, 1.6603e-08]),
indices=tensor([  0, 319, 836, 316]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 97 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7913e-04, 8.6293e-05, 8.8341e-01,
3.2134e-02, 4.1811e-02, 9.7735e-03,
        2.6053e-02, 1.7804e-05, 3.3291e-05, 1.8934e-04, 1.0295e-04, 1.0767e-06,
        3.3377e-04, 4.6690e-03, 4.9456e-05, 5.4452e-05, 1.5043e-04, 2.5317e-04,
        1.9789e-04, 4.7595e-04])

max probability is torch.return_types.max(
values=tensor(0.8834),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4895e-06, 1.8860e-08, 1.8859e-08, 1.8855e-08]),
indices=tensor([  0, 748, 150, 910]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 98 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0290e-03, 5.9137e-03, 9.3116e-04,
5.7347e-04, 2.9876e-03, 5.0372e-03,
        3.0294e-03, 2.1373e-01, 4.4250e-01, 1.3390e-02, 7.2720e-02, 1.6834e-04,
        2.5180e-02, 1.3519e-02, 2.0080e-05, 4.2347e-02, 1.9010e-02, 3.8750e-04,
        3.0061e-02, 1.0649e-01])

max probability is torch.return_types.max(
values=tensor(0.4425),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.0695e-04, 3.9776e-07, 3.9758e-07, 3.9758e-07]),
indices=tensor([  0, 319, 771, 316]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 99 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.4179e-04, 2.7575e-06, 4.2798e-05,
1.5626e-05, 1.2972e-04, 7.2264e-06,
        4.4279e-06, 1.6764e-04, 1.6353e-04, 1.9605e-04, 8.5818e-04, 9.9720e-04,
        1.4471e-04, 1.3512e-02, 1.9866e-04, 3.4284e-04, 1.8607e-02, 4.2377e-02,
        8.4690e-01, 7.4992e-02])

max probability is torch.return_types.max(
values=tensor(0.8469),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.2980e-05, 3.3594e-09, 3.3594e-09, 3.3584e-09]),
indices=tensor([  0,  19, 158, 230]))
End**************************************************************************

currently at 100 current time is 3.2242431640625
**************************************************************************
Predicting Test Sample 100 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.2786e-02, 1.7596e-04, 2.7856e-02,
2.3244e-02, 1.8393e-02, 1.1934e-02,
        3.2072e-02, 7.1152e-03, 1.5364e-02, 7.9274e-01, 1.3534e-03, 8.0479e-05,
        4.4438e-03, 2.1919e-02, 4.9924e-03, 4.5580e-04, 1.4014e-03, 1.4789e-02,
```

```
        5.1045e-03, 2.6973e-03])

max probability is torch.return_types.max(
values=tensor(0.7927),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6272e-05, 1.1230e-06, 1.1230e-06, 1.1226e-06]),
indices=tensor([  0, 771, 323,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 101 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.9435e-05, 5.9032e-06, 3.7079e-01,
4.8039e-01, 1.3905e-01, 7.1693e-03,
        2.5285e-03, 6.4633e-07, 3.4874e-07, 2.3328e-05, 4.2943e-07, 9.1075e-10,
        1.6193e-06, 1.3012e-05, 7.1915e-07, 1.6812e-08, 5.5776e-08, 1.2824e-06,
        2.6305e-06, 3.4751e-06])

max probability is torch.return_types.max(
values=tensor(0.4804),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8095e-08, 1.0320e-10, 1.0313e-10, 1.0312e-10]),
indices=tensor([  0, 771, 910, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 102 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.2286e-04, 5.8411e-07, 4.1704e-06,
2.4550e-06, 3.9015e-05, 5.9114e-06,
        4.6607e-06, 3.3104e-05, 7.9141e-05, 7.8696e-05, 3.5917e-05, 4.3130e-05,
        4.9455e-05, 4.4308e-04, 1.0243e-01, 1.8830e-05, 3.5532e-05, 7.8268e-01,
        1.1308e-01, 7.1412e-04])

max probability is torch.return_types.max(
values=tensor(0.7827),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2463e-07, 3.2158e-10, 3.2157e-10, 3.2132e-10]),
indices=tensor([  0, 337, 748, 914]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 103 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.3658e-04, 1.1224e-03, 1.8421e-03,
2.0714e-04, 1.0671e-03, 2.6526e-04,
        6.5285e-04, 5.6096e-03, 1.5894e-02, 1.3179e-04, 2.1310e-02, 4.4717e-04,
        9.4261e-01, 4.3549e-03, 1.5311e-04, 1.4782e-03, 8.7483e-04, 2.2015e-04,
        2.6326e-04, 4.4809e-04])

max probability is torch.return_types.max(
values=tensor(0.9426),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5766e-06, 1.1262e-07, 1.1261e-07, 1.1258e-07]),
indices=tensor([  0,  44, 319, 910]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 104 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.3159e-04, 6.9649e-04, 2.5331e-01,
2.3551e-01, 4.3775e-01, 4.6583e-02,
        2.3193e-02, 5.9868e-05, 4.0508e-05, 1.2895e-04, 1.0337e-04, 4.7689e-06,
        1.8953e-04, 2.8110e-04, 2.1897e-04, 2.1411e-05, 2.0443e-05, 1.7797e-04,
        7.4810e-04, 1.2600e-04])

max probability is torch.return_types.max(
values=tensor(0.4377),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.9945e-06, 6.3050e-07, 6.3032e-07, 6.3008e-07]),
indices=tensor([  0, 316, 319, 957]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 105 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0126, 0.0048, 0.0210, 0.0182, 0.0480,
0.0223, 0.0571, 0.2252, 0.2226,
        0.0569, 0.0049, 0.0060, 0.0442, 0.0258, 0.0529, 0.0014, 0.0135, 0.0364,
        0.0512, 0.0130])
```

```
max probability is torch.return_types.max(
values=tensor(0.2252),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.1217e-04, 6.3449e-05, 6.3440e-05, 6.3436e-05]),
indices=tensor([  0, 717, 197, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 106 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.7935e-07, 3.3821e-07, 1.6066e-05,
2.3870e-05, 1.0549e-04, 1.6289e-06,
        1.3288e-06, 1.5510e-04, 8.0404e-05, 1.6513e-04, 1.7556e-04, 1.2554e-06,
        2.2130e-05, 3.6037e-03, 5.3031e-06, 5.1709e-05, 1.1221e-03, 4.6520e-04,
        8.9394e-01, 9.9956e-02])

max probability is torch.return_types.max(
values=tensor(0.8939),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1045e-04, 2.1755e-10, 2.1733e-10, 2.1732e-10]),
indices=tensor([  0, 337, 197, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 107 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.9697e-03, 3.7609e-03, 7.9584e-05,
7.9772e-06, 9.3212e-05, 2.1866e-05,
        4.0381e-05, 7.9603e-01, 1.7307e-01, 1.2011e-04, 5.8196e-04, 9.3512e-04,
        2.0452e-02, 8.0206e-05, 5.8128e-05, 2.3533e-04, 8.9817e-04, 2.2757e-05,
        4.4958e-04, 7.6094e-05])

max probability is torch.return_types.max(
values=tensor(0.7960),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.3078e-07, 1.1573e-08, 1.1568e-08, 1.1561e-08]),
indices=tensor([  0, 316, 836, 391]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 108 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.6585e-05, 6.9833e-05, 2.4081e-04,
1.6682e-04, 7.6957e-04, 1.2097e-04,
        7.8829e-05, 1.0891e-03, 1.2799e-03, 4.4906e-04, 1.6145e-03, 1.7024e-05,
        1.1461e-04, 6.1773e-03, 3.2774e-06, 2.9830e-04, 2.8909e-02, 1.0516e-03,
        1.4478e-01, 8.0536e-01])

max probability is torch.return_types.max(
values=tensor(0.8054),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.3395e-03, 2.4286e-08, 2.4283e-08, 2.4283e-08]),
indices=tensor([  0, 337, 655, 914]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 109 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.5395e-03, 2.0553e-04, 5.1544e-04,
3.4525e-04, 1.0359e-03, 1.4132e-04,
        9.6244e-05, 1.4414e-03, 2.3960e-03, 1.2152e-03, 8.0768e-03, 5.6371e-04,
        1.6892e-03, 9.2331e-02, 1.2422e-04, 2.0161e-03, 5.5845e-02, 1.2304e-02,
        3.1752e-01, 4.9526e-01])

max probability is torch.return_types.max(
values=tensor(0.4953),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.7951e-03, 5.6709e-07, 5.6708e-07, 5.6693e-07]),
indices=tensor([  0, 337, 914, 158]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 110 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3845e-04, 9.8108e-01, 2.7790e-04,
1.4319e-04, 4.8402e-04, 1.9167e-03,
        1.9626e-03, 7.1190e-05, 9.9772e-05, 9.8766e-06, 8.6981e-03, 3.8714e-05,
        2.5226e-03, 7.3738e-05, 5.6194e-06, 2.3718e-03, 7.6137e-05, 1.4333e-06,
        1.9389e-05, 9.7084e-06])

max probability is torch.return_types.max(
```

```
values=tensor(0.9811),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1609e-07, 2.4207e-09, 2.4128e-09, 2.4126e-09]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 111 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.7430e-07, 9.9996e-01, 4.8895e-07,
2.6686e-08, 6.7736e-07, 1.0031e-07,
        3.0210e-07, 2.5480e-05, 3.7987e-06, 4.9112e-10, 1.5624e-08, 1.2098e-09,
        4.7093e-06, 5.4768e-09, 2.9339e-10, 8.9117e-08, 5.5622e-07, 1.9255e-11,
        2.1999e-09, 1.7701e-09])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3634e-12, 1.3388e-16, 1.3371e-16, 1.3354e-16]),
indices=tensor([  0, 316, 319, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 112 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0048, 0.0082, 0.1212, 0.0453, 0.1143,
0.0374, 0.0730, 0.0566, 0.0491,
        0.0800, 0.0058, 0.0010, 0.0098, 0.0344, 0.0051, 0.0034, 0.0167, 0.0170,
        0.2358, 0.0436])

max probability is torch.return_types.max(
values=tensor(0.2358),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5264e-03, 3.6570e-05, 3.6564e-05, 3.6564e-05]),
indices=tensor([  0, 914, 197,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 113 :   Prediction is Correct?
```

```
No
first 20 classes probability: tensor([8.3931e-04, 4.7975e-04, 3.6651e-04,
8.9638e-05, 4.4094e-04, 1.1429e-03,
        5.6010e-04, 9.5221e-04, 2.5388e-03, 4.4014e-04, 7.3709e-02, 3.2142e-01,
        2.5012e-01, 3.2032e-02, 1.0859e-03, 5.2144e-03, 2.8833e-01, 1.8935e-03,
        7.4117e-03, 1.0598e-02])

max probability is torch.return_types.max(
values=tensor(0.3214),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.7500e-05, 2.7524e-07, 2.7507e-07, 2.7505e-07]),
indices=tensor([  0, 914, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 114 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.8259e-03, 1.2351e-04, 4.1740e-04,
4.7366e-04, 2.3620e-03, 4.6630e-03,
        8.7670e-04, 3.5962e-03, 3.5509e-02, 3.2663e-02, 2.8251e-02, 1.3112e-03,
        1.8766e-02, 5.4297e-01, 2.5944e-04, 1.7080e-02, 1.2430e-01, 7.1056e-03,
        9.4636e-02, 7.8040e-02])

max probability is torch.return_types.max(
values=tensor(0.5430),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0360e-04, 5.9920e-07, 5.9907e-07, 5.9861e-07]),
indices=tensor([  0, 771, 914, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 115 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.1444e-05, 9.5991e-07, 3.2181e-05,
2.8476e-05, 2.4105e-04, 3.6167e-05,
        1.1667e-05, 7.0318e-03, 1.2907e-02, 2.0162e-02, 9.3672e-05, 1.0346e-06,
        6.0541e-05, 3.3168e-03, 1.3854e-05, 2.8382e-04, 1.1418e-03, 3.0846e-04,
        9.4870e-01, 5.6115e-03])

max probability is torch.return_types.max(
values=tensor(0.9487),
indices=tensor(18))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.9647e-06, 4.5697e-10, 4.5687e-10, 4.5677e-10]),
indices=tensor([  0, 896, 914, 197]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 116 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0083, 0.0004, 0.0249, 0.0363, 0.1200,
0.0798, 0.1870, 0.0027, 0.0071,
        0.0709, 0.0024, 0.0008, 0.0061, 0.0846, 0.1161, 0.0017, 0.0027, 0.1933,
        0.0363, 0.0127])

max probability is torch.return_types.max(
values=tensor(0.1933),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0131e-04, 6.0643e-06, 6.0641e-06, 6.0633e-06]),
indices=tensor([  0, 323, 748, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 117 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.6404e-05, 1.8630e-05, 4.5394e-06,
2.8931e-06, 7.5223e-06, 1.6474e-05,
        8.7594e-06, 2.4329e-04, 3.8620e-05, 2.6496e-05, 2.6814e-03, 9.9047e-01,
        7.3074e-04, 1.8141e-04, 9.7551e-05, 5.3611e-05, 2.8432e-03, 1.5077e-04,
        1.9494e-03, 3.9971e-04])

max probability is torch.return_types.max(
values=tensor(0.9905),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8444e-06, 4.0233e-09, 4.0226e-09, 4.0216e-09]),
indices=tensor([  0, 914, 691, 382]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 118 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.6422e-03, 8.1707e-02, 2.8331e-02,
```

```
9.5746e-02, 2.2166e-02, 6.8478e-01,
        8.1693e-02, 5.7458e-07, 1.4740e-06, 2.8058e-05, 1.6981e-03, 2.7701e-06,
        1.6056e-05, 1.3234e-04, 7.4989e-07, 2.2649e-05, 1.7244e-06, 1.0130e-06,
        1.3313e-05, 1.4224e-05])

max probability is torch.return_types.max(
values=tensor(0.6848),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7395e-08, 1.9104e-10, 1.9059e-10, 1.9053e-10]),
indices=tensor([  0, 319, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 119 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.6873e-05, 6.0117e-06, 1.3783e-04,
1.1539e-04, 3.4515e-04, 2.0586e-04,
        1.6695e-04, 8.8982e-02, 1.7789e-01, 7.2558e-01, 2.5265e-04, 2.4522e-07,
        7.8813e-04, 1.0242e-03, 3.1857e-06, 6.9706e-04, 2.1480e-04, 7.2619e-06,
        3.3674e-03, 1.5408e-04])

max probability is torch.return_types.max(
values=tensor(0.7256),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.6042e-08, 2.9210e-10, 2.9172e-10, 2.9169e-10]),
indices=tensor([  0, 896, 316, 363]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 120 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.5916e-05, 3.1781e-07, 2.9360e-07,
5.5133e-07, 2.6798e-06, 1.8400e-06,
        1.7844e-06, 4.3916e-06, 1.6457e-06, 1.0816e-05, 7.9305e-06, 1.0009e-02,
        2.7876e-04, 9.3732e-05, 9.8192e-01, 5.6976e-06, 2.9829e-04, 5.0945e-03,
        2.1703e-03, 5.0690e-05])

max probability is torch.return_types.max(
values=tensor(0.9819),
indices=tensor(14))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([2.3030e-08, 3.0768e-10, 3.0767e-10, 3.0759e-10]),
indices=tensor([  0, 691, 748, 230]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 121 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0969, 0.0004, 0.0014, 0.0010, 0.0033,
0.0033, 0.0028, 0.0129, 0.0275,
        0.1694, 0.0077, 0.1135, 0.0807, 0.0333, 0.0920, 0.0133, 0.1493, 0.0663,
        0.1086, 0.0123])

max probability is torch.return_types.max(
values=tensor(0.1694),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.5061e-05, 4.2468e-06, 4.2454e-06, 4.2442e-06]),
indices=tensor([  0, 323, 914, 910]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 122 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.3939e-09, 5.3786e-09, 3.8389e-11,
8.3668e-12, 3.5869e-11, 2.0772e-10,
        9.4753e-11, 6.8050e-08, 1.1064e-09, 2.6151e-10, 7.4834e-07, 9.9994e-01,
        7.0044e-06, 8.3446e-09, 3.0646e-07, 4.1411e-09, 5.1434e-05, 6.2118e-09,
        1.7631e-07, 1.6615e-08])

max probability is torch.return_types.max(
values=tensor(0.9999),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.7506e-13, 7.0692e-17, 7.0633e-17, 7.0620e-17]),
indices=tensor([  0, 691, 382,  82]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 123 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.3608e-02, 7.4797e-01, 1.8643e-03,
6.8527e-04, 2.2066e-03, 4.1825e-03,
        3.4259e-03, 9.7206e-03, 9.5149e-03, 4.1453e-04, 1.4915e-02, 2.6810e-02,
```

```
        7.9106e-02, 2.1366e-02, 1.1734e-02, 3.8515e-03, 1.3780e-02, 7.1532e-03,
        3.4231e-03, 2.8981e-03])

max probability is torch.return_types.max(
values=tensor(0.7480),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.2671e-05, 1.4041e-06, 1.4028e-06, 1.4027e-06]),
indices=tensor([  0, 323, 910, 888]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 124 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.9752e-04, 1.5379e-03, 2.7479e-01,
2.3548e-01, 4.1296e-01, 4.4668e-02,
        2.4478e-02, 1.3636e-04, 1.0244e-04, 2.6663e-04, 2.4483e-04, 9.9833e-06,
        4.6143e-04, 5.1634e-04, 3.0284e-04, 5.9551e-05, 5.9547e-05, 3.1454e-04,
        1.0687e-03, 2.7259e-04])

max probability is torch.return_types.max(
values=tensor(0.4130),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6937e-05, 1.8383e-06, 1.8383e-06, 1.8376e-06]),
indices=tensor([  0, 316, 319, 466]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 125 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.0818e-03, 9.4935e-04, 9.5345e-03,
3.8039e-03, 1.7202e-02, 2.9379e-03,
        1.8731e-03, 7.8730e-03, 2.6940e-02, 1.2973e-02, 6.8923e-02, 1.6923e-04,
        1.9210e-02, 4.3594e-01, 1.8086e-04, 9.3558e-02, 5.9061e-02, 9.2431e-03,
        1.0897e-01, 1.1214e-01])

max probability is torch.return_types.max(
values=tensor(0.4359),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.7295e-04, 1.9818e-06, 1.9804e-06, 1.9802e-06]),
```

```
indices=tensor([  0, 771, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 126 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0364, 0.0020, 0.1040, 0.0372, 0.0938,
0.0456, 0.0442, 0.0124, 0.0497,
        0.0769, 0.0280, 0.0008, 0.0310, 0.2553, 0.0021, 0.0176, 0.0315, 0.0208,
        0.0517, 0.0417])

max probability is torch.return_types.max(
values=tensor(0.2553),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.0365e-04, 1.7485e-05, 1.7480e-05, 1.7479e-05]),
indices=tensor([  0, 771, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 127 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1631e-03, 4.0954e-04, 2.2512e-02,
2.9193e-01, 1.5312e-01, 4.3112e-01,
        9.7807e-02, 1.3399e-05, 1.6364e-05, 1.8682e-04, 2.2886e-05, 5.0585e-06,
        3.2142e-05, 2.4351e-04, 6.4817e-04, 7.5577e-06, 2.5653e-06, 1.5949e-04,
        4.4315e-04, 1.0925e-04])

max probability is torch.return_types.max(
values=tensor(0.4311),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0485e-06, 4.8552e-08, 4.8545e-08, 4.8516e-08]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 128 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.7204e-04, 3.2176e-02, 5.3347e-05,
1.1173e-05, 1.1483e-04, 2.2032e-03,
        1.0426e-03, 2.0745e-01, 7.1243e-01, 2.0833e-04, 6.6395e-03, 4.1943e-05,
        2.9562e-02, 2.9427e-04, 3.9904e-06, 7.1731e-03, 1.1292e-04, 3.4250e-06,
        6.1907e-05, 3.7179e-05])
```

```
max probability is torch.return_types.max(
values=tensor(0.7124),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.0890e-07, 1.1018e-08, 1.1012e-08, 1.1003e-08]),
indices=tensor([  0, 836, 319, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 129 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0079, 0.0007, 0.0072, 0.0109, 0.0323,
0.0270, 0.0120, 0.0441, 0.1179,
        0.1611, 0.0135, 0.0005, 0.0060, 0.0622, 0.0005, 0.0055, 0.0220, 0.0080,
        0.3162, 0.1341])

max probability is torch.return_types.max(
values=tensor(0.3162),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3854e-03, 8.5432e-06, 8.5412e-06, 8.5372e-06]),
indices=tensor([  0, 914, 771,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 130 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.8900e-03, 1.2204e-02, 2.5093e-04,
7.2357e-05, 5.8846e-04, 3.7428e-03,
        4.7993e-03, 3.6171e-01, 5.7903e-01, 1.2964e-03, 3.5805e-03, 1.1375e-03,
        2.4300e-02, 5.9405e-04, 7.5023e-05, 3.0437e-03, 8.8836e-04, 5.7036e-05,
        4.1607e-04, 1.2711e-04])

max probability is torch.return_types.max(
values=tensor(0.5790),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.4373e-06, 2.0794e-07, 2.0789e-07, 2.0785e-07]),
indices=tensor([  0, 836, 316, 957]))
End*****************************************************************************
```

```
**************************************************************************
Predicting Test Sample 131 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1117e-11, 5.0045e-11, 3.4052e-14,
4.3593e-15, 4.7781e-14, 1.9297e-13,
        1.5251e-13, 7.3827e-10, 3.7614e-12, 1.3288e-13, 2.2920e-08, 1.0000e+00,
        2.6347e-07, 5.3601e-11, 1.3341e-09, 2.6630e-12, 6.2637e-07, 2.7851e-11,
        1.4939e-09, 9.8662e-11])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.9973e-16, 1.1607e-21, 1.1596e-21, 1.1590e-21]),
indices=tensor([  0, 691,  82, 382]))
End*************************************************************************

**************************************************************************
Predicting Test Sample 132 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.2354e-02, 4.0951e-03, 2.3473e-01,
1.1063e-01, 1.1231e-01, 2.0985e-01,
        9.4753e-02, 4.8138e-03, 2.0417e-02, 8.9349e-02, 2.0158e-02, 1.8297e-04,
        8.6370e-03, 2.2994e-02, 2.7524e-04, 3.7731e-03, 3.1086e-03, 2.2399e-03,
        6.3116e-03, 1.1993e-02])

max probability is torch.return_types.max(
values=tensor(0.2347),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1436e-04, 7.1983e-06, 7.1952e-06, 7.1923e-06]),
indices=tensor([  0, 771, 319, 896]))
End*************************************************************************

**************************************************************************
Predicting Test Sample 133 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.1887e-03, 5.5855e-01, 1.3553e-04,
5.9379e-05, 3.3068e-04, 1.7791e-03,
        5.8327e-04, 1.4990e-03, 1.2612e-03, 1.1914e-04, 9.2510e-02, 2.6278e-03,
        1.4186e-02, 1.2727e-03, 3.9860e-04, 3.1763e-01, 3.6438e-03, 1.2042e-04,
        1.2479e-03, 7.4712e-04])

max probability is torch.return_types.max(
```

```
values=tensor(0.5585),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.2924e-06, 1.1180e-07, 1.1157e-07, 1.1155e-07]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 134 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.5037e-03, 6.8278e-04, 1.0837e-01,
3.7396e-02, 8.9561e-02, 2.2202e-02,
        7.1083e-02, 1.0218e-03, 5.7395e-03, 2.3553e-02, 4.5719e-02, 6.1870e-05,
        2.8563e-02, 5.2295e-01, 1.9389e-04, 5.5841e-03, 5.7811e-03, 3.7469e-03,
        2.1422e-03, 1.6624e-02])

max probability is torch.return_types.max(
values=tensor(0.5230),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.5414e-05, 5.0224e-07, 5.0200e-07, 5.0197e-07]),
indices=tensor([  0, 771, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 135 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.9946e-03, 4.2532e-03, 1.4529e-03,
1.4198e-04, 1.1075e-03, 7.8901e-04,
        4.0493e-04, 2.0185e-02, 3.1906e-02, 3.3772e-03, 1.0867e-01, 2.4940e-03,
        2.4505e-01, 7.8799e-03, 5.9355e-04, 4.5166e-01, 9.2799e-02, 8.1071e-04,
        1.1336e-02, 8.6104e-03])

max probability is torch.return_types.max(
values=tensor(0.4517),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5412e-05, 4.8161e-07, 4.8141e-07, 4.8121e-07]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 136 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.4451e-05, 4.5893e-04, 8.3064e-06,
5.0022e-07, 5.6706e-06, 2.5460e-05,
        3.2286e-05, 1.4197e-02, 9.7213e-02, 2.5344e-06, 1.1110e-03, 4.4725e-05,
        8.8658e-01, 1.0006e-04, 1.1514e-06, 1.0289e-04, 1.5568e-05, 2.6249e-07,
        5.4962e-07, 1.2607e-06])

max probability is torch.return_types.max(
values=tensor(0.8866),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8797e-09, 4.2605e-11, 4.2590e-11, 4.2534e-11]),
indices=tensor([  0, 836, 319,  44]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 137 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.9663e-03, 7.8156e-01, 1.5194e-02,
4.6029e-03, 1.3313e-02, 1.0256e-02,
        3.1197e-02, 4.7561e-02, 5.8526e-02, 6.4428e-04, 1.0961e-03, 2.1037e-04,
        2.5010e-02, 1.1964e-03, 4.6973e-04, 8.1575e-04, 9.8772e-04, 1.8107e-04,
        3.2644e-04, 1.9028e-04])

max probability is torch.return_types.max(
values=tensor(0.7816),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2739e-05, 7.2330e-07, 7.2313e-07, 7.2278e-07]),
indices=tensor([  0, 319, 316,  44]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 138 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2629e-07, 2.5883e-09, 1.9790e-09,
2.6528e-08, 5.8637e-08, 7.8899e-08,
        5.9397e-08, 8.4697e-08, 6.3577e-09, 1.6311e-07, 1.4420e-08, 5.9257e-05,
        9.3317e-07, 1.6052e-07, 9.9968e-01, 3.1283e-09, 9.4918e-08, 1.9547e-04,
        6.0674e-05, 1.3444e-07])

max probability is torch.return_types.max(
values=tensor(0.9997),
```

```
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2662e-11, 1.9726e-13, 1.9724e-13, 1.9721e-13]),
indices=tensor([  0, 337, 255, 748]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 139 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.5101e-05, 9.9181e-01, 5.2457e-06,
1.1564e-06, 1.6262e-05, 9.7123e-06,
        2.4804e-05, 5.0554e-03, 9.1370e-04, 7.5044e-07, 6.2919e-05, 1.3608e-05,
        1.8750e-03, 4.7813e-06, 1.1261e-06, 3.9369e-05, 1.4068e-04, 9.1409e-08,
        5.0371e-06, 4.5019e-06])

max probability is torch.return_types.max(
values=tensor(0.9918),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.7842e-08, 3.9998e-11, 3.9965e-11, 3.9927e-11]),
indices=tensor([  0, 316, 319, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 140 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.0702e-05, 4.1237e-03, 3.5345e-05,
8.4939e-06, 7.3574e-05, 1.9717e-04,
        8.3056e-05, 9.3687e-04, 2.6606e-03, 6.8000e-05, 2.4288e-01, 1.4976e-03,
        4.9010e-02, 2.2662e-03, 8.9723e-05, 6.8856e-01, 5.2684e-03, 1.3192e-04,
        8.8448e-04, 1.1121e-03])

max probability is torch.return_types.max(
values=tensor(0.6886),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7550e-06, 4.9104e-08, 4.8984e-08, 4.8983e-08]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 141 :   Prediction is Correct?
```

```
Yes
first 20 classes probability: tensor([3.4791e-04, 2.9226e-05, 1.1652e-04,
1.3340e-04, 9.0763e-04, 2.1667e-04,
        7.9419e-05, 1.7823e-03, 2.5883e-03, 2.9548e-03, 7.9628e-04, 2.2425e-04,
        3.0747e-04, 5.6723e-03, 1.7589e-03, 1.5385e-03, 2.5167e-03, 1.9740e-02,
        9.4421e-01, 1.3882e-02])

max probability is torch.return_types.max(
values=tensor(0.9442),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.6716e-05, 1.3132e-07, 1.3128e-07, 1.3124e-07]),
indices=tensor([  0, 914, 337, 263]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 142 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7456e-02, 2.2235e-04, 5.1191e-03,
3.8165e-04, 2.0448e-03, 4.7523e-04,
        4.7836e-04, 2.7860e-01, 5.9356e-01, 7.4878e-02, 5.0074e-04, 3.9823e-06,
        1.5501e-02, 2.9628e-03, 1.2022e-05, 1.8486e-03, 1.7891e-03, 6.6374e-05,
        3.6484e-03, 4.4204e-04])

max probability is torch.return_types.max(
values=tensor(0.5936),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.1806e-07, 9.9450e-09, 9.9394e-09, 9.9385e-09]),
indices=tensor([  0, 896,  85, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 143 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.2013e-06, 9.9984e-01, 8.8925e-06,
3.0441e-06, 1.5806e-05, 2.8794e-05,
        3.0994e-05, 1.7458e-05, 1.1583e-05, 9.4822e-08, 4.1845e-06, 6.8164e-08,
        2.1316e-05, 3.1419e-07, 3.6128e-08, 4.3187e-06, 7.4860e-07, 8.3263e-09,
        2.6062e-07, 8.9149e-08])

max probability is torch.return_types.max(
values=tensor(0.9998),
indices=tensor(1))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4135e-10, 4.7957e-13, 4.7907e-13, 4.7828e-13]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 144 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7132e-02, 2.8303e-03, 1.0797e-01,
2.5948e-01, 1.7707e-01, 2.6646e-01,
        1.1665e-01, 6.8422e-04, 1.6549e-03, 1.6861e-02, 1.9001e-03, 7.9731e-05,
        1.5524e-03, 1.2341e-02, 7.1507e-04, 8.7483e-04, 4.4153e-04, 1.1754e-03,
        2.4753e-03, 3.0809e-03])

max probability is torch.return_types.max(
values=tensor(0.2665),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.5949e-05, 8.9135e-06, 8.9096e-06, 8.9066e-06]),
indices=tensor([  0, 319, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 145 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0390, 0.0020, 0.0775, 0.0279, 0.0713,
0.0233, 0.0360, 0.0074, 0.0190,
        0.0404, 0.0122, 0.0014, 0.0208, 0.1412, 0.0416, 0.0046, 0.0118, 0.2956,
        0.0514, 0.0367])

max probability is torch.return_types.max(
values=tensor(0.2956),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.6641e-04, 3.9636e-05, 3.9632e-05, 3.9631e-05]),
indices=tensor([  0, 748, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 146 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.3442e-05, 2.3068e-04, 9.8096e-04,
```

```
1.2605e-04, 2.1754e-03, 2.1109e-04,
        6.1193e-04, 3.2201e-02, 4.4100e-02, 1.8455e-03, 3.9488e-03, 1.8023e-03,
        4.2245e-02, 2.6012e-02, 3.3267e-04, 2.7863e-02, 7.1279e-01, 2.1427e-03,
        6.3290e-02, 3.6721e-02])

max probability is torch.return_types.max(
values=tensor(0.7128),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0927e-04, 8.4804e-08, 8.4800e-08, 8.4767e-08]),
indices=tensor([  0, 197, 910, 726]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 147 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0415, 0.0018, 0.0102, 0.0141, 0.0330,
0.1588, 0.0429, 0.0105, 0.0844,
        0.1980, 0.0573, 0.0019, 0.0324, 0.1537, 0.0017, 0.0228, 0.0206, 0.0109,
        0.0445, 0.0428])

max probability is torch.return_types.max(
values=tensor(0.1980),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.5812e-04, 1.6663e-05, 1.6652e-05, 1.6650e-05]),
indices=tensor([  0, 771, 319, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 148 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.3408e-02, 1.0093e-03, 1.9812e-03,
1.5341e-03, 3.8484e-03, 4.2370e-03,
        6.0952e-03, 2.8569e-03, 2.4937e-03, 4.1292e-03, 2.1386e-04, 5.3012e-03,
        6.2883e-03, 4.9047e-04, 7.9693e-01, 1.0217e-04, 2.7630e-04, 1.0329e-01,
        4.2078e-03, 4.1952e-04])

max probability is torch.return_types.max(
values=tensor(0.7969),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([3.5039e-06, 9.2774e-07, 9.2732e-07, 9.2710e-07]),
indices=tensor([  0, 323, 748, 255]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 149 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.3934, 0.0014, 0.0089, 0.0116, 0.0279,
0.0639, 0.0351, 0.0013, 0.0120,
        0.0973, 0.0215, 0.0016, 0.0279, 0.2431, 0.0012, 0.0078, 0.0113, 0.0064,
        0.0066, 0.0170])

max probability is torch.return_types.max(
values=tensor(0.3934),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.9693e-05, 3.0965e-06, 3.0953e-06, 3.0948e-06]),
indices=tensor([  0, 771, 323, 910]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 150 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.2681e-05, 1.0837e-08, 5.3012e-07,
2.4064e-07, 4.5242e-06, 2.1097e-07,
        2.5737e-07, 1.0560e-06, 1.5699e-06, 4.4186e-06, 3.6402e-06, 1.9273e-05,
        6.0918e-06, 7.2445e-05, 3.5430e-02, 7.0002e-07, 6.8702e-06, 9.4058e-01,
        2.3624e-02, 2.1074e-04])

max probability is torch.return_types.max(
values=tensor(0.9406),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.3244e-09, 1.1570e-12, 1.1566e-12, 1.1561e-12]),
indices=tensor([  0, 748, 337, 158]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 151 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.8554e-05, 6.5655e-05, 4.0024e-01,
3.6129e-01, 2.2261e-01, 1.1700e-02,
        3.9819e-03, 2.3930e-06, 1.2932e-06, 1.5838e-05, 2.0522e-06, 1.3841e-08,
        6.7143e-06, 1.3480e-05, 2.1687e-06, 1.2303e-07, 3.5956e-07, 3.0288e-06,
```

```
        8.8819e-06, 6.1257e-06])

max probability is torch.return_types.max(
values=tensor(0.4002),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4284e-07, 1.8914e-09, 1.8906e-09, 1.8903e-09]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 152 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.4181e-04, 2.5941e-04, 1.8517e-05,
1.1458e-05, 6.3826e-05, 4.8090e-05,
        1.1951e-05, 3.4667e-04, 7.8062e-04, 3.9492e-04, 7.2507e-01, 7.6597e-03,
        3.4194e-02, 7.2316e-03, 8.6204e-05, 1.5748e-01, 4.3194e-02, 5.3434e-04,
        1.1908e-02, 1.0096e-02])

max probability is torch.return_types.max(
values=tensor(0.7251),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7525e-05, 5.1061e-08, 5.1046e-08, 5.0990e-08]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 153 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.8529e-03, 1.7105e-03, 6.4924e-02,
1.1603e-01, 1.4262e-01, 2.7265e-01,
        3.6025e-01, 6.3315e-04, 1.3944e-03, 1.7734e-02, 8.7043e-04, 8.2246e-05,
        1.0993e-03, 4.2722e-03, 2.5369e-03, 6.9031e-04, 2.6899e-04, 1.3104e-03,
        2.8187e-03, 1.0024e-03])

max probability is torch.return_types.max(
values=tensor(0.3602),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7921e-05, 3.3851e-06, 3.3850e-06, 3.3848e-06]),
indices=tensor([  0, 316, 896, 319]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 154 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.9788e-02, 7.9304e-03, 2.0855e-02,
8.8792e-03, 3.1675e-02, 5.8386e-02,
        4.5601e-02, 6.5147e-02, 4.3151e-01, 6.4470e-02, 4.7823e-02, 3.8925e-04,
        6.3547e-02, 4.8943e-02, 4.4401e-04, 2.9818e-02, 6.8299e-03, 2.3388e-03,
        8.9127e-03, 8.8172e-03])

max probability is torch.return_types.max(
values=tensor(0.4315),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3133e-04, 1.8701e-05, 1.8684e-05, 1.8683e-05]),
indices=tensor([  0, 319, 836, 896]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 155 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0105, 0.0044, 0.0168, 0.0130, 0.0214,
0.0228, 0.0128, 0.0089, 0.0152,
        0.0175, 0.1098, 0.0436, 0.0421, 0.1198, 0.0051, 0.0289, 0.1143, 0.0176,
        0.0825, 0.0793])

max probability is torch.return_types.max(
values=tensor(0.1198),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([0.0057, 0.0002, 0.0002, 0.0002]),
indices=tensor([  0, 914, 771, 323]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 156 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.8483e-01, 1.7055e-04, 6.2963e-04,
7.1990e-05, 6.0773e-04, 3.2619e-04,
        4.3417e-04, 9.9661e-04, 5.6976e-03, 2.2393e-03, 1.6087e-04, 4.7991e-05,
        1.8163e-03, 2.5886e-04, 1.2216e-04, 4.4265e-04, 1.2958e-04, 4.9387e-04,
        3.5260e-04, 1.2106e-04])
```

```
max probability is torch.return_types.max(
values=tensor(0.9848),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.7589e-07, 5.5826e-08, 5.5818e-08, 5.5809e-08]),
indices=tensor([  0, 319, 910,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 157 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0051, 0.0017, 0.0028, 0.0016, 0.0025,
0.0033, 0.0018, 0.0013, 0.0028,
        0.0054, 0.6686, 0.0460, 0.0407, 0.0662, 0.0012, 0.0309, 0.0448, 0.0096,
        0.0239, 0.0330])

max probability is torch.return_types.max(
values=tensor(0.6686),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.0198e-04, 6.6730e-06, 6.6712e-06, 6.6691e-06]),
indices=tensor([  0, 771, 914, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 158 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.0989e-06, 1.9993e-04, 4.3663e-05,
6.3998e-06, 2.3198e-05, 1.7057e-05,
        2.0734e-05, 2.5127e-04, 7.0371e-04, 3.0119e-06, 7.1900e-03, 8.9409e-05,
        9.9079e-01, 2.7247e-04, 6.6428e-06, 3.1350e-04, 5.7187e-05, 1.0471e-06,
        2.8271e-06, 4.8665e-06])

max probability is torch.return_types.max(
values=tensor(0.9908),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.0069e-09, 1.8102e-10, 1.8101e-10, 1.8094e-10]),
indices=tensor([  0, 319,  44, 910]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 159 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.8326e-04, 1.8934e-05, 4.0518e-05,
1.3609e-05, 1.5092e-04, 3.5004e-05,
        3.1026e-05, 2.2089e-05, 7.5371e-05, 5.8865e-05, 3.6814e-03, 2.9063e-04,
        5.7984e-04, 1.2195e-02, 8.9358e-03, 7.4780e-04, 4.4506e-04, 9.4072e-01,
        2.7343e-02, 4.1264e-03])

max probability is torch.return_types.max(
values=tensor(0.9407),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0896e-06, 1.2083e-09, 1.2078e-09, 1.2074e-09]),
indices=tensor([  0, 748, 337, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 160 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0854, 0.0010, 0.0176, 0.0168, 0.0355,
0.0611, 0.0686, 0.0050, 0.0338,
        0.3974, 0.0241, 0.0017, 0.0406, 0.1385, 0.0016, 0.0080, 0.0278, 0.0062,
        0.0080, 0.0135])

max probability is torch.return_types.max(
values=tensor(0.3974),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2976e-04, 8.1412e-06, 8.1378e-06, 8.1357e-06]),
indices=tensor([  0, 771, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 161 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0022, 0.0004, 0.0169, 0.0256, 0.0434,
0.0422, 0.0340, 0.0043, 0.0157,
        0.1403, 0.0548, 0.0018, 0.0443, 0.3789, 0.0029, 0.0165, 0.0555, 0.0077,
        0.0685, 0.0398])

max probability is torch.return_types.max(
values=tensor(0.3789),
indices=tensor(13))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7940e-04, 4.2934e-06, 4.2924e-06, 4.2909e-06]),
indices=tensor([  0, 771, 914, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 162 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9998e-01, 8.5355e-07, 2.5770e-06,
4.4978e-07, 8.1212e-07, 9.1087e-06,
        3.6743e-06, 7.2374e-08, 2.6910e-07, 1.6330e-06, 1.0043e-08, 3.7924e-07,
        5.5335e-07, 5.8134e-08, 1.0474e-08, 4.4586e-10, 5.3546e-08, 8.5657e-08,
        2.2777e-08, 1.2321e-07])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.6644e-12, 8.7208e-15, 8.7170e-15, 8.7169e-15]),
indices=tensor([  0, 323, 271, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 163 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.9787e-04, 1.0788e-04, 3.2992e-03,
2.4312e-03, 6.0301e-03, 8.4015e-04,
        4.8121e-04, 2.1425e-03, 2.9864e-03, 6.5205e-03, 7.6901e-03, 7.0713e-04,
        5.1548e-03, 1.6171e-01, 1.0043e-03, 5.3409e-03, 8.4308e-02, 1.7540e-02,
        4.6284e-01, 2.2494e-01])

max probability is torch.return_types.max(
values=tensor(0.4628),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9617e-03, 1.4367e-06, 1.4362e-06, 1.4355e-06]),
indices=tensor([  0, 914, 337, 197]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 164 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0091, 0.0030, 0.0022, 0.0010, 0.0041,
```

```
0.0247, 0.0054, 0.0017, 0.0056,
        0.0033, 0.2466, 0.2005, 0.2028, 0.0365, 0.0031, 0.0392, 0.1279, 0.0078,
        0.0390, 0.0337])

max probability is torch.return_types.max(
values=tensor(0.2466),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2997e-04, 2.6857e-06, 2.6856e-06, 2.6837e-06]),
indices=tensor([  0, 914, 771, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 165 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.1375e-03, 1.0559e-02, 3.8673e-04,
4.1571e-05, 3.2135e-04, 2.5471e-04,
        3.3339e-04, 3.9179e-03, 6.5945e-03, 1.3874e-04, 4.3642e-02, 3.2693e-02,
        7.9335e-01, 9.1702e-03, 9.8807e-03, 5.4653e-02, 2.6011e-02, 2.6367e-03,
        1.3611e-03, 1.3145e-03])

max probability is torch.return_types.max(
values=tensor(0.7934),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.4899e-06, 6.2880e-07, 6.2843e-07, 6.2798e-07]),
indices=tensor([  0, 910, 323, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 166 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.3703e-03, 2.3263e-03, 1.3410e-02,
1.8327e-02, 6.3103e-02, 4.1167e-01,
        3.0754e-01, 2.3300e-03, 2.4589e-02, 6.3560e-02, 1.1488e-02, 3.9908e-04,
        2.1666e-02, 3.1399e-02, 5.6783e-04, 7.8876e-03, 5.7324e-03, 5.9634e-04,
        3.4179e-03, 2.3892e-03])

max probability is torch.return_types.max(
values=tensor(0.4117),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([2.6001e-05, 1.2840e-06, 1.2836e-06, 1.2834e-06]),
indices=tensor([  0, 319, 896, 771]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 167 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.5159e-03, 1.8447e-02, 1.3446e-03,
3.9179e-04, 2.6840e-03, 1.1242e-03,
        2.1962e-03, 6.1255e-01, 3.2607e-01, 1.5787e-03, 2.5274e-03, 5.4928e-04,
        1.5979e-02, 1.6312e-03, 1.1753e-04, 1.7023e-03, 3.0320e-03, 2.5051e-04,
        2.9390e-03, 1.5657e-03])

max probability is torch.return_types.max(
values=tensor(0.6125),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.6192e-05, 7.9857e-07, 7.9857e-07, 7.9821e-07]),
indices=tensor([  0, 957, 316, 910]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 168 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.9208e-03, 1.3231e-03, 6.1755e-03,
6.8782e-03, 2.4538e-02, 7.7518e-01,
        1.5279e-01, 2.0368e-04, 3.2551e-03, 3.7115e-03, 3.7324e-03, 2.3562e-04,
        5.9442e-03, 8.0403e-03, 1.6938e-04, 1.4881e-03, 8.6695e-04, 1.3524e-04,
        1.4860e-03, 7.7151e-04])

max probability is torch.return_types.max(
values=tensor(0.7752),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.6378e-06, 1.5964e-07, 1.5963e-07, 1.5963e-07]),
indices=tensor([  0, 896, 771, 319]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 169 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0467, 0.0097, 0.0149, 0.0042, 0.0119,
0.0141, 0.0123, 0.2837, 0.4228,
        0.1069, 0.0058, 0.0005, 0.0232, 0.0071, 0.0011, 0.0066, 0.0067, 0.0024,
```

```
        0.0092, 0.0033])

max probability is torch.return_types.max(
values=tensor(0.4228),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.6324e-05, 7.3663e-06, 7.3655e-06, 7.3655e-06]),
indices=tensor([  0, 896,  85, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 170 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.5819e-03, 7.9945e-03, 1.2265e-03,
4.0948e-04, 2.9488e-03, 8.3614e-02,
        3.9836e-02, 2.5723e-02, 7.1466e-01, 6.4716e-03, 2.6893e-02, 8.7620e-05,
        5.7214e-02, 6.3408e-03, 4.3754e-05, 2.2304e-02, 5.7683e-04, 1.0436e-04,
        2.8613e-04, 4.0442e-04])

max probability is torch.return_types.max(
values=tensor(0.7147),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6090e-06, 2.9567e-07, 2.9532e-07, 2.9513e-07]),
indices=tensor([  0, 319, 836, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 171 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.3170e-06, 5.1739e-06, 3.5640e-01,
6.0263e-01, 3.9363e-02, 1.2879e-03,
        3.0557e-04, 1.8583e-08, 5.0058e-09, 6.6526e-07, 1.5121e-08, 8.9930e-12,
        8.9466e-08, 4.3407e-07, 8.5899e-09, 1.1652e-10, 1.2201e-09, 1.3013e-08,
        3.1615e-08, 1.3260e-07])

max probability is torch.return_types.max(
values=tensor(0.6026),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4202e-10, 2.4249e-13, 2.4242e-13, 2.4234e-13]),
indices=tensor([  0, 771, 910, 319]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 172 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7647e-05, 1.6654e-06, 1.3830e-04,
1.7625e-04, 4.6310e-04, 1.3222e-05,
        2.1789e-05, 1.9761e-05, 4.9477e-05, 1.5754e-04, 1.9444e-04, 3.5432e-06,
        9.9279e-05, 6.8225e-01, 3.3722e-06, 1.9058e-04, 1.0565e-01, 1.1735e-03,
        9.5881e-03, 1.9958e-01])

max probability is torch.return_types.max(
values=tensor(0.6823),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0726e-04, 1.2598e-09, 1.2590e-09, 1.2589e-09]),
indices=tensor([  0, 655, 691, 337]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 173 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0877e-06, 9.3562e-08, 6.5282e-06,
6.2982e-06, 2.8593e-05, 3.0139e-06,
        2.1739e-06, 1.2926e-06, 1.7987e-05, 1.4942e-05, 2.2259e-04, 2.1757e-07,
        8.9198e-05, 9.6865e-01, 6.1699e-08, 3.4948e-05, 1.3680e-02, 3.5080e-05,
        3.8919e-04, 1.6814e-02])

max probability is torch.return_types.max(
values=tensor(0.9687),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1726e-06, 1.0238e-11, 1.0229e-11, 1.0229e-11]),
indices=tensor([  0, 914, 323, 771]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 174 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.5743e-04, 1.0843e-03, 3.3302e-01,
2.4474e-01, 3.4882e-01, 3.7245e-02,
        2.7423e-02, 1.5869e-04, 1.2531e-04, 5.2495e-04, 2.7884e-04, 1.0082e-05,
        6.4811e-04, 8.2781e-04, 5.2042e-04, 9.1544e-05, 8.0778e-05, 5.4748e-04,
        1.0982e-03, 3.4647e-04])
```

```
max probability is torch.return_types.max(
values=tensor(0.3488),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5072e-05, 1.9240e-06, 1.9239e-06, 1.9238e-06]),
indices=tensor([  0, 316, 910, 466]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 175 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.8734e-03, 2.0065e-03, 3.3492e-03,
1.2495e-03, 5.5137e-03, 4.0748e-03,
        2.9542e-03, 6.3211e-03, 1.8115e-02, 5.8225e-03, 4.8579e-02, 1.3509e-03,
        6.8774e-03, 9.2315e-02, 1.5416e-04, 1.2630e-02, 1.7357e-01, 1.2567e-02,
        1.1472e-01, 4.7423e-01])

max probability is torch.return_types.max(
values=tensor(0.4742),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.2506e-03, 3.6456e-06, 3.6442e-06, 3.6424e-06]),
indices=tensor([  0, 914, 771, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 176 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.2916e-04, 9.1471e-04, 1.0891e-01,
2.4568e-01, 4.5895e-01, 1.4014e-01,
        4.2097e-02, 4.0705e-05, 4.0251e-05, 1.4828e-04, 9.4824e-05, 4.1584e-06,
        7.7349e-05, 3.0941e-04, 1.6434e-04, 2.9742e-05, 1.3425e-05, 1.2486e-04,
        9.8791e-04, 1.8635e-04])

max probability is torch.return_types.max(
values=tensor(0.4590),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1165e-05, 5.8229e-07, 5.8207e-07, 5.8163e-07]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************
```

```
**************************************************************************
Predicting Test Sample 177 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.8342e-04, 3.0825e-04, 5.0585e-02,
2.3059e-01, 4.1343e-01, 2.4805e-01,
        5.5745e-02, 1.0118e-05, 1.3956e-05, 7.7913e-05, 2.2021e-05, 8.2088e-07,
        1.6853e-05, 1.2183e-04, 8.3460e-05, 8.1566e-06, 1.7843e-06, 5.2369e-05,
        3.8929e-04, 5.7387e-05])

max probability is torch.return_types.max(
values=tensor(0.4134),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3068e-06, 5.0487e-08, 5.0453e-08, 5.0413e-08]),
indices=tensor([  0, 319, 316, 794]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 178 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.1315e-03, 1.4885e-02, 2.2981e-03,
1.5307e-03, 8.4848e-03, 3.0562e-01,
        6.2618e-02, 2.5361e-02, 4.8578e-01, 1.7107e-02, 1.3311e-02, 3.1716e-04,
        2.7892e-02, 1.1582e-02, 9.9390e-05, 7.3765e-03, 2.6153e-03, 2.4957e-04,
        2.8244e-03, 2.1412e-03])

max probability is torch.return_types.max(
values=tensor(0.4858),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.8652e-05, 1.8426e-06, 1.8412e-06, 1.8407e-06]),
indices=tensor([  0, 319, 896, 316]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 179 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.5986e-02, 4.0453e-05, 3.1316e-03,
1.9588e-03, 4.2308e-03, 2.0000e-03,
        3.7369e-03, 4.8105e-03, 1.7047e-02, 8.1311e-01, 1.5343e-03, 5.3606e-04,
        9.1402e-03, 2.3601e-02, 3.6480e-03, 1.5299e-03, 5.6840e-03, 1.2034e-02,
        1.3573e-02, 2.1868e-03])
```

```
max probability is torch.return_types.max(
values=tensor(0.8131),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.7449e-06, 4.9590e-07, 4.9582e-07, 4.9555e-07]),
indices=tensor([  0, 323, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 180 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0044, 0.0024, 0.0325, 0.0531, 0.0391,
0.1975, 0.2800, 0.0028, 0.0112,
        0.2899, 0.0281, 0.0006, 0.0170, 0.0293, 0.0008, 0.0028, 0.0029, 0.0010,
        0.0020, 0.0018])

max probability is torch.return_types.max(
values=tensor(0.2899),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5620e-05, 1.0987e-06, 1.0982e-06, 1.0981e-06]),
indices=tensor([  0, 771, 319, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 181 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1406e-03, 2.6742e-05, 2.8204e-04,
1.3899e-04, 8.8035e-04, 9.5815e-05,
        5.9935e-05, 8.0328e-04, 1.9954e-03, 1.7395e-03, 3.2296e-03, 8.3543e-05,
        4.8067e-04, 5.3676e-02, 1.3601e-04, 6.5750e-04, 6.7120e-03, 3.4014e-02,
        6.7962e-01, 2.1365e-01])

max probability is torch.return_types.max(
values=tensor(0.6796),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.2574e-04, 5.2541e-08, 5.2531e-08, 5.2519e-08]),
indices=tensor([  0, 914, 337, 158]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 182 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.8411e-03, 8.9067e-05, 1.3669e-03,
8.7328e-04, 3.6216e-03, 1.3111e-03,
        2.1665e-03, 9.6345e-04, 1.5560e-03, 8.3067e-03, 2.1155e-03, 1.5553e-02,
        1.1755e-02, 6.3598e-02, 5.5699e-01, 4.1186e-03, 5.6834e-02, 1.8729e-01,
        7.0363e-02, 6.6584e-03])

max probability is torch.return_types.max(
values=tensor(0.5570),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.2762e-05, 2.6895e-06, 2.6879e-06, 2.6879e-06]),
indices=tensor([  0, 748, 230,  82]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 183 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.8019e-05, 5.7423e-05, 1.0078e-03,
7.9274e-04, 2.2614e-03, 3.8451e-04,
        3.6885e-04, 4.1879e-04, 9.8911e-04, 1.2945e-03, 2.1990e-03, 1.2047e-05,
        4.9719e-04, 8.3170e-02, 4.3669e-06, 1.0730e-03, 1.0756e-01, 7.6448e-04,
        2.4947e-02, 7.6784e-01])

max probability is torch.return_types.max(
values=tensor(0.7678),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.1997e-03, 6.4043e-08, 6.4034e-08, 6.4031e-08]),
indices=tensor([  0, 914, 337, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 184 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.9520e-03, 2.3031e-03, 4.8130e-01,
4.4060e-02, 1.0218e-01, 3.4347e-02,
        4.0912e-02, 6.5912e-03, 1.9246e-02, 1.2287e-02, 1.2878e-02, 3.1152e-04,
        3.7638e-02, 8.0409e-02, 5.9640e-04, 5.5147e-03, 1.8220e-02, 9.9021e-03,
        2.0712e-02, 5.6970e-02])

max probability is torch.return_types.max(
values=tensor(0.4813),
```

```
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.7648e-04, 6.0780e-06, 6.0768e-06, 6.0765e-06]),
indices=tensor([  0, 914, 910, 726]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 185 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0101, 0.0041, 0.1410, 0.0539, 0.0678,
0.0385, 0.0448, 0.0081, 0.0093,
        0.0244, 0.0136, 0.0103, 0.0234, 0.0627, 0.0638, 0.0063, 0.0204, 0.0864,
        0.0511, 0.0280])

max probability is torch.return_types.max(
values=tensor(0.1410),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([0.0020, 0.0002, 0.0002, 0.0002]),
indices=tensor([  0, 748, 150, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 186 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.3189e-03, 3.8966e-03, 4.4928e-04,
5.8008e-04, 3.6781e-03, 4.5246e-01,
        2.2627e-02, 5.2360e-03, 3.0678e-01, 1.3008e-02, 6.5901e-02, 3.2813e-04,
        4.1403e-02, 4.9494e-02, 3.1211e-05, 8.7437e-03, 3.7174e-03, 2.6339e-04,
        4.2629e-03, 7.6070e-03])

max probability is torch.return_types.max(
values=tensor(0.4525),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7655e-05, 1.9955e-07, 1.9940e-07, 1.9930e-07]),
indices=tensor([  0, 319, 771, 144]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 187 :   Prediction is Correct?
Yes
```

```
first 20 classes probability: tensor([0.0223, 0.0030, 0.0066, 0.0112, 0.0336,
0.4111, 0.1183, 0.0037, 0.0415,
        0.1071, 0.0726, 0.0018, 0.0302, 0.0638, 0.0015, 0.0200, 0.0094, 0.0042,
        0.0150, 0.0143])

max probability is torch.return_types.max(
values=tensor(0.4111),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6300e-04, 9.0988e-06, 9.0983e-06, 9.0919e-06]),
indices=tensor([  0, 319, 771, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 188 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.7190e-04, 6.6590e-05, 6.6847e-04,
3.2029e-04, 2.3219e-03, 1.2004e-04,
        7.2267e-05, 2.2366e-03, 1.9388e-03, 1.1553e-03, 1.2065e-03, 2.7528e-05,
        2.6364e-04, 5.5039e-03, 7.0472e-05, 7.3472e-04, 4.9493e-03, 5.1552e-03,
        8.4773e-01, 1.2405e-01])

max probability is torch.return_types.max(
values=tensor(0.8477),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.6224e-04, 7.8345e-08, 7.8338e-08, 7.8330e-08]),
indices=tensor([  0, 337, 197, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 189 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.7045e-02, 6.5794e-04, 7.3866e-03,
2.9403e-03, 1.0622e-02, 7.2392e-03,
        5.8242e-03, 6.6413e-04, 4.4273e-03, 7.2800e-03, 8.7259e-02, 2.2165e-03,
        2.5482e-02, 5.9092e-01, 5.1270e-04, 1.1473e-02, 2.5434e-02, 2.7139e-02,
        1.7414e-02, 9.6890e-02])

max probability is torch.return_types.max(
values=tensor(0.5909),
indices=tensor(13))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([2.1097e-04, 1.0111e-06, 1.0108e-06, 1.0107e-06]),
indices=tensor([  0, 771, 914, 323]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 190 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.6741e-05, 4.1571e-04, 1.6422e-01,
1.9562e-01, 5.7517e-01, 5.2307e-02,
        1.1783e-02, 7.4422e-06, 4.8725e-06, 1.1547e-05, 1.3068e-05, 3.3777e-07,
        1.5422e-05, 2.6986e-05, 2.0026e-05, 1.4080e-06, 1.5930e-06, 1.9908e-05,
        2.0996e-04, 2.1971e-05])

max probability is torch.return_types.max(
values=tensor(0.5752),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2948e-06, 3.1793e-08, 3.1788e-08, 3.1768e-08]),
indices=tensor([  0, 319, 316, 957]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 191 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.1273e-05, 1.4018e-06, 8.5842e-04,
1.0320e-03, 2.2167e-03, 2.2569e-04,
        2.3918e-04, 1.4545e-02, 9.0026e-03, 7.5437e-01, 1.0589e-04, 9.4729e-07,
        9.3416e-05, 6.9079e-03, 1.2671e-05, 1.0786e-04, 1.0272e-03, 1.0892e-04,
        2.0279e-01, 6.3289e-03])

max probability is torch.return_types.max(
values=tensor(0.7544),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.4171e-06, 1.3937e-09, 1.3929e-09, 1.3928e-09]),
indices=tensor([  0,  91, 771, 914]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 192 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.5074e-03, 1.1159e-03, 2.0329e-02,
1.0935e-01, 2.1024e-01, 1.7758e-01,
```

```
        4.1056e-01, 9.7136e-04, 2.0607e-03, 1.8311e-02, 1.7770e-03, 1.7196e-04,
        2.2545e-03, 2.2535e-02, 3.9789e-03, 1.6321e-03, 7.4818e-04, 2.1412e-03,
        6.4121e-03, 2.0645e-03])

max probability is torch.return_types.max(
values=tensor(0.4106),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.2280e-05, 4.4122e-06, 4.4121e-06, 4.4119e-06]),
indices=tensor([  0, 910, 319, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 193 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.1557, 0.0082, 0.0335, 0.0848, 0.0933,
0.4225, 0.1262, 0.0013, 0.0035,
        0.0091, 0.0052, 0.0013, 0.0041, 0.0237, 0.0012, 0.0013, 0.0011, 0.0025,
        0.0047, 0.0071])

max probability is torch.return_types.max(
values=tensor(0.4225),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5369e-04, 1.0201e-05, 1.0198e-05, 1.0197e-05]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 194 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.2012e-09, 1.0000e+00, 1.5462e-07,
2.6371e-08, 1.6734e-07, 2.3484e-08,
        2.3254e-08, 1.2891e-08, 1.3844e-09, 7.3546e-12, 2.2458e-09, 1.8966e-11,
        6.8364e-08, 9.1226e-11, 1.0366e-11, 2.5928e-09, 1.0769e-09, 3.2218e-13,
        4.2940e-11, 4.4922e-12])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6260e-15, 3.5585e-19, 3.5545e-19, 3.5489e-19]),
```

```
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 195 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.9786e-03, 7.0405e-01, 3.0418e-03,
6.7623e-04, 3.9220e-03, 4.1270e-03,
        4.6681e-03, 6.2946e-02, 4.5835e-02, 7.1807e-04, 1.7608e-02, 4.0699e-03,
        5.8806e-02, 8.8098e-03, 8.6006e-04, 2.0750e-02, 3.0354e-02, 8.9397e-04,
        5.6798e-03, 8.0044e-03])

max probability is torch.return_types.max(
values=tensor(0.7041),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.7893e-04, 6.1512e-06, 6.1510e-06, 6.1495e-06]),
indices=tensor([  0, 910, 316, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 196 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.7631e-01, 2.6276e-05, 2.5152e-03,
3.4571e-04, 1.7263e-03, 3.9568e-04,
        4.7788e-04, 7.3346e-04, 2.4703e-03, 7.4148e-03, 7.3285e-05, 3.6092e-05,
        6.3302e-04, 1.2896e-03, 9.1933e-05, 3.1899e-05, 1.8575e-04, 2.8048e-03,
        1.8567e-03, 5.6153e-04])

max probability is torch.return_types.max(
values=tensor(0.9763),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2341e-06, 2.0235e-08, 2.0231e-08, 2.0229e-08]),
indices=tensor([  0, 323, 910, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 197 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.8800e-01, 1.2804e-05, 1.8372e-03,
2.0920e-04, 6.1366e-04, 3.1089e-04,
        3.7222e-04, 1.2032e-04, 8.2183e-04, 5.7006e-03, 4.2249e-05, 1.0110e-05,
        7.9533e-04, 4.7490e-04, 4.6415e-05, 1.6763e-05, 5.1280e-05, 4.0940e-04,
```

```
        8.2856e-05, 6.2349e-05])

max probability is torch.return_types.max(
values=tensor(0.9880),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0611e-07, 6.6444e-09, 6.6438e-09, 6.6437e-09]),
indices=tensor([  0, 771, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 198 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.0186e-04, 1.0603e-03, 1.5940e-01,
2.1409e-01, 4.7867e-01, 1.0578e-01,
        3.7323e-02, 6.3797e-05, 6.5832e-05, 1.4685e-04, 1.1271e-04, 5.4920e-06,
        1.3767e-04, 2.5743e-04, 1.6612e-04, 2.8342e-05, 1.8682e-05, 1.4873e-04,
        9.2866e-04, 1.5417e-04])

max probability is torch.return_types.max(
values=tensor(0.4787),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3283e-05, 8.6695e-07, 8.6675e-07, 8.6615e-07]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 199 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.8689e-04, 5.8016e-04, 3.9552e-03,
4.2947e-03, 1.9986e-02, 4.8138e-03,
        3.8451e-03, 1.3528e-02, 1.8157e-02, 6.7102e-03, 4.3875e-03, 1.2513e-04,
        1.3774e-03, 2.3247e-02, 7.9766e-05, 1.1797e-03, 1.7651e-02, 6.4821e-03,
        2.9951e-01, 5.5855e-01])

max probability is torch.return_types.max(
values=tensor(0.5585),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.6686e-03, 1.1428e-06, 1.1425e-06, 1.1422e-06]),
indices=tensor([  0, 914, 337, 197]))
```

```
End**********************************************************************

currently at 200 current time is 6.324920654296875
**********************************************************************
Predicting Test Sample 200 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.3083e-03, 5.6590e-03, 1.9683e-03,
8.7864e-04, 3.4439e-03, 4.8427e-02,
        7.8094e-03, 4.5255e-03, 4.8107e-02, 3.7799e-03, 1.4714e-01, 8.4735e-03,
        5.7055e-01, 6.4375e-02, 3.1471e-04, 1.7375e-02, 4.8304e-02, 4.1730e-04,
        3.1402e-03, 7.7542e-03])

max probability is torch.return_types.max(
values=tensor(0.5706),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.3986e-05, 1.2516e-06, 1.2513e-06, 1.2511e-06]),
indices=tensor([  0, 771,  44, 319]))
End**********************************************************************

**********************************************************************
Predicting Test Sample 201 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.6593e-04, 5.0646e-03, 2.3772e-04,
4.4070e-05, 1.8073e-04, 1.8435e-04,
        3.2398e-04, 1.1649e-02, 1.4456e-02, 2.7168e-05, 1.2495e-02, 2.8571e-03,
        9.5031e-01, 4.4753e-04, 1.4277e-04, 8.5803e-04, 3.8587e-04, 2.8860e-05,
        5.1798e-05, 4.9683e-05])

max probability is torch.return_types.max(
values=tensor(0.9503),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.1600e-07, 4.7144e-08, 4.7143e-08, 4.7141e-08]),
indices=tensor([  0,  44, 319, 836]))
End**********************************************************************

**********************************************************************
Predicting Test Sample 202 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.6132e-01, 9.1251e-04, 2.1676e-03,
3.7274e-04, 1.1451e-03, 1.9873e-03,
        2.5241e-03, 4.8797e-03, 1.7469e-02, 2.1247e-03, 1.8464e-04, 9.5268e-05,
        3.6293e-03, 4.0441e-04, 1.3997e-04, 6.7466e-05, 5.1596e-05, 2.8899e-04,
```

```
        7.8592e-05, 7.9223e-05])

max probability is torch.return_types.max(
values=tensor(0.9613),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.3376e-07, 7.8422e-08, 7.8389e-08, 7.8384e-08]),
indices=tensor([  0, 319, 910, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 203 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0468, 0.0014, 0.0044, 0.0018, 0.0076,
0.0060, 0.0038, 0.0064, 0.0264,
        0.0219, 0.0607, 0.0183, 0.1233, 0.2912, 0.0033, 0.0365, 0.2020, 0.0277,
        0.0377, 0.0631])

max probability is torch.return_types.max(
values=tensor(0.2912),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.4096e-04, 9.6937e-06, 9.6937e-06, 9.6923e-06]),
indices=tensor([  0, 771, 323, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 204 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.8049e-09, 4.8279e-09, 1.1631e-08,
2.9921e-09, 5.7324e-08, 3.3604e-10,
        1.7131e-10, 3.7319e-08, 2.8444e-08, 1.9323e-09, 1.5696e-06, 6.2870e-11,
        1.6111e-08, 1.2494e-05, 1.6324e-13, 1.0451e-08, 4.5375e-04, 2.3177e-08,
        2.4983e-04, 9.9912e-01])

max probability is torch.return_types.max(
values=tensor(0.9991),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5983e-04, 3.1422e-16, 3.1421e-16, 3.1375e-16]),
indices=tensor([  0, 337, 655, 691]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 205 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9992e-01, 9.2797e-09, 2.5305e-05,
3.9251e-07, 4.2207e-06, 1.5598e-07,
        5.4220e-07, 2.6119e-07, 3.8210e-06, 2.9324e-05, 5.5883e-08, 1.0140e-08,
        9.2325e-06, 2.8647e-06, 2.5551e-08, 1.5780e-08, 8.5938e-08, 1.9421e-06,
        2.2737e-07, 1.5574e-07])

max probability is torch.return_types.max(
values=tensor(0.9999),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.1773e-12, 4.1187e-14, 4.1173e-14, 4.1142e-14]),
indices=tensor([  0, 910, 323, 771]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 206 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9697e-01, 1.2801e-05, 2.0156e-04,
1.6344e-05, 1.7221e-04, 8.9039e-05,
        9.8109e-05, 4.7587e-05, 6.0743e-04, 3.5111e-04, 5.5659e-05, 1.7762e-05,
        6.9122e-04, 2.3614e-04, 1.2132e-05, 2.0282e-05, 2.7464e-05, 2.3743e-04,
        6.5047e-05, 7.1502e-05])

max probability is torch.return_types.max(
values=tensor(0.9970),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.1712e-08, 2.5649e-09, 2.5644e-09, 2.5641e-09]),
indices=tensor([  0, 910, 323, 771]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 207 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.5318e-04, 3.1093e-04, 5.2380e-03,
4.6018e-03, 7.4327e-03, 1.9802e-03,
        8.5989e-04, 2.9038e-03, 6.6747e-03, 7.0214e-03, 8.0584e-02, 5.2377e-04,
        7.1440e-02, 5.2486e-01, 2.6056e-04, 2.2293e-02, 6.8859e-02, 2.4015e-03,
        6.8042e-02, 1.2198e-01])
```

```
max probability is torch.return_types.max(
values=tensor(0.5249),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.9649e-04, 8.3198e-07, 8.3167e-07, 8.3130e-07]),
indices=tensor([  0, 771, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 208 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.5697e-03, 5.7893e-04, 1.6073e-02,
1.1974e-02, 6.8165e-02, 6.2520e-01,
        2.2187e-01, 2.1046e-04, 2.4404e-03, 6.5778e-03, 4.4559e-03, 2.5039e-04,
        6.1924e-03, 1.8288e-02, 8.7303e-04, 3.2070e-03, 1.5380e-03, 8.4885e-04,
        5.3782e-03, 1.9288e-03])

max probability is torch.return_types.max(
values=tensor(0.6252),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1852e-05, 3.9586e-07, 3.9577e-07, 3.9576e-07]),
indices=tensor([  0, 771, 896, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 209 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.0276e-04, 7.8705e-04, 8.9881e-02,
2.1226e-01, 5.0421e-01, 1.6009e-01,
        3.0906e-02, 1.4914e-05, 1.3283e-05, 4.9678e-05, 3.8472e-05, 2.2514e-06,
        3.2933e-05, 8.9882e-05, 1.2355e-04, 9.6806e-06, 4.2458e-06, 9.5992e-05,
        6.7223e-04, 8.6969e-05])

max probability is torch.return_types.max(
values=tensor(0.5042),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5162e-06, 1.3107e-07, 1.3102e-07, 1.3093e-07]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 210 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.5274e-02, 1.0786e-02, 7.7057e-04,
6.9630e-05, 2.4122e-04, 5.7924e-04,
        5.1954e-04, 4.7526e-01, 4.7065e-01, 6.6486e-04, 2.3581e-04, 6.5472e-05,
        1.4473e-02, 7.8278e-05, 2.2295e-05, 2.0624e-04, 5.2778e-05, 6.3818e-06,
        2.3341e-05, 1.0986e-05])

max probability is torch.return_types.max(
values=tensor(0.4753),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2938e-07, 1.2088e-08, 1.2082e-08, 1.2075e-08]),
indices=tensor([  0, 836, 319, 794]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 211 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.8731e-02, 9.8860e-04, 2.3807e-02,
2.0057e-02, 4.4133e-02, 2.1024e-01,
        8.8312e-02, 5.4146e-03, 4.2015e-02, 4.3310e-01, 7.3284e-03, 2.9625e-04,
        1.3733e-02, 1.9287e-02, 1.0882e-03, 7.7777e-03, 4.2212e-03, 3.5099e-03,
        8.6550e-03, 5.3639e-03])

max probability is torch.return_types.max(
values=tensor(0.4331),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.2319e-05, 2.0220e-06, 2.0217e-06, 2.0212e-06]),
indices=tensor([  0, 771, 319, 896]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 212 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.8410e-04, 1.1910e-04, 1.2762e-04,
7.7459e-05, 3.3401e-04, 9.2918e-05,
        6.2263e-05, 1.2148e-03, 6.3556e-04, 3.8227e-04, 6.0256e-03, 3.9646e-02,
        8.8596e-04, 1.1387e-02, 8.2505e-04, 5.9930e-04, 1.0273e-01, 3.5341e-02,
        3.7425e-01, 4.2038e-01])

max probability is torch.return_types.max(
```

```
values=tensor(0.4204),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.0552e-03, 2.5387e-07, 2.5387e-07, 2.5381e-07]),
indices=tensor([  0, 691, 230,  19]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 213 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.1077e-03, 5.6282e-04, 1.1235e-02,
4.5146e-02, 9.5640e-02, 3.7205e-01,
        3.3793e-01, 1.2991e-03, 7.3988e-03, 7.4285e-02, 6.7011e-03, 1.3728e-04,
        3.8601e-03, 2.6828e-02, 4.7874e-04, 2.5495e-03, 1.4304e-03, 8.5427e-04,
        4.8630e-03, 3.9008e-03])

max probability is torch.return_types.max(
values=tensor(0.3721),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8023e-05, 7.5578e-07, 7.5561e-07, 7.5550e-07]),
indices=tensor([  0, 319, 771, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 214 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0019, 0.0018, 0.0016, 0.0006, 0.0031,
0.0089, 0.0101, 0.1647, 0.4828,
        0.0046, 0.0166, 0.0051, 0.2712, 0.0078, 0.0019, 0.0030, 0.0029, 0.0031,
        0.0043, 0.0024])

max probability is torch.return_types.max(
values=tensor(0.4828),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.0264e-05, 1.9360e-06, 1.9356e-06, 1.9345e-06]),
indices=tensor([  0,  44, 717, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 215 :   Prediction is Correct?
```

```
No
first 20 classes probability: tensor([0.0093, 0.0054, 0.0254, 0.0338, 0.0871,
0.2853, 0.3549, 0.0036, 0.0116,
        0.0124, 0.0032, 0.0019, 0.0211, 0.0070, 0.0828, 0.0014, 0.0007, 0.0431,
        0.0045, 0.0015])

max probability is torch.return_types.max(
values=tensor(0.3549),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3114e-05, 4.1228e-06, 4.1225e-06, 4.1220e-06]),
indices=tensor([  0, 319, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 216 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.6870e-04, 6.0697e-04, 1.5658e-02,
1.7460e-01, 2.6702e-01, 3.3549e-01,
        2.0318e-01, 3.8045e-05, 7.7009e-05, 4.6328e-04, 8.2729e-05, 3.7684e-06,
        8.1914e-05, 8.1381e-04, 2.1605e-04, 5.3322e-05, 9.8389e-06, 8.3352e-05,
        6.3479e-04, 1.6073e-04])

max probability is torch.return_types.max(
values=tensor(0.3355),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5122e-06, 1.7444e-07, 1.7431e-07, 1.7416e-07]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 217 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0267, 0.0069, 0.0633, 0.0118, 0.0251,
0.0115, 0.0180, 0.0472, 0.0470,
        0.0488, 0.0978, 0.0448, 0.0533, 0.0412, 0.0017, 0.0238, 0.2376, 0.0113,
        0.0491, 0.0464])

max probability is torch.return_types.max(
values=tensor(0.2376),
indices=tensor(16))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([2.9543e-03, 8.7120e-05, 8.7117e-05, 8.7117e-05]),
indices=tensor([  0, 323, 914, 158]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 218 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.3361e-04, 1.4641e-04, 1.6577e-05,
7.9494e-06, 1.8728e-05, 8.1937e-05,
        4.9236e-05, 1.8212e-03, 5.3018e-04, 1.5555e-04, 1.7002e-03, 9.7412e-01,
        7.0360e-03, 3.8624e-04, 4.6065e-04, 1.5528e-04, 1.1589e-02, 1.5205e-04,
        9.4058e-04, 2.6472e-04])

max probability is torch.return_types.max(
values=tensor(0.9741),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6504e-06, 3.6133e-08, 3.6130e-08, 3.6120e-08]),
indices=tensor([  0, 914, 323, 391]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 219 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.6656e-04, 3.1336e-04, 1.1853e-01,
2.2305e-01, 5.3314e-01, 9.8468e-02,
        2.5542e-02, 1.1079e-05, 1.0819e-05, 4.1519e-05, 2.7296e-05, 6.1541e-07,
        2.4163e-05, 8.3314e-05, 4.6076e-05, 6.3745e-06, 2.5083e-06, 4.0286e-05,
        3.7932e-04, 5.0041e-05])

max probability is torch.return_types.max(
values=tensor(0.5331),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0291e-06, 6.9961e-08, 6.9929e-08, 6.9871e-08]),
indices=tensor([  0, 319, 316,  85]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 220 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.2074e-03, 2.0045e-03, 4.9570e-01,
1.7033e-01, 2.1655e-01, 1.9835e-02,
```

```
        2.3751e-02, 1.7413e-03, 1.4977e-03, 4.3346e-03, 3.7229e-03, 1.6764e-04,
        9.3768e-03, 1.1363e-02, 1.6446e-03, 1.4018e-03, 2.8490e-03, 3.0311e-03,
        5.9694e-03, 3.8381e-03])

max probability is torch.return_types.max(
values=tensor(0.4957),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9612e-04, 1.9215e-05, 1.9214e-05, 1.9209e-05]),
indices=tensor([  0, 910, 466, 726]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 221 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.0247e-04, 4.5964e-04, 2.7211e-02,
1.4953e-01, 3.3203e-01, 4.1585e-01,
        7.2873e-02, 1.3673e-05, 2.4897e-05, 9.0178e-05, 3.3578e-05, 2.6363e-06,
        3.1752e-05, 1.3661e-04, 1.6709e-04, 1.2717e-05, 2.7051e-06, 8.2047e-05,
        5.0121e-04, 6.5788e-05])

max probability is torch.return_types.max(
values=tensor(0.4159),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5466e-06, 8.2257e-08, 8.2208e-08, 8.2167e-08]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 222 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.5601e-06, 1.2319e-06, 3.7922e-05,
6.9224e-05, 3.7877e-04, 1.7129e-05,
        1.9000e-05, 2.1648e-04, 2.0643e-04, 2.4490e-04, 1.5198e-04, 4.6713e-07,
        7.7662e-06, 1.7703e-03, 7.7234e-07, 1.0281e-05, 1.4348e-03, 2.0871e-04,
        3.9441e-01, 5.9797e-01])

max probability is torch.return_types.max(
values=tensor(0.5980),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([2.8407e-03, 9.1698e-10, 9.1636e-10, 9.1635e-10]),
indices=tensor([  0, 337, 655, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 223 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1362e-03, 4.5739e-03, 3.5141e-04,
1.3061e-04, 8.9462e-04, 1.8433e-03,
        1.7775e-03, 3.7654e-01, 5.9560e-01, 1.4763e-03, 1.6768e-03, 3.9494e-05,
        1.0374e-02, 1.0818e-03, 5.8146e-05, 1.2175e-03, 1.7451e-04, 1.1177e-04,
        6.0128e-04, 2.1266e-04])

max probability is torch.return_types.max(
values=tensor(0.5956),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.3626e-06, 1.3298e-07, 1.3296e-07, 1.3295e-07]),
indices=tensor([  0, 836, 319, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 224 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7350e-03, 3.2789e-03, 5.3520e-04,
1.1513e-04, 5.7542e-04, 1.7454e-04,
        4.8108e-04, 6.9850e-01, 2.8294e-01, 3.7885e-03, 3.6572e-04, 7.5135e-05,
        6.2922e-03, 2.1586e-04, 4.5284e-05, 1.2962e-04, 2.7294e-04, 2.9318e-05,
        3.6872e-04, 4.4876e-05])

max probability is torch.return_types.max(
values=tensor(0.6985),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.5087e-07, 4.0877e-08, 4.0865e-08, 4.0852e-08]),
indices=tensor([  0, 316, 319, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 225 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.1293e-08, 1.7600e-10, 1.8356e-08,
8.4090e-09, 1.3774e-07, 1.5655e-09,
        1.8864e-09, 2.2920e-08, 4.5236e-08, 6.3647e-08, 8.5582e-08, 3.6421e-08,
```

```
        3.1747e-08, 1.9797e-04, 4.3430e-03, 2.4967e-07, 1.2500e-05, 9.3144e-01,
        6.2906e-02, 1.0993e-03])

max probability is torch.return_types.max(
values=tensor(0.9314),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.2603e-09, 2.0121e-15, 2.0119e-15, 2.0105e-15]),
indices=tensor([  0, 748, 337, 230]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 226 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.4194e-07, 7.3027e-11, 1.1168e-08,
2.7039e-09, 7.9792e-08, 1.4846e-09,
        3.2573e-09, 5.1336e-09, 2.5103e-08, 5.1691e-08, 4.5366e-08, 1.2482e-08,
        2.8215e-08, 5.2506e-05, 1.8525e-03, 1.1676e-08, 2.4452e-07, 9.9676e-01,
        1.3069e-03, 2.6299e-05])

max probability is torch.return_types.max(
values=tensor(0.9968),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.4397e-11, 3.7304e-16, 3.7276e-16, 3.7263e-16]),
indices=tensor([  0, 748, 337, 500]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 227 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0128, 0.0131, 0.0134, 0.0026, 0.0092,
0.0127, 0.0114, 0.2411, 0.4612,
        0.0251, 0.0144, 0.0015, 0.1269, 0.0092, 0.0008, 0.0151, 0.0091, 0.0011,
        0.0046, 0.0033])

max probability is torch.return_types.max(
values=tensor(0.4612),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2402e-04, 1.1775e-05, 1.1775e-05, 1.1773e-05]),
indices=tensor([  0, 910, 316, 896]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 228 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.6519e-06, 3.8051e-05, 1.2374e-05,
1.1143e-06, 9.7811e-06, 2.9140e-05,
        4.1145e-05, 2.6488e-04, 3.3020e-03, 1.7976e-06, 3.1594e-03, 2.3368e-05,
        9.9292e-01, 6.4167e-05, 1.6280e-06, 1.1160e-04, 6.3449e-06, 4.5731e-07,
        5.8289e-07, 6.4812e-07])

max probability is torch.return_types.max(
values=tensor(0.9929),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.6695e-10, 2.4411e-11, 2.4391e-11, 2.4386e-11]),
indices=tensor([  0, 319,  44, 836]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 229 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0825e-03, 7.1037e-05, 1.7550e-04,
3.2994e-04, 7.6074e-04, 6.4697e-04,
        6.8804e-04, 3.9246e-04, 3.0056e-04, 1.2073e-03, 1.8132e-04, 2.4371e-03,
        2.4526e-03, 1.1264e-03, 9.2416e-01, 2.3322e-04, 3.6905e-04, 5.3813e-02,
        8.2239e-03, 5.3492e-04])

max probability is torch.return_types.max(
values=tensor(0.9242),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.8190e-06, 8.4968e-07, 8.4964e-07, 8.4937e-07]),
indices=tensor([  0, 748, 337, 323]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 230 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.0641e-05, 1.8578e-06, 4.4994e-05,
2.9782e-04, 2.6315e-04, 1.5081e-04,
        1.0077e-04, 2.4080e-04, 4.0846e-05, 1.2677e-03, 1.9550e-05, 1.0043e-03,
        1.3548e-04, 7.7582e-04, 9.0082e-01, 2.6728e-05, 2.2953e-04, 1.2043e-02,
        8.1794e-02, 7.1310e-04])
```

```
max probability is torch.return_types.max(
values=tensor(0.9008),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.6911e-07, 8.8350e-09, 8.8340e-09, 8.8333e-09]),
indices=tensor([  0, 748, 337, 230]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 231 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.1887e-03, 8.3012e-01, 1.5802e-02,
8.4395e-03, 5.4189e-03, 1.3073e-02,
        1.9789e-02, 7.0501e-03, 6.9437e-03, 2.1074e-03, 6.7152e-03, 1.7381e-03,
        7.4650e-02, 6.8437e-04, 5.1857e-04, 5.0905e-04, 5.1343e-04, 8.5614e-05,
        1.2987e-04, 7.9551e-05])

max probability is torch.return_types.max(
values=tensor(0.8301),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.7734e-06, 4.7291e-07, 4.7252e-07, 4.7249e-07]),
indices=tensor([  0, 319, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 232 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.1077e-03, 1.0474e-02, 5.5390e-04,
3.2985e-04, 1.0508e-03, 1.3201e-03,
        1.1075e-03, 5.0241e-02, 4.6105e-02, 2.7281e-03, 4.9358e-01, 5.5479e-02,
        1.4106e-01, 1.4303e-02, 5.8546e-04, 9.9327e-02, 3.7565e-02, 1.4906e-03,
        1.2789e-02, 1.6481e-02])

max probability is torch.return_types.max(
values=tensor(0.4936),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.6668e-04, 1.1504e-05, 1.1501e-05, 1.1500e-05]),
indices=tensor([  0, 319, 836, 316]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 233 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.2718e-03, 2.1191e-02, 7.4523e-05,
1.7982e-05, 1.7061e-04, 6.0547e-05,
        8.7657e-05, 8.5195e-01, 1.1498e-01, 3.3089e-04, 7.7611e-04, 1.2337e-03,
        4.6685e-03, 9.2881e-05, 6.0532e-05, 2.7879e-04, 1.0162e-03, 2.4289e-05,
        1.5168e-03, 1.6701e-04])

max probability is torch.return_types.max(
values=tensor(0.8519),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.7298e-06, 3.4918e-08, 3.4893e-08, 3.4888e-08]),
indices=tensor([  0, 316, 391, 294]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 234 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.1126e-02, 3.7457e-03, 1.2078e-03,
1.9118e-04, 1.0959e-03, 9.2093e-04,
        1.7210e-03, 4.7647e-01, 4.5703e-01, 4.1686e-03, 2.0625e-03, 1.3553e-03,
        3.1394e-02, 1.1870e-03, 1.2076e-04, 8.8356e-04, 2.3694e-03, 2.7356e-04,
        1.6449e-03, 5.1198e-04])

max probability is torch.return_types.max(
values=tensor(0.4765),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3285e-05, 5.3219e-07, 5.3209e-07, 5.3206e-07]),
indices=tensor([  0, 910, 316, 957]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 235 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.2408e-03, 3.7884e-03, 7.4376e-04,
9.9571e-05, 8.1244e-04, 6.4077e-04,
        3.1348e-04, 8.7174e-04, 7.2297e-03, 3.4095e-04, 3.2864e-01, 3.1412e-04,
        3.8086e-02, 4.1147e-01, 2.5685e-06, 5.0554e-02, 1.2128e-01, 2.7519e-04,
        1.7575e-03, 2.9480e-02])
```

```
max probability is torch.return_types.max(
values=tensor(0.4115),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.8676e-05, 2.6885e-08, 2.6882e-08, 2.6876e-08]),
indices=tensor([  0, 771, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 236 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.2543e-04, 1.1302e-04, 5.5436e-04,
9.6591e-04, 6.8243e-03, 1.7631e-03,
        7.2716e-04, 3.3806e-03, 5.8428e-03, 5.0933e-03, 4.8918e-03, 1.1269e-04,
        5.7892e-04, 2.3371e-02, 1.7467e-04, 1.4234e-03, 1.9069e-03, 8.3535e-03,
        8.0805e-01, 1.2429e-01])

max probability is torch.return_types.max(
values=tensor(0.8081),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.9024e-04, 2.7196e-07, 2.7186e-07, 2.7186e-07]),
indices=tensor([  0, 914, 771, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 237 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.4320e-05, 9.9964e-01, 2.3141e-05,
1.0406e-05, 3.1521e-05, 7.2511e-05,
        8.0568e-05, 7.9947e-06, 5.5160e-06, 4.5060e-07, 4.5725e-05, 7.6809e-07,
        2.8071e-05, 1.4471e-06, 3.0561e-08, 8.5740e-06, 4.2525e-06, 1.0880e-08,
        3.3679e-07, 1.9588e-07])

max probability is torch.return_types.max(
values=tensor(0.9996),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1484e-09, 4.7759e-12, 4.7664e-12, 4.7663e-12]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 238 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.5759e-03, 2.2784e-04, 5.9973e-03,
9.2781e-03, 3.9495e-03, 1.2202e-02,
        1.3889e-02, 5.2366e-03, 5.8554e-03, 7.2019e-01, 9.6745e-02, 1.6381e-02,
        1.8610e-02, 2.3595e-02, 2.8684e-04, 2.8507e-03, 4.4175e-02, 9.0796e-04,
        9.7500e-03, 7.9281e-03])

max probability is torch.return_types.max(
values=tensor(0.7202),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.3766e-05, 3.6195e-07, 3.6177e-07, 3.6175e-07]),
indices=tensor([  0, 914, 771, 323]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 239 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.4128e-08, 1.3003e-08, 5.8828e-10,
6.1069e-10, 3.8867e-09, 5.6792e-09,
        9.3853e-10, 1.8630e-09, 2.3173e-08, 5.0841e-08, 9.9990e-01, 5.4007e-07,
        6.8342e-06, 1.1937e-05, 5.4964e-11, 7.0252e-05, 1.7112e-06, 5.2914e-08,
        3.7683e-06, 2.3990e-06])

max probability is torch.return_types.max(
values=tensor(0.9999),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2802e-11, 9.2061e-16, 9.2043e-16, 9.1673e-16]),
indices=tensor([  0, 771, 319, 363]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 240 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.8899e-05, 2.0247e-05, 6.4055e-04,
1.4838e-03, 4.6292e-03, 5.2438e-04,
        4.0341e-04, 3.1222e-03, 4.3479e-03, 1.9503e-02, 1.9478e-03, 8.3701e-06,
        3.7451e-04, 3.0735e-02, 3.9687e-05, 6.8006e-04, 5.1215e-03, 1.4973e-03,
        6.6168e-01, 2.6165e-01])

max probability is torch.return_types.max(
```

```
values=tensor(0.6617),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3942e-03, 1.1633e-07, 1.1631e-07, 1.1629e-07]),
indices=tensor([  0, 914, 771, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 241 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.7603e-03, 4.2548e-04, 8.4122e-04,
7.9440e-05, 1.7706e-03, 1.7532e-04,
        1.6862e-04, 8.9085e-03, 1.7637e-02, 3.7689e-03, 2.8730e-02, 7.3014e-04,
        6.8255e-03, 5.9355e-02, 6.9934e-05, 2.0551e-01, 4.8943e-01, 6.8311e-03,
        1.2412e-01, 3.9753e-02])

max probability is torch.return_types.max(
values=tensor(0.4894),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.8660e-05, 4.3719e-08, 4.3712e-08, 4.3696e-08]),
indices=tensor([  0, 323, 910, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 242 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.8914e-05, 2.8537e-05, 7.4381e-06,
2.0844e-06, 1.2538e-05, 1.6944e-06,
        8.3794e-07, 9.6519e-05, 1.0740e-04, 7.6103e-06, 1.4660e-03, 7.4909e-05,
        1.6679e-04, 1.4657e-02, 3.0605e-07, 8.3581e-05, 8.2685e-02, 2.3067e-04,
        1.9083e-02, 8.7830e-01])

max probability is torch.return_types.max(
values=tensor(0.8783),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9467e-03, 1.5605e-09, 1.5601e-09, 1.5595e-09]),
indices=tensor([  0, 655, 691, 450]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 243 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0055, 0.0010, 0.0117, 0.0082, 0.0280,
0.0078, 0.0041, 0.0160, 0.0185,
        0.0277, 0.0148, 0.0009, 0.0040, 0.0576, 0.0005, 0.0064, 0.0373, 0.0134,
        0.3980, 0.3224])

max probability is torch.return_types.max(
values=tensor(0.3980),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.8355e-03, 9.8534e-06, 9.8499e-06, 9.8473e-06]),
indices=tensor([  0, 914, 337, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 244 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.5401e-03, 3.6228e-04, 6.9348e-04,
5.6629e-04, 1.4177e-03, 1.6151e-03,
        8.6252e-04, 4.2020e-04, 1.9113e-03, 2.1284e-03, 7.8018e-01, 5.0107e-03,
        5.2337e-02, 9.2608e-02, 5.0735e-04, 2.3637e-02, 8.9652e-03, 5.7092e-03,
        8.9737e-03, 1.0032e-02])

max probability is torch.return_types.max(
values=tensor(0.7802),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5586e-05, 5.1592e-07, 5.1543e-07, 5.1529e-07]),
indices=tensor([  0, 771, 319, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 245 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0061, 0.0058, 0.0013, 0.0006, 0.0030,
0.0042, 0.0021, 0.1614, 0.3055,
        0.0287, 0.0420, 0.0068, 0.0709, 0.0347, 0.0017, 0.0907, 0.1029, 0.0037,
        0.0888, 0.0224])

max probability is torch.return_types.max(
values=tensor(0.3055),
indices=tensor(8))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.1613e-04, 1.6812e-05, 1.6807e-05, 1.6804e-05]),
indices=tensor([  0, 316, 896, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 246 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.3823e-04, 5.6304e-04, 4.8780e-04,
1.9621e-04, 1.0403e-03, 2.7257e-04,
        5.7109e-04, 6.4798e-01, 3.2527e-01, 1.0727e-02, 1.4051e-03, 1.9420e-05,
        3.5857e-03, 8.7497e-04, 2.2068e-05, 8.7641e-04, 3.3428e-04, 8.2934e-05,
        5.2024e-03, 3.2009e-04])

max probability is torch.return_types.max(
values=tensor(0.6480),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6133e-06, 2.5472e-08, 2.5470e-08, 2.5466e-08]),
indices=tensor([  0, 896, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 247 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.6135e-03, 1.1139e-06, 2.6105e-03,
8.8517e-04, 2.7784e-03, 5.1335e-05,
        1.7438e-04, 1.5850e-04, 5.9771e-04, 7.1762e-03, 1.9406e-04, 1.1228e-06,
        5.6023e-04, 9.3639e-01, 1.7587e-06, 5.9113e-05, 1.0477e-02, 9.1019e-04,
        4.4256e-03, 2.9930e-02])

max probability is torch.return_types.max(
values=tensor(0.9364),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.3019e-06, 8.1981e-10, 8.1955e-10, 8.1946e-10]),
indices=tensor([  0, 914, 158, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 248 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0000e+00, 8.7527e-10, 2.4283e-08,
```

```
4.0914e-10, 4.9783e-09, 7.8482e-09,
        8.5655e-09, 9.1769e-10, 2.1570e-08, 1.1188e-08, 1.3676e-11, 4.9515e-11,
        1.1248e-08, 2.1667e-10, 8.9083e-12, 2.3217e-12, 1.2946e-11, 1.4986e-10,
        1.3236e-11, 2.6364e-11])

max probability is torch.return_types.max(
values=tensor(1.),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.9970e-17, 1.7133e-19, 1.7125e-19, 1.7107e-19]),
indices=tensor([  0, 910, 323,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 249 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.4011e-03, 1.7073e-03, 1.5614e-03,
1.6294e-03, 4.2630e-03, 1.8186e-02,
        8.4801e-03, 2.9487e-03, 1.7413e-02, 3.6607e-02, 5.5443e-01, 1.2527e-02,
        8.2322e-02, 7.3685e-02, 2.3466e-04, 4.6447e-02, 1.0669e-01, 1.2816e-03,
        9.4483e-03, 1.4260e-02])

max probability is torch.return_types.max(
values=tensor(0.5544),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3762e-04, 2.4884e-06, 2.4872e-06, 2.4858e-06]),
indices=tensor([  0, 771, 319, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 250 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0017, 0.0006, 0.0021, 0.0015, 0.0093,
0.0089, 0.0046, 0.0115, 0.0291,
        0.0188, 0.0121, 0.0064, 0.0099, 0.0337, 0.0051, 0.0153, 0.0867, 0.0296,
        0.6356, 0.0709])

max probability is torch.return_types.max(
values=tensor(0.6356),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([1.1543e-03, 5.5710e-06, 5.5668e-06, 5.5665e-06]),
indices=tensor([  0, 914, 197, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 251 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.2163e-05, 9.6067e-06, 1.5150e-05,
4.0656e-06, 2.4288e-05, 1.5559e-06,
        7.3484e-07, 6.6779e-05, 9.3076e-05, 1.7258e-05, 8.6664e-04, 3.4911e-05,
        1.1400e-04, 5.4063e-02, 7.0440e-07, 1.7854e-04, 1.4913e-01, 6.6328e-04,
        5.0757e-02, 7.4230e-01])

max probability is torch.return_types.max(
values=tensor(0.7423),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6349e-03, 1.2080e-09, 1.2076e-09, 1.2070e-09]),
indices=tensor([  0, 655, 691, 450]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 252 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.4723e-03, 1.3056e-03, 1.5055e-02,
1.3798e-01, 2.3970e-01, 3.5254e-01,
        2.3009e-01, 2.8863e-04, 7.0005e-04, 3.5356e-03, 4.8162e-04, 2.8951e-05,
        4.2196e-04, 5.2459e-03, 1.7881e-03, 6.9467e-04, 1.0111e-04, 7.3845e-04,
        3.1240e-03, 9.3641e-04])

max probability is torch.return_types.max(
values=tensor(0.3525),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8073e-05, 2.8880e-06, 2.8855e-06, 2.8835e-06]),
indices=tensor([  0, 319, 316, 144]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 253 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.9803e-04, 2.3414e-01, 1.9460e-04,
4.9371e-05, 4.9957e-04, 2.0492e-04,
        3.7864e-04, 6.3870e-01, 1.1444e-01, 1.3421e-04, 5.0016e-04, 8.2357e-05,
```

```
        8.1249e-03, 9.8023e-05, 6.6531e-05, 9.2213e-04, 4.0389e-04, 1.0150e-05,
        2.7876e-04, 5.1860e-05])

max probability is torch.return_types.max(
values=tensor(0.6387),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3393e-06, 2.6688e-08, 2.6672e-08, 2.6672e-08]),
indices=tensor([  0, 316, 836, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 254 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.0279e-03, 1.1146e-04, 8.4932e-04,
1.6273e-04, 1.0118e-03, 2.0861e-04,
        1.2725e-04, 1.8755e-03, 1.0811e-02, 2.6551e-03, 6.5198e-03, 1.5028e-04,
        1.9339e-02, 6.2280e-01, 2.1565e-05, 9.8198e-03, 2.9933e-01, 6.1345e-04,
        6.5499e-03, 1.4869e-02])

max probability is torch.return_types.max(
values=tensor(0.6228),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.9763e-05, 1.0475e-07, 1.0474e-07, 1.0468e-07]),
indices=tensor([  0, 910, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 255 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.8486e-03, 1.5570e-04, 3.7615e-02,
9.5480e-03, 3.1267e-02, 5.4594e-03,
        1.7518e-02, 1.4193e-03, 4.7438e-03, 2.1557e-02, 5.9023e-03, 7.0077e-04,
        5.6252e-03, 5.1749e-01, 5.2958e-03, 1.6493e-03, 1.5093e-02, 2.1804e-01,
        5.9543e-02, 3.7489e-02])

max probability is torch.return_types.max(
values=tensor(0.5175),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6406e-04, 9.2004e-07, 9.1987e-07, 9.1969e-07]),
```

```
indices=tensor([  0, 158, 150, 748]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 256 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.5836e-04, 3.7561e-05, 2.0941e-04,
2.6882e-04, 1.1391e-03, 3.6311e-04,
        7.1353e-05, 1.1604e-03, 1.0172e-03, 1.7193e-03, 6.8947e-03, 1.8405e-03,
        1.3423e-02, 1.6462e-02, 2.6210e-03, 2.8601e-03, 7.8157e-03, 1.0905e-02,
        8.0480e-01, 1.2603e-01])

max probability is torch.return_types.max(
values=tensor(0.8048),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5057e-04, 5.3422e-08, 5.3392e-08, 5.3366e-08]),
indices=tensor([  0, 914, 771, 420]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 257 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0182, 0.0020, 0.0041, 0.0031, 0.0094,
0.0052, 0.0036, 0.0186, 0.0251,
        0.0231, 0.0325, 0.0259, 0.0151, 0.0762, 0.0029, 0.0126, 0.1309, 0.0262,
        0.2932, 0.1914])

max probability is torch.return_types.max(
values=tensor(0.2932),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.6263e-03, 7.3979e-05, 7.3939e-05, 7.3932e-05]),
indices=tensor([  0, 914, 158, 127]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 258 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.8151e-04, 3.4364e-06, 8.4974e-06,
9.1801e-06, 5.4783e-05, 1.0578e-05,
        9.6806e-06, 2.2982e-04, 1.8304e-04, 1.9327e-04, 2.1916e-04, 1.6084e-03,
        2.7798e-04, 9.7745e-04, 3.2203e-01, 7.2860e-05, 2.4821e-04, 4.7830e-01,
        1.9362e-01, 1.6579e-03])
```

```
max probability is torch.return_types.max(
values=tensor(0.4783),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6331e-06, 5.5514e-09, 5.5504e-09, 5.5480e-09]),
indices=tensor([  0, 748, 337, 230]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 259 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.6464e-02, 1.7901e-01, 1.6183e-03,
2.4526e-03, 6.2418e-03, 4.8391e-01,
        5.0092e-02, 7.9259e-05, 1.1460e-03, 4.8783e-04, 2.4539e-01, 1.2013e-04,
        3.0010e-03, 1.2389e-03, 3.6359e-06, 8.3845e-03, 9.0130e-05, 5.3906e-06,
        1.4629e-04, 1.0506e-04])

max probability is torch.return_types.max(
values=tensor(0.4839),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.1263e-07, 1.9804e-08, 1.9723e-08, 1.9715e-08]),
indices=tensor([  0, 319, 794, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 260 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.7154e-06, 2.0704e-07, 1.1527e-04,
2.6191e-04, 3.8002e-04, 1.3293e-05,
        2.6024e-05, 5.6625e-06, 2.4149e-05, 1.4305e-04, 1.2289e-04, 1.2877e-07,
        4.6808e-05, 9.0820e-01, 3.1382e-07, 2.2204e-05, 8.2936e-03, 1.1689e-04,
        1.3412e-03, 8.0850e-02])

max probability is torch.return_types.max(
values=tensor(0.9082),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8516e-05, 1.3791e-10, 1.3784e-10, 1.3784e-10]),
indices=tensor([  0, 914, 337, 655]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 261 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.5655e-03, 5.5301e-01, 2.5604e-03,
5.7500e-04, 2.4414e-03, 2.5600e-03,
        2.6477e-03, 3.8856e-03, 4.1331e-03, 4.9726e-04, 1.0329e-01, 3.0987e-02,
        4.9469e-02, 2.6412e-02, 3.6081e-04, 2.1823e-02, 1.6116e-01, 9.4096e-04,
        3.7674e-03, 1.6291e-02])

max probability is torch.return_types.max(
values=tensor(0.5530),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.9722e-04, 3.3099e-06, 3.3096e-06, 3.3071e-06]),
indices=tensor([  0, 910, 323,  82]))
End********************************************************************************

**********************************************************************************
Predicting Test Sample 262 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.1200e-03, 3.4849e-04, 2.2023e-03,
3.4667e-03, 7.1478e-03, 9.8708e-03,
        5.0974e-03, 5.4467e-04, 3.7771e-03, 1.1389e-02, 4.3030e-02, 6.6026e-04,
        1.8190e-02, 7.8120e-01, 1.0836e-04, 4.5919e-03, 5.5235e-02, 1.5427e-03,
        5.7639e-03, 4.1536e-02])

max probability is torch.return_types.max(
values=tensor(0.7812),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1702e-04, 1.0084e-06, 1.0082e-06, 1.0078e-06]),
indices=tensor([  0, 771, 914, 323]))
End********************************************************************************

**********************************************************************************
Predicting Test Sample 263 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9998e-01, 2.5170e-07, 3.3131e-06,
2.4066e-07, 1.5986e-06, 2.4868e-06,
        1.8209e-06, 4.5073e-07, 3.9638e-06, 4.3906e-06, 3.2984e-08, 5.0495e-08,
        1.9585e-06, 3.6324e-07, 1.8994e-08, 8.8042e-09, 3.4823e-08, 3.3226e-07,
        4.7514e-08, 1.1547e-07])
```

```
max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.1484e-12, 5.7621e-14, 5.7619e-14, 5.7544e-14]),
indices=tensor([  0, 910, 323,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 264 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.1881e-06, 3.2092e-07, 2.0508e-04,
1.4898e-04, 1.9430e-04, 3.8847e-06,
        5.2953e-06, 7.9705e-06, 2.0216e-05, 8.5634e-05, 6.2303e-04, 1.8962e-06,
        7.4216e-04, 9.6714e-01, 3.0336e-06, 1.1975e-04, 1.3853e-02, 3.1106e-04,
        1.9316e-03, 1.4595e-02])

max probability is torch.return_types.max(
values=tensor(0.9671),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.4335e-06, 1.8750e-10, 1.8738e-10, 1.8736e-10]),
indices=tensor([  0, 914, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 265 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0036, 0.0028, 0.0056, 0.0011, 0.0032,
0.0027, 0.0028, 0.0074, 0.0119,
        0.0053, 0.0441, 0.0978, 0.0544, 0.0894, 0.0036, 0.0290, 0.5320, 0.0139,
        0.0363, 0.0336])

max probability is torch.return_types.max(
values=tensor(0.5320),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2413e-03, 1.9103e-05, 1.9102e-05, 1.9102e-05]),
indices=tensor([  0, 914, 691, 230]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 266 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0998, 0.0470, 0.0441, 0.0418, 0.0401,
0.1512, 0.0948, 0.0239, 0.0252,
        0.0342, 0.0207, 0.0324, 0.0200, 0.0127, 0.0221, 0.0062, 0.0058, 0.0206,
        0.0208, 0.0095])

max probability is torch.return_types.max(
values=tensor(0.1512),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([0.0009, 0.0002, 0.0002, 0.0002]),
indices=tensor([  0, 914, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 267 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0781e-04, 2.1581e-04, 3.0147e-01,
5.6583e-01, 1.1242e-01, 1.8231e-02,
        1.7206e-03, 1.8613e-07, 8.1125e-08, 1.6505e-06, 1.2138e-07, 2.1620e-09,
        6.0697e-07, 8.0963e-07, 2.0513e-07, 2.5897e-09, 1.0804e-08, 1.4450e-07,
        5.9369e-07, 3.8343e-07])

max probability is torch.return_types.max(
values=tensor(0.5658),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.3764e-09, 2.3496e-11, 2.3481e-11, 2.3470e-11]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 268 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4537e-04, 6.4329e-05, 1.4108e-04,
4.2010e-06, 7.6980e-05, 1.2714e-06,
        2.8178e-06, 7.6007e-01, 1.1312e-01, 1.2212e-03, 9.7137e-04, 8.0058e-05,
        7.8298e-03, 8.0179e-04, 6.4665e-07, 2.1353e-03, 9.8941e-02, 6.3620e-06,
        1.2443e-02, 1.9474e-03])

max probability is torch.return_types.max(
values=tensor(0.7601),
indices=tensor(7))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.7558e-06, 6.2327e-10, 6.2315e-10, 6.2303e-10]),
indices=tensor([  0, 910,  85, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 269 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9460e-01, 8.3394e-05, 5.8399e-04,
8.0063e-05, 2.6449e-04, 3.0170e-04,
        1.9735e-04, 2.2396e-04, 6.7115e-04, 1.3972e-03, 5.7132e-05, 1.2833e-04,
        5.8443e-04, 1.3765e-04, 2.3519e-05, 2.6293e-05, 2.2527e-04, 1.6440e-04,
        1.4441e-04, 9.8263e-05])

max probability is torch.return_types.max(
values=tensor(0.9946),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.4781e-07, 1.0701e-08, 1.0701e-08, 1.0694e-08]),
indices=tensor([  0, 910, 323, 391]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 270 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.0192e-04, 2.6855e-02, 3.7595e-05,
9.0776e-06, 1.1522e-04, 1.4363e-04,
        3.8739e-04, 7.1023e-01, 2.5018e-01, 9.7177e-05, 3.7182e-04, 2.7558e-04,
        9.9425e-03, 7.4293e-05, 1.3515e-05, 1.3396e-04, 2.8768e-04, 5.7372e-06,
        9.7677e-05, 2.8827e-05])

max probability is torch.return_types.max(
values=tensor(0.7102),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.6853e-07, 1.0322e-08, 1.0319e-08, 1.0318e-08]),
indices=tensor([  0, 316, 836, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 271 :   Prediction is Correct?
No
```

```
first 20 classes probability: tensor([0.0066, 0.0042, 0.0248, 0.0082, 0.0340,
0.0322, 0.0283, 0.0572, 0.2425,
        0.0477, 0.0933, 0.0005, 0.0445, 0.0622, 0.0007, 0.0358, 0.0206, 0.0163,
        0.0984, 0.1346])

max probability is torch.return_types.max(
values=tensor(0.2425),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3495e-03, 6.5457e-06, 6.5401e-06, 6.5398e-06]),
indices=tensor([  0, 771, 319, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 272 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.6783e-04, 4.1890e-02, 1.5428e-04,
3.2074e-05, 2.9287e-04, 5.9702e-04,
        4.0650e-04, 1.5903e-03, 4.1364e-03, 1.3304e-04, 1.4451e-01, 1.1426e-03,
        6.5324e-02, 4.7219e-03, 4.2963e-04, 7.2696e-01, 3.8115e-03, 4.2700e-04,
        1.4448e-03, 1.1399e-03])

max probability is torch.return_types.max(
values=tensor(0.7270),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.8965e-06, 2.0423e-07, 2.0375e-07, 2.0373e-07]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 273 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0063, 0.0030, 0.0109, 0.0042, 0.0106,
0.0056, 0.0057, 0.0143, 0.0247,
        0.0088, 0.0897, 0.0333, 0.3952, 0.1121, 0.0109, 0.0216, 0.1154, 0.0162,
        0.0419, 0.0360])

max probability is torch.return_types.max(
values=tensor(0.3952),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([9.8041e-04, 3.3979e-05, 3.3976e-05, 3.3976e-05]),
indices=tensor([  0, 914, 771, 910]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 274 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0454, 0.0040, 0.1009, 0.0263, 0.0865,
0.0251, 0.0221, 0.0207, 0.0572,
        0.0458, 0.0783, 0.0006, 0.0453, 0.1015, 0.0006, 0.0245, 0.0232, 0.0196,
        0.0698, 0.1820])

max probability is torch.return_types.max(
values=tensor(0.1820),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4491e-03, 1.9035e-05, 1.9021e-05, 1.9019e-05]),
indices=tensor([  0, 771, 914, 910]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 275 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.5186e-09, 2.7260e-09, 6.7575e-08,
5.9372e-08, 8.5274e-07, 3.2512e-09,
        2.7251e-09, 2.6109e-07, 1.1587e-07, 8.0090e-08, 7.8742e-07, 1.2271e-11,
        6.9767e-09, 1.8883e-05, 8.5003e-13, 1.1671e-08, 1.2137e-04, 6.7692e-08,
        1.3081e-03, 9.9835e-01])

max probability is torch.return_types.max(
values=tensor(0.9984),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9690e-04, 1.6323e-15, 1.6319e-15, 1.6287e-15]),
indices=tensor([  0, 655, 337, 158]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 276 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.5313e-06, 2.2024e-05, 6.8111e-06,
2.7510e-07, 4.9115e-06, 1.5237e-07,
        1.8494e-07, 6.0687e-04, 1.0483e-04, 5.8808e-06, 2.0746e-04, 1.0991e-04,
        8.8728e-05, 8.9050e-04, 4.3273e-08, 1.9689e-04, 9.4810e-01, 8.3974e-06,
```

```
        1.8809e-03, 4.7633e-02])

max probability is torch.return_types.max(
values=tensor(0.9481),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2233e-04, 1.0477e-10, 1.0475e-10, 1.0468e-10]),
indices=tensor([  0, 691, 230, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 277 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.5665e-04, 5.0381e-04, 1.0060e-01,
2.7544e-01, 4.4238e-01, 1.3932e-01,
        4.0161e-02, 2.6201e-05, 3.0061e-05, 1.3181e-04, 3.6758e-05, 1.0193e-06,
        4.2250e-05, 2.0890e-04, 6.0551e-05, 1.0261e-05, 5.1781e-06, 4.4915e-05,
        3.8504e-04, 9.0345e-05])

max probability is torch.return_types.max(
values=tensor(0.4424),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.1347e-06, 1.7643e-07, 1.7633e-07, 1.7620e-07]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 278 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1765e-03, 9.4371e-01, 9.0158e-03,
8.7220e-03, 1.6671e-02, 7.7401e-03,
        6.1350e-03, 2.0510e-03, 7.7022e-04, 1.3046e-04, 2.1283e-04, 1.8513e-05,
        7.9333e-04, 3.5222e-04, 3.3552e-04, 9.7701e-04, 2.4416e-04, 1.6734e-04,
        4.9142e-04, 2.0443e-04])

max probability is torch.return_types.max(
values=tensor(0.9437),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6434e-06, 7.8109e-08, 7.8062e-08, 7.8033e-08]),
indices=tensor([  0, 316, 319, 323]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 279 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.4475e-07, 1.5846e-09, 4.1276e-09,
3.9149e-08, 8.6462e-08, 5.1788e-08,
        5.9270e-08, 7.9970e-08, 7.0693e-09, 2.6636e-07, 3.6386e-09, 3.6470e-06,
        5.7899e-07, 7.4632e-08, 9.9961e-01, 6.4637e-10, 1.1240e-08, 3.7203e-04,
        1.4287e-05, 4.3985e-08])

max probability is torch.return_types.max(
values=tensor(0.9996),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.0923e-12, 8.0836e-14, 8.0831e-14, 8.0772e-14]),
indices=tensor([  0, 748, 337, 323]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 280 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0280, 0.0103, 0.0043, 0.0013, 0.0040,
0.0088, 0.0054, 0.0296, 0.0483,
        0.0054, 0.3207, 0.2241, 0.1523, 0.0186, 0.0017, 0.0159, 0.0375, 0.0070,
        0.0311, 0.0227])

max probability is torch.return_types.max(
values=tensor(0.3207),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.0467e-04, 2.3324e-05, 2.3324e-05, 2.3323e-05]),
indices=tensor([  0,  44, 771, 914]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 281 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.5765e-03, 4.1851e-04, 1.4011e-03,
2.0907e-03, 3.4187e-03, 3.1739e-03,
        3.7276e-03, 7.4952e-03, 2.0785e-02, 1.3115e-01, 2.2985e-02, 4.1793e-04,
        6.8729e-03, 5.3247e-01, 2.1085e-05, 2.7700e-03, 1.8161e-01, 6.5451e-04,
        1.5710e-02, 6.0631e-02])
```

```
max probability is torch.return_types.max(
values=tensor(0.5325),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9592e-04, 3.3950e-07, 3.3940e-07, 3.3938e-07]),
indices=tensor([  0, 914, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 282 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.4201e-05, 3.3021e-06, 1.9791e-06,
8.1121e-07, 2.3823e-06, 4.7386e-06,
        3.9814e-06, 2.6294e-05, 4.9502e-06, 1.1441e-05, 2.3220e-04, 9.8977e-01,
        7.2785e-04, 1.6472e-04, 3.0862e-04, 7.6817e-06, 8.1944e-03, 8.8610e-05,
        3.0851e-04, 7.1316e-05])

max probability is torch.return_types.max(
values=tensor(0.9898),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5228e-07, 4.1185e-10, 4.1159e-10, 4.1128e-10]),
indices=tensor([  0, 691,  82, 382]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 283 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.8247e-04, 2.3836e-04, 2.9515e-04,
3.2316e-04, 8.2086e-04, 5.2951e-04,
        2.3096e-04, 3.1148e-04, 1.0512e-03, 1.6662e-03, 5.5172e-01, 8.1212e-04,
        1.2858e-02, 2.1623e-01, 1.4933e-05, 3.5042e-02, 5.0718e-02, 8.7255e-04,
        1.3723e-02, 1.1173e-01])

max probability is torch.return_types.max(
values=tensor(0.5517),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5993e-04, 7.5945e-08, 7.5850e-08, 7.5844e-08]),
indices=tensor([  0, 771, 914, 323]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 284 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.3905e-01, 5.9860e-05, 2.0687e-03,
6.3522e-04, 3.5080e-03, 2.2254e-03,
        2.9801e-03, 2.2865e-03, 7.5988e-03, 8.5859e-02, 5.7149e-04, 1.9328e-03,
        2.7949e-03, 6.1855e-03, 3.5117e-02, 5.5956e-04, 2.0230e-03, 2.6363e-01,
        3.8150e-02, 2.4346e-03])

max probability is torch.return_types.max(
values=tensor(0.5391),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.0635e-06, 3.3292e-07, 3.3283e-07, 3.3267e-07]),
indices=tensor([  0, 323, 748, 311]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 285 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2438e-04, 1.1058e-03, 7.6866e-05,
1.6583e-05, 6.8471e-05, 1.8833e-04,
        1.5495e-04, 3.9891e-02, 7.6797e-02, 4.9619e-05, 8.8152e-03, 9.3480e-04,
        8.7017e-01, 4.2026e-04, 2.4078e-05, 8.9652e-04, 2.0367e-04, 6.0522e-06,
        2.7393e-05, 2.3544e-05])

max probability is torch.return_types.max(
values=tensor(0.8702),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6230e-07, 7.5064e-09, 7.5019e-09, 7.4980e-09]),
indices=tensor([  0, 836, 319, 794]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 286 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.3557e-03, 1.4314e-03, 2.4744e-03,
4.4440e-04, 1.8093e-03, 9.0346e-04,
        1.0369e-03, 1.8511e-02, 1.9054e-02, 4.9072e-03, 2.3501e-02, 4.2362e-02,
        3.2146e-02, 2.9178e-02, 1.5129e-04, 6.3540e-03, 7.6718e-01, 1.3125e-03,
        1.6400e-02, 2.2133e-02])

max probability is torch.return_types.max(
```

```
values=tensor(0.7672),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.1557e-04, 3.7034e-06, 3.7032e-06, 3.7025e-06]),
indices=tensor([  0, 691, 323, 382]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 287 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0181, 0.0023, 0.0624, 0.0739, 0.0845,
0.0508, 0.0441, 0.0131, 0.0308,
        0.2000, 0.0243, 0.0008, 0.0179, 0.1097, 0.0011, 0.0112, 0.0364, 0.0082,
        0.0489, 0.0956])

max probability is torch.return_types.max(
values=tensor(0.2000),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.3193e-03, 6.5449e-05, 6.5436e-05, 6.5411e-05]),
indices=tensor([  0, 771, 914,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 288 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.1582e-03, 5.7752e-04, 2.7978e-01,
2.7240e-01, 2.1087e-01, 4.7447e-02,
        1.1571e-01, 6.5458e-04, 9.0966e-04, 1.1278e-02, 1.4151e-03, 1.7678e-05,
        1.4240e-03, 4.0997e-02, 4.1194e-04, 6.2289e-04, 1.2881e-03, 1.2027e-03,
        3.1675e-03, 6.0299e-03])

max probability is torch.return_types.max(
values=tensor(0.2798),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3100e-04, 2.6246e-06, 2.6245e-06, 2.6244e-06]),
indices=tensor([  0, 910, 771, 726]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 289 :   Prediction is Correct?
```

```
Yes
first 20 classes probability: tensor([6.6091e-04, 9.8541e-05, 9.4679e-03,
8.2006e-02, 8.1107e-02, 1.9764e-02,
        4.6593e-02, 9.5950e-04, 2.8830e-03, 2.7173e-02, 3.5054e-03, 3.7983e-05,
        3.1592e-03, 6.9310e-01, 5.1449e-04, 1.1189e-03, 2.6159e-03, 2.3537e-03,
        7.3615e-03, 1.4821e-02])

max probability is torch.return_types.max(
values=tensor(0.6931),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.9817e-05, 6.5572e-07, 6.5571e-07, 6.5563e-07]),
indices=tensor([  0, 771, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 290 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.4290e-04, 1.2417e-04, 3.4778e-03,
2.8784e-03, 8.9475e-03, 1.9386e-03,
        1.7547e-03, 9.9345e-03, 1.4343e-02, 4.4678e-02, 6.9724e-03, 2.1336e-03,
        7.5379e-03, 1.6365e-01, 2.0048e-03, 6.0039e-03, 1.4845e-01, 1.1023e-02,
        4.7872e-01, 7.8758e-02])

max probability is torch.return_types.max(
values=tensor(0.4787),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2707e-03, 4.6511e-06, 4.6484e-06, 4.6480e-06]),
indices=tensor([  0, 914, 337, 197]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 291 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.1124e-05, 9.9880e-01, 1.6685e-05,
6.8598e-06, 4.1651e-05, 1.0894e-04,
        6.1750e-05, 3.8213e-04, 1.9781e-04, 1.3832e-06, 3.0234e-05, 5.1271e-07,
        2.1543e-04, 2.9030e-06, 6.2755e-07, 9.5722e-05, 9.1187e-06, 8.1666e-08,
        4.7483e-06, 1.5263e-06])

max probability is torch.return_types.max(
values=tensor(0.9988),
indices=tensor(1))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.7408e-09, 3.3287e-11, 3.3275e-11, 3.3188e-11]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 292 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.8802e-07, 9.9983e-01, 1.7091e-07,
1.8819e-08, 6.9616e-07, 1.3847e-07,
        1.7463e-07, 1.5254e-04, 9.2272e-06, 5.7082e-10, 3.0750e-08, 1.0687e-09,
        3.5625e-06, 5.6924e-09, 1.4855e-09, 7.8888e-08, 7.2189e-08, 6.2175e-11,
        2.4195e-08, 1.8727e-09])

max probability is torch.return_types.max(
values=tensor(0.9998),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1328e-12, 1.7243e-16, 1.7202e-16, 1.7167e-16]),
indices=tensor([  0, 316, 319, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 293 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.3478e-03, 1.3396e-03, 1.0443e-01,
2.0606e-01, 4.1640e-01, 1.9333e-01,
        7.1456e-02, 8.9819e-05, 1.2929e-04, 3.3379e-04, 2.0079e-04, 1.3659e-05,
        2.0476e-04, 5.4643e-04, 3.4182e-04, 8.4749e-05, 3.3599e-05, 2.5918e-04,
        1.4230e-03, 2.8558e-04])

max probability is torch.return_types.max(
values=tensor(0.4164),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0593e-05, 1.7573e-06, 1.7566e-06, 1.7555e-06]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 294 :   Prediction is Correct?
No
```

```
first 20 classes probability: tensor([1.2462e-03, 1.0427e-03, 2.7672e-02,
1.7352e-01, 2.0909e-01, 3.9176e-01,
        1.9278e-01, 4.3661e-05, 9.5528e-05, 4.2562e-04, 8.9236e-05, 7.8322e-06,
        1.3773e-04, 5.6977e-04, 3.7171e-04, 6.9953e-05, 1.2295e-05, 1.2969e-04,
        5.0162e-04, 1.3418e-04])

max probability is torch.return_types.max(
values=tensor(0.3918),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.7358e-06, 3.2150e-07, 3.2131e-07, 3.2109e-07]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 295 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.6220e-01, 3.4843e-04, 1.9231e-03,
1.1638e-04, 7.6592e-04, 5.4341e-04,
        1.1665e-03, 6.3484e-04, 8.2862e-03, 8.1760e-04, 2.6465e-04, 9.9000e-05,
        2.1847e-02, 4.4395e-04, 7.3880e-05, 7.0817e-05, 8.6413e-05, 2.0195e-04,
        4.2169e-05, 4.8531e-05])

max probability is torch.return_types.max(
values=tensor(0.9622),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5097e-07, 2.6926e-08, 2.6923e-08, 2.6910e-08]),
indices=tensor([  0, 910, 319, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 296 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0073, 0.0030, 0.0042, 0.0038, 0.0099,
0.0054, 0.0035, 0.0228, 0.0235,
        0.0263, 0.0829, 0.0048, 0.0152, 0.0281, 0.0010, 0.0125, 0.0475, 0.0143,
        0.2486, 0.3983])

max probability is torch.return_types.max(
values=tensor(0.3983),
indices=tensor(19))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([1.1584e-02, 2.6915e-05, 2.6914e-05, 2.6902e-05]),
indices=tensor([  0, 914, 771, 337]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 297 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0014, 0.0004, 0.0367, 0.0776, 0.1003,
0.0234, 0.0266, 0.0034, 0.0053,
        0.0368, 0.0265, 0.0007, 0.0261, 0.3611, 0.0029, 0.0059, 0.0177, 0.0134,
        0.1069, 0.1206])

max probability is torch.return_types.max(
values=tensor(0.3611),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.8333e-04, 5.5402e-06, 5.5394e-06, 5.5363e-06]),
indices=tensor([  0, 914, 771, 323]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 298 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.6770e-06, 1.5623e-07, 5.8645e-08,
3.0052e-08, 6.9264e-08, 2.0660e-07,
        1.0118e-07, 1.6358e-06, 1.9573e-07, 4.2894e-07, 1.2205e-04, 9.9905e-01,
        1.5198e-04, 9.2966e-06, 1.9255e-05, 1.8228e-07, 5.7339e-04, 9.7077e-06,
        4.0291e-05, 1.1441e-05])

max probability is torch.return_types.max(
values=tensor(0.9991),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.0362e-09, 1.4239e-12, 1.4237e-12, 1.4230e-12]),
indices=tensor([  0, 691, 914,  82]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 299 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.5336e-03, 4.0613e-04, 1.1519e-03,
8.5579e-04, 1.6621e-03, 1.0754e-03,
        6.8064e-04, 3.8715e-04, 1.1469e-03, 1.4472e-03, 9.3734e-02, 7.4538e-03,
```

```
        5.2438e-02, 5.5125e-01, 2.3814e-04, 5.2473e-03, 2.0366e-01, 2.0774e-03,
        6.2550e-03, 6.5767e-02])

max probability is torch.return_types.max(
values=tensor(0.5513),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2905e-04, 3.1415e-07, 3.1409e-07, 3.1401e-07]),
indices=tensor([  0, 914, 323, 771]))
End*****************************************************************************

currently at 300 current time is 9.389639139175415
*****************************************************************************
Predicting Test Sample 300 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.1113e-03, 2.8543e-04, 2.4617e-01,
6.5632e-02, 1.0092e-01, 1.3076e-02,
        1.6458e-02, 1.7198e-02, 3.5669e-02, 1.7614e-01, 5.3983e-03, 6.9236e-05,
        1.7718e-02, 1.6761e-01, 8.5737e-04, 4.6931e-03, 1.0424e-02, 7.2799e-03,
        7.8433e-02, 2.6675e-02])

max probability is torch.return_types.max(
values=tensor(0.2462),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6708e-04, 4.1249e-06, 4.1243e-06, 4.1241e-06]),
indices=tensor([  0, 771, 914,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 301 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0968e-03, 9.8569e-04, 7.5780e-05,
3.1524e-05, 6.1990e-05, 1.9641e-04,
        1.8400e-04, 1.0923e-03, 2.7492e-04, 9.8816e-05, 3.5482e-03, 9.6803e-01,
        8.3657e-03, 4.3761e-04, 6.9761e-04, 2.4906e-04, 1.3449e-02, 1.7467e-04,
        5.6734e-04, 2.7968e-04])

max probability is torch.return_types.max(
values=tensor(0.9680),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([4.5617e-06, 9.9568e-08, 9.9567e-08, 9.9552e-08]),
indices=tensor([  0, 382, 691,  82]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 302 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0187, 0.0074, 0.0016, 0.0009, 0.0037,
0.0127, 0.0067, 0.0063, 0.0137,
        0.0043, 0.0527, 0.3289, 0.1187, 0.0822, 0.0357, 0.0227, 0.1701, 0.0277,
        0.0419, 0.0205])

max probability is torch.return_types.max(
values=tensor(0.3289),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.8732e-04, 2.3514e-05, 2.3509e-05, 2.3506e-05]),
indices=tensor([  0, 323, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 303 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.6175e-04, 5.9690e-02, 3.4134e-05,
7.5549e-06, 9.6580e-05, 2.7275e-05,
        3.1722e-05, 8.7856e-01, 5.8416e-02, 6.1287e-05, 1.6525e-04, 2.8639e-05,
        1.4970e-03, 1.9499e-05, 1.6780e-05, 1.4152e-04, 6.8279e-05, 4.3403e-06,
        2.4156e-04, 2.6839e-05])

max probability is torch.return_types.max(
values=tensor(0.8786),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9197e-07, 2.8771e-09, 2.8732e-09, 2.8730e-09]),
indices=tensor([  0, 316, 319, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 304 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.5779e-03, 4.0740e-04, 5.8912e-02,
5.1184e-03, 2.1503e-02, 4.2100e-03,
        4.3263e-03, 1.7430e-03, 1.1201e-02, 8.5805e-03, 1.9439e-02, 1.5614e-04,
        5.3611e-02, 7.0126e-01, 3.9792e-04, 1.0080e-02, 3.8924e-02, 1.5531e-02,
```

```
        9.3560e-03, 2.7125e-02])

max probability is torch.return_types.max(
values=tensor(0.7013),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.5145e-05, 4.9542e-07, 4.9542e-07, 4.9538e-07]),
indices=tensor([  0, 910, 771, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 305 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.0607e-05, 8.9821e-06, 1.6542e-03,
2.1465e-03, 2.6686e-03, 3.5907e-04,
        3.6289e-04, 5.1820e-05, 2.3946e-04, 1.4336e-03, 3.5015e-03, 8.2057e-06,
        3.0151e-03, 9.4933e-01, 1.2120e-05, 6.0478e-04, 1.8115e-02, 3.4994e-04,
        1.1844e-03, 1.4847e-02])

max probability is torch.return_types.max(
values=tensor(0.9493),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5727e-05, 9.6216e-09, 9.6216e-09, 9.6214e-09]),
indices=tensor([  0, 771, 323, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 306 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.7776e-03, 3.2569e-03, 7.7686e-04,
8.8303e-04, 5.3336e-03, 3.1895e-02,
        1.5568e-02, 6.0013e-02, 5.2179e-01, 3.0812e-02, 6.1525e-02, 1.7548e-03,
        1.0208e-01, 6.9727e-02, 1.0073e-04, 1.8520e-02, 4.2679e-02, 5.3699e-04,
        9.8342e-03, 1.6762e-02])

max probability is torch.return_types.max(
values=tensor(0.5218),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3330e-04, 2.2799e-06, 2.2788e-06, 2.2785e-06]),
indices=tensor([  0, 319, 771, 316]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 307 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0054, 0.0007, 0.0310, 0.0277, 0.0535,
0.0171, 0.0160, 0.0247, 0.0396,
        0.1364, 0.0098, 0.0013, 0.0131, 0.1824, 0.0057, 0.0090, 0.0385, 0.0256,
        0.2351, 0.0805])

max probability is torch.return_types.max(
values=tensor(0.2351),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3143e-03, 4.6485e-05, 4.6469e-05, 4.6467e-05]),
indices=tensor([  0, 914, 771,  91]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 308 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.7150e-06, 1.0695e-03, 6.6011e-07,
1.4015e-07, 3.9046e-06, 5.6854e-05,
        1.8514e-05, 3.5512e-05, 4.4039e-04, 2.2982e-06, 1.7360e-01, 9.2952e-05,
        6.7625e-03, 8.7445e-04, 1.6306e-06, 8.1561e-01, 1.2757e-03, 4.0518e-06,
        3.8088e-05, 1.0611e-04])

max probability is torch.return_types.max(
values=tensor(0.8156),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5084e-08, 1.2706e-10, 1.2655e-10, 1.2654e-10]),
indices=tensor([  0, 319, 771, 316]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 309 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.3435e-04, 2.1547e-03, 5.8841e-04,
8.8412e-05, 8.0411e-04, 3.3848e-04,
        2.0507e-04, 7.1810e-03, 1.6201e-02, 1.5105e-03, 1.4245e-01, 8.9289e-04,
        5.0373e-02, 2.8553e-02, 1.0122e-04, 5.7830e-01, 1.3063e-01, 9.4401e-04,
        1.5288e-02, 2.2028e-02])
```

```
max probability is torch.return_types.max(
values=tensor(0.5783),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.2956e-05, 3.8212e-07, 3.8197e-07, 3.8186e-07]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 310 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.7085e-04, 2.3663e-04, 8.3823e-01,
4.2182e-02, 2.0045e-02, 3.3788e-02,
        5.9524e-02, 8.6731e-05, 2.1181e-04, 3.1237e-03, 1.6994e-04, 6.5425e-06,
        5.0288e-04, 5.4236e-04, 8.0149e-05, 5.6597e-05, 9.2293e-05, 1.8108e-04,
        1.5665e-04, 1.6544e-04])

max probability is torch.return_types.max(
values=tensor(0.8382),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5838e-06, 5.0806e-08, 5.0797e-08, 5.0790e-08]),
indices=tensor([  0, 914, 910, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 311 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.2250, 0.0022, 0.0158, 0.0094, 0.0390,
0.0586, 0.0364, 0.0099, 0.0774,
        0.0920, 0.0534, 0.0012, 0.0377, 0.1712, 0.0014, 0.0245, 0.0159, 0.0266,
        0.0278, 0.0634])

max probability is torch.return_types.max(
values=tensor(0.2250),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.5005e-04, 1.1019e-05, 1.1014e-05, 1.1008e-05]),
indices=tensor([  0, 771, 319, 323]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 312 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3790e-02, 1.2853e-03, 5.0994e-03,
5.0854e-04, 3.8299e-03, 6.9713e-04,
        7.3542e-04, 8.3618e-02, 1.0872e-01, 2.8002e-02, 4.3256e-03, 4.4112e-03,
        3.7023e-02, 3.0605e-02, 1.2281e-03, 2.4818e-02, 5.3119e-01, 5.1086e-03,
        1.0143e-01, 1.1302e-02])

max probability is torch.return_types.max(
values=tensor(0.5312),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8585e-04, 2.2015e-06, 2.2012e-06, 2.2012e-06]),
indices=tensor([  0, 197, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 313 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0245, 0.0136, 0.0036, 0.0028, 0.0124,
0.0751, 0.0248, 0.0070, 0.0499,
        0.0199, 0.1502, 0.0189, 0.1730, 0.0744, 0.0084, 0.2142, 0.0663, 0.0050,
        0.0174, 0.0096])

max probability is torch.return_types.max(
values=tensor(0.2142),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3569e-04, 3.0615e-05, 3.0590e-05, 3.0583e-05]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 314 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.2672e-04, 3.3222e-03, 5.1363e-01,
4.1335e-02, 4.5961e-02, 1.2443e-01,
        2.6759e-01, 3.8961e-05, 1.4430e-04, 2.0990e-04, 2.7583e-04, 8.0621e-06,
        7.8897e-04, 8.9298e-04, 1.5528e-04, 1.4596e-04, 5.1165e-05, 1.6053e-04,
        9.7347e-05, 8.4968e-05])

max probability is torch.return_types.max(
values=tensor(0.5136),
indices=tensor(2))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0812e-06, 4.5891e-08, 4.5876e-08, 4.5875e-08]),
indices=tensor([  0, 910, 323, 726]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 315 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.3460e-03, 3.6004e-03, 4.3433e-04,
1.8269e-04, 1.2557e-03, 1.3500e-02,
        5.3685e-03, 5.5277e-03, 8.3455e-02, 2.1857e-03, 5.5933e-01, 1.1956e-03,
        2.0223e-01, 1.0115e-02, 1.0724e-04, 1.0390e-01, 1.9668e-03, 4.3454e-04,
        1.7644e-03, 1.8102e-03])

max probability is torch.return_types.max(
values=tensor(0.5593),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.1517e-06, 3.0322e-07, 3.0262e-07, 3.0260e-07]),
indices=tensor([  0, 319, 836, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 316 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.7544e-04, 6.9157e-03, 2.0325e-04,
6.6813e-05, 5.1375e-04, 5.8208e-03,
        4.7728e-03, 2.6840e-01, 6.9634e-01, 1.9038e-03, 1.6151e-03, 8.7577e-05,
        1.0727e-02, 4.3991e-04, 1.4395e-05, 5.3204e-04, 2.4184e-04, 2.5796e-05,
        2.1870e-04, 1.2791e-04])

max probability is torch.return_types.max(
values=tensor(0.6963),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0595e-06, 5.4496e-08, 5.4496e-08, 5.4491e-08]),
indices=tensor([  0, 957, 319, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 317 :   Prediction is Correct?
No
```

```
first 20 classes probability: tensor([0.0466, 0.0057, 0.0109, 0.0042, 0.0158,
0.0375, 0.0569, 0.0227, 0.1553,
        0.0300, 0.0233, 0.0035, 0.5363, 0.0268, 0.0024, 0.0065, 0.0067, 0.0019,
        0.0012, 0.0016])

max probability is torch.return_types.max(
values=tensor(0.5363),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6396e-05, 4.1838e-06, 4.1828e-06, 4.1821e-06]),
indices=tensor([  0, 319, 910, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 318 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.3716e-05, 2.3728e-05, 9.3025e-02,
6.8707e-01, 1.8402e-01, 2.9674e-02,
        5.9753e-03, 1.4220e-06, 9.9326e-07, 4.3820e-05, 1.1115e-06, 2.2876e-09,
        1.3387e-06, 6.8421e-05, 6.7742e-07, 9.1775e-08, 9.6653e-08, 1.1088e-06,
        7.9452e-06, 1.3737e-05])

max probability is torch.return_types.max(
values=tensor(0.6871),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.5225e-08, 3.4136e-10, 3.4106e-10, 3.4095e-10]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 319 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.9938e-03, 8.6619e-03, 3.1256e-02,
4.1849e-03, 2.3674e-02, 1.2536e-02,
        2.7503e-02, 2.5100e-01, 4.0261e-01, 1.5615e-02, 1.9980e-02, 3.1488e-03,
        3.6013e-02, 2.0808e-02, 1.1547e-04, 3.8522e-03, 8.1479e-02, 1.4857e-03,
        2.5366e-02, 2.1371e-02])

max probability is torch.return_types.max(
values=tensor(0.4026),
indices=tensor(8))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([1.0513e-03, 5.5567e-06, 5.5557e-06, 5.5555e-06]),
indices=tensor([  0, 197, 717, 910]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 320 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0122, 0.0016, 0.0257, 0.0176, 0.0599,
0.0657, 0.0474, 0.0105, 0.0641,
        0.0490, 0.0160, 0.0005, 0.0223, 0.4051, 0.0059, 0.0150, 0.0141, 0.0393,
        0.0724, 0.0430])

max probability is torch.return_types.max(
values=tensor(0.4051),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.7406e-04, 1.2968e-05, 1.2963e-05, 1.2961e-05]),
indices=tensor([  0, 771, 914, 323]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 321 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.7314e-04, 5.0936e-04, 1.0112e-04,
5.2074e-05, 2.5401e-04, 6.3734e-04,
        1.7433e-04, 1.5561e-04, 2.9593e-03, 2.8616e-04, 7.5995e-01, 1.0869e-04,
        5.6909e-02, 1.2623e-01, 3.5150e-06, 4.4564e-02, 4.3614e-03, 6.6200e-05,
        2.9448e-04, 1.8938e-03])

max probability is torch.return_types.max(
values=tensor(0.7600),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8632e-06, 1.0877e-08, 1.0868e-08, 1.0846e-08]),
indices=tensor([  0, 319, 771, 316]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 322 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.0680e-03, 2.0582e-03, 1.1439e-01,
2.3371e-01, 3.5791e-01, 1.7930e-01,
        9.6485e-02, 2.6570e-04, 4.3416e-04, 1.3008e-03, 4.3559e-04, 2.5965e-05,
```

```
        6.0371e-04, 1.9155e-03, 5.7475e-04, 2.2759e-04, 9.6033e-05, 4.5712e-04,
        1.8714e-03, 6.0860e-04])

max probability is torch.return_types.max(
values=tensor(0.3579),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.9029e-05, 5.4721e-06, 5.4696e-06, 5.4658e-06]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 323 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.4604e-04, 4.9855e-04, 2.1398e-04,
6.9267e-05, 1.9673e-04, 3.0856e-04,
        3.4312e-04, 3.0305e-04, 1.7379e-03, 3.5280e-04, 9.1497e-01, 1.2360e-03,
        6.2127e-02, 5.7302e-03, 4.6631e-06, 9.1804e-03, 1.8542e-03, 5.4136e-05,
        1.8750e-04, 3.7450e-04])

max probability is torch.return_types.max(
values=tensor(0.9150),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1462e-06, 1.5854e-08, 1.5849e-08, 1.5828e-08]),
indices=tensor([  0, 319, 771,  44]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 324 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.9586e-04, 2.7429e-03, 5.6949e-05,
7.6720e-06, 9.4631e-05, 3.0211e-05,
        3.7229e-05, 7.7657e-01, 2.1548e-01, 1.0385e-04, 1.1948e-04, 2.8005e-05,
        4.0685e-03, 4.6335e-05, 2.7978e-06, 7.0882e-05, 1.3697e-04, 2.5416e-06,
        1.7844e-04, 2.0417e-05])

max probability is torch.return_types.max(
values=tensor(0.7766),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1153e-07, 1.3418e-09, 1.3407e-09, 1.3403e-09]),
```

```
indices=tensor([  0, 316, 957, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 325 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0085, 0.0027, 0.0196, 0.0137, 0.0274,
0.0074, 0.0090, 0.0390, 0.0586,
        0.0559, 0.1130, 0.0037, 0.0619, 0.1964, 0.0008, 0.0311, 0.1207, 0.0075,
        0.0995, 0.0890])

max probability is torch.return_types.max(
values=tensor(0.1964),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6101e-03, 3.3444e-05, 3.3440e-05, 3.3437e-05]),
indices=tensor([  0, 771, 914, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 326 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.5371e-04, 1.3860e-05, 8.7216e-04,
8.7905e-04, 2.9027e-03, 5.3706e-04,
        6.8979e-04, 1.1084e-01, 8.3944e-02, 6.7430e-01, 2.3184e-04, 1.2231e-05,
        6.9767e-04, 6.7678e-03, 7.2283e-05, 3.2157e-04, 2.2176e-03, 3.9046e-04,
        1.1110e-01, 3.0304e-03])

max probability is torch.return_types.max(
values=tensor(0.6743),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.0886e-06, 9.8077e-09, 9.8074e-09, 9.8049e-09]),
indices=tensor([  0,  91, 896, 197]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 327 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.8988e-04, 9.7430e-01, 3.4633e-04,
4.5439e-05, 3.6542e-04, 1.4376e-04,
        1.3810e-04, 5.1686e-03, 2.1640e-03, 2.4235e-05, 1.3917e-03, 1.7379e-04,
        9.9797e-03, 3.7838e-04, 3.9925e-05, 1.4809e-03, 2.7073e-03, 1.8154e-05,
        1.8312e-04, 1.5374e-04])
```

```
max probability is torch.return_types.max(
values=tensor(0.9743),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6836e-06, 9.8775e-09, 9.8752e-09, 9.8749e-09]),
indices=tensor([  0, 316, 910, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 328 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0059, 0.0020, 0.0054, 0.0052, 0.0268,
0.0743, 0.0335, 0.0263, 0.2761,
        0.0941, 0.0552, 0.0004, 0.0434, 0.2036, 0.0027, 0.0505, 0.0101, 0.0150,
        0.0409, 0.0223])

max probability is torch.return_types.max(
values=tensor(0.2761),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8493e-04, 6.7608e-06, 6.7592e-06, 6.7560e-06]),
indices=tensor([  0, 319, 771, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 329 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.3908e-05, 1.7779e-07, 5.6276e-05,
6.3982e-05, 2.1183e-04, 1.0572e-05,
        1.1490e-05, 1.9475e-06, 1.7459e-05, 1.7246e-04, 2.6770e-04, 4.4624e-07,
        1.2351e-04, 9.8882e-01, 4.0487e-06, 1.1993e-04, 2.9549e-03, 8.6018e-04,
        5.4764e-04, 5.7108e-03])

max probability is torch.return_types.max(
values=tensor(0.9888),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.9877e-07, 1.4720e-10, 1.4717e-10, 1.4713e-10]),
indices=tensor([  0, 323, 771, 914]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 330 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.6358e-05, 1.4470e-06, 2.5624e-04,
3.6225e-04, 7.8695e-04, 1.0478e-04,
        1.3226e-04, 9.4557e-06, 9.4972e-05, 4.0996e-04, 7.1821e-04, 7.0053e-07,
        5.2598e-04, 9.8422e-01, 7.3641e-07, 1.0078e-04, 4.1801e-03, 6.7967e-05,
        2.1206e-04, 7.7390e-03])

max probability is torch.return_types.max(
values=tensor(0.9842),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6181e-06, 5.0951e-10, 5.0945e-10, 5.0944e-10]),
indices=tensor([  0, 771, 910, 914]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 331 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.5155e-05, 3.8932e-05, 8.6562e-07,
2.8872e-07, 1.3089e-06, 2.1223e-06,
        1.5322e-06, 3.5444e-04, 3.0558e-05, 4.8712e-06, 9.6510e-04, 9.9494e-01,
        3.3924e-04, 2.0350e-05, 2.5178e-05, 2.9899e-05, 2.5474e-03, 2.1829e-05,
        5.5748e-04, 8.6090e-05])

max probability is torch.return_types.max(
values=tensor(0.9949),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.7529e-07, 5.1511e-10, 5.1504e-10, 5.1484e-10]),
indices=tensor([  0, 382, 691, 230]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 332 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.5864e-04, 5.7938e-04, 1.3112e-01,
2.2223e-01, 5.0058e-01, 1.1355e-01,
        3.0353e-02, 2.0956e-05, 2.0730e-05, 6.4960e-05, 4.3504e-05, 1.6213e-06,
        4.3832e-05, 1.1071e-04, 7.8399e-05, 1.0541e-05, 5.2334e-06, 6.6665e-05,
        5.0723e-04, 7.1794e-05])

max probability is torch.return_types.max(
```

```
values=tensor(0.5006),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.0663e-06, 1.9112e-07, 1.9105e-07, 1.9091e-07]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 333 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.3417e-02, 1.5223e-03, 1.7343e-02,
1.3175e-02, 3.0369e-02, 6.5422e-01,
        1.0003e-01, 1.8478e-03, 2.0427e-02, 7.2348e-02, 9.4398e-03, 5.4416e-04,
        1.3350e-02, 1.1952e-02, 7.4143e-04, 5.2778e-03, 1.7596e-03, 1.4962e-03,
        5.7366e-03, 3.6800e-03])

max probability is torch.return_types.max(
values=tensor(0.6542),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3244e-05, 1.3855e-06, 1.3850e-06, 1.3850e-06]),
indices=tensor([  0, 771, 319, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 334 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0106, 0.0034, 0.0629, 0.1505, 0.0987,
0.2465, 0.2311, 0.0014, 0.0030,
        0.0629, 0.0311, 0.0014, 0.0165, 0.0524, 0.0014, 0.0038, 0.0046, 0.0013,
        0.0055, 0.0065])

max probability is torch.return_types.max(
values=tensor(0.2465),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.4296e-05, 4.6188e-06, 4.6175e-06, 4.6172e-06]),
indices=tensor([  0, 771, 323, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 335 :   Prediction is Correct?
```

```
No
first 20 classes probability: tensor([5.6244e-04, 5.9774e-04, 6.3578e-02,
2.2776e-01, 4.2501e-01, 2.2888e-01,
        5.2161e-02, 1.9163e-05, 2.5219e-05, 1.0124e-04, 4.5287e-05, 2.0779e-06,
        3.9002e-05, 1.8972e-04, 1.1627e-04, 1.3761e-05, 4.8781e-06, 7.7490e-05,
        5.5371e-04, 9.9107e-05])

max probability is torch.return_types.max(
values=tensor(0.4250),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6669e-06, 1.6943e-07, 1.6932e-07, 1.6921e-07]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 336 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0184e-03, 1.2235e-05, 4.6153e-04,
3.5323e-04, 9.2233e-04, 5.0292e-04,
        1.1921e-03, 6.3824e-03, 2.2747e-02, 7.1598e-01, 1.7964e-03, 6.9891e-04,
        8.2973e-03, 3.0094e-02, 6.5518e-05, 1.5599e-03, 1.9917e-01, 3.2673e-04,
        5.4954e-03, 2.8862e-03])

max probability is torch.return_types.max(
values=tensor(0.7160),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.2410e-06, 3.5228e-08, 3.5206e-08, 3.5201e-08]),
indices=tensor([  0, 323, 914, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 337 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.0210e-04, 4.0582e-04, 1.3292e-02,
1.9191e-03, 4.9811e-03, 8.2619e-01,
        1.5167e-01, 9.7000e-06, 3.8651e-04, 1.7638e-04, 5.5290e-05, 3.2121e-06,
        1.6089e-04, 6.8878e-05, 5.2320e-06, 3.2957e-05, 4.5004e-06, 5.7167e-06,
        1.8636e-05, 5.6567e-06])

max probability is torch.return_types.max(
values=tensor(0.8262),
indices=tensor(5))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7225e-08, 5.4344e-10, 5.4341e-10, 5.4328e-10]),
indices=tensor([  0, 896, 319, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 338 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0182, 0.0748, 0.0057, 0.0021, 0.0082,
0.0278, 0.0152, 0.0120, 0.0472,
        0.0045, 0.2289, 0.0131, 0.3863, 0.0570, 0.0012, 0.0352, 0.0272, 0.0022,
        0.0062, 0.0115])

max probability is torch.return_types.max(
values=tensor(0.3863),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.4543e-04, 1.6123e-05, 1.6120e-05, 1.6118e-05]),
indices=tensor([  0, 319, 771,  44]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 339 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.9819e-03, 8.5001e-04, 2.5421e-04,
1.0844e-04, 1.5372e-04, 1.2066e-03,
        6.1171e-04, 9.9847e-04, 4.3158e-04, 4.8837e-04, 1.8711e-02, 9.4853e-01,
        5.4527e-03, 5.9333e-04, 4.6521e-04, 1.8687e-04, 9.3275e-03, 4.9327e-04,
        2.1302e-03, 8.5347e-04])

max probability is torch.return_types.max(
values=tensor(0.9485),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1244e-05, 1.7026e-07, 1.7018e-07, 1.7017e-07]),
indices=tensor([  0, 914, 323, 691]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 340 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.8789e-03, 2.9055e-05, 2.6849e-05,
```

```
1.6677e-05, 3.2987e-05, 2.3490e-04,
        8.9491e-05, 1.5207e-04, 8.8281e-05, 1.2479e-04, 3.4521e-03, 9.7484e-01,
        2.1405e-03, 1.5654e-03, 3.4543e-04, 2.1792e-05, 9.4965e-03, 6.1858e-04,
        2.5915e-03, 1.2430e-03])

max probability is torch.return_types.max(
values=tensor(0.9748),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.8155e-06, 1.0502e-08, 1.0495e-08, 1.0491e-08]),
indices=tensor([  0, 914, 691, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 341 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.3866e-04, 1.0837e-04, 1.3903e-03,
8.5668e-04, 2.3519e-03, 2.3990e-04,
        2.0034e-04, 4.7618e-04, 7.0436e-04, 6.1292e-04, 9.6334e-03, 2.5925e-05,
        1.1554e-03, 7.6157e-02, 3.0511e-06, 7.2618e-04, 3.3104e-02, 6.7660e-04,
        3.0398e-02, 8.3575e-01])

max probability is torch.return_types.max(
values=tensor(0.8357),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.2031e-03, 9.6004e-08, 9.5985e-08, 9.5960e-08]),
indices=tensor([  0, 914, 337, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 342 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.6345e-04, 2.6894e-03, 2.9797e-04,
3.9695e-05, 2.7128e-04, 2.7805e-03,
        2.7998e-03, 1.5123e-02, 1.6109e-01, 9.0716e-05, 1.0565e-02, 2.7057e-04,
        8.0167e-01, 5.7054e-04, 1.6739e-05, 1.3603e-03, 1.4953e-04, 6.8436e-06,
        1.9575e-05, 2.0209e-05])

max probability is torch.return_types.max(
values=tensor(0.8017),
indices=tensor(12))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([1.4883e-07, 8.2697e-09, 8.2667e-09, 8.2635e-09]),
indices=tensor([  0, 319, 836, 794]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 343 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9524e-01, 5.6900e-05, 3.7705e-04,
1.9846e-04, 1.6298e-04, 3.3008e-03,
        4.6114e-04, 2.5254e-06, 1.3493e-05, 1.3279e-04, 1.3861e-06, 1.4075e-06,
        5.8710e-06, 7.4426e-06, 4.9327e-07, 2.1424e-07, 1.6526e-06, 6.1426e-06,
        3.1324e-06, 2.3082e-05])

max probability is torch.return_types.max(
values=tensor(0.9952),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.4322e-09, 3.4352e-11, 3.4342e-11, 3.4340e-11]),
indices=tensor([  0, 323, 910, 291]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 344 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.1802, 0.0633, 0.0141, 0.0020, 0.0071,
0.0053, 0.0083, 0.1090, 0.0880,
        0.0040, 0.0123, 0.2384, 0.1202, 0.0134, 0.0033, 0.0014, 0.0950, 0.0033,
        0.0073, 0.0060])

max probability is torch.return_types.max(
values=tensor(0.2384),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.2579e-04, 1.8398e-05, 1.8397e-05, 1.8395e-05]),
indices=tensor([  0, 391, 691, 382]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 345 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.8341e-02, 5.8073e-01, 4.4493e-02,
2.0407e-02, 2.6288e-02, 1.9643e-02,
        3.5552e-02, 9.6276e-02, 8.9669e-02, 2.6740e-02, 6.5466e-04, 1.9546e-05,
```

```
        7.7940e-03, 1.5431e-03, 6.1846e-05, 5.5405e-04, 4.5988e-04, 7.0789e-05,
        3.2350e-04, 2.2016e-04])

max probability is torch.return_types.max(
values=tensor(0.5807),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.1740e-06, 1.6416e-07, 1.6403e-07, 1.6395e-07]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 346 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0184, 0.0045, 0.0207, 0.0232, 0.0934,
0.0646, 0.0274, 0.0829, 0.1407,
        0.0718, 0.0121, 0.0007, 0.0094, 0.0504, 0.0004, 0.0070, 0.0168, 0.0069,
        0.2026, 0.1343])

max probability is torch.return_types.max(
values=tensor(0.2026),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5765e-03, 9.7727e-06, 9.7699e-06, 9.7665e-06]),
indices=tensor([  0, 914, 771,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 347 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.2450e-04, 2.8447e-04, 3.1144e-06,
8.1923e-07, 1.2474e-06, 1.0811e-05,
        8.6643e-06, 7.4552e-05, 5.0781e-06, 2.4849e-06, 3.7711e-04, 9.9772e-01,
        2.0739e-04, 4.6886e-06, 1.8723e-05, 1.4465e-06, 1.0002e-03, 4.1252e-06,
        3.5185e-05, 1.2438e-05])

max probability is torch.return_types.max(
values=tensor(0.9977),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.4448e-08, 6.4422e-11, 6.4401e-11, 6.4346e-11]),
indices=tensor([  0, 691, 382,  82]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 348 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.7040e-04, 1.6786e-04, 1.4422e-01,
3.4599e-01, 4.3379e-01, 4.4490e-02,
        2.4351e-02, 1.0082e-04, 1.2152e-04, 8.8961e-04, 9.9769e-05, 6.5458e-07,
        1.4525e-04, 2.6012e-03, 5.2429e-05, 3.8891e-05, 2.8028e-05, 1.4271e-04,
        8.2801e-04, 7.3779e-04])

max probability is torch.return_types.max(
values=tensor(0.4338),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6559e-05, 3.3494e-07, 3.3473e-07, 3.3468e-07]),
indices=tensor([  0, 319, 771, 316]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 349 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.0998e-03, 1.3628e-03, 3.4202e-02,
6.8464e-02, 7.8506e-02, 4.6934e-01,
        3.2863e-01, 1.8343e-04, 8.5487e-04, 7.1504e-03, 3.1212e-04, 1.8554e-05,
        5.6589e-04, 2.5406e-03, 1.4069e-03, 3.6065e-04, 3.8121e-05, 8.0247e-04,
        7.1146e-04, 2.4828e-04])

max probability is torch.return_types.max(
values=tensor(0.4693),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9278e-06, 2.1091e-07, 2.1080e-07, 2.1076e-07]),
indices=tensor([  0, 319, 896, 910]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 350 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.3554e-05, 2.2351e-04, 1.8380e-01,
2.0375e-01, 5.7118e-01, 3.3176e-02,
        7.6448e-03, 3.6291e-06, 2.1268e-06, 5.8432e-06, 5.8955e-06, 9.8511e-08,
        7.7540e-06, 1.2883e-05, 9.2615e-06, 5.3479e-07, 6.7632e-07, 1.0427e-05,
        1.0182e-04, 1.1502e-05])
```

```
max probability is torch.return_types.max(
values=tensor(0.5712),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.3562e-07, 9.2202e-09, 9.2178e-09, 9.2123e-09]),
indices=tensor([  0, 319, 316, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 351 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0176, 0.0019, 0.2558, 0.0653, 0.1305,
0.1492, 0.1576, 0.0039, 0.0136,
        0.0710, 0.0086, 0.0007, 0.0203, 0.0307, 0.0059, 0.0040, 0.0055, 0.0195,
        0.0129, 0.0092])

max probability is torch.return_types.max(
values=tensor(0.2558),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9757e-04, 1.6728e-05, 1.6724e-05, 1.6722e-05]),
indices=tensor([  0, 771, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 352 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0034, 0.0144, 0.0041, 0.0016, 0.0075,
0.0094, 0.0061, 0.0093, 0.0326,
        0.0043, 0.2071, 0.0041, 0.1074, 0.1393, 0.0020, 0.2510, 0.0758, 0.0110,
        0.0321, 0.0590])

max probability is torch.return_types.max(
values=tensor(0.2510),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0087e-03, 1.8503e-05, 1.8503e-05, 1.8492e-05]),
indices=tensor([  0, 319, 771, 323]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 353 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.1758e-07, 1.7139e-07, 5.5885e-08,
2.8128e-09, 3.7974e-08, 5.9419e-09,
        4.0644e-09, 4.5545e-06, 6.4604e-07, 1.0147e-07, 6.4331e-04, 7.9972e-01,
        1.6541e-04, 3.2901e-05, 1.0451e-07, 8.2232e-06, 1.9925e-01, 1.2512e-06,
        1.1266e-04, 6.1951e-05])

max probability is torch.return_types.max(
values=tensor(0.7997),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9904e-08, 2.2944e-13, 2.2939e-13, 2.2935e-13]),
indices=tensor([  0, 691, 230, 382]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 354 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.2095e-06, 3.9077e-05, 9.1635e-07,
6.9468e-07, 1.6761e-06, 5.5373e-06,
        2.7350e-06, 3.5125e-07, 1.1696e-06, 2.1066e-06, 9.9899e-01, 1.0888e-04,
        1.5325e-04, 6.7688e-05, 1.8487e-08, 5.2026e-04, 5.5637e-05, 8.4029e-07,
        1.1384e-05, 3.6395e-05])

max probability is torch.return_types.max(
values=tensor(0.9990),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8819e-08, 1.3075e-11, 1.3074e-11, 1.3041e-11]),
indices=tensor([  0, 319, 771, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 355 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.1173, 0.0030, 0.0040, 0.0005, 0.0055,
0.0061, 0.0034, 0.0108, 0.1014,
        0.0145, 0.0631, 0.0073, 0.2168, 0.1612, 0.0028, 0.0844, 0.1147, 0.0280,
        0.0277, 0.0228])

max probability is torch.return_types.max(
values=tensor(0.2168),
indices=tensor(12))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4776e-04, 4.9477e-06, 4.9471e-06, 4.9468e-06]),
indices=tensor([  0, 771, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 356 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.0974e-04, 1.2216e-02, 3.2710e-05,
4.9349e-06, 5.6330e-05, 1.7917e-04,
        1.7547e-04, 5.7082e-01, 4.0904e-01, 9.5062e-05, 2.8740e-04, 2.9570e-05,
        5.9063e-03, 4.6084e-05, 5.8946e-06, 4.6118e-04, 8.1182e-05, 2.1031e-06,
        3.9153e-05, 1.0156e-05])

max probability is torch.return_types.max(
values=tensor(0.5708),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4378e-07, 4.3505e-09, 4.3471e-09, 4.3467e-09]),
indices=tensor([  0, 836, 319, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 357 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.9681e-01, 1.4294e-03, 8.6668e-03,
1.3654e-03, 4.7740e-03, 1.7127e-03,
        2.3095e-03, 1.4352e-03, 1.1360e-03, 2.4376e-03, 1.2200e-03, 3.0231e-02,
        3.3355e-03, 9.6951e-04, 1.5514e-03, 1.3213e-04, 7.5040e-03, 1.5738e-02,
        7.8932e-03, 8.5136e-03])

max probability is torch.return_types.max(
values=tensor(0.8968),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.3315e-05, 7.8599e-07, 7.8572e-07, 7.8551e-07]),
indices=tensor([  0, 691, 382,  19]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 358 :   Prediction is Correct?
No
```

```
first 20 classes probability: tensor([1.0682e-04, 3.1331e-07, 7.1112e-06,
3.8418e-06, 4.9334e-05, 2.3089e-06,
        1.4894e-06, 8.7767e-06, 1.8983e-05, 2.4252e-05, 1.8462e-04, 1.3328e-05,
        4.6493e-06, 4.3347e-03, 9.9906e-05, 2.1572e-05, 1.6557e-03, 2.1005e-01,
        6.1419e-01, 1.6918e-01])

max probability is torch.return_types.max(
values=tensor(0.6142),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.4254e-05, 7.5175e-11, 7.5173e-11, 7.5113e-11]),
indices=tensor([  0, 158, 337,  19]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 359 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.1908e-06, 3.5701e-06, 9.5987e-01,
1.5390e-02, 1.1794e-02, 1.4896e-03,
        1.1290e-02, 6.2089e-07, 9.8094e-07, 2.8896e-05, 2.2703e-06, 6.7097e-09,
        1.0048e-05, 8.2995e-05, 7.7038e-07, 4.7640e-07, 3.3207e-06, 5.5932e-06,
        3.0737e-06, 1.1022e-05])

max probability is torch.return_types.max(
values=tensor(0.9599),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3838e-08, 5.9579e-11, 5.9571e-11, 5.9570e-11]),
indices=tensor([  0, 748, 150, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 360 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0220, 0.0074, 0.0025, 0.0012, 0.0026,
0.0058, 0.0028, 0.0143, 0.0167,
        0.0096, 0.1455, 0.3909, 0.0630, 0.0412, 0.0014, 0.0256, 0.1784, 0.0040,
        0.0198, 0.0224])

max probability is torch.return_types.max(
values=tensor(0.3909),
indices=tensor(11))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([7.7977e-04, 2.3028e-05, 2.3021e-05, 2.3020e-05]),
indices=tensor([  0, 323, 914, 910]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 361 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.4994e-05, 6.1138e-05, 3.2980e-01,
4.2749e-01, 2.1644e-01, 1.7061e-02,
        8.9555e-03, 4.4103e-06, 2.3970e-06, 6.8619e-05, 2.5776e-06, 2.0495e-08,
        8.9334e-06, 3.4618e-05, 9.0563e-06, 2.0117e-07, 3.8326e-07, 9.4041e-06,
        1.5759e-05, 8.1681e-06])

max probability is torch.return_types.max(
values=tensor(0.4275),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0957e-07, 1.8990e-09, 1.8984e-09, 1.8983e-09]),
indices=tensor([  0, 771, 319, 316]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 362 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.5185e-03, 1.3399e-04, 4.1726e-03,
7.3664e-03, 1.1648e-02, 1.0743e-02,
        1.1287e-02, 6.9663e-03, 2.9774e-02, 4.5255e-01, 5.9572e-03, 4.6334e-04,
        8.6985e-03, 3.0562e-01, 8.4655e-04, 2.5623e-03, 3.6887e-02, 8.1313e-03,
        3.5804e-02, 4.8746e-02])

max probability is torch.return_types.max(
values=tensor(0.4526),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4060e-04, 1.9794e-06, 1.9792e-06, 1.9786e-06]),
indices=tensor([  0, 914, 771, 323]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 363 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.1879e-03, 9.6181e-01, 4.9136e-04,
1.0792e-04, 5.4874e-04, 2.7693e-04,
```

```
        4.3615e-04, 2.0874e-02, 5.7849e-03, 4.8847e-05, 3.4893e-04, 5.6394e-04,
        4.3406e-03, 1.5620e-04, 4.8269e-05, 2.1061e-04, 1.5553e-03, 1.0462e-05,
        1.2416e-04, 5.2646e-05])

max probability is torch.return_types.max(
values=tensor(0.9618),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7824e-06, 2.5521e-08, 2.5503e-08, 2.5502e-08]),
indices=tensor([  0, 316, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 364 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.5885e-05, 1.0881e-04, 6.6290e-01,
1.8086e-01, 1.2521e-01, 1.1317e-02,
        1.9035e-02, 1.0119e-05, 9.3142e-06, 1.2516e-04, 1.1031e-05, 9.2092e-08,
        3.8246e-05, 1.5971e-04, 1.5329e-05, 2.7695e-06, 6.5309e-06, 3.2225e-05,
        3.7308e-05, 3.5081e-05])

max probability is torch.return_types.max(
values=tensor(0.6629),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.2683e-07, 1.3398e-08, 1.3395e-08, 1.3395e-08]),
indices=tensor([  0, 910, 771, 466]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 365 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.1305e-11, 2.6067e-11, 1.1579e-10,
2.7715e-11, 1.3473e-09, 2.8278e-12,
        2.0715e-12, 5.6738e-10, 2.5141e-10, 2.3088e-11, 6.0506e-09, 5.9132e-14,
        2.5685e-11, 1.3318e-07, 3.1979e-17, 7.3571e-12, 2.6512e-05, 2.4291e-10,
        4.5808e-06, 9.9995e-01])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([1.5886e-05, 2.1424e-20, 2.1393e-20, 2.1393e-20]),
indices=tensor([  0, 655, 691, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 366 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.5049e-03, 1.6977e-04, 8.7032e-01,
4.0110e-02, 2.4543e-02, 5.8120e-03,
        3.2807e-02, 7.9581e-04, 1.9486e-03, 1.6142e-02, 4.2807e-04, 3.2149e-06,
        3.2853e-03, 1.3778e-03, 2.7222e-05, 3.6079e-05, 1.9826e-04, 1.9460e-04,
        7.8579e-05, 1.8789e-04])

max probability is torch.return_types.max(
values=tensor(0.8703),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5216e-06, 3.6284e-08, 3.6283e-08, 3.6277e-08]),
indices=tensor([  0, 572, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 367 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0948, 0.0048, 0.0103, 0.0136, 0.0267,
0.0734, 0.0329, 0.0028, 0.0143,
        0.0319, 0.1126, 0.0111, 0.0360, 0.3606, 0.0020, 0.0218, 0.0291, 0.0123,
        0.0238, 0.0527])

max probability is torch.return_types.max(
values=tensor(0.3606),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.9761e-04, 3.3078e-05, 3.3069e-05, 3.3063e-05]),
indices=tensor([  0, 771, 914, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 368 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.4554e-01, 1.0306e-03, 4.3400e-01,
6.1533e-02, 2.6932e-02, 2.7873e-02,
        1.1505e-02, 6.5005e-03, 1.1557e-02, 1.5702e-01, 8.6091e-04, 6.6363e-05,
        6.2951e-03, 2.7491e-03, 1.8325e-04, 1.8518e-04, 8.6225e-04, 7.9452e-04,
```

```
        1.7714e-03, 2.0398e-03])

max probability is torch.return_types.max(
values=tensor(0.4340),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7950e-05, 7.3840e-07, 7.3809e-07, 7.3787e-07]),
indices=tensor([  0, 771, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 369 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.7538e-05, 2.5380e-05, 6.7536e-01,
2.3679e-01, 8.1485e-02, 4.3003e-03,
        1.9143e-03, 1.4354e-06, 8.3447e-07, 4.5300e-05, 3.4573e-07, 2.1441e-09,
        1.1857e-06, 4.1427e-06, 5.9472e-07, 1.3791e-08, 2.0718e-07, 2.3759e-06,
        3.5654e-06, 3.1140e-06])

max probability is torch.return_types.max(
values=tensor(0.6754),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2254e-08, 7.4507e-11, 7.4470e-11, 7.4469e-11]),
indices=tensor([  0, 771, 910, 466]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 370 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0070, 0.0033, 0.0595, 0.0284, 0.0560,
0.2175, 0.3148, 0.0201, 0.1075,
        0.1429, 0.0054, 0.0004, 0.0140, 0.0073, 0.0008, 0.0025, 0.0019, 0.0021,
        0.0032, 0.0020])

max probability is torch.return_types.max(
values=tensor(0.3148),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.4744e-05, 3.6009e-06, 3.6005e-06, 3.5996e-06]),
indices=tensor([  0, 896, 319, 771]))
End*****************************************************************************
```

```
**************************************************************************
Predicting Test Sample 371 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0135, 0.0336, 0.0279, 0.0056, 0.0206,
0.0189, 0.0132, 0.0201, 0.0386,
        0.0078, 0.0762, 0.0086, 0.0478, 0.0542, 0.0066, 0.0908, 0.1034, 0.0564,
        0.1361, 0.1684])

max probability is torch.return_types.max(
values=tensor(0.1684),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.7374e-03, 4.9460e-05, 4.9459e-05, 4.9457e-05]),
indices=tensor([  0, 323, 914, 197]))
End*************************************************************************

**************************************************************************
Predicting Test Sample 372 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0058, 0.0005, 0.0063, 0.0089, 0.0258,
0.0231, 0.0162, 0.0042, 0.0157,
        0.0788, 0.0360, 0.0048, 0.0259, 0.3636, 0.0100, 0.0194, 0.0712, 0.0482,
        0.1434, 0.0801])

max probability is torch.return_types.max(
values=tensor(0.3636),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.2970e-04, 1.1932e-05, 1.1931e-05, 1.1926e-05]),
indices=tensor([  0, 914, 771, 323]))
End*************************************************************************

**************************************************************************
Predicting Test Sample 373 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4955e-05, 2.1801e-06, 6.8892e-07,
1.2068e-07, 2.9767e-07, 7.2331e-07,
        3.8598e-07, 2.8691e-06, 3.3772e-07, 2.5128e-07, 1.7360e-04, 9.9638e-01,
        9.0132e-04, 1.9092e-05, 7.1188e-05, 4.2965e-07, 2.3694e-03, 5.5412e-06,
        4.6239e-05, 1.2181e-05])

max probability is torch.return_types.max(
values=tensor(0.9964),
```

```
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0704e-08, 6.3575e-12, 6.3519e-12, 6.3480e-12]),
indices=tensor([  0, 691,  82, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 374 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.2001e-02, 3.3279e-03, 1.9694e-02,
2.5482e-02, 4.3929e-02, 6.5867e-01,
        8.8602e-02, 5.6590e-04, 4.3927e-03, 1.1412e-02, 2.3819e-02, 1.5977e-03,
        1.5853e-02, 2.7126e-02, 4.4769e-04, 2.7089e-03, 3.5547e-03, 1.0260e-03,
        6.8962e-03, 7.6367e-03])

max probability is torch.return_types.max(
values=tensor(0.6587),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.6844e-05, 1.2770e-06, 1.2764e-06, 1.2762e-06]),
indices=tensor([  0, 771, 914, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 375 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9987e-01, 3.1744e-06, 2.4398e-05,
9.0200e-06, 5.9682e-06, 7.2771e-05,
        1.0388e-05, 1.4495e-07, 6.9570e-07, 6.2614e-06, 7.6895e-09, 1.2333e-08,
        2.7028e-07, 1.0026e-07, 7.7763e-09, 1.1421e-09, 7.0104e-09, 4.6971e-08,
        1.5861e-08, 6.6613e-08])

max probability is torch.return_types.max(
values=tensor(0.9999),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.3938e-12, 1.6191e-14, 1.6186e-14, 1.6181e-14]),
indices=tensor([  0, 323, 910, 291]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 376 :   Prediction is Correct?
```

```
No
first 20 classes probability: tensor([4.7658e-03, 6.6348e-01, 1.3936e-02,
1.3692e-03, 5.1591e-03, 3.7567e-03,
        8.7992e-03, 2.0646e-02, 2.2297e-02, 6.2043e-04, 2.6931e-02, 1.1332e-02,
        1.3500e-01, 9.0704e-03, 1.0208e-03, 2.2411e-02, 3.5338e-02, 1.0358e-03,
        4.1699e-03, 5.8486e-03])

max probability is torch.return_types.max(
values=tensor(0.6635),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8165e-04, 2.9919e-06, 2.9908e-06, 2.9897e-06]),
indices=tensor([  0, 910, 323, 726]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 377 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0012, 0.0007, 0.0013, 0.0011, 0.0036,
0.0023, 0.0009, 0.0035, 0.0049,
        0.0040, 0.0414, 0.0354, 0.0483, 0.0779, 0.0489, 0.0226, 0.0663, 0.0933,
        0.4325, 0.1025])

max probability is torch.return_types.max(
values=tensor(0.4325),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.2247e-04, 6.5977e-06, 6.5937e-06, 6.5925e-06]),
indices=tensor([  0, 914, 771, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 378 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.6650e-04, 7.0132e-04, 1.6961e-05,
6.5738e-06, 8.4932e-05, 3.2603e-04,
        7.8690e-05, 2.5632e-04, 3.7804e-03, 1.5331e-04, 4.4541e-01, 2.5876e-04,
        2.2580e-02, 2.8015e-02, 1.0157e-05, 4.7294e-01, 1.8044e-02, 1.5447e-04,
        1.5915e-03, 5.2987e-03])

max probability is torch.return_types.max(
values=tensor(0.4729),
indices=tensor(15))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.9272e-06, 2.1977e-08, 2.1937e-08, 2.1918e-08]),
indices=tensor([  0, 319, 771, 316]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 379 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9941e-01, 1.9184e-05, 1.9385e-04,
4.5723e-05, 9.8725e-06, 2.7501e-04,
        3.9387e-05, 1.2552e-07, 3.6125e-07, 3.5817e-06, 1.2624e-08, 3.7101e-07,
        4.2461e-07, 3.2006e-08, 3.1409e-08, 3.7259e-10, 2.1269e-08, 9.4464e-08,
        2.6279e-08, 5.6264e-08])

max probability is torch.return_types.max(
values=tensor(0.9994),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1511e-11, 5.2652e-14, 5.2619e-14, 5.2597e-14]),
indices=tensor([  0, 271, 323, 914]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 380 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0000e+00, 3.5168e-11, 7.4357e-08,
1.3103e-09, 3.0992e-08, 2.0071e-09,
        4.2462e-09, 2.1542e-10, 5.9481e-09, 6.4514e-08, 3.2788e-11, 1.5148e-11,
        9.1662e-09, 9.6637e-09, 2.5180e-10, 4.7341e-12, 5.0741e-11, 5.5813e-08,
        6.8324e-10, 8.5856e-10])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1758e-15, 1.0546e-18, 1.0536e-18, 1.0521e-18]),
indices=tensor([  0, 323, 910, 771]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 381 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.3304e-05, 1.4605e-05, 1.9810e-03,
```

```
8.5112e-04, 1.9767e-03, 8.7167e-05,
        7.5948e-05, 3.0617e-04, 7.4369e-04, 6.3777e-04, 2.4767e-03, 1.6116e-05,
        2.7963e-03, 7.7525e-01, 2.8793e-05, 1.1130e-03, 4.9721e-02, 2.1032e-03,
        3.8606e-02, 1.2084e-01])

max probability is torch.return_types.max(
values=tensor(0.7752),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6545e-04, 3.2555e-08, 3.2536e-08, 3.2535e-08]),
indices=tensor([  0, 914, 726, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 382 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.8378e-03, 4.1065e-04, 1.5629e-02,
6.6349e-03, 1.9896e-02, 6.5530e-03,
        1.0248e-02, 3.0850e-03, 8.3999e-03, 2.6973e-02, 2.1328e-02, 3.1076e-03,
        1.0750e-02, 1.2523e-01, 3.4095e-02, 7.7599e-03, 1.8426e-02, 5.0125e-01,
        1.1678e-01, 4.8058e-02])

max probability is torch.return_types.max(
values=tensor(0.5012),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.4277e-04, 8.4546e-06, 8.4542e-06, 8.4534e-06]),
indices=tensor([  0, 337, 748, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 383 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.2106, 0.0009, 0.0154, 0.0089, 0.0195,
0.0136, 0.0279, 0.0093, 0.0295,
        0.3144, 0.0234, 0.0087, 0.0932, 0.0826, 0.0194, 0.0064, 0.0253, 0.0471,
        0.0171, 0.0152])

max probability is torch.return_types.max(
values=tensor(0.3144),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([1.2333e-04, 1.1828e-05, 1.1827e-05, 1.1822e-05]),
indices=tensor([  0, 323, 771, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 384 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.0988e-07, 5.5195e-08, 1.2486e-06,
1.5741e-06, 2.4919e-05, 2.3555e-07,
        1.7218e-07, 6.1038e-06, 2.1296e-06, 5.4242e-06, 8.4730e-06, 2.0995e-09,
        1.7448e-07, 1.3990e-04, 1.7424e-10, 2.2419e-07, 2.3765e-04, 1.9893e-06,
        8.3095e-03, 9.9084e-01])

max probability is torch.return_types.max(
values=tensor(0.9908),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.1654e-04, 5.5722e-13, 5.5693e-13, 5.5615e-13]),
indices=tensor([  0, 655, 337, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 385 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.9261e-03, 2.1486e-05, 6.7867e-04,
2.7957e-04, 1.6633e-03, 2.0574e-04,
        2.1531e-04, 2.8159e-04, 1.0389e-03, 1.8849e-03, 1.6178e-03, 2.0599e-04,
        1.4346e-03, 1.3434e-01, 1.3547e-02, 1.2926e-03, 6.4442e-03, 7.1121e-01,
        9.6856e-02, 2.4700e-02])

max probability is torch.return_types.max(
values=tensor(0.7112),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.1783e-05, 1.2041e-07, 1.2037e-07, 1.2033e-07]),
indices=tensor([  0, 748, 337, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 386 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.1691e-05, 2.9278e-07, 2.6167e-06,
2.0225e-06, 2.9128e-05, 6.2278e-06,
        7.3683e-06, 4.9032e-06, 1.8772e-05, 2.2343e-05, 2.3197e-05, 1.7015e-05,
```

```
        2.4599e-05, 1.9306e-03, 4.0515e-02, 4.6268e-05, 1.3434e-04, 9.3939e-01,
        1.6872e-02, 9.2340e-04])

max probability is torch.return_types.max(
values=tensor(0.9394),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3186e-07, 9.2641e-11, 9.2629e-11, 9.2544e-11]),
indices=tensor([  0, 748, 337, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 387 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0028, 0.0005, 0.0072, 0.0282, 0.0498,
0.0166, 0.0166, 0.0041, 0.0051,
        0.0298, 0.0032, 0.0004, 0.0013, 0.1029, 0.0035, 0.0022, 0.0109, 0.0362,
        0.3316, 0.3259])

max probability is torch.return_types.max(
values=tensor(0.3316),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.7757e-03, 1.5137e-05, 1.5136e-05, 1.5134e-05]),
indices=tensor([  0, 337, 914,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 388 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9126e-01, 2.0072e-05, 1.7165e-03,
4.5033e-04, 1.7793e-03, 1.4560e-03,
        1.2413e-03, 3.2620e-05, 2.6234e-04, 7.9675e-04, 1.5819e-05, 3.9714e-06,
        1.0697e-04, 4.5039e-04, 5.5737e-06, 5.3017e-06, 1.2151e-05, 1.8483e-04,
        4.9529e-05, 1.5397e-04])

max probability is torch.return_types.max(
values=tensor(0.9913),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1385e-07, 1.0382e-09, 1.0381e-09, 1.0379e-09]),
indices=tensor([  0, 323, 910, 771]))
```

```
End**************************************************************************

******************************************************************************
Predicting Test Sample 389 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.5240e-04, 8.8352e-01, 7.2329e-06,
1.0566e-06, 2.8946e-05, 9.5904e-06,
        5.6090e-06, 1.0893e-01, 7.0174e-03, 5.9991e-06, 1.1122e-05, 1.5250e-07,
        1.2263e-04, 4.0101e-06, 2.2284e-07, 1.0256e-04, 2.7666e-05, 8.0380e-08,
        4.5664e-05, 4.0659e-06])

max probability is torch.return_types.max(
values=tensor(0.8835),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.6523e-09, 3.7431e-12, 3.7331e-12, 3.7311e-12]),
indices=tensor([  0, 316, 319, 263]))
End**************************************************************************

******************************************************************************
Predicting Test Sample 390 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0191, 0.0046, 0.0028, 0.0033, 0.0053,
0.0201, 0.0092, 0.0061, 0.0101,
        0.0383, 0.1475, 0.3635, 0.0656, 0.0412, 0.0141, 0.0247, 0.1083, 0.0080,
        0.0470, 0.0144])

max probability is torch.return_types.max(
values=tensor(0.3635),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.2559e-04, 4.8822e-05, 4.8821e-05, 4.8806e-05]),
indices=tensor([  0, 914, 323, 771]))
End**************************************************************************

******************************************************************************
Predicting Test Sample 391 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0053, 0.3630, 0.0040, 0.0009, 0.0037,
0.0100, 0.0078, 0.0092, 0.0186,
        0.0009, 0.1615, 0.0159, 0.1244, 0.0288, 0.0012, 0.1741, 0.0431, 0.0016,
        0.0063, 0.0100])

max probability is torch.return_types.max(
```

```
values=tensor(0.3630),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9708e-04, 9.9538e-06, 9.9495e-06, 9.9487e-06]),
indices=tensor([  0, 910, 323, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 392 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.2721e-04, 1.3344e-04, 9.0818e-04,
3.2666e-04, 3.3320e-03, 2.0256e-04,
        1.5378e-04, 1.5142e-03, 1.4014e-03, 3.6243e-04, 2.7589e-03, 3.0642e-05,
        8.1062e-04, 8.0127e-03, 1.2694e-05, 4.5759e-04, 9.7681e-03, 4.3918e-03,
        8.7252e-02, 8.7394e-01])

max probability is torch.return_types.max(
values=tensor(0.8739),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5616e-03, 5.1791e-08, 5.1773e-08, 5.1769e-08]),
indices=tensor([  0, 337, 655, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 393 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.8845e-05, 7.7763e-06, 6.9661e-06,
4.0986e-06, 5.7365e-05, 1.6591e-05,
        6.5782e-06, 3.6717e-04, 1.1479e-03, 1.8947e-04, 1.2857e-03, 3.4273e-05,
        4.3119e-05, 1.7865e-03, 3.8204e-04, 2.1844e-03, 1.3686e-03, 4.8474e-02,
        9.2499e-01, 1.7574e-02])

max probability is torch.return_types.max(
values=tensor(0.9250),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0223e-05, 2.2084e-09, 2.2081e-09, 2.2072e-09]),
indices=tensor([  0, 914, 337, 771]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 394 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.0000e-08, 5.1832e-09, 5.7463e-08,
8.5421e-08, 1.9025e-06, 8.2489e-08,
        2.1576e-08, 3.3610e-05, 2.4815e-05, 3.0029e-05, 1.9507e-06, 6.3297e-09,
        4.8654e-08, 5.4384e-06, 1.1796e-07, 6.2567e-07, 3.5113e-06, 1.3149e-05,
        9.9833e-01, 1.5566e-03])

max probability is torch.return_types.max(
values=tensor(0.9983),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.1414e-07, 1.0356e-13, 1.0351e-13, 1.0351e-13]),
indices=tensor([  0, 914, 197, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 395 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.5642e-10, 2.2743e-10, 2.1308e-10,
1.0928e-10, 1.1410e-09, 1.5407e-11,
        1.2383e-11, 5.5583e-09, 1.4315e-09, 4.0750e-10, 6.4714e-07, 2.0735e-09,
        1.0050e-09, 1.9642e-06, 9.3997e-14, 1.0837e-09, 1.0725e-03, 2.7549e-08,
        1.1970e-04, 9.9870e-01])

max probability is torch.return_types.max(
values=tensor(0.9987),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0386e-04, 2.2553e-17, 2.2544e-17, 2.2504e-17]),
indices=tensor([  0, 655, 691, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 396 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.6176e-01, 1.2252e-03, 1.2754e-03,
9.7457e-05, 4.5104e-04, 7.2761e-04,
        5.8234e-04, 1.2572e-03, 7.8011e-03, 8.1862e-04, 3.1731e-04, 8.6955e-04,
        2.1200e-02, 5.1940e-04, 1.1644e-04, 1.8730e-04, 4.9887e-04, 1.0650e-04,
        5.7083e-05, 9.0320e-05])

max probability is torch.return_types.max(
values=tensor(0.9618),
```

```
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.9640e-07, 4.3253e-08, 4.3244e-08, 4.3217e-08]),
indices=tensor([  0, 910, 323, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 397 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.2401e-03, 5.0084e-04, 7.9725e-02,
4.8314e-01, 1.3801e-01, 2.3091e-01,
        5.8829e-02, 5.7460e-05, 6.1526e-05, 1.5451e-03, 4.4877e-05, 5.1351e-06,
        7.8877e-05, 5.3080e-04, 3.7648e-04, 7.5266e-06, 6.4153e-06, 1.8579e-04,
        3.2500e-04, 2.0105e-04])

max probability is torch.return_types.max(
values=tensor(0.4831),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.2625e-06, 2.3083e-07, 2.3075e-07, 2.3073e-07]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 398 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0060, 0.0017, 0.0025, 0.0009, 0.0048,
0.0015, 0.0006, 0.0676, 0.0668,
        0.0237, 0.0308, 0.0030, 0.0125, 0.0243, 0.0008, 0.0588, 0.1011, 0.0094,
        0.5272, 0.0539])

max probability is torch.return_types.max(
values=tensor(0.5272),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.5369e-04, 1.5151e-06, 1.5149e-06, 1.5148e-06]),
indices=tensor([  0, 323, 914, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 399 :   Prediction is Correct?
No
```

```
first 20 classes probability: tensor([0.0984, 0.0087, 0.0327, 0.0215, 0.0554,
0.1272, 0.0842, 0.0150, 0.0493,
        0.0404, 0.0108, 0.0055, 0.0340, 0.0519, 0.1214, 0.0161, 0.0089, 0.0768,
        0.0316, 0.0098])

max probability is torch.return_types.max(
values=tensor(0.1272),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([0.0003, 0.0001, 0.0001, 0.0001]),
indices=tensor([  0, 323, 910, 771]))
End*****************************************************************************

currently at 400 current time is 12.552746534347534
*****************************************************************************
Predicting Test Sample 400 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0325, 0.0007, 0.0056, 0.0048, 0.0098,
0.0183, 0.0108, 0.0062, 0.0261,
        0.2713, 0.1532, 0.0219, 0.2285, 0.0686, 0.0030, 0.0183, 0.0756, 0.0050,
        0.0217, 0.0129])

max probability is torch.return_types.max(
values=tensor(0.2713),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.7636e-05, 5.3116e-06, 5.3074e-06, 5.3068e-06]),
indices=tensor([  0, 771, 914, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 401 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.2339e-03, 2.3013e-02, 3.5981e-03,
4.1747e-04, 4.1050e-03, 3.8630e-03,
        2.5741e-03, 1.2490e-02, 3.7543e-02, 2.2110e-03, 1.5382e-01, 7.7930e-03,
        6.4712e-02, 4.1601e-02, 5.8734e-04, 4.5347e-01, 1.1770e-01, 6.0964e-03,
        2.9302e-02, 2.3269e-02])

max probability is torch.return_types.max(
values=tensor(0.4535),
indices=tensor(15))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([1.8837e-04, 2.5602e-06, 2.5597e-06, 2.5587e-06]),
indices=tensor([  0, 910, 323, 771]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 402 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.8848e-03, 4.6018e-03, 5.0254e-04,
8.9187e-05, 7.5346e-04, 1.0355e-02,
        3.6704e-03, 2.9405e-02, 5.6656e-01, 1.3970e-03, 5.4461e-02, 1.7775e-04,
        2.8093e-01, 1.1284e-02, 3.4503e-05, 3.1283e-02, 7.9145e-04, 6.6736e-05,
        2.6861e-04, 3.8746e-04])

max probability is torch.return_types.max(
values=tensor(0.5666),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7309e-06, 1.0221e-07, 1.0213e-07, 1.0203e-07]),
indices=tensor([  0, 319, 836, 794]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 403 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.3700e-02, 6.8079e-03, 2.1624e-01,
1.7127e-01, 2.5534e-01, 1.1457e-01,
        6.9585e-02, 2.0186e-02, 3.2298e-02, 5.6955e-02, 2.2456e-03, 4.8445e-05,
        6.1974e-03, 8.1877e-03, 8.6099e-04, 1.2833e-03, 4.1439e-04, 2.3263e-03,
        5.7958e-03, 2.3848e-03])

max probability is torch.return_types.max(
values=tensor(0.2553),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.9321e-05, 3.4430e-06, 3.4414e-06, 3.4410e-06]),
indices=tensor([  0, 319, 316, 896]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 404 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.5004e-04, 1.3402e-03, 2.4958e-01,
2.3975e-01, 4.3470e-01, 4.9782e-02,
```

```
        2.1231e-02, 7.7454e-05, 6.1294e-05, 1.7700e-04, 1.2071e-04, 4.6027e-06,
        1.9025e-04, 2.5482e-04, 1.8853e-04, 3.2367e-05, 2.4262e-05, 2.1705e-04,
        8.1551e-04, 1.7475e-04])

max probability is torch.return_types.max(
values=tensor(0.4347),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3550e-05, 7.6211e-07, 7.6204e-07, 7.6164e-07]),
indices=tensor([  0, 319, 316, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 405 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.6002e-04, 1.3789e-04, 6.9393e-03,
6.4153e-03, 1.6686e-02, 6.5860e-03,
        4.4020e-03, 3.3385e-02, 9.8972e-02, 2.2755e-01, 3.2347e-03, 3.2731e-05,
        1.6000e-03, 6.0134e-02, 5.7105e-05, 2.7841e-03, 2.4416e-02, 1.4761e-03,
        4.1395e-01, 8.8904e-02])

max probability is torch.return_types.max(
values=tensor(0.4139),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.5674e-04, 6.6068e-07, 6.6045e-07, 6.6039e-07]),
indices=tensor([  0, 914, 896, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 406 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.7589e-03, 1.1305e-02, 2.2631e-03,
7.3179e-05, 9.6939e-04, 5.8121e-04,
        5.5146e-04, 5.7703e-02, 8.7120e-02, 2.3398e-03, 4.2401e-02, 2.2253e-04,
        7.3003e-02, 8.9681e-03, 9.3830e-06, 6.1417e-01, 9.0493e-02, 5.5137e-05,
        1.6643e-03, 3.3269e-03])

max probability is torch.return_types.max(
values=tensor(0.6142),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([5.3805e-06, 2.1802e-08, 2.1795e-08, 2.1792e-08]),
indices=tensor([  0, 910, 319, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 407 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0046, 0.0016, 0.0022, 0.0018, 0.0056,
0.0050, 0.0038, 0.0030, 0.0122,
        0.0220, 0.0868, 0.0036, 0.0279, 0.3584, 0.0006, 0.0703, 0.3168, 0.0046,
        0.0197, 0.0422])

max probability is torch.return_types.max(
values=tensor(0.3584),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.7383e-04, 7.2994e-06, 7.2992e-06, 7.2960e-06]),
indices=tensor([  0, 323, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 408 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.7980e-03, 4.7056e-05, 1.6317e-03,
7.1978e-04, 2.8478e-03, 7.4108e-04,
        3.9284e-04, 2.2755e-01, 2.9433e-01, 3.9380e-01, 6.6518e-04, 4.5222e-06,
        2.7862e-03, 7.5808e-03, 6.7250e-05, 2.3588e-03, 1.2486e-03, 2.8615e-04,
        5.8064e-02, 2.0721e-03])

max probability is torch.return_types.max(
values=tensor(0.3938),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3585e-06, 1.1343e-08, 1.1333e-08, 1.1331e-08]),
indices=tensor([  0, 896, 316, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 409 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.7929e-04, 1.0196e-04, 4.1081e-03,
3.3103e-03, 1.0266e-02, 3.2004e-03,
        2.4170e-03, 9.6537e-02, 1.4572e-01, 3.5540e-01, 2.0132e-03, 7.4102e-05,
        2.7823e-03, 1.9101e-02, 4.4187e-04, 2.8796e-03, 7.0092e-03, 2.9196e-03,
```

```
        3.2653e-01, 1.4080e-02])

max probability is torch.return_types.max(
values=tensor(0.3554),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.6605e-05, 3.6451e-07, 3.6437e-07, 3.6430e-07]),
indices=tensor([  0, 896,  91, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 410 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.6674e-08, 2.8821e-06, 1.4005e-09,
1.5057e-10, 1.2925e-08, 7.1257e-09,
        1.8592e-09, 1.9359e-05, 1.8876e-05, 1.0437e-07, 3.3506e-03, 5.2560e-06,
        2.1902e-04, 7.2047e-06, 1.4141e-07, 9.9536e-01, 9.5900e-04, 3.2906e-07,
        4.1211e-05, 1.5416e-05])

max probability is torch.return_types.max(
values=tensor(0.9954),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6343e-10, 1.2238e-13, 1.2219e-13, 1.2209e-13]),
indices=tensor([  0, 319, 316, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 411 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.4261e-06, 1.6314e-06, 1.9704e-07,
7.4338e-08, 1.1435e-07, 5.7108e-07,
        2.5473e-07, 1.4355e-05, 1.2454e-06, 1.2672e-06, 3.6592e-04, 9.9864e-01,
        2.6998e-04, 5.6785e-06, 1.2101e-05, 4.2237e-06, 6.3683e-04, 2.9608e-06,
        3.1118e-05, 5.4674e-06])

max probability is torch.return_types.max(
values=tensor(0.9986),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.0901e-09, 1.0983e-11, 1.0982e-11, 1.0976e-11]),
indices=tensor([  0, 914, 323, 691]))
```

```
End**********************************************************************

**********************************************************************
Predicting Test Sample 412 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.2964e-04, 3.8531e-04, 1.5728e-01,
2.1115e-01, 5.4528e-01, 7.0303e-02,
        1.5011e-02, 7.0771e-06, 4.9691e-06, 1.4530e-05, 1.4839e-05, 4.7444e-07,
        1.6121e-05, 3.1815e-05, 2.9832e-05, 2.0723e-06, 1.6058e-06, 2.4143e-05,
        2.5052e-04, 2.4181e-05])

max probability is torch.return_types.max(
values=tensor(0.5453),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2893e-06, 4.0410e-08, 4.0404e-08, 4.0372e-08]),
indices=tensor([  0, 319, 316, 957]))
End**********************************************************************

**********************************************************************
Predicting Test Sample 413 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.3431e-03, 2.4722e-02, 5.3189e-04,
1.5146e-04, 7.5286e-04, 9.5998e-03,
        9.4733e-03, 1.9033e-01, 7.1879e-01, 1.5244e-03, 2.1472e-03, 1.0753e-04,
        3.3168e-02, 3.4909e-04, 2.9425e-05, 7.7209e-04, 7.8654e-05, 1.6108e-05,
        5.0631e-05, 2.7782e-05])

max probability is torch.return_types.max(
values=tensor(0.7188),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.2147e-07, 4.0875e-08, 4.0870e-08, 4.0840e-08]),
indices=tensor([  0, 836, 319, 794]))
End**********************************************************************

**********************************************************************
Predicting Test Sample 414 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.2550e-04, 7.0579e-04, 1.7684e-01,
2.1697e-01, 5.1756e-01, 6.7084e-02,
        1.9589e-02, 2.0101e-05, 1.5033e-05, 3.7244e-05, 4.0657e-05, 1.3312e-06,
        4.5942e-05, 7.9416e-05, 6.0951e-05, 6.9767e-06, 5.4441e-06, 5.3985e-05,
        4.2678e-04, 5.6104e-05])
```

```
max probability is torch.return_types.max(
values=tensor(0.5176),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.1337e-06, 1.7716e-07, 1.7712e-07, 1.7701e-07]),
indices=tensor([  0, 319, 316, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 415 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.7300e-05, 2.3947e-04, 2.9095e-01,
2.9948e-01, 3.8448e-01, 1.9987e-02,
        4.6874e-03, 3.0046e-06, 1.5856e-06, 7.1877e-06, 2.1206e-06, 1.9988e-08,
        4.8535e-06, 1.0325e-05, 2.4692e-06, 1.6409e-07, 4.9238e-07, 3.5046e-06,
        2.1900e-05, 8.8558e-06])

max probability is torch.return_types.max(
values=tensor(0.3845),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.0808e-07, 3.2056e-09, 3.2040e-09, 3.2019e-09]),
indices=tensor([  0, 319, 316, 720]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 416 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0542, 0.4935, 0.0075, 0.0020, 0.0051,
0.0229, 0.0086, 0.0104, 0.0209,
        0.0022, 0.0580, 0.0472, 0.1166, 0.0272, 0.0036, 0.0381, 0.0425, 0.0027,
        0.0055, 0.0072])

max probability is torch.return_types.max(
values=tensor(0.4935),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.3320e-04, 2.5213e-05, 2.5210e-05, 2.5192e-05]),
indices=tensor([  0, 910, 323,  44]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 417 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.6548e-05, 2.0008e-04, 1.6715e-01,
3.5942e-01, 4.0417e-01, 5.2484e-02,
        1.6273e-02, 4.9410e-06, 4.6461e-06, 3.2326e-05, 6.3781e-06, 3.7996e-08,
        8.4186e-06, 5.3099e-05, 7.5157e-06, 1.1027e-06, 6.7491e-07, 8.7467e-06,
        5.4301e-05, 2.0062e-05])

max probability is torch.return_types.max(
values=tensor(0.4042),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.0125e-07, 9.6127e-09, 9.6042e-09, 9.5964e-09]),
indices=tensor([  0, 319, 316, 771]))
End********************************************************************************

**********************************************************************************
Predicting Test Sample 418 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1686e-03, 2.8630e-02, 1.0104e-04,
1.9717e-05, 1.4621e-04, 1.1553e-04,
        1.2122e-04, 7.9493e-01, 1.7146e-01, 1.1219e-04, 1.5025e-04, 4.1528e-05,
        2.7418e-03, 3.6306e-05, 8.9744e-06, 8.1830e-05, 7.3389e-05, 2.0037e-06,
        3.9908e-05, 9.0782e-06])

max probability is torch.return_types.max(
values=tensor(0.7949),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9013e-07, 5.0039e-09, 5.0033e-09, 5.0024e-09]),
indices=tensor([  0, 316, 836, 319]))
End********************************************************************************

**********************************************************************************
Predicting Test Sample 419 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.4031e-04, 6.3432e-04, 7.6188e-05,
9.3412e-06, 7.0820e-05, 3.1322e-05,
        5.1951e-05, 9.8793e-03, 4.7781e-03, 2.0175e-04, 5.6473e-02, 4.9284e-01,
        2.8848e-02, 2.2560e-03, 1.3237e-05, 2.0017e-03, 3.9712e-01, 8.3137e-05,
        1.3967e-03, 2.2894e-03])

max probability is torch.return_types.max(
```

```
values=tensor(0.4928),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.7143e-05, 7.5018e-08, 7.4992e-08, 7.4930e-08]),
indices=tensor([  0, 323, 910, 197]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 420 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.8056e-05, 2.0739e-05, 9.0157e-05,
1.0909e-04, 4.0409e-04, 3.1075e-05,
        2.3563e-05, 2.7204e-04, 2.2443e-04, 1.4719e-04, 2.6754e-03, 1.4599e-05,
        9.5823e-05, 6.2229e-03, 2.3327e-06, 1.1191e-04, 4.9280e-03, 6.2259e-04,
        1.6074e-01, 8.1821e-01])

max probability is torch.return_types.max(
values=tensor(0.8182),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.0271e-03, 1.0651e-08, 1.0644e-08, 1.0643e-08]),
indices=tensor([  0, 337, 655, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 421 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0091, 0.0030, 0.1407, 0.1262, 0.2147,
0.0727, 0.0422, 0.0104, 0.0145,
        0.0247, 0.0223, 0.0015, 0.0269, 0.0650, 0.0036, 0.0104, 0.0081, 0.0173,
        0.0786, 0.0539])

max probability is torch.return_types.max(
values=tensor(0.2147),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8017e-03, 5.4940e-05, 5.4940e-05, 5.4925e-05]),
indices=tensor([  0, 914, 771, 466]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 422 :   Prediction is Correct?
```

```
Yes
first 20 classes probability: tensor([1.2207e-02, 1.2759e-03, 6.9051e-01,
3.8787e-02, 8.1587e-02, 4.5578e-02,
        5.9436e-02, 1.6058e-03, 5.4780e-03, 9.9939e-03, 2.6325e-03, 1.0017e-04,
        8.9347e-03, 1.9928e-02, 2.1704e-04, 1.4734e-03, 3.8190e-03, 2.4384e-03,
        4.1329e-03, 7.9628e-03])

max probability is torch.return_types.max(
values=tensor(0.6905),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0830e-04, 1.8865e-06, 1.8865e-06, 1.8862e-06]),
indices=tensor([  0, 910, 914, 726]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 423 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4637e-03, 1.4101e-02, 2.0156e-04,
1.8616e-05, 1.9101e-04, 4.5349e-05,
        1.8880e-04, 6.3181e-01, 3.3122e-01, 7.3497e-05, 1.4846e-04, 6.3145e-05,
        2.0182e-02, 5.4863e-05, 2.2288e-05, 5.3400e-05, 1.2642e-04, 3.1945e-06,
        3.0011e-05, 8.3949e-06])

max probability is torch.return_types.max(
values=tensor(0.6318),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1990e-07, 2.5538e-09, 2.5535e-09, 2.5534e-09]),
indices=tensor([  0, 319, 316, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 424 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.1803e-08, 7.7477e-10, 1.9567e-10,
5.0923e-09, 1.2261e-08, 2.2416e-08,
        1.7451e-08, 2.1405e-08, 1.4515e-09, 5.4245e-08, 1.7510e-09, 6.4901e-06,
        2.6721e-07, 3.0793e-08, 9.9991e-01, 4.4788e-10, 5.0516e-09, 7.3837e-05,
        5.9617e-06, 9.0044e-09])

max probability is torch.return_types.max(
values=tensor(0.9999),
indices=tensor(14))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.1323e-13, 8.1954e-15, 8.1950e-15, 8.1937e-15]),
indices=tensor([  0, 337, 748, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 425 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0272, 0.0092, 0.0072, 0.0046, 0.0141,
0.0059, 0.0048, 0.0246, 0.0299,
        0.0161, 0.1232, 0.0175, 0.1079, 0.0767, 0.0314, 0.1038, 0.0357, 0.0847,
        0.1159, 0.0512])

max probability is torch.return_types.max(
values=tensor(0.1232),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([0.0015, 0.0001, 0.0001, 0.0001]),
indices=tensor([  0, 771, 319, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 426 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.7711e-04, 1.1837e-03, 3.5423e-03,
1.2955e-03, 7.6020e-03, 6.4904e-04,
        8.5358e-04, 2.4271e-03, 2.2612e-03, 7.8183e-04, 2.0034e-03, 1.3831e-05,
        6.3824e-04, 1.3374e-02, 1.2909e-06, 4.9256e-04, 3.9648e-02, 2.8922e-04,
        1.5130e-02, 8.9512e-01])

max probability is torch.return_types.max(
values=tensor(0.8951),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2182e-02, 1.3876e-07, 1.3864e-07, 1.3859e-07]),
indices=tensor([  0, 655, 337, 691]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 427 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.6387e-04, 7.6623e-03, 4.0152e-03,
```

```
1.2239e-03, 3.5310e-03, 1.5799e-02,
        2.1477e-02, 1.8642e-03, 6.5781e-03, 1.7457e-03, 7.6039e-01, 2.2901e-02,
        8.8890e-02, 2.4433e-02, 5.0318e-04, 2.2739e-02, 6.6203e-03, 2.0780e-03,
        2.8911e-03, 2.9242e-03])

max probability is torch.return_types.max(
values=tensor(0.7604),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7136e-05, 7.9222e-07, 7.9203e-07, 7.9201e-07]),
indices=tensor([  0, 771,  44, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 428 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.4640e-05, 1.0908e-03, 9.4321e-06,
1.2939e-06, 2.0628e-05, 1.5100e-05,
        1.0485e-05, 7.4070e-01, 2.5359e-01, 9.4210e-05, 1.8265e-04, 9.0301e-06,
        3.3356e-03, 5.9789e-05, 6.2176e-07, 3.7516e-04, 2.6958e-04, 7.7382e-07,
        1.1603e-04, 2.8817e-05])

max probability is torch.return_types.max(
values=tensor(0.7407),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5683e-07, 6.4414e-10, 6.4388e-10, 6.4377e-10]),
indices=tensor([  0, 316, 836, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 429 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.1301e-06, 4.0241e-07, 3.1337e-05,
1.6657e-05, 4.2036e-05, 2.3959e-06,
        1.4747e-06, 4.0128e-06, 1.9541e-05, 2.2812e-05, 3.1861e-04, 1.8102e-07,
        4.8980e-05, 5.7089e-01, 2.1836e-08, 5.0571e-05, 7.2665e-02, 5.2462e-05,
        1.8889e-03, 3.5383e-01])

max probability is torch.return_types.max(
values=tensor(0.5709),
indices=tensor(13))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([1.0659e-04, 3.2709e-11, 3.2694e-11, 3.2681e-11]),
indices=tensor([  0, 914, 655, 337]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 430 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.1032e-03, 4.7367e-02, 1.2339e-03,
2.0853e-04, 9.0548e-04, 1.4309e-02,
        8.6252e-03, 9.2957e-02, 6.8713e-01, 1.2265e-03, 4.3789e-03, 7.2434e-05,
        1.3442e-01, 6.7956e-04, 3.5215e-05, 3.9832e-03, 1.6007e-04, 1.4936e-05,
        5.7917e-05, 4.9537e-05])

max probability is torch.return_types.max(
values=tensor(0.6871),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.8644e-07, 8.1544e-08, 8.1514e-08, 8.1474e-08]),
indices=tensor([  0, 319, 836, 794]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 431 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.8144e-04, 2.6819e-03, 2.2714e-04,
5.5060e-05, 3.9019e-04, 4.1131e-04,
        2.9944e-04, 1.3196e-03, 4.2067e-03, 6.3222e-04, 3.6169e-01, 2.5230e-03,
        2.2650e-02, 3.2180e-02, 2.1824e-05, 3.7154e-01, 1.6344e-01, 4.4849e-04,
        3.9274e-03, 3.0515e-02])

max probability is torch.return_types.max(
values=tensor(0.3715),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.8339e-05, 1.7736e-07, 1.7731e-07, 1.7725e-07]),
indices=tensor([  0, 319, 771, 323]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 432 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.6195e-03, 2.1075e-04, 2.2117e-02,
5.3449e-02, 4.3589e-02, 6.9197e-03,
```

```
        1.9107e-02, 1.2997e-02, 1.2243e-02, 3.2106e-01, 6.4721e-03, 3.8865e-05,
        2.3234e-03, 6.7399e-02, 6.2800e-05, 5.4623e-04, 1.6061e-02, 1.8720e-03,
        4.9646e-02, 3.5588e-01])

max probability is torch.return_types.max(
values=tensor(0.3559),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.5824e-03, 1.8913e-06, 1.8913e-06, 1.8908e-06]),
indices=tensor([  0, 914,  91, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 433 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.8705e-03, 2.1129e-04, 1.1588e-02,
5.1361e-02, 6.9726e-02, 2.8919e-02,
        3.3024e-02, 9.6940e-04, 2.8775e-03, 2.4960e-02, 4.3956e-03, 2.1609e-04,
        3.2185e-03, 6.4441e-01, 4.6690e-03, 1.7153e-03, 4.6508e-03, 3.2424e-02,
        3.0453e-02, 4.1395e-02])

max probability is torch.return_types.max(
values=tensor(0.6444),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2644e-04, 2.8382e-06, 2.8378e-06, 2.8377e-06]),
indices=tensor([  0, 771, 914, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 434 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0305e-04, 9.9737e-01, 1.9766e-04,
4.7453e-05, 1.8774e-04, 3.7819e-04,
        3.7783e-04, 3.6514e-04, 2.8350e-04, 4.6813e-06, 3.7612e-05, 3.8810e-06,
        4.6741e-04, 2.0822e-05, 9.7635e-06, 7.9261e-05, 4.6878e-05, 1.5541e-06,
        1.5790e-05, 3.9652e-06])

max probability is torch.return_types.max(
values=tensor(0.9974),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([3.8164e-08, 3.2060e-10, 3.2044e-10, 3.2037e-10]),
indices=tensor([  0, 316, 323, 319]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 435 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.9573e-02, 1.2571e-03, 2.9679e-03,
7.2310e-04, 3.2683e-03, 5.5405e-03,
        2.4009e-03, 2.4019e-03, 2.3474e-02, 8.5010e-03, 9.8319e-02, 2.5637e-03,
        1.1913e-01, 3.6084e-01, 9.5499e-05, 1.8292e-02, 2.9087e-01, 1.4083e-03,
        5.0222e-03, 3.1483e-02])

max probability is torch.return_types.max(
values=tensor(0.3608),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4811e-04, 1.7133e-06, 1.7125e-06, 1.7123e-06]),
indices=tensor([  0, 771, 910, 323]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 436 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.3151e-05, 9.9985e-01, 1.2439e-05,
2.6791e-06, 1.9374e-05, 8.8835e-06,
        1.3805e-05, 2.8931e-05, 8.6706e-06, 1.1329e-07, 8.0274e-06, 5.0384e-07,
        1.2898e-05, 4.6206e-07, 1.1832e-08, 1.9630e-06, 8.0060e-06, 7.6417e-09,
        4.7065e-07, 1.8491e-07])

max probability is torch.return_types.max(
values=tensor(0.9998),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.6069e-10, 8.3889e-13, 8.3882e-13, 8.3868e-13]),
indices=tensor([  0, 316, 319, 910]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 437 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.4818e-04, 4.0866e-04, 5.6912e-04,
4.4761e-04, 5.4972e-04, 7.3354e-04,
        4.6641e-04, 3.4365e-05, 2.1123e-04, 5.7559e-04, 9.7618e-01, 9.1409e-05,
```

```
        8.1779e-03, 6.0508e-03, 1.3671e-06, 4.2146e-03, 4.5868e-04, 2.2540e-05,
        8.1688e-05, 2.7249e-04])

max probability is torch.return_types.max(
values=tensor(0.9762),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.9889e-07, 6.3044e-09, 6.3006e-09, 6.2861e-09]),
indices=tensor([  0, 319, 771, 572]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 438 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.8042e-03, 1.4986e-03, 9.4498e-04,
1.4109e-04, 1.1341e-03, 9.2815e-04,
        7.3099e-04, 3.2446e-03, 2.0788e-02, 7.3931e-04, 1.2298e-01, 3.0057e-03,
        7.4485e-01, 3.4657e-02, 1.6580e-03, 3.9612e-02, 4.5750e-03, 5.4934e-03,
        2.8277e-03, 2.8882e-03])

max probability is torch.return_types.max(
values=tensor(0.7449),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0042e-05, 5.1633e-07, 5.1624e-07, 5.1605e-07]),
indices=tensor([  0, 319, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 439 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.5191e-04, 5.7296e-05, 2.8019e-05,
1.2583e-05, 5.8597e-05, 4.8649e-05,
        2.8623e-05, 2.6971e-04, 2.9332e-04, 3.0606e-04, 1.1577e-02, 2.1493e-01,
        2.3768e-03, 5.0027e-02, 6.2045e-04, 1.1909e-03, 5.5191e-01, 1.8661e-02,
        5.5110e-02, 9.1423e-02])

max probability is torch.return_types.max(
values=tensor(0.5519),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6991e-04, 4.6313e-08, 4.6303e-08, 4.6288e-08]),
```

```
indices=tensor([  0, 691, 230, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 440 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9470e-03, 6.3373e-03, 2.4497e-03,
2.2680e-04, 1.9262e-03, 1.4839e-03,
        9.3602e-04, 6.1987e-03, 2.3749e-02, 2.8246e-03, 9.9200e-02, 2.7234e-03,
        1.1231e-01, 2.9635e-02, 8.3152e-04, 5.6491e-01, 1.0584e-01, 3.8297e-03,
        1.0626e-02, 1.2150e-02])

max probability is torch.return_types.max(
values=tensor(0.5649),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.4865e-05, 1.9043e-06, 1.9039e-06, 1.9038e-06]),
indices=tensor([  0, 319, 910, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 441 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.3509e-08, 1.4528e-09, 7.3160e-09,
2.7906e-08, 8.4396e-08, 2.5110e-08,
        2.2601e-08, 5.3255e-08, 8.6410e-09, 1.4547e-07, 6.6625e-08, 1.4320e-05,
        2.1171e-06, 1.7469e-06, 9.9854e-01, 8.6541e-08, 6.8651e-07, 1.2256e-03,
        2.1056e-04, 7.0020e-07])

max probability is torch.return_types.max(
values=tensor(0.9985),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.5903e-11, 3.6290e-13, 3.6277e-13, 3.6240e-13]),
indices=tensor([  0, 748, 337, 255]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 442 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.1982e-03, 3.0734e-04, 6.4617e-01,
2.1075e-01, 4.6843e-02, 3.5431e-02,
        4.8186e-02, 2.4142e-04, 3.5723e-04, 7.5608e-03, 2.4102e-04, 3.6181e-06,
        6.7462e-04, 1.2466e-03, 4.6366e-05, 1.4105e-05, 7.0801e-05, 1.2259e-04,
```

```
        1.0681e-04, 3.3317e-04])

max probability is torch.return_types.max(
values=tensor(0.6462),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.1981e-06, 1.0318e-07, 1.0315e-07, 1.0314e-07]),
indices=tensor([  0, 771, 914, 572]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 443 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0902, 0.1014, 0.0477, 0.0036, 0.0211,
0.0164, 0.0111, 0.0641, 0.1766,
        0.0078, 0.0410, 0.0013, 0.1888, 0.0708, 0.0004, 0.0695, 0.0514, 0.0020,
        0.0104, 0.0179])

max probability is torch.return_types.max(
values=tensor(0.1888),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8697e-04, 6.6546e-06, 6.6494e-06, 6.6483e-06]),
indices=tensor([  0, 910, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 444 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0155, 0.0018, 0.1678, 0.1932, 0.2034,
0.1660, 0.1175, 0.0024, 0.0056,
        0.0623, 0.0032, 0.0003, 0.0080, 0.0154, 0.0034, 0.0020, 0.0015, 0.0048,
        0.0077, 0.0043])

max probability is torch.return_types.max(
values=tensor(0.2034),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1951e-04, 1.4390e-05, 1.4387e-05, 1.4384e-05]),
indices=tensor([  0, 771, 910, 319]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 445 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.3335e-03, 3.6699e-03, 4.4542e-03,
1.0498e-03, 4.4493e-03, 2.0014e-02,
        9.8225e-03, 5.3693e-03, 4.1722e-02, 1.8096e-03, 1.1322e-01, 5.3108e-03,
        7.0788e-01, 4.4941e-02, 5.8411e-04, 1.5652e-02, 9.9388e-03, 1.0097e-03,
        1.9439e-03, 3.4546e-03])

max probability is torch.return_types.max(
values=tensor(0.7079),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6738e-05, 1.4192e-06, 1.4190e-06, 1.4187e-06]),
indices=tensor([  0,  44, 771, 319]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 446 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.7385e-03, 8.2943e-05, 3.8583e-03,
3.1195e-03, 1.3103e-02, 8.9030e-04,
        6.5084e-04, 1.9040e-03, 2.4655e-03, 5.0436e-03, 3.5037e-03, 1.3688e-04,
        1.3494e-03, 1.0021e-01, 2.3782e-04, 1.1626e-03, 1.6944e-02, 1.4144e-02,
        4.3963e-01, 3.8622e-01])

max probability is torch.return_types.max(
values=tensor(0.4396),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8532e-03, 7.8752e-07, 7.8745e-07, 7.8716e-07]),
indices=tensor([  0, 337, 914, 158]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 447 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.3660e-03, 6.7911e-04, 6.8692e-03,
1.1045e-02, 3.2275e-02, 6.3126e-02,
        2.4764e-02, 6.0368e-04, 4.9188e-03, 9.0809e-03, 8.6021e-02, 5.0948e-04,
        1.8675e-02, 6.3200e-01, 4.1342e-04, 1.7422e-02, 2.0938e-02, 3.0273e-03,
        1.9860e-02, 4.4373e-02])

max probability is torch.return_types.max(
```

```
values=tensor(0.6320),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7291e-04, 9.1338e-07, 9.1269e-07, 9.1249e-07]),
indices=tensor([  0, 771, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 448 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.1817e-02, 1.3661e-03, 1.2130e-01,
7.3993e-02, 5.5675e-02, 1.1958e-01,
        5.9313e-02, 1.5494e-02, 5.4539e-02, 4.0285e-01, 3.7051e-03, 5.1662e-05,
        6.9997e-03, 1.2677e-02, 1.8203e-04, 1.2126e-03, 8.3129e-04, 9.4735e-04,
        2.8021e-03, 3.5730e-03])

max probability is torch.return_types.max(
values=tensor(0.4029),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6341e-05, 1.1395e-06, 1.1391e-06, 1.1388e-06]),
indices=tensor([  0, 771, 319, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 449 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.7608e-04, 1.3657e-04, 2.0527e-01,
3.6724e-01, 3.2159e-01, 5.3192e-02,
        5.1359e-02, 1.0415e-05, 1.0688e-05, 1.7107e-04, 1.0537e-05, 1.4445e-07,
        2.2101e-05, 3.3534e-04, 6.4657e-05, 6.6823e-06, 3.1832e-06, 7.1438e-05,
        1.2277e-04, 7.3577e-05])

max probability is torch.return_types.max(
values=tensor(0.3672),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1497e-06, 3.3320e-08, 3.3303e-08, 3.3290e-08]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 450 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.7519e-07, 9.9994e-01, 1.1517e-07,
1.4216e-08, 4.8020e-07, 8.1537e-08,
        3.3505e-07, 5.0830e-05, 5.8526e-06, 3.1162e-10, 3.4723e-08, 3.8878e-09,
        3.7608e-06, 9.0456e-09, 1.3201e-09, 8.9287e-08, 1.6500e-07, 5.5414e-11,
        4.6770e-09, 1.1385e-09])

max probability is torch.return_types.max(
values=tensor(0.9999),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.1783e-13, 1.9989e-16, 1.9956e-16, 1.9920e-16]),
indices=tensor([  0, 316, 319, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 451 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0074, 0.0004, 0.0006, 0.0004, 0.0021,
0.0009, 0.0005, 0.0042, 0.0144,
        0.0078, 0.0493, 0.0019, 0.0089, 0.1781, 0.0126, 0.0481, 0.0307, 0.3048,
        0.2698, 0.0546])

max probability is torch.return_types.max(
values=tensor(0.3048),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5583e-04, 2.2671e-06, 2.2671e-06, 2.2660e-06]),
indices=tensor([  0, 323, 771, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 452 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.3554e-03, 3.1952e-03, 5.2402e-04,
9.5026e-05, 4.2924e-04, 1.3552e-04,
        3.6095e-04, 6.5784e-01, 3.2424e-01, 1.4965e-03, 2.3314e-04, 2.4434e-06,
        7.6428e-03, 8.7809e-05, 1.6443e-05, 2.9530e-04, 7.9437e-06, 5.6401e-06,
        3.0610e-05, 6.8909e-06])

max probability is torch.return_types.max(
values=tensor(0.6578),
indices=tensor(7))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.9064e-08, 3.7529e-09, 3.7522e-09, 3.7495e-09]),
indices=tensor([  0, 836, 319, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 453 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.8962e-05, 1.1683e-06, 2.4899e-06,
1.8895e-05, 2.7078e-05, 2.7281e-05,
        2.8095e-05, 5.8348e-06, 1.8173e-06, 3.5740e-05, 3.3749e-06, 2.9297e-04,
        1.6045e-04, 5.0371e-05, 9.9509e-01, 4.1502e-06, 6.9109e-06, 3.8575e-03,
        3.2716e-04, 7.3167e-06])

max probability is torch.return_types.max(
values=tensor(0.9951),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.3345e-09, 7.2314e-10, 7.2303e-10, 7.2302e-10]),
indices=tensor([  0, 748, 323, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 454 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.2246e-04, 7.2415e-05, 1.5111e-03,
9.3560e-04, 2.9235e-03, 2.8278e-04,
        3.2884e-04, 3.0991e-02, 1.8889e-02, 2.8040e-02, 5.9976e-03, 1.5309e-03,
        3.3478e-03, 2.9989e-02, 3.7776e-04, 3.2555e-03, 6.6148e-02, 4.4556e-03,
        7.5258e-01, 4.7360e-02])

max probability is torch.return_types.max(
values=tensor(0.7526),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.2435e-04, 3.5146e-07, 3.5145e-07, 3.5144e-07]),
indices=tensor([  0, 914, 337, 197]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 455 :   Prediction is Correct?
Yes
```

```
first 20 classes probability: tensor([1.4453e-03, 9.4026e-01, 1.4918e-03,
6.4340e-04, 3.1313e-03, 4.3605e-03,
        3.3610e-03, 1.3885e-02, 1.1296e-02, 2.2166e-04, 1.5785e-03, 1.1522e-04,
        7.6897e-03, 1.0181e-03, 4.7540e-04, 5.6421e-03, 1.1109e-03, 2.0955e-04,
        1.1847e-03, 5.2047e-04])

max probability is torch.return_types.max(
values=tensor(0.9403),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4025e-05, 3.7529e-07, 3.7513e-07, 3.7479e-07]),
indices=tensor([  0, 316, 319, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 456 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.7124e-03, 7.2451e-04, 1.4149e-02,
8.6960e-03, 4.3908e-02, 4.3291e-03,
        2.7110e-03, 1.2718e-02, 9.7849e-03, 9.3273e-03, 4.3314e-03, 2.7506e-04,
        1.9353e-03, 2.0941e-02, 7.5783e-04, 2.2546e-03, 1.3855e-02, 1.4346e-02,
        6.2314e-01, 2.0209e-01])

max probability is torch.return_types.max(
values=tensor(0.6231),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.5345e-03, 3.6392e-06, 3.6391e-06, 3.6384e-06]),
indices=tensor([  0, 337, 197,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 457 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.4875e-03, 1.0541e-03, 9.5181e-05,
1.3529e-05, 1.0829e-04, 3.2531e-05,
        8.0108e-05, 8.0684e-01, 1.8582e-01, 2.4044e-04, 4.5352e-05, 2.4421e-05,
        1.0064e-03, 2.0322e-05, 7.7393e-06, 2.7225e-05, 2.4250e-05, 6.2669e-06,
        5.3114e-05, 6.7839e-06])

max probability is torch.return_types.max(
values=tensor(0.8068),
indices=tensor(7))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5535e-08, 6.5139e-10, 6.5101e-10, 6.5091e-10]),
indices=tensor([  0, 316, 836, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 458 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.0012e-03, 1.4143e-03, 3.0516e-02,
1.3801e-01, 2.6449e-01, 4.1597e-01,
        1.4306e-01, 7.5766e-05, 1.8478e-04, 4.9514e-04, 1.8390e-04, 1.5234e-05,
        2.0684e-04, 7.1800e-04, 3.6195e-04, 1.0854e-04, 2.3599e-05, 1.8031e-04,
        1.0317e-03, 2.3312e-04])

max probability is torch.return_types.max(
values=tensor(0.4160),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.0396e-06, 7.4785e-07, 7.4741e-07, 7.4701e-07]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 459 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.7701e-02, 7.0883e-01, 1.9355e-02,
3.0037e-03, 1.0355e-02, 1.1809e-02,
        1.1421e-02, 2.5190e-02, 3.6082e-02, 2.0579e-03, 1.0485e-02, 7.3005e-04,
        5.3239e-02, 7.3047e-03, 5.2280e-04, 5.5297e-02, 7.5357e-03, 4.9263e-04,
        2.1150e-03, 2.1903e-03])

max probability is torch.return_types.max(
values=tensor(0.7088),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.5730e-05, 4.4919e-06, 4.4905e-06, 4.4892e-06]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 460 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.1629, 0.0021, 0.2600, 0.0566, 0.1609,
```

```
0.0371, 0.0442, 0.0081, 0.0160,
        0.0308, 0.0027, 0.0004, 0.0108, 0.0466, 0.0072, 0.0021, 0.0027, 0.0711,
        0.0387, 0.0281])

max probability is torch.return_types.max(
values=tensor(0.2600),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.4841e-04, 1.0916e-05, 1.0915e-05, 1.0914e-05]),
indices=tensor([  0, 748, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 461 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.8801e-04, 5.5452e-04, 3.0864e-01,
2.3429e-01, 4.0785e-01, 3.1111e-02,
        1.5596e-02, 4.2652e-05, 3.0171e-05, 1.2299e-04, 6.4177e-05, 2.1022e-06,
        1.3296e-04, 1.8096e-04, 1.1423e-04, 1.2075e-05, 1.6873e-05, 1.3581e-04,
        4.2862e-04, 9.9427e-05])

max probability is torch.return_types.max(
values=tensor(0.4078),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.6283e-06, 2.9436e-07, 2.9433e-07, 2.9424e-07]),
indices=tensor([  0, 316, 319, 466]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 462 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.1941e-03, 2.8329e-04, 1.1426e-04,
6.9196e-05, 3.0189e-04, 4.0821e-04,
        1.0161e-04, 3.6911e-03, 7.9106e-03, 4.1233e-03, 4.0600e-02, 3.6460e-02,
        4.9033e-03, 5.3642e-02, 2.5284e-04, 2.1042e-02, 2.7069e-01, 9.9349e-03,
        3.5422e-01, 1.8469e-01])

max probability is torch.return_types.max(
values=tensor(0.3542),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([8.7615e-04, 5.1579e-07, 5.1529e-07, 5.1519e-07]),
indices=tensor([  0, 914, 323, 771]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 463 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.0572e-03, 4.6031e-04, 2.2681e-04,
3.0970e-05, 1.7490e-04, 4.5069e-04,
        6.2247e-04, 1.6805e-01, 7.8622e-01, 6.0665e-03, 7.9197e-04, 5.2516e-06,
        3.2944e-02, 2.2230e-04, 5.3207e-06, 5.7201e-04, 4.2242e-05, 4.1353e-06,
        3.5516e-05, 1.3560e-05])

max probability is torch.return_types.max(
values=tensor(0.7862),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.6804e-08, 3.3016e-09, 3.3006e-09, 3.2972e-09]),
indices=tensor([  0, 319, 836, 794]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 464 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0047, 0.0035, 0.0042, 0.0025, 0.0129,
0.0933, 0.0170, 0.0038, 0.0621,
        0.0071, 0.1207, 0.0016, 0.1794, 0.3628, 0.0009, 0.0568, 0.0249, 0.0035,
        0.0166, 0.0200])

max probability is torch.return_types.max(
values=tensor(0.3628),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0396e-04, 1.7913e-06, 1.7909e-06, 1.7897e-06]),
indices=tensor([  0, 771, 319, 910]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 465 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4039e-03, 3.7438e-03, 4.7993e-04,
5.8313e-05, 3.5933e-04, 2.3947e-04,
        3.4946e-04, 1.1716e-02, 2.0528e-02, 1.4019e-04, 2.2840e-02, 4.1303e-03,
        9.1569e-01, 1.9848e-03, 4.5744e-04, 1.2418e-02, 2.2643e-03, 2.0785e-04,
```

```
        3.7205e-04, 3.1735e-04])

max probability is torch.return_types.max(
values=tensor(0.9157),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5236e-06, 3.1594e-07, 3.1594e-07, 3.1583e-07]),
indices=tensor([  0, 836, 319, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 466 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.7433e-04, 9.3725e-05, 2.2621e-03,
2.8070e-03, 1.2243e-02, 2.0893e-03,
        9.4899e-04, 6.0372e-03, 7.6412e-03, 1.1109e-02, 5.4009e-03, 1.7136e-04,
        1.4160e-03, 7.2182e-02, 1.5188e-04, 1.4712e-03, 2.0507e-02, 5.1053e-03,
        6.1895e-01, 2.2634e-01])

max probability is torch.return_types.max(
values=tensor(0.6190),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7632e-03, 5.6103e-07, 5.6067e-07, 5.6058e-07]),
indices=tensor([  0, 914, 337, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 467 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.5906e-04, 1.8721e-04, 2.7020e-02,
3.9740e-01, 3.1477e-01, 1.9086e-01,
        6.7981e-02, 8.6013e-06, 9.2752e-06, 1.0515e-04, 3.3231e-05, 7.5910e-07,
        1.5122e-05, 5.8001e-04, 7.7687e-05, 9.0275e-06, 2.7496e-06, 4.7381e-05,
        4.2153e-04, 1.6456e-04])

max probability is torch.return_types.max(
values=tensor(0.3974),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3832e-06, 4.9544e-08, 4.9502e-08, 4.9453e-08]),
indices=tensor([  0, 319, 316,  85]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 468 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0173, 0.0033, 0.1156, 0.0803, 0.1347,
0.0866, 0.1294, 0.0141, 0.0423,
        0.1192, 0.0241, 0.0006, 0.0245, 0.0806, 0.0010, 0.0125, 0.0210, 0.0070,
        0.0160, 0.0420])

max probability is torch.return_types.max(
values=tensor(0.1347),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0638e-03, 2.8148e-05, 2.8138e-05, 2.8135e-05]),
indices=tensor([  0, 771, 910, 323]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 469 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.0178e-03, 1.9886e-03, 2.3625e-03,
2.6403e-04, 1.5941e-03, 4.7978e-04,
        2.8527e-04, 1.0154e-02, 1.7488e-02, 5.6235e-04, 7.3207e-02, 9.3003e-03,
        6.5895e-01, 3.6615e-02, 9.7468e-04, 1.8963e-02, 9.4929e-02, 5.8367e-03,
        2.8168e-02, 3.4979e-02])

max probability is torch.return_types.max(
values=tensor(0.6590),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1422e-04, 7.0360e-07, 7.0359e-07, 7.0345e-07]),
indices=tensor([  0, 910, 771,  44]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 470 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2307e-05, 3.9518e-05, 3.4052e-01,
3.3375e-01, 3.1637e-01, 7.2436e-03,
        2.0520e-03, 7.0145e-07, 2.4449e-07, 2.4503e-06, 4.5973e-07, 1.7461e-09,
        1.3346e-06, 4.0585e-06, 8.6383e-07, 1.4304e-08, 8.0138e-08, 1.2080e-06,
        5.1028e-06, 2.0079e-06])
```

```
max probability is torch.return_types.max(
values=tensor(0.3405),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.3732e-08, 1.8165e-10, 1.8158e-10, 1.8155e-10]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 471 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0068, 0.0017, 0.0530, 0.1054, 0.1243,
0.0897, 0.0775, 0.0026, 0.0029,
        0.0194, 0.0053, 0.0026, 0.0101, 0.0515, 0.2063, 0.0039, 0.0039, 0.1071,
        0.0601, 0.0163])

max probability is torch.return_types.max(
values=tensor(0.2063),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.1691e-04, 5.1319e-05, 5.1317e-05, 5.1309e-05]),
indices=tensor([  0, 337, 771, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 472 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0043, 0.0006, 0.0528, 0.0110, 0.0217,
0.0032, 0.0109, 0.1604, 0.1765,
        0.3287, 0.0067, 0.0015, 0.0804, 0.0301, 0.0008, 0.0027, 0.0720, 0.0024,
        0.0253, 0.0050])

max probability is torch.return_types.max(
values=tensor(0.3287),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0492e-04, 3.0241e-06, 3.0237e-06, 3.0237e-06]),
indices=tensor([  0, 323, 910, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 473 :   Prediction is Correct?
```

```
No
first 20 classes probability: tensor([3.8511e-03, 7.6913e-04, 6.1487e-01,
3.4383e-02, 1.0042e-01, 4.5200e-02,
        1.3938e-01, 2.6102e-03, 1.1087e-02, 1.6481e-02, 7.8070e-04, 3.2556e-05,
        4.9703e-03, 1.2495e-02, 5.6192e-04, 8.2424e-04, 1.8312e-03, 3.8849e-03,
        3.3822e-03, 1.5420e-03])

max probability is torch.return_types.max(
values=tensor(0.6149),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0028e-05, 6.5650e-07, 6.5647e-07, 6.5641e-07]),
indices=tensor([  0, 910, 726, 748]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 474 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.7088e-04, 4.0744e-06, 4.7617e-04,
4.9165e-04, 1.8649e-03, 2.0865e-04,
        1.3593e-04, 2.2847e-03, 3.5945e-03, 4.5038e-02, 2.9027e-04, 1.2516e-04,
        1.0165e-03, 2.4422e-02, 8.1029e-03, 7.2820e-04, 3.5471e-03, 1.8104e-02,
        8.7833e-01, 1.0158e-02])

max probability is torch.return_types.max(
values=tensor(0.8783),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1296e-05, 8.7735e-08, 8.7729e-08, 8.7726e-08]),
indices=tensor([  0, 914, 337, 197]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 475 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.8222e-05, 2.0175e-05, 2.8752e-06,
4.6419e-06, 1.8982e-05, 3.5314e-05,
        8.7249e-06, 1.0267e-05, 5.1915e-05, 1.4983e-04, 9.8421e-01, 5.6819e-05,
        4.1794e-04, 8.8189e-04, 6.1647e-07, 1.2120e-02, 1.8290e-04, 3.7886e-05,
        9.1759e-04, 7.8029e-04])

max probability is torch.return_types.max(
values=tensor(0.9842),
indices=tensor(10))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.8924e-07, 8.2038e-10, 8.1964e-10, 8.1783e-10]),
indices=tensor([  0, 319, 771, 144]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 476 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.9949e-04, 4.8378e-04, 1.6185e-01,
2.0090e-01, 5.5059e-01, 6.9366e-02,
        1.5978e-02, 1.2252e-05, 9.0483e-06, 2.1143e-05, 2.0694e-05, 8.5604e-07,
        2.6222e-05, 4.1883e-05, 4.2647e-05, 2.5307e-06, 2.6486e-06, 4.0001e-05,
        3.0842e-04, 3.4900e-05])

max probability is torch.return_types.max(
values=tensor(0.5506),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2280e-06, 7.6439e-08, 7.6425e-08, 7.6373e-08]),
indices=tensor([  0, 319, 316, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 477 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.9306e-06, 3.9415e-05, 5.2772e-08,
1.8449e-09, 9.2163e-08, 9.3796e-09,
        1.5049e-08, 9.4999e-01, 4.9916e-02, 5.5027e-07, 4.5067e-07, 2.2033e-09,
        3.6823e-05, 3.3458e-08, 2.0336e-09, 9.5596e-06, 1.6063e-07, 5.4333e-10,
        3.4723e-07, 2.5851e-08])

max probability is torch.return_types.max(
values=tensor(0.9500),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.8276e-12, 1.0266e-14, 1.0265e-14, 1.0255e-14]),
indices=tensor([  0, 836, 316, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 478 :   Prediction is Correct?
Yes
```

```
first 20 classes probability: tensor([9.9458e-01, 1.6533e-04, 3.6822e-04,
2.2437e-04, 1.7212e-04, 3.6375e-03,
        4.7392e-04, 8.9813e-06, 3.7122e-05, 2.4397e-04, 1.1274e-05, 7.9768e-06,
        1.9822e-05, 1.0837e-05, 2.1701e-06, 1.1252e-06, 2.3054e-06, 9.9839e-06,
        7.4401e-06, 1.4850e-05])

max probability is torch.return_types.max(
values=tensor(0.9946),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5477e-08, 2.5908e-10, 2.5906e-10, 2.5904e-10]),
indices=tensor([  0, 319, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 479 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.3271, 0.0044, 0.0088, 0.0032, 0.0157,
0.0437, 0.0346, 0.0301, 0.0966,
        0.1107, 0.0156, 0.0378, 0.0321, 0.0107, 0.0122, 0.0079, 0.0304, 0.0754,
        0.0743, 0.0151])

max probability is torch.return_types.max(
values=tensor(0.3271),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6030e-04, 1.4212e-05, 1.4209e-05, 1.4209e-05]),
indices=tensor([  0, 914, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 480 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0071, 0.0009, 0.0173, 0.0476, 0.0690,
0.0899, 0.0861, 0.0066, 0.0195,
        0.3899, 0.0142, 0.0009, 0.0068, 0.0798, 0.0024, 0.0083, 0.0295, 0.0135,
        0.0636, 0.0384])

max probability is torch.return_types.max(
values=tensor(0.3899),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([4.9008e-04, 8.6502e-06, 8.6489e-06, 8.6450e-06]),
indices=tensor([  0, 771, 914, 323]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 481 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0090, 0.0059, 0.0127, 0.0094, 0.0246,
0.0506, 0.0259, 0.0186, 0.0701,
        0.0465, 0.0836, 0.0023, 0.0554, 0.1393, 0.0013, 0.0392, 0.0883, 0.0097,
        0.0521, 0.1977])

max probability is torch.return_types.max(
values=tensor(0.1977),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.2129e-03, 5.5220e-05, 5.5191e-05, 5.5167e-05]),
indices=tensor([  0, 771, 914,  44]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 482 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0330, 0.0020, 0.0018, 0.0014, 0.0049,
0.0071, 0.0049, 0.0027, 0.0075,
        0.0098, 0.3456, 0.0708, 0.0240, 0.1259, 0.0020, 0.0154, 0.0916, 0.0595,
        0.0583, 0.1216])

max probability is torch.return_types.max(
values=tensor(0.3456),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2931e-03, 9.2715e-06, 9.2686e-06, 9.2677e-06]),
indices=tensor([  0, 914, 771, 323]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 483 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0164, 0.0004, 0.1171, 0.0976, 0.1291,
0.0738, 0.0627, 0.0021, 0.0055,
        0.1594, 0.0048, 0.0003, 0.0092, 0.2368, 0.0039, 0.0033, 0.0138, 0.0090,
        0.0284, 0.0217])
```

```
max probability is torch.return_types.max(
values=tensor(0.2368),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7088e-04, 4.9121e-06, 4.9111e-06, 4.9110e-06]),
indices=tensor([  0, 771, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 484 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0147, 0.0159, 0.0076, 0.0029, 0.0163,
0.0159, 0.0162, 0.1542, 0.3055,
        0.0189, 0.0403, 0.0020, 0.0429, 0.0510, 0.0120, 0.0367, 0.0101, 0.0672,
        0.1072, 0.0281])

max probability is torch.return_types.max(
values=tensor(0.3055),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.0084e-04, 3.5306e-05, 3.5304e-05, 3.5302e-05]),
indices=tensor([  0, 323, 910, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 485 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.4895e-05, 9.9723e-01, 3.2309e-05,
1.0956e-05, 3.0373e-05, 2.7966e-04,
        1.2799e-04, 2.2499e-06, 3.9063e-06, 4.1667e-07, 1.9814e-03, 7.9893e-06,
        1.6378e-04, 6.9911e-06, 3.2565e-08, 6.5695e-05, 9.1547e-06, 3.0497e-08,
        6.6768e-07, 5.9449e-07])

max probability is torch.return_types.max(
values=tensor(0.9972),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8616e-09, 6.8879e-12, 6.8730e-12, 6.8689e-12]),
indices=tensor([  0, 319, 910, 323]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 486 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0842, 0.0037, 0.0025, 0.0016, 0.0074,
0.0134, 0.0056, 0.0033, 0.0286,
        0.0179, 0.1353, 0.0033, 0.0873, 0.3820, 0.0022, 0.1205, 0.0445, 0.0109,
        0.0105, 0.0249])

max probability is torch.return_types.max(
values=tensor(0.3820),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8630e-04, 1.0828e-05, 1.0825e-05, 1.0819e-05]),
indices=tensor([  0, 319, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 487 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.6198e-09, 5.5644e-09, 3.0518e-08,
1.9035e-08, 1.1528e-07, 1.1593e-09,
        1.0777e-09, 1.6854e-07, 1.1230e-07, 5.0251e-08, 2.3516e-06, 1.7321e-10,
        2.5547e-08, 4.5418e-05, 1.2484e-12, 3.4757e-08, 2.6568e-03, 4.2927e-08,
        4.7618e-04, 9.9632e-01])

max probability is torch.return_types.max(
values=tensor(0.9963),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.0079e-04, 5.2652e-15, 5.2616e-15, 5.2573e-15]),
indices=tensor([  0, 655, 337, 691]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 488 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0495, 0.0078, 0.0215, 0.0542, 0.0967,
0.4848, 0.1586, 0.0006, 0.0028,
        0.0056, 0.0126, 0.0013, 0.0115, 0.0661, 0.0015, 0.0034, 0.0024, 0.0012,
        0.0058, 0.0049])

max probability is torch.return_types.max(
values=tensor(0.4848),
indices=tensor(5))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0326e-04, 7.4312e-06, 7.4308e-06, 7.4303e-06]),
indices=tensor([  0, 319, 910, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 489 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.7130e-05, 1.2898e-05, 5.7760e-01,
3.6903e-01, 3.2568e-02, 1.5085e-02,
        5.5865e-03, 4.2232e-07, 3.0740e-07, 2.3695e-05, 1.0807e-06, 2.2291e-08,
        3.7834e-06, 2.4824e-05, 5.2367e-06, 3.9350e-08, 1.6418e-07, 7.0971e-06,
        2.6847e-06, 5.5856e-06])

max probability is torch.return_types.max(
values=tensor(0.5776),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4767e-08, 1.3339e-10, 1.3337e-10, 1.3332e-10]),
indices=tensor([  0, 771, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 490 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0128, 0.0313, 0.0055, 0.0014, 0.0127,
0.0320, 0.0245, 0.0475, 0.2734,
        0.0187, 0.0417, 0.0019, 0.0663, 0.0702, 0.0004, 0.1951, 0.1214, 0.0027,
        0.0146, 0.0209])

max probability is torch.return_types.max(
values=tensor(0.2734),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9093e-04, 4.9154e-06, 4.9142e-06, 4.9137e-06]),
indices=tensor([  0, 910, 896, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 491 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.4796e-07, 2.8283e-11, 8.8974e-10,
4.1446e-10, 2.4164e-08, 2.7368e-10,
```

```
        1.5910e-10, 8.7428e-09, 9.9255e-09, 2.0846e-08, 2.1293e-07, 3.7966e-07,
        2.3002e-08, 2.2344e-06, 6.9884e-03, 3.5866e-08, 3.4672e-07, 9.4138e-01,
        5.1590e-02, 3.5347e-05])

max probability is torch.return_types.max(
values=tensor(0.9414),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.2593e-11, 6.9870e-17, 6.9833e-17, 6.9736e-17]),
indices=tensor([  0, 748, 337, 230]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 492 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.8431e-05, 2.6043e-04, 1.9753e-01,
1.9841e-01, 5.6730e-01, 2.9183e-02,
        7.1138e-03, 3.5391e-06, 2.2232e-06, 6.0539e-06, 5.3646e-06, 6.8411e-08,
        7.2493e-06, 1.2127e-05, 7.1018e-06, 5.7005e-07, 6.5349e-07, 9.2334e-06,
        8.6599e-05, 9.9198e-06])

max probability is torch.return_types.max(
values=tensor(0.5673),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.3432e-07, 6.8795e-09, 6.8771e-09, 6.8729e-09]),
indices=tensor([  0, 319, 316, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 493 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0741e-08, 2.2675e-09, 4.4866e-07,
5.5005e-07, 1.8726e-06, 1.0058e-08,
        1.1179e-08, 4.6447e-08, 6.3690e-08, 1.7857e-07, 5.4970e-06, 1.2887e-10,
        9.2979e-08, 5.0187e-03, 4.1762e-12, 6.0286e-08, 1.1871e-03, 4.2558e-07,
        1.4233e-04, 9.9357e-01])

max probability is torch.return_types.max(
values=tensor(0.9936),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([7.0098e-05, 8.3122e-15, 8.3025e-15, 8.2991e-15]),
indices=tensor([  0, 655, 337, 691]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 494 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.4729e-04, 5.2333e-04, 3.2788e-01,
2.4706e-01, 3.8438e-01, 2.5888e-02,
        1.2884e-02, 3.3722e-05, 2.2538e-05, 9.0196e-05, 4.7277e-05, 1.2643e-06,
        1.1475e-04, 1.2642e-04, 8.0173e-05, 8.2956e-06, 1.1189e-05, 9.7692e-05,
        2.7471e-04, 7.2092e-05])

max probability is torch.return_types.max(
values=tensor(0.3844),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.2576e-06, 1.6706e-07, 1.6706e-07, 1.6701e-07]),
indices=tensor([  0, 319, 316, 466]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 495 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.6255e-03, 1.3984e-05, 3.7203e-04,
1.8114e-04, 8.4624e-04, 2.5288e-04,
        1.9691e-04, 8.3323e-04, 8.7517e-04, 2.5414e-03, 5.3277e-04, 3.3677e-03,
        3.0101e-03, 8.7557e-03, 3.8784e-01, 2.4441e-04, 3.1872e-03, 4.7876e-01,
        1.0103e-01, 5.3727e-03])

max probability is torch.return_types.max(
values=tensor(0.4788),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3042e-05, 1.6641e-07, 1.6636e-07, 1.6632e-07]),
indices=tensor([  0, 748, 428, 230]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 496 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.6649e-07, 2.5639e-08, 3.3636e-06,
3.9875e-06, 4.2499e-05, 3.0934e-07,
        1.7465e-07, 1.3335e-05, 1.1133e-05, 1.4474e-05, 1.3523e-05, 1.3124e-07,
```

```
        2.6220e-06, 1.5862e-03, 3.5547e-06, 3.8669e-06, 2.1129e-04, 6.8293e-04,
        9.0820e-01, 8.9167e-02])

max probability is torch.return_types.max(
values=tensor(0.9082),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5957e-05, 1.3210e-11, 1.3197e-11, 1.3195e-11]),
indices=tensor([  0, 337, 914, 197]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 497 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0122, 0.0008, 0.0584, 0.0681, 0.1241,
0.1177, 0.0431, 0.0089, 0.0385,
        0.1605, 0.0096, 0.0002, 0.0112, 0.1481, 0.0010, 0.0081, 0.0095, 0.0085,
        0.0890, 0.0728])

max probability is torch.return_types.max(
values=tensor(0.1605),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.5445e-04, 9.4312e-06, 9.4283e-06, 9.4242e-06]),
indices=tensor([  0, 771, 914, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 498 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.9086e-04, 2.5976e-03, 4.0255e-02,
8.3034e-03, 3.8394e-02, 1.9117e-01,
        7.1026e-01, 1.2323e-04, 8.1836e-04, 8.8289e-04, 9.3860e-04, 6.1296e-05,
        2.6405e-03, 1.2387e-03, 3.5978e-04, 6.9415e-04, 1.3472e-04, 2.0562e-04,
        1.5089e-04, 4.9410e-05])

max probability is torch.return_types.max(
values=tensor(0.7103),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.2463e-07, 3.4429e-08, 3.4421e-08, 3.4413e-08]),
indices=tensor([  0, 910, 323, 896]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 499 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9999e-01, 1.5858e-07, 3.0068e-06,
1.9301e-07, 6.5386e-07, 8.4935e-07,
        7.1299e-07, 5.2796e-08, 5.1571e-07, 1.1651e-06, 2.8679e-09, 4.9936e-09,
        7.2959e-07, 4.9640e-08, 1.8158e-09, 5.6002e-10, 4.2153e-09, 1.5835e-08,
        1.8621e-09, 7.0848e-09])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5831e-13, 1.3158e-15, 1.3157e-15, 1.3137e-15]),
indices=tensor([  0, 910, 323,  85]))
End**************************************************************************

currently at 500 current time is 15.708709001541138
**************************************************************************
Predicting Test Sample 500 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.1138e-03, 8.8918e-04, 1.7208e-04,
1.2349e-04, 2.4792e-04, 1.3813e-03,
        6.2936e-04, 1.9242e-03, 1.2457e-03, 1.5133e-03, 1.6421e-02, 9.0265e-01,
        1.5171e-02, 2.0013e-03, 3.0176e-03, 3.9176e-03, 3.4263e-02, 1.1625e-03,
        4.7674e-03, 1.2859e-03])

max probability is torch.return_types.max(
values=tensor(0.9027),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1708e-05, 1.1370e-06, 1.1366e-06, 1.1365e-06]),
indices=tensor([  0, 323, 914, 910]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 501 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.5051e-05, 4.7803e-05, 1.6870e-03,
1.5755e-03, 1.7813e-02, 4.5759e-04,
        2.6201e-04, 1.1584e-01, 2.8810e-02, 2.3853e-02, 1.1056e-04, 2.0322e-06,
        1.1958e-04, 5.8590e-03, 3.9427e-05, 1.4324e-04, 1.0611e-03, 5.4974e-04,
```

```
        7.7624e-01, 2.5377e-02])

max probability is torch.return_types.max(
values=tensor(0.7762),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.1587e-05, 5.2643e-09, 5.2635e-09, 5.2601e-09]),
indices=tensor([  0,  91, 197, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 502 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0000e+00, 3.4563e-11, 1.7509e-08,
3.7961e-10, 7.9851e-09, 1.3390e-09,
        1.7927e-09, 1.4069e-10, 3.8629e-09, 9.8544e-09, 5.2121e-12, 4.4642e-12,
        2.6518e-09, 3.4187e-10, 5.2232e-11, 8.6255e-13, 2.8766e-12, 3.4548e-09,
        7.6909e-11, 8.1818e-11])

max probability is torch.return_types.max(
values=tensor(1.),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2405e-16, 1.6685e-19, 1.6678e-19, 1.6654e-19]),
indices=tensor([  0, 323, 910,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 503 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.2288e-03, 1.2682e-03, 2.6174e-03,
5.5527e-04, 3.7529e-03, 5.6218e-04,
        4.2526e-04, 2.4763e-03, 3.4943e-03, 5.5490e-04, 2.6094e-02, 1.0434e-03,
        3.1398e-03, 1.9564e-02, 9.0214e-05, 2.1645e-03, 2.7246e-02, 1.9127e-02,
        1.8819e-01, 6.8302e-01])

max probability is torch.return_types.max(
values=tensor(0.6830),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.4701e-03, 9.5170e-07, 9.5167e-07, 9.5163e-07]),
indices=tensor([  0, 337, 914, 158]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 504 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0133, 0.0014, 0.1999, 0.0918, 0.1659,
0.0352, 0.0376, 0.0173, 0.0267,
        0.1415, 0.0052, 0.0004, 0.0089, 0.0600, 0.0015, 0.0044, 0.0317, 0.0135,
        0.0627, 0.0507])

max probability is torch.return_types.max(
values=tensor(0.1999),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8765e-03, 2.9615e-05, 2.9615e-05, 2.9610e-05]),
indices=tensor([  0, 914,  91, 771]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 505 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.3259e-04, 9.2164e-03, 4.6541e-04,
2.1372e-04, 2.5017e-03, 4.4374e-04,
        1.9423e-04, 6.1704e-02, 2.8583e-02, 1.4237e-03, 1.2357e-01, 2.0720e-04,
        8.7992e-03, 3.6942e-03, 4.6646e-05, 1.6872e-01, 1.3081e-02, 6.6795e-04,
        4.1928e-01, 1.5588e-01])

max probability is torch.return_types.max(
values=tensor(0.4193),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.8276e-04, 2.0243e-07, 2.0226e-07, 2.0224e-07]),
indices=tensor([  0, 316, 319, 771]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 506 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.0851e-04, 8.4062e-04, 9.7246e-03,
9.9825e-02, 1.9035e-01, 4.3782e-01,
        2.5862e-01, 3.2411e-05, 1.0504e-04, 4.4887e-04, 7.7490e-05, 3.6948e-06,
        9.3604e-05, 5.8338e-04, 1.3330e-04, 6.3096e-05, 8.0386e-06, 4.7121e-05,
        3.5756e-04, 8.2533e-05])
```

```
max probability is torch.return_types.max(
values=tensor(0.4378),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5258e-06, 8.5461e-08, 8.5393e-08, 8.5322e-08]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 507 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.2276e-03, 1.2588e-03, 2.6509e-03,
4.1671e-03, 1.6544e-02, 7.7317e-02,
        2.4754e-02, 2.6663e-02, 2.6554e-01, 1.4820e-01, 4.8274e-02, 1.6924e-04,
        1.1038e-02, 1.5668e-01, 3.0280e-04, 4.9509e-02, 1.8938e-02, 3.4904e-03,
        6.9285e-02, 6.9903e-02])

max probability is torch.return_types.max(
values=tensor(0.2655),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.4194e-04, 2.8152e-06, 2.8145e-06, 2.8145e-06]),
indices=tensor([  0, 771, 896, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 508 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.1489e-05, 2.1316e-04, 1.9588e-01,
1.9296e-01, 5.7104e-01, 3.2647e-02,
        7.0515e-03, 3.3379e-06, 2.1113e-06, 4.8811e-06, 4.6365e-06, 9.7878e-08,
        7.7109e-06, 9.4016e-06, 8.3384e-06, 3.9240e-07, 5.5495e-07, 9.5523e-06,
        8.6214e-05, 8.6700e-06])

max probability is torch.return_types.max(
values=tensor(0.5710),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.8733e-07, 6.6436e-09, 6.6422e-09, 6.6383e-09]),
indices=tensor([  0, 319, 316, 957]))
End*****************************************************************************
```

```
**************************************************************************
Predicting Test Sample 509 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.7499e-03, 1.2247e-04, 5.1580e-03,
2.4627e-03, 9.9731e-03, 1.9677e-03,
        2.3549e-03, 7.2720e-04, 2.6681e-03, 5.9956e-03, 4.2113e-03, 2.9991e-04,
        3.0369e-03, 4.7709e-01, 8.9364e-03, 3.4009e-03, 2.7410e-02, 2.9684e-01,
        6.5234e-02, 7.7559e-02])

max probability is torch.return_types.max(
values=tensor(0.4771),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6220e-04, 1.5057e-06, 1.5054e-06, 1.5051e-06]),
indices=tensor([  0, 748, 337, 914]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 510 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.2832e-04, 1.3906e-04, 2.1127e-01,
6.4244e-01, 8.6934e-02, 3.9495e-02,
        1.8575e-02, 1.1488e-05, 7.4227e-06, 2.7556e-04, 1.3739e-05, 3.7652e-07,
        3.3567e-05, 2.2822e-04, 2.7168e-05, 8.0139e-07, 2.0330e-06, 2.0817e-05,
        3.0592e-05, 4.9470e-05])

max probability is torch.return_types.max(
values=tensor(0.6424),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.2217e-07, 2.0817e-08, 2.0806e-08, 2.0805e-08]),
indices=tensor([  0, 771, 910, 319]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 511 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.2656e-06, 4.9444e-06, 4.7596e-01,
4.4519e-01, 7.7009e-02, 1.3588e-03,
        4.6799e-04, 4.2122e-08, 1.6044e-08, 8.2956e-07, 2.2778e-08, 7.8707e-12,
        2.1757e-07, 4.1527e-07, 7.2068e-09, 3.9071e-10, 1.8830e-09, 1.2349e-08,
        4.1200e-08, 8.6571e-08])

max probability is torch.return_types.max(
```

```
values=tensor(0.4760),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.2282e-10, 6.7491e-13, 6.7447e-13, 6.7426e-13]),
indices=tensor([  0, 319, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 512 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.1895e-02, 6.0297e-03, 2.5551e-02,
6.1495e-03, 1.7426e-02, 1.1592e-02,
        1.8784e-02, 2.4139e-01, 4.6791e-01, 1.3233e-02, 1.0721e-02, 3.7268e-04,
        1.4867e-01, 9.6314e-03, 2.2182e-04, 2.6566e-03, 1.4772e-03, 5.9623e-04,
        1.6432e-03, 1.3939e-03])

max probability is torch.return_types.max(
values=tensor(0.4679),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5994e-05, 2.7877e-06, 2.7873e-06, 2.7865e-06]),
indices=tensor([  0, 319, 836, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 513 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.5782e-05, 2.9275e-03, 1.1312e-06,
2.7032e-07, 4.7359e-06, 1.6665e-05,
        5.3124e-06, 5.0572e-05, 1.3355e-04, 8.1414e-06, 1.8563e-01, 8.2203e-04,
        3.1142e-03, 4.3231e-04, 4.4509e-06, 7.9834e-01, 7.9638e-03, 1.2684e-05,
        2.7848e-04, 2.2613e-04])

max probability is torch.return_types.max(
values=tensor(0.7983),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5485e-07, 5.9442e-10, 5.9246e-10, 5.9218e-10]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 514 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0180, 0.0040, 0.0029, 0.0024, 0.0040,
0.0131, 0.0068, 0.0064, 0.0042,
        0.0047, 0.0142, 0.7096, 0.0338, 0.0138, 0.0364, 0.0037, 0.0521, 0.0094,
        0.0297, 0.0067])

max probability is torch.return_types.max(
values=tensor(0.7096),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9100e-04, 2.5029e-05, 2.5029e-05, 2.5028e-05]),
indices=tensor([  0, 914, 691, 573]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 515 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.2070e-03, 1.4126e-03, 9.2115e-02,
2.1562e-01, 4.0648e-01, 2.0367e-01,
        7.3666e-02, 8.4138e-05, 1.1080e-04, 3.4879e-04, 1.8750e-04, 1.4005e-05,
        1.7853e-04, 6.0466e-04, 3.9050e-04, 8.5026e-05, 3.3237e-05, 2.6548e-04,
        1.5826e-03, 3.1566e-04])

max probability is torch.return_types.max(
values=tensor(0.4065),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1276e-05, 1.6900e-06, 1.6895e-06, 1.6882e-06]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 516 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.7975e-01, 5.9032e-05, 7.1863e-04,
3.1143e-05, 6.0205e-04, 1.2376e-05,
        2.5377e-05, 2.7525e-03, 5.5564e-03, 3.7453e-04, 1.7401e-04, 8.3719e-06,
        2.1559e-03, 1.5279e-04, 7.7840e-05, 1.4260e-04, 1.9292e-05, 1.2675e-03,
        5.9890e-03, 1.3250e-04])

max probability is torch.return_types.max(
values=tensor(0.9797),
indices=tensor(0))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2643e-07, 1.5720e-09, 1.5710e-09, 1.5708e-09]),
indices=tensor([  0, 910,  85, 859]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 517 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.2544e-03, 2.6743e-03, 1.5611e-01,
2.4535e-01, 4.1266e-01, 1.1820e-01,
        5.1569e-02, 1.9064e-04, 1.9793e-04, 5.6652e-04, 3.9967e-04, 2.1670e-05,
        3.7562e-04, 9.7451e-04, 5.5014e-04, 1.7073e-04, 8.4128e-05, 4.9865e-04,
        2.2915e-03, 5.8544e-04])

max probability is torch.return_types.max(
values=tensor(0.4127),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.0186e-05, 5.4646e-06, 5.4624e-06, 5.4590e-06]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 518 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.2845e-04, 1.3923e-04, 2.9664e-03,
3.1116e-03, 6.5439e-03, 5.6538e-04,
        4.2092e-04, 1.9174e-03, 1.5528e-03, 1.7231e-03, 4.5476e-03, 6.1742e-05,
        2.0944e-03, 1.3908e-01, 2.8157e-05, 1.8518e-03, 6.6622e-02, 1.5526e-03,
        5.3092e-02, 7.0544e-01])

max probability is torch.return_types.max(
values=tensor(0.7054),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.0473e-03, 4.3045e-07, 4.3040e-07, 4.3037e-07]),
indices=tensor([  0, 914, 655, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 519 :   Prediction is Correct?
Yes
```

```
first 20 classes probability: tensor([2.0426e-06, 2.1410e-06, 2.4858e-08,
9.1931e-09, 2.6058e-08, 9.9684e-08,
        7.3367e-08, 1.1447e-05, 5.1252e-07, 1.1337e-07, 2.7603e-05, 9.9919e-01,
        2.3688e-04, 9.3134e-07, 1.6493e-05, 5.4389e-07, 5.0037e-04, 6.6763e-07,
        5.7058e-06, 1.0083e-06])

max probability is torch.return_types.max(
values=tensor(0.9992),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.5434e-10, 1.2785e-12, 1.2780e-12, 1.2778e-12]),
indices=tensor([  0, 691, 382,  82]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 520 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0103, 0.0120, 0.0193, 0.0048, 0.0194,
0.0058, 0.0045, 0.1856, 0.2090,
        0.0195, 0.0419, 0.0015, 0.1867, 0.0299, 0.0016, 0.0869, 0.0262, 0.0047,
        0.0736, 0.0345])

max probability is torch.return_types.max(
values=tensor(0.2090),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.6562e-04, 2.2807e-05, 2.2798e-05, 2.2796e-05]),
indices=tensor([  0, 316, 319,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 521 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2371e-01, 3.0139e-03, 1.3688e-02,
1.3388e-03, 9.1489e-03, 5.5349e-02,
        3.1431e-02, 3.0629e-02, 6.6892e-01, 1.7987e-02, 3.7640e-03, 1.1399e-05,
        3.3622e-02, 2.5919e-03, 3.5101e-05, 3.4414e-03, 1.6310e-04, 2.9162e-04,
        3.4942e-04, 4.7568e-04])

max probability is torch.return_types.max(
values=tensor(0.6689),
indices=tensor(8))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([1.1351e-06, 4.4286e-08, 4.4239e-08, 4.4219e-08]),
indices=tensor([  0, 319, 836, 316]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 522 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.2577e-09, 3.9077e-07, 4.7715e-10,
4.6328e-10, 4.7203e-09, 2.2624e-08,
        2.5780e-09, 7.6098e-09, 9.0190e-08, 2.4415e-08, 9.8082e-01, 4.0241e-07,
        4.5538e-05, 1.9350e-05, 3.6829e-10, 1.9111e-02, 5.4711e-06, 1.5823e-08,
        8.5486e-07, 1.7827e-06])

max probability is torch.return_types.max(
values=tensor(0.9808),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4582e-11, 2.8725e-15, 2.8621e-15, 2.8536e-15]),
indices=tensor([  0, 319, 771, 869]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 523 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.0868e-03, 1.7738e-04, 3.2167e-03,
3.2274e-03, 1.3385e-02, 4.3844e-03,
        2.4586e-03, 4.6376e-03, 1.0696e-02, 2.5685e-02, 3.9721e-03, 9.3857e-04,
        3.0772e-03, 5.2902e-02, 2.2480e-02, 6.7315e-03, 8.5665e-03, 1.7476e-01,
        6.1666e-01, 3.3696e-02])

max probability is torch.return_types.max(
values=tensor(0.6167),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.1460e-04, 4.1290e-06, 4.1283e-06, 4.1275e-06]),
indices=tensor([  0, 914, 337, 771]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 524 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.0483e-05, 6.8075e-07, 5.7175e-06,
3.2221e-05, 3.2284e-05, 4.2149e-05,
```

```
        5.5869e-05, 1.7116e-05, 4.7931e-06, 2.2685e-04, 5.2935e-06, 3.5220e-03,
        1.9226e-04, 6.6955e-05, 9.9076e-01, 1.4597e-06, 5.8067e-05, 4.0366e-03,
        8.2503e-04, 2.2280e-05])

max probability is torch.return_types.max(
values=tensor(0.9908),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9001e-08, 1.0354e-09, 1.0353e-09, 1.0351e-09]),
indices=tensor([  0, 691, 748, 478]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 525 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.4407e-05, 1.6922e-03, 1.4522e-05,
1.9453e-06, 2.6849e-05, 2.1330e-05,
        2.2978e-05, 7.7867e-01, 2.1700e-01, 1.3148e-04, 8.9740e-05, 2.7219e-06,
        1.7720e-03, 8.5661e-06, 1.2929e-06, 3.8497e-04, 2.2758e-05, 4.3520e-07,
        4.4895e-05, 4.3415e-06])

max probability is torch.return_types.max(
values=tensor(0.7787),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2413e-08, 3.4628e-10, 3.4628e-10, 3.4605e-10]),
indices=tensor([  0, 836, 316, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 526 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.9653e-08, 2.0442e-08, 2.4265e-07,
2.7541e-07, 1.3361e-06, 2.5344e-08,
        1.8613e-08, 5.6468e-07, 6.3946e-07, 5.1738e-07, 8.7750e-06, 5.2398e-09,
        1.7621e-07, 8.2364e-04, 1.1430e-09, 6.9185e-07, 1.4624e-03, 1.7991e-05,
        4.5912e-03, 9.9271e-01])

max probability is torch.return_types.max(
values=tensor(0.9927),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([3.8408e-04, 4.3491e-13, 4.3488e-13, 4.3423e-13]),
indices=tensor([  0, 655, 337, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 527 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.4289e-06, 4.7523e-06, 1.5605e-08,
2.5636e-08, 1.5790e-07, 7.8060e-07,
        1.0689e-07, 4.6992e-07, 1.2821e-06, 1.5901e-06, 9.9527e-01, 1.1541e-04,
        3.4378e-05, 3.2806e-05, 3.3250e-08, 4.2956e-03, 5.8993e-05, 2.0527e-06,
        8.9894e-05, 9.1865e-05])

max probability is torch.return_types.max(
values=tensor(0.9953),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3781e-08, 4.3638e-12, 4.3603e-12, 4.3481e-12]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 528 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0772e-06, 1.5483e-06, 3.2211e-08,
1.1040e-08, 5.6031e-08, 1.1754e-07,
        6.4141e-08, 7.8537e-06, 9.9778e-07, 2.7448e-07, 4.8810e-04, 9.9654e-01,
        2.8998e-04, 7.2491e-06, 3.0276e-06, 9.6005e-06, 2.6125e-03, 1.4310e-06,
        2.8849e-05, 6.4811e-06])

max probability is torch.return_types.max(
values=tensor(0.9965),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.0510e-09, 6.1030e-12, 6.1014e-12, 6.0984e-12]),
indices=tensor([  0, 323, 914, 382]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 529 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.5965e-04, 3.9082e-05, 2.8407e-03,
2.0830e-03, 1.1109e-02, 4.4816e-04,
        2.8868e-04, 1.1968e-02, 6.1801e-03, 1.7814e-02, 7.2322e-04, 1.5284e-05,
```

```
        6.1957e-04, 1.4667e-02, 1.6612e-04, 7.5713e-04, 3.3599e-03, 1.6245e-03,
        8.4934e-01, 7.5059e-02])

max probability is torch.return_types.max(
values=tensor(0.8493),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.7763e-04, 1.6551e-07, 1.6550e-07, 1.6547e-07]),
indices=tensor([  0,  91, 197, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 530 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3763e-03, 3.2350e-04, 2.9273e-04,
7.0245e-05, 7.1531e-04, 3.6487e-04,
        1.8656e-04, 1.9364e-03, 9.6708e-03, 1.9268e-03, 2.9490e-02, 2.8625e-03,
        1.6021e-02, 1.3094e-01, 2.0456e-04, 9.9920e-02, 6.4592e-01, 3.1784e-03,
        3.4612e-02, 1.9507e-02])

max probability is torch.return_types.max(
values=tensor(0.6459),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1233e-04, 3.8890e-07, 3.8880e-07, 3.8873e-07]),
indices=tensor([  0, 323, 910, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 531 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.1619e-01, 5.0941e-05, 5.9059e-03,
2.3935e-03, 1.8103e-03, 7.6985e-03,
        2.2374e-03, 6.7273e-04, 2.7359e-03, 1.5765e-01, 2.3483e-04, 2.0713e-05,
        7.8650e-04, 6.5434e-04, 3.3933e-05, 4.8994e-05, 9.2260e-05, 1.7919e-04,
        2.8999e-04, 2.9793e-04])

max probability is torch.return_types.max(
values=tensor(0.8162),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.9399e-07, 1.5061e-08, 1.5047e-08, 1.5047e-08]),
```

```
indices=tensor([  0, 771, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 532 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.9462e-04, 2.2908e-04, 3.6183e-05,
1.1765e-05, 3.1992e-05, 6.8427e-05,
        3.0810e-05, 3.2403e-04, 1.4016e-04, 5.8382e-05, 7.9805e-03, 9.2505e-01,
        2.4065e-02, 1.3344e-03, 1.5625e-03, 1.1564e-03, 3.5063e-02, 4.5755e-04,
        1.4628e-03, 6.9027e-04])

max probability is torch.return_types.max(
values=tensor(0.9251),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.7385e-06, 5.2767e-08, 5.2760e-08, 5.2759e-08]),
indices=tensor([  0, 323, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 533 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4301e-03, 1.5450e-04, 6.8490e-04,
4.9716e-04, 1.5252e-03, 3.8581e-04,
        3.6187e-04, 4.4611e-04, 1.1206e-03, 1.7171e-03, 8.0580e-03, 1.7781e-04,
        1.0487e-03, 1.2963e-01, 3.4971e-05, 1.5239e-03, 8.9454e-02, 8.4639e-03,
        2.9073e-02, 7.1924e-01])

max probability is torch.return_types.max(
values=tensor(0.7192),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.5723e-03, 4.2256e-07, 4.2249e-07, 4.2244e-07]),
indices=tensor([  0, 655, 914, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 534 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0038, 0.0009, 0.0447, 0.0327, 0.1553,
0.0226, 0.0191, 0.0175, 0.0198,
        0.0410, 0.0028, 0.0006, 0.0031, 0.0338, 0.0047, 0.0032, 0.0166, 0.0353,
        0.4904, 0.0413])
```

```
max probability is torch.return_types.max(
values=tensor(0.4904),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2730e-03, 9.9233e-06, 9.9227e-06, 9.9223e-06]),
indices=tensor([  0, 197, 337,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 535 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.5809e-02, 1.8156e-03, 5.9485e-01,
1.5447e-01, 3.6597e-02, 8.6202e-02,
        3.4346e-02, 1.0938e-03, 2.5477e-03, 1.9104e-02, 1.5357e-04, 2.6267e-05,
        1.8071e-03, 3.8418e-04, 1.3024e-04, 2.6539e-05, 3.1297e-05, 2.2499e-04,
        1.3971e-04, 1.4595e-04])

max probability is torch.return_types.max(
values=tensor(0.5949),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2737e-06, 9.4858e-08, 9.4830e-08, 9.4814e-08]),
indices=tensor([  0, 771, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 536 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.3640e-05, 6.3310e-05, 1.8778e-01,
6.1847e-01, 1.0402e-01, 6.4563e-03,
        2.1611e-02, 4.3297e-05, 2.7413e-05, 1.0220e-03, 4.2820e-04, 1.6933e-06,
        8.8498e-04, 5.5927e-02, 4.2590e-04, 5.5463e-05, 1.0647e-04, 6.5514e-04,
        5.8467e-04, 1.3687e-03])

max probability is torch.return_types.max(
values=tensor(0.6185),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.8852e-06, 2.3894e-08, 2.3893e-08, 2.3892e-08]),
indices=tensor([  0, 910, 726, 323]))
End*****************************************************************************
```

```
**************************************************************************
Predicting Test Sample 537 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0286, 0.0076, 0.0022, 0.0014, 0.0071,
0.0131, 0.0053, 0.0105, 0.0421,
        0.0161, 0.0511, 0.0207, 0.0654, 0.1463, 0.0167, 0.1166, 0.1969, 0.0322,
        0.0894, 0.0575])

max probability is torch.return_types.max(
values=tensor(0.1969),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6449e-03, 7.4889e-05, 7.4882e-05, 7.4878e-05]),
indices=tensor([  0, 771, 323, 910]))
End*************************************************************************

**************************************************************************
Predicting Test Sample 538 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.4219e-04, 4.3489e-05, 3.8538e-04,
2.0673e-04, 1.1533e-03, 2.2601e-04,
        9.6503e-05, 8.2482e-04, 1.0223e-03, 1.4189e-03, 2.6639e-03, 3.9489e-03,
        1.6125e-03, 1.3590e-02, 8.5505e-03, 3.8491e-03, 4.2973e-02, 4.9723e-02,
        8.2678e-01, 3.9796e-02])

max probability is torch.return_types.max(
values=tensor(0.8268),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8457e-04, 4.2646e-07, 4.2638e-07, 4.2629e-07]),
indices=tensor([  0, 914, 337, 230]))
End*************************************************************************

**************************************************************************
Predicting Test Sample 539 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.8157e-05, 3.4996e-05, 1.6097e-06,
2.1323e-07, 1.3260e-06, 9.8680e-07,
        4.8887e-07, 8.3808e-05, 1.6124e-05, 2.0906e-06, 3.8445e-03, 9.5970e-01,
        6.0727e-03, 1.3673e-04, 1.0350e-04, 1.0286e-03, 2.8629e-02, 2.4920e-05,
        2.2098e-04, 8.2289e-05])

max probability is torch.return_types.max(
```

```
values=tensor(0.9597),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7831e-07, 5.5402e-10, 5.5394e-10, 5.5301e-10]),
indices=tensor([  0, 910, 323,  82]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 540 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7713e-03, 1.1096e-03, 4.7876e-01,
3.2788e-01, 1.2557e-01, 4.9417e-02,
        1.4905e-02, 1.9843e-05, 2.2542e-05, 1.9877e-04, 2.6242e-05, 7.1743e-07,
        6.5440e-05, 6.1964e-05, 1.9352e-05, 3.7461e-06, 5.0561e-06, 2.8982e-05,
        4.4833e-05, 4.8780e-05])

max probability is torch.return_types.max(
values=tensor(0.4788),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2501e-06, 3.8206e-08, 3.8186e-08, 3.8176e-08]),
indices=tensor([  0, 319, 771, 466]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 541 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.3653e-07, 3.0775e-08, 6.3768e-01,
2.9818e-01, 6.3839e-02, 2.0737e-04,
        8.4830e-05, 1.3133e-09, 4.4288e-10, 9.1188e-08, 9.7086e-10, 3.3766e-14,
        6.8159e-09, 1.3513e-07, 3.2823e-10, 8.1667e-12, 1.9190e-10, 2.6383e-09,
        8.4265e-09, 3.8164e-08])

max probability is torch.return_types.max(
values=tensor(0.6377),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3865e-11, 3.1063e-15, 3.1055e-15, 3.1044e-15]),
indices=tensor([  0, 771, 910, 466]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 542 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.5303e-04, 6.0352e-04, 4.1184e-01,
2.4359e-01, 2.7385e-01, 3.8765e-02,
        2.7825e-02, 1.0139e-04, 1.1180e-04, 6.0839e-04, 1.0972e-04, 1.8813e-06,
        2.6400e-04, 4.7000e-04, 7.7596e-05, 2.9265e-05, 3.5180e-05, 1.1824e-04,
        3.3902e-04, 2.0431e-04])

max probability is torch.return_types.max(
values=tensor(0.4118),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1353e-05, 5.2173e-07, 5.2168e-07, 5.2152e-07]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 543 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.3119e-03, 9.5809e-05, 6.7177e-04,
9.8274e-05, 4.6556e-04, 1.2001e-04,
        1.3436e-04, 8.9924e-04, 1.0387e-03, 8.4831e-04, 4.6984e-03, 1.5240e-01,
        1.5495e-02, 1.1162e-01, 2.5138e-03, 1.7666e-03, 6.3163e-01, 1.7432e-02,
        3.2418e-02, 2.0985e-02])

max probability is torch.return_types.max(
values=tensor(0.6316),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1600e-04, 2.5279e-07, 2.5272e-07, 2.5270e-07]),
indices=tensor([  0, 691,  82, 230]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 544 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.7615e-03, 2.8378e-04, 4.8928e-05,
1.4004e-05, 1.5576e-04, 1.2115e-04,
        3.1429e-05, 1.1258e-03, 5.9487e-03, 3.0411e-03, 1.0119e-01, 5.3633e-03,
        2.1783e-02, 3.5732e-02, 1.7366e-04, 2.7024e-01, 4.9792e-01, 1.9896e-03,
        2.9930e-02, 1.9003e-02])

max probability is torch.return_types.max(
values=tensor(0.4979),
```

```
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.2782e-05, 1.0593e-07, 1.0589e-07, 1.0587e-07]),
indices=tensor([  0, 771, 323, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 545 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.0756e-01, 4.1677e-01, 6.9507e-02,
1.5270e-02, 1.3654e-02, 6.2769e-02,
        5.6379e-02, 4.3897e-02, 7.1984e-02, 2.4124e-02, 2.7025e-03, 2.6799e-04,
        9.9370e-03, 5.2543e-04, 1.3916e-04, 2.1123e-03, 6.8072e-04, 1.6526e-04,
        5.0790e-04, 4.3192e-04])

max probability is torch.return_types.max(
values=tensor(0.4168),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.2542e-06, 6.4353e-07, 6.4291e-07, 6.4287e-07]),
indices=tensor([  0, 319, 316, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 546 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0147, 0.0258, 0.0088, 0.0029, 0.0102,
0.0151, 0.0158, 0.0640, 0.1281,
        0.0077, 0.0572, 0.0229, 0.3769, 0.0461, 0.0070, 0.0413, 0.0362, 0.0084,
        0.0114, 0.0144])

max probability is torch.return_types.max(
values=tensor(0.3769),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.0962e-04, 8.8315e-05, 8.8311e-05, 8.8287e-05]),
indices=tensor([  0, 910,  44, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 547 :   Prediction is Correct?
No
```

```
first 20 classes probability: tensor([9.6316e-05, 2.2344e-06, 3.3134e-05,
6.2680e-05, 8.5225e-05, 3.6073e-05,
        3.6913e-05, 2.5023e-05, 1.7958e-05, 8.4382e-05, 1.0472e-05, 5.4592e-05,
        5.2472e-05, 1.4697e-03, 7.0464e-01, 5.6995e-06, 3.8281e-05, 2.8807e-01,
        4.9592e-03, 2.1531e-04])

max probability is torch.return_types.max(
values=tensor(0.7046),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4067e-07, 1.8645e-09, 1.8630e-09, 1.8628e-09]),
indices=tensor([  0, 748, 428, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 548 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.2025e-06, 1.5332e-06, 1.0396e-05,
1.9899e-05, 4.7009e-05, 1.6687e-05,
        1.2650e-05, 4.4072e-05, 1.6695e-05, 1.0818e-04, 4.9689e-05, 1.4304e-03,
        4.4357e-04, 3.7836e-04, 9.5701e-01, 8.6611e-05, 3.3888e-04, 1.2305e-02,
        2.7313e-02, 3.4140e-04])

max probability is torch.return_types.max(
values=tensor(0.9570),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.9662e-07, 1.4365e-08, 1.4360e-08, 1.4360e-08]),
indices=tensor([  0, 748, 337, 255]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 549 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.1791e-03, 1.6109e-03, 2.9364e-04,
1.3688e-04, 9.3742e-04, 4.5254e-04,
        2.9042e-04, 6.1502e-01, 3.4869e-01, 9.1389e-03, 8.8013e-04, 2.3948e-05,
        3.8217e-03, 1.2295e-03, 8.9446e-05, 1.4036e-03, 3.2489e-04, 2.2967e-04,
        1.2559e-02, 6.1162e-04])

max probability is torch.return_types.max(
values=tensor(0.6150),
indices=tensor(7))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.0268e-06, 7.6138e-08, 7.6111e-08, 7.6104e-08]),
indices=tensor([  0, 316, 896,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 550 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2603e-03, 1.0373e-03, 7.1325e-05,
2.8621e-05, 6.4112e-05, 1.1494e-04,
        9.3950e-05, 3.1677e-03, 5.1360e-04, 1.1293e-04, 3.5499e-03, 9.6706e-01,
        4.1714e-03, 4.9734e-04, 5.2375e-04, 2.4650e-04, 1.4211e-02, 2.9619e-04,
        1.9735e-03, 8.0403e-04])

max probability is torch.return_types.max(
values=tensor(0.9671),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6559e-05, 1.8631e-07, 1.8627e-07, 1.8619e-07]),
indices=tensor([  0, 691, 382, 450]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 551 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.1121, 0.0172, 0.0064, 0.0015, 0.0074,
0.0229, 0.0080, 0.0175, 0.0872,
        0.0133, 0.0565, 0.0167, 0.1201, 0.1244, 0.0004, 0.0139, 0.3180, 0.0025,
        0.0082, 0.0308])

max probability is torch.return_types.max(
values=tensor(0.3180),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.2671e-04, 1.4863e-05, 1.4862e-05, 1.4856e-05]),
indices=tensor([  0, 910, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 552 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9854e-01, 7.6881e-05, 2.6756e-04,
5.1061e-05, 7.3595e-05, 5.0947e-04,
```

```
        2.4526e-04, 1.3780e-05, 4.1573e-05, 1.1421e-04, 8.4727e-07, 1.1366e-05,
        1.7098e-05, 1.1496e-06, 6.5933e-06, 1.8326e-07, 1.5924e-06, 1.7086e-05,
        3.5216e-06, 2.6448e-06])

max probability is torch.return_types.max(
values=tensor(0.9985),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6168e-09, 8.2641e-11, 8.2633e-11, 8.2606e-11]),
indices=tensor([  0, 323, 271, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 553 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.3821e-05, 7.8718e-08, 3.4629e-06,
6.7157e-07, 1.8150e-05, 3.5246e-07,
        2.7639e-07, 1.1242e-05, 1.6770e-05, 1.7607e-05, 7.1650e-05, 5.1712e-05,
        1.2647e-05, 1.1794e-03, 6.0038e-04, 2.9715e-05, 1.4623e-03, 2.9164e-01,
        6.8941e-01, 1.5430e-02])

max probability is torch.return_types.max(
values=tensor(0.6894),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6329e-06, 2.2420e-11, 2.2418e-11, 2.2409e-11]),
indices=tensor([  0, 337, 158,  19]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 554 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.5174e-05, 3.9376e-04, 2.0117e-01,
2.1224e-01, 5.3861e-01, 3.7218e-02,
        9.9215e-03, 7.4719e-06, 4.5021e-06, 1.2081e-05, 1.2927e-05, 2.8719e-07,
        1.7392e-05, 2.6518e-05, 1.9268e-05, 1.4319e-06, 1.7574e-06, 2.0403e-05,
        1.6705e-04, 2.1238e-05])

max probability is torch.return_types.max(
values=tensor(0.5386),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([1.2595e-06, 3.0500e-08, 3.0495e-08, 3.0477e-08]),
indices=tensor([  0, 319, 316, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 555 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0739e-03, 8.7814e-03, 2.7054e-04,
6.4232e-05, 5.4880e-04, 3.8780e-03,
        4.6874e-03, 2.1231e-01, 7.5485e-01, 1.6025e-03, 7.4189e-04, 2.9879e-05,
        9.8200e-03, 3.6458e-04, 9.8579e-06, 5.8795e-04, 1.9237e-04, 1.1930e-05,
        9.5762e-05, 5.4387e-05])

max probability is torch.return_types.max(
values=tensor(0.7549),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.1607e-07, 2.5927e-08, 2.5925e-08, 2.5923e-08]),
indices=tensor([  0, 319, 957, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 556 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.3773e-03, 1.7152e-02, 4.5887e-03,
5.3834e-04, 2.5857e-03, 1.0772e-03,
        1.2151e-03, 2.5518e-02, 3.1329e-02, 2.2995e-03, 4.7379e-02, 2.4206e-02,
        1.6792e-01, 1.0793e-01, 3.4265e-04, 8.3978e-03, 5.1270e-01, 1.5157e-03,
        9.2887e-03, 2.1113e-02])

max probability is torch.return_types.max(
values=tensor(0.5127),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.3174e-04, 4.0916e-06, 4.0915e-06, 4.0896e-06]),
indices=tensor([  0, 910, 323, 726]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 557 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.0127e-03, 5.4690e-04, 5.4468e-04,
3.0553e-04, 1.6912e-03, 5.9527e-04,
        2.5812e-04, 4.1226e-03, 2.5279e-03, 1.5289e-03, 2.4651e-02, 6.1274e-03,
```

```
        1.7344e-03, 1.6868e-02, 2.6532e-04, 3.2369e-03, 5.4663e-02, 1.6111e-02,
        3.2519e-01, 5.3032e-01])

max probability is torch.return_types.max(
values=tensor(0.5303),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.7322e-03, 1.0134e-06, 1.0133e-06, 1.0133e-06]),
indices=tensor([  0, 337, 158, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 558 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1462e-03, 2.0012e-03, 1.1331e-03,
7.1747e-04, 1.3249e-03, 1.9577e-03,
        1.2147e-03, 1.4834e-04, 7.0752e-04, 8.8400e-04, 9.3177e-01, 6.2560e-04,
        1.9840e-02, 1.4988e-02, 2.7524e-05, 1.7645e-02, 1.7801e-03, 2.5314e-04,
        7.4430e-04, 9.8518e-04])

max probability is torch.return_types.max(
values=tensor(0.9318),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.3301e-06, 1.0788e-07, 1.0779e-07, 1.0759e-07]),
indices=tensor([  0, 319, 771, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 559 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4978e-04, 1.3683e-05, 3.5354e-04,
2.8203e-04, 8.9872e-04, 2.5789e-04,
        1.1799e-04, 1.2483e-04, 8.7548e-04, 6.9292e-04, 3.1735e-03, 4.0202e-05,
        9.2051e-03, 9.4890e-01, 6.2625e-05, 2.3062e-03, 1.6761e-02, 1.4801e-03,
        3.0401e-03, 1.1246e-02])

max probability is torch.return_types.max(
values=tensor(0.9489),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.6812e-06, 7.9139e-09, 7.9136e-09, 7.9133e-09]),
```

```
indices=tensor([  0, 910, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 560 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0736, 0.0274, 0.0450, 0.0114, 0.0393,
0.0212, 0.0156, 0.0283, 0.0476,
        0.0253, 0.0509, 0.0062, 0.0371, 0.0509, 0.0070, 0.0671, 0.0604, 0.0537,
        0.1486, 0.0679])

max probability is torch.return_types.max(
values=tensor(0.1486),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([0.0029, 0.0001, 0.0001, 0.0001]),
indices=tensor([  0, 771, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 561 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0600, 0.0098, 0.0046, 0.0024, 0.0132,
0.1352, 0.0766, 0.0350, 0.4255,
        0.0269, 0.0353, 0.0018, 0.1324, 0.0162, 0.0007, 0.0113, 0.0034, 0.0021,
        0.0022, 0.0029])

max probability is torch.return_types.max(
values=tensor(0.4255),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.4297e-05, 2.6285e-06, 2.6255e-06, 2.6250e-06]),
indices=tensor([  0, 319, 836, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 562 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.0958e-02, 2.1845e-04, 3.5312e-03,
2.2543e-03, 9.4314e-03, 4.6409e-03,
        7.2596e-03, 1.3176e-02, 5.4478e-02, 1.9832e-01, 2.4847e-02, 7.7594e-04,
        5.0726e-03, 1.0292e-01, 2.2937e-03, 1.1529e-02, 2.4953e-02, 1.1736e-01,
        3.4759e-01, 5.7314e-02])
```

```
max probability is torch.return_types.max(
values=tensor(0.3476),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9348e-04, 9.2825e-07, 9.2820e-07, 9.2810e-07]),
indices=tensor([  0, 771, 914, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 563 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.9828e-03, 5.8706e-01, 1.0189e-01,
1.1025e-01, 7.8162e-02, 2.8172e-02,
        6.4820e-02, 5.5552e-03, 2.8769e-03, 2.8350e-03, 8.8924e-04, 9.4224e-05,
        2.5542e-03, 3.5580e-03, 3.1113e-04, 1.6396e-04, 1.3065e-03, 3.4322e-04,
        1.0396e-03, 9.7741e-04])

max probability is torch.return_types.max(
values=tensor(0.5871),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.6118e-05, 2.1830e-06, 2.1828e-06, 2.1827e-06]),
indices=tensor([  0, 323, 726, 620]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 564 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.1283e-02, 3.0078e-03, 2.4475e-03,
1.5802e-03, 7.6730e-03, 6.0906e-02,
        1.0759e-02, 1.1600e-02, 1.3997e-01, 3.9627e-02, 2.7228e-01, 1.7014e-03,
        6.2430e-02, 1.5178e-01, 1.4264e-04, 5.2471e-02, 3.8714e-02, 2.9061e-03,
        2.7960e-02, 7.6504e-02])

max probability is torch.return_types.max(
values=tensor(0.2723),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.7157e-04, 4.0246e-06, 4.0230e-06, 4.0202e-06]),
indices=tensor([  0, 771, 319, 316]))
End*****************************************************************************
```

```
**************************************************************************
Predicting Test Sample 565 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.4017e-04, 2.8285e-04, 9.0185e-02,
2.9769e-02, 1.1331e-01, 1.0902e-01,
        6.5163e-01, 4.2452e-05, 1.5526e-04, 9.9022e-04, 8.0370e-05, 5.5572e-06,
        2.1383e-04, 2.2892e-03, 5.1366e-04, 8.8978e-05, 6.3367e-05, 5.3995e-04,
        3.3512e-04, 1.2374e-04])

max probability is torch.return_types.max(
values=tensor(0.6516),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.3658e-07, 2.7086e-08, 2.7081e-08, 2.7079e-08]),
indices=tensor([  0, 910, 726, 748]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 566 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3673e-04, 1.0509e-05, 6.7463e-01,
2.3467e-01, 7.3309e-02, 8.2424e-03,
        8.1741e-03, 7.5544e-06, 8.8164e-06, 5.9393e-04, 4.2552e-06, 1.5699e-08,
        1.3875e-05, 1.3024e-04, 4.3350e-06, 6.6365e-07, 1.4603e-06, 1.9906e-05,
        2.0186e-05, 2.5864e-05])

max probability is torch.return_types.max(
values=tensor(0.6746),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2935e-07, 1.4744e-09, 1.4735e-09, 1.4734e-09]),
indices=tensor([  0, 771, 572, 466]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 567 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.5579e-04, 7.3199e-04, 3.5819e-01,
2.3384e-01, 3.1846e-01, 4.3572e-02,
        3.8570e-02, 1.4935e-04, 1.5827e-04, 8.5302e-04, 1.2730e-04, 5.3044e-06,
        3.6695e-04, 7.7217e-04, 3.9982e-04, 5.4892e-05, 5.5660e-05, 5.6666e-04,
        7.4483e-04, 2.7430e-04])

max probability is torch.return_types.max(
```

```
values=tensor(0.3582),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6056e-05, 1.1967e-06, 1.1966e-06, 1.1965e-06]),
indices=tensor([  0, 316, 319, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 568 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0209, 0.0052, 0.0021, 0.0019, 0.0053,
0.0054, 0.0104, 0.0113, 0.0116,
        0.0062, 0.0053, 0.0155, 0.0992, 0.0066, 0.7236, 0.0032, 0.0023, 0.0412,
        0.0039, 0.0013])

max probability is torch.return_types.max(
values=tensor(0.7236),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6244e-05, 1.8462e-05, 1.8457e-05, 1.8447e-05]),
indices=tensor([  0, 323, 910, 748]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 569 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.9778e-04, 5.1483e-04, 1.8526e-01,
2.3672e-01, 5.0565e-01, 5.5927e-02,
        1.5081e-02, 1.3755e-05, 9.8581e-06, 3.2741e-05, 2.2444e-05, 4.9867e-07,
        2.4269e-05, 6.5894e-05, 3.2536e-05, 3.6365e-06, 3.4550e-06, 3.6290e-05,
        2.7791e-04, 4.8564e-05])

max probability is torch.return_types.max(
values=tensor(0.5056),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8410e-06, 8.4216e-08, 8.4192e-08, 8.4130e-08]),
indices=tensor([  0, 319, 316, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 570 :   Prediction is Correct?
```

```
Yes
first 20 classes probability: tensor([2.7374e-04, 8.4585e-03, 7.6789e-05,
1.2773e-05, 1.0244e-04, 3.6080e-05,
        4.0452e-05, 8.6356e-01, 1.2588e-01, 1.1592e-04, 4.5570e-05, 1.0253e-06,
        1.3088e-03, 6.8363e-06, 1.5261e-06, 5.0704e-05, 4.2810e-06, 4.4775e-07,
        1.7602e-05, 2.5058e-06])

max probability is torch.return_types.max(
values=tensor(0.8636),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6329e-08, 3.0001e-10, 2.9989e-10, 2.9982e-10]),
indices=tensor([  0, 316, 319, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 571 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.9454e-04, 4.0507e-05, 9.1927e-05,
8.1079e-05, 2.5280e-04, 1.1528e-04,
        9.0733e-05, 4.8470e-04, 2.7584e-04, 5.0187e-04, 6.9824e-04, 1.4847e-02,
        2.9666e-03, 4.9497e-03, 7.0510e-01, 4.1304e-04, 3.9032e-03, 2.1678e-01,
        4.4296e-02, 3.3413e-03])

max probability is torch.return_types.max(
values=tensor(0.7051),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3015e-05, 2.6637e-07, 2.6633e-07, 2.6627e-07]),
indices=tensor([  0, 748, 230, 428]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 572 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.9578e-04, 1.0635e-03, 1.0418e-01,
2.2781e-01, 4.4040e-01, 1.7756e-01,
        4.6295e-02, 3.8907e-05, 4.0085e-05, 1.1731e-04, 5.2334e-05, 3.6396e-06,
        6.2408e-05, 1.7505e-04, 1.5716e-04, 1.6315e-05, 8.2813e-06, 1.0104e-04,
        7.7940e-04, 1.1031e-04])

max probability is torch.return_types.max(
values=tensor(0.4404),
indices=tensor(4))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.3452e-06, 3.4394e-07, 3.4388e-07, 3.4362e-07]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 573 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4555e-03, 2.5639e-04, 4.5038e-03,
1.8711e-01, 3.6310e-02, 6.9601e-01,
        7.3423e-02, 2.5412e-06, 3.3364e-06, 7.8745e-05, 4.4346e-06, 5.3142e-06,
        7.8034e-06, 9.0062e-05, 5.2451e-04, 5.7505e-07, 3.3140e-07, 6.4974e-05,
        1.3041e-04, 2.4410e-05])

max probability is torch.return_types.max(
values=tensor(0.6960),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.7690e-08, 2.3955e-09, 2.3949e-09, 2.3947e-09]),
indices=tensor([  0, 316, 319, 144]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 574 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.4456e-04, 4.3430e-04, 6.7383e-04,
1.3637e-04, 7.1547e-04, 2.3431e-04,
        3.0605e-04, 5.9493e-01, 3.7674e-01, 5.0062e-03, 5.2108e-04, 8.0652e-05,
        1.5460e-02, 6.0224e-04, 8.5634e-05, 4.3231e-04, 4.6548e-04, 1.4112e-04,
        1.7943e-03, 2.5295e-04])

max probability is torch.return_types.max(
values=tensor(0.5949),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1790e-06, 4.4191e-08, 4.4182e-08, 4.4173e-08]),
indices=tensor([  0, 316,  85, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 575 :   Prediction is Correct?
Yes
```

```
first 20 classes probability: tensor([1.1168e-05, 5.3777e-05, 2.6410e-07,
1.3949e-07, 9.2573e-07, 1.6559e-06,
        3.5185e-07, 1.0088e-06, 5.1778e-06, 4.0695e-06, 9.8283e-01, 8.0062e-04,
        5.3727e-04, 8.0444e-04, 6.2800e-07, 9.8168e-03, 4.3006e-03, 2.9632e-05,
        5.3580e-04, 2.6478e-04])

max probability is torch.return_types.max(
values=tensor(0.9828),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.3072e-08, 3.2000e-11, 3.1996e-11, 3.1955e-11]),
indices=tensor([  0, 319, 771, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 576 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3940e-03, 1.7988e-03, 6.8728e-02,
1.3121e-01, 1.9732e-01, 1.2279e-01,
        4.4754e-01, 5.2056e-04, 6.4548e-04, 6.2650e-03, 3.7828e-04, 6.8313e-05,
        1.1153e-03, 4.1458e-03, 7.6517e-03, 4.9848e-04, 2.1845e-04, 2.6979e-03,
        2.0173e-03, 6.2054e-04])

max probability is torch.return_types.max(
values=tensor(0.4475),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8991e-05, 2.4741e-06, 2.4737e-06, 2.4736e-06]),
indices=tensor([  0, 910, 323, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 577 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.1217e-04, 3.6429e-05, 7.2412e-01,
1.2530e-01, 1.1415e-01, 1.3576e-02,
        2.1370e-02, 2.2059e-05, 3.3007e-05, 7.6028e-04, 1.2703e-05, 1.7143e-07,
        4.6615e-05, 1.3250e-04, 2.1079e-05, 3.6279e-06, 8.4655e-06, 7.1278e-05,
        6.4365e-05, 3.6902e-05])

max probability is torch.return_types.max(
values=tensor(0.7241),
indices=tensor(2))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.0128e-07, 1.2276e-08, 1.2275e-08, 1.2273e-08]),
indices=tensor([  0, 771, 910,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 578 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.1929e-08, 6.7057e-10, 5.7048e-10,
1.0947e-08, 3.9717e-08, 8.4729e-08,
        6.1726e-08, 4.9868e-09, 8.6299e-10, 3.0287e-08, 3.0537e-09, 2.3253e-06,
        3.3951e-07, 7.2002e-08, 9.9973e-01, 4.3761e-09, 6.8669e-09, 2.6301e-04,
        3.4712e-06, 1.0814e-08])

max probability is torch.return_types.max(
values=tensor(0.9997),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5204e-13, 1.5422e-14, 1.5421e-14, 1.5417e-14]),
indices=tensor([  0, 337, 812, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 579 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.1838e-06, 1.4391e-05, 1.2856e-05,
1.3449e-05, 2.3978e-05, 4.8897e-05,
        1.9320e-05, 2.0162e-06, 2.0004e-05, 5.3876e-05, 9.9053e-01, 1.0944e-05,
        3.6546e-03, 2.8331e-03, 1.3036e-07, 2.3798e-03, 2.4203e-04, 3.0859e-06,
        2.7762e-05, 1.0639e-04])

max probability is torch.return_types.max(
values=tensor(0.9905),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.4121e-08, 7.2294e-11, 7.2280e-11, 7.2070e-11]),
indices=tensor([  0, 319, 771, 363]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 580 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.2765e-03, 1.7190e-03, 1.2600e-01,
```

```
2.1551e-01, 4.3402e-01, 1.4930e-01,
        6.1625e-02, 1.8489e-04, 2.2632e-04, 5.0858e-04, 3.6643e-04, 2.2591e-05,
        3.8918e-04, 9.8271e-04, 4.6145e-04, 1.2743e-04, 6.7536e-05, 3.9552e-04,
        2.1996e-03, 5.1523e-04])

max probability is torch.return_types.max(
values=tensor(0.4340),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.8456e-05, 4.2503e-06, 4.2491e-06, 4.2462e-06]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 581 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0045, 0.0028, 0.0116, 0.0019, 0.0066,
0.0017, 0.0013, 0.0055, 0.0106,
        0.0023, 0.2917, 0.0083, 0.1376, 0.0689, 0.0039, 0.0701, 0.1166, 0.0482,
        0.0942, 0.1073])

max probability is torch.return_types.max(
values=tensor(0.2917),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.6076e-04, 3.8923e-06, 3.8910e-06, 3.8908e-06]),
indices=tensor([  0, 771, 323, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 582 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0051, 0.0023, 0.0007, 0.0003, 0.0013,
0.0018, 0.0007, 0.0024, 0.0075,
        0.0040, 0.3037, 0.0518, 0.1186, 0.0823, 0.0024, 0.1567, 0.2144, 0.0058,
        0.0163, 0.0162])

max probability is torch.return_types.max(
values=tensor(0.3037),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9292e-04, 5.9277e-06, 5.9267e-06, 5.9253e-06]),
```

```
indices=tensor([  0, 771, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 583 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.3255e-01, 7.1043e-03, 7.1173e-02,
4.1841e-02, 3.2397e-02, 5.3766e-01,
        5.8466e-02, 4.7965e-04, 4.2214e-03, 7.0927e-03, 1.2077e-03, 5.8201e-05,
        1.9345e-03, 1.9703e-03, 7.4521e-05, 3.6326e-04, 8.4693e-05, 1.6733e-04,
        3.4251e-04, 4.6394e-04])

max probability is torch.return_types.max(
values=tensor(0.5377),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.5820e-06, 3.6087e-07, 3.6048e-07, 3.6034e-07]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 584 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2447e-02, 2.3251e-02, 6.6125e-03,
1.3917e-03, 5.5954e-03, 4.2208e-03,
        9.4062e-03, 4.0777e-01, 4.8994e-01, 5.0115e-03, 1.6353e-03, 1.3228e-04,
        2.9055e-02, 8.1974e-04, 1.2801e-04, 9.4373e-04, 3.7334e-04, 1.2727e-04,
        4.2695e-04, 1.8088e-04])

max probability is torch.return_types.max(
values=tensor(0.4899),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.9770e-06, 5.6292e-07, 5.6281e-07, 5.6270e-07]),
indices=tensor([  0, 319, 836, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 585 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.3089e-05, 1.7407e-07, 3.7591e-07,
1.2968e-06, 6.9494e-06, 7.9686e-06,
        4.3115e-06, 7.4924e-06, 6.9880e-06, 3.4399e-05, 3.0859e-06, 5.5643e-05,
        2.2577e-05, 2.5188e-05, 9.5510e-01, 2.7067e-06, 2.5933e-06, 4.1129e-02,
```

```
        3.5173e-03, 1.4261e-05])

max probability is torch.return_types.max(
values=tensor(0.9551),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.6130e-09, 2.6819e-10, 2.6813e-10, 2.6809e-10]),
indices=tensor([  0, 337, 748, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 586 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.3260e-02, 2.9092e-05, 2.0142e-02,
6.7936e-03, 9.9449e-03, 2.8663e-03,
        3.6645e-03, 2.2282e-03, 5.0478e-03, 7.0265e-01, 1.5009e-03, 2.5540e-04,
        2.8429e-03, 5.5999e-02, 1.2114e-04, 5.9275e-04, 4.5323e-02, 5.7047e-03,
        1.8222e-02, 2.2675e-02])

max probability is torch.return_types.max(
values=tensor(0.7027),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.7036e-05, 1.0612e-07, 1.0609e-07, 1.0608e-07]),
indices=tensor([  0, 914, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 587 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.6356e-03, 7.8679e-04, 4.1066e-04,
3.3699e-04, 1.6793e-03, 2.7922e-03,
        1.8046e-03, 2.2284e-03, 5.1438e-03, 2.5567e-03, 1.4856e-02, 4.3407e-02,
        9.6251e-02, 5.8968e-02, 5.1615e-01, 2.7498e-02, 5.5058e-02, 7.9747e-02,
        7.3938e-02, 1.1519e-02])

max probability is torch.return_types.max(
values=tensor(0.5162),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.6066e-05, 3.3194e-06, 3.3193e-06, 3.3180e-06]),
indices=tensor([  0, 323, 910, 771]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 588 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.8179e-05, 3.3637e-04, 2.5708e-01,
2.1941e-01, 4.8995e-01, 2.4569e-02,
        8.3205e-03, 7.2670e-06, 4.3536e-06, 1.3213e-05, 9.6278e-06, 1.7536e-07,
        2.0141e-05, 2.5229e-05, 1.6001e-05, 1.1346e-06, 1.7152e-06, 1.8857e-05,
        1.0729e-04, 1.5960e-05])

max probability is torch.return_types.max(
values=tensor(0.4900),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.1236e-07, 1.7908e-08, 1.7905e-08, 1.7894e-08]),
indices=tensor([  0, 319, 316, 957]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 589 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.1040, 0.0097, 0.0174, 0.0195, 0.0234,
0.0507, 0.0293, 0.0072, 0.0060,
        0.0358, 0.0645, 0.4068, 0.0246, 0.0280, 0.0084, 0.0095, 0.0284, 0.0171,
        0.0443, 0.0208])

max probability is torch.return_types.max(
values=tensor(0.4068),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.6094e-04, 4.6105e-05, 4.6092e-05, 4.6083e-05]),
indices=tensor([  0, 914, 323, 158]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 590 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.2946e-03, 9.3027e-04, 3.8516e-01,
3.6209e-01, 1.9800e-01, 3.2052e-02,
        1.7052e-02, 1.3601e-04, 1.3329e-04, 9.7669e-04, 1.0824e-04, 1.3701e-06,
        3.3488e-04, 6.9979e-04, 5.8254e-05, 2.1823e-05, 2.7983e-05, 9.3343e-05,
        1.7989e-04, 2.3458e-04])
```

```
max probability is torch.return_types.max(
values=tensor(0.3852),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.6482e-06, 4.2927e-07, 4.2916e-07, 4.2897e-07]),
indices=tensor([  0, 319, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 591 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.3096e-03, 4.1102e-04, 7.2365e-02,
3.0841e-01, 1.4153e-01, 2.9622e-01,
        1.0237e-01, 4.1898e-04, 7.3998e-04, 2.2578e-02, 1.1759e-03, 1.3269e-04,
        7.9379e-04, 1.9689e-02, 4.4166e-03, 3.7079e-04, 3.3620e-04, 4.4726e-03,
        1.1349e-02, 5.3568e-03])

max probability is torch.return_types.max(
values=tensor(0.3084),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.6302e-05, 2.6211e-06, 2.6204e-06, 2.6195e-06]),
indices=tensor([  0, 771, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 592 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.6434e-02, 1.5296e-03, 1.3766e-01,
1.9045e-02, 2.5380e-02, 1.3073e-02,
        2.3458e-02, 1.4998e-01, 3.0285e-01, 2.7408e-01, 3.1069e-03, 2.6256e-05,
        2.1646e-02, 5.6540e-03, 1.2998e-04, 1.6596e-03, 7.3421e-04, 4.7504e-04,
        1.7870e-03, 8.2900e-04])

max probability is torch.return_types.max(
values=tensor(0.3029),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.6842e-06, 4.8054e-07, 4.8031e-07, 4.8030e-07]),
indices=tensor([  0, 896,  85, 771]))
End*****************************************************************************
```

```
**************************************************************************
Predicting Test Sample 593 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.0446e-02, 6.5342e-04, 1.0489e-02,
1.0610e-02, 3.3033e-02, 2.0013e-02,
        3.2324e-02, 4.8506e-03, 2.8065e-02, 7.4662e-02, 1.9785e-02, 4.5232e-04,
        2.6332e-02, 6.4415e-01, 6.1481e-04, 1.1341e-02, 2.9768e-02, 4.5146e-03,
        1.0761e-02, 2.4242e-02])

max probability is torch.return_types.max(
values=tensor(0.6441),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5456e-04, 2.8827e-06, 2.8826e-06, 2.8821e-06]),
indices=tensor([  0, 771, 323, 910]))
End*************************************************************************

**************************************************************************
Predicting Test Sample 594 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0081, 0.0097, 0.0119, 0.0134, 0.0448,
0.4230, 0.2020, 0.0019, 0.0156,
        0.0119, 0.0432, 0.0030, 0.0856, 0.0896, 0.0016, 0.0132, 0.0090, 0.0012,
        0.0060, 0.0039])

max probability is torch.return_types.max(
values=tensor(0.4230),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.0705e-05, 1.7295e-06, 1.7295e-06, 1.7292e-06]),
indices=tensor([  0, 771, 319, 910]))
End*************************************************************************

**************************************************************************
Predicting Test Sample 595 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0307, 0.0016, 0.0183, 0.0073, 0.0241,
0.0059, 0.0037, 0.0345, 0.0426,
        0.0535, 0.0299, 0.0058, 0.0431, 0.0723, 0.0039, 0.0215, 0.0812, 0.0216,
        0.3684, 0.0987])

max probability is torch.return_types.max(
values=tensor(0.3684),
indices=tensor(18))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1802e-03, 3.0498e-05, 3.0493e-05, 3.0484e-05]),
indices=tensor([  0, 914, 771, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 596 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.2510e-03, 9.7749e-03, 4.8036e-04,
8.7118e-05, 4.6587e-04, 4.4844e-04,
        4.1328e-04, 6.7523e-01, 2.8962e-01, 9.8528e-04, 1.0224e-03, 3.7429e-04,
        1.1934e-02, 2.0711e-04, 5.4253e-05, 4.9592e-04, 4.4369e-04, 3.2929e-05,
        4.4081e-04, 9.8059e-05])

max probability is torch.return_types.max(
values=tensor(0.6752),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8080e-06, 1.4866e-07, 1.4865e-07, 1.4859e-07]),
indices=tensor([  0, 836, 316, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 597 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.8552e-05, 1.8094e-04, 2.4300e-06,
5.7324e-07, 8.8273e-06, 1.6653e-05,
        3.5036e-06, 4.5956e-04, 1.0106e-03, 3.8777e-05, 7.7114e-02, 4.1396e-05,
        3.7382e-03, 5.0075e-04, 7.1502e-06, 9.1333e-01, 6.3644e-04, 4.7253e-05,
        2.2761e-03, 5.5350e-04])

max probability is torch.return_types.max(
values=tensor(0.9133),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3213e-07, 4.7337e-10, 4.7248e-10, 4.7217e-10]),
indices=tensor([  0, 319, 316, 476]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 598 :   Prediction is Correct?
No
```

```
first 20 classes probability: tensor([0.0052, 0.0008, 0.1718, 0.0478, 0.0887,
0.0093, 0.0098, 0.0051, 0.0056,
        0.0191, 0.0106, 0.0013, 0.0212, 0.2058, 0.0017, 0.0027, 0.0600, 0.0254,
        0.1077, 0.1951])

max probability is torch.return_types.max(
values=tensor(0.2058),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9665e-03, 3.5394e-06, 3.5389e-06, 3.5385e-06]),
indices=tensor([  0, 150, 158, 691]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 599 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.5851e-04, 1.3698e-04, 3.6059e-02,
4.8104e-01, 1.8540e-01, 8.4085e-02,
        8.7160e-02, 1.0713e-04, 2.3948e-04, 4.6090e-03, 3.8624e-04, 1.3522e-06,
        1.6571e-04, 1.1221e-01, 1.2609e-04, 1.2705e-04, 1.0859e-04, 3.8081e-04,
        1.9034e-03, 4.9607e-03])

max probability is torch.return_types.max(
values=tensor(0.4810),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9646e-05, 1.2645e-07, 1.2641e-07, 1.2640e-07]),
indices=tensor([  0, 771, 319, 910]))
End*****************************************************************************

currently at 600 current time is 18.799937963485718
*****************************************************************************
Predicting Test Sample 600 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4621e-07, 2.1381e-10, 1.0831e-07,
1.9551e-08, 5.4598e-07, 1.6058e-09,
        4.2039e-09, 1.0358e-07, 1.9498e-07, 1.6251e-07, 1.5810e-07, 2.1899e-08,
        9.0708e-08, 4.4793e-04, 3.8450e-04, 1.1106e-07, 1.2664e-05, 8.8985e-01,
        1.0432e-01, 4.9879e-03])

max probability is torch.return_types.max(
values=tensor(0.8898),
indices=tensor(17))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8005e-08, 2.5791e-15, 2.5789e-15, 2.5784e-15]),
indices=tensor([  0, 500, 337,  19]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 601 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.1864e-04, 4.5413e-04, 2.2742e-02,
1.9336e-01, 3.0053e-01, 3.8883e-01,
        9.2710e-02, 8.4700e-06, 1.4212e-05, 8.9933e-05, 2.4439e-05, 1.2512e-06,
        1.8588e-05, 1.4437e-04, 8.2992e-05, 1.2037e-05, 2.0880e-06, 3.3140e-05,
        3.1772e-04, 5.4005e-05])

max probability is torch.return_types.max(
values=tensor(0.3888),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0937e-06, 4.6718e-08, 4.6683e-08, 4.6651e-08]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 602 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.8428e-04, 6.9756e-06, 1.8730e-02,
1.5382e-02, 9.2169e-03, 1.7804e-03,
        2.0994e-03, 1.7423e-04, 2.1492e-04, 8.3672e-03, 1.3062e-03, 3.1443e-04,
        1.4661e-03, 2.8290e-01, 6.0138e-02, 2.3451e-04, 3.1378e-03, 4.9499e-01,
        6.8721e-02, 3.0077e-02])

max probability is torch.return_types.max(
values=tensor(0.4950),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7904e-05, 4.2441e-08, 4.2421e-08, 4.2421e-08]),
indices=tensor([  0, 748, 150, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 603 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0067, 0.0032, 0.2539, 0.0304, 0.1103,
```

```
0.2254, 0.2187, 0.0015, 0.0122,
        0.0166, 0.0136, 0.0003, 0.0403, 0.0377, 0.0007, 0.0084, 0.0089, 0.0019,
        0.0039, 0.0037])

max probability is torch.return_types.max(
values=tensor(0.2539),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.2892e-05, 1.5850e-06, 1.5850e-06, 1.5844e-06]),
indices=tensor([  0, 771, 910, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 604 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.0061e-04, 7.6894e-04, 3.1539e-05,
6.4322e-06, 8.9726e-05, 1.0889e-04,
        5.0097e-05, 3.1830e-03, 5.8451e-03, 3.1060e-04, 1.9917e-02, 4.2883e-05,
        5.9654e-03, 1.0537e-03, 3.4042e-05, 9.5908e-01, 1.3966e-03, 6.3046e-05,
        1.4165e-03, 4.2765e-04])

max probability is torch.return_types.max(
values=tensor(0.9591),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.7933e-07, 1.1599e-08, 1.1590e-08, 1.1585e-08]),
indices=tensor([  0, 319, 316, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 605 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.4152e-05, 2.1554e-04, 5.9190e-05,
4.3432e-06, 5.4227e-05, 4.7517e-05,
        8.2945e-05, 3.9761e-01, 5.6382e-01, 1.1001e-04, 2.2163e-04, 4.9976e-06,
        3.7345e-02, 6.4686e-05, 3.9941e-06, 2.5555e-04, 3.0697e-05, 3.5122e-06,
        2.3426e-05, 1.3393e-05])

max probability is torch.return_types.max(
values=tensor(0.5638),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([2.3508e-08, 2.4083e-10, 2.4074e-10, 2.4067e-10]),
indices=tensor([  0, 836, 319, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 606 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.7101e-03, 3.4316e-01, 8.4915e-05,
8.0568e-06, 8.4884e-05, 9.6654e-05,
        1.8459e-04, 4.5156e-01, 1.8583e-01, 5.8299e-05, 1.0209e-04, 3.6638e-05,
        1.5297e-02, 3.2309e-05, 4.7213e-05, 5.1837e-04, 1.6488e-04, 2.5578e-06,
        1.6736e-05, 5.5844e-06])

max probability is torch.return_types.max(
values=tensor(0.4516),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.2772e-08, 1.2899e-09, 1.2899e-09, 1.2896e-09]),
indices=tensor([  0, 316, 319, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 607 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0416, 0.3094, 0.0041, 0.0010, 0.0043,
0.0191, 0.0077, 0.0184, 0.0605,
        0.0022, 0.0943, 0.0149, 0.2075, 0.0665, 0.0008, 0.0577, 0.0525, 0.0011,
        0.0055, 0.0133])

max probability is torch.return_types.max(
values=tensor(0.3094),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.6856e-04, 1.8073e-05, 1.8066e-05, 1.8065e-05]),
indices=tensor([  0, 910, 319, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 608 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.3723e-03, 2.0902e-03, 2.3838e-04,
2.7438e-04, 1.2641e-03, 5.2339e-03,
        1.8416e-03, 7.3052e-03, 4.0152e-02, 1.8944e-02, 5.4734e-01, 1.0415e-03,
        3.0242e-02, 1.2220e-02, 2.8852e-04, 3.0408e-01, 6.4601e-03, 9.0851e-04,
```

```
        8.6279e-03, 6.2123e-03])

max probability is torch.return_types.max(
values=tensor(0.5473),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.9457e-05, 1.9561e-06, 1.9523e-06, 1.9519e-06]),
indices=tensor([  0, 319, 836, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 609 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.7575e-02, 2.5420e-03, 7.4136e-02,
2.8760e-02, 8.5197e-02, 2.6463e-01,
        1.4313e-01, 9.7168e-03, 1.0405e-01, 1.0050e-01, 3.0778e-02, 2.5720e-04,
        2.8535e-02, 5.2070e-02, 9.4219e-04, 1.2926e-02, 4.7810e-03, 8.4693e-03,
        1.0301e-02, 1.3833e-02])

max probability is torch.return_types.max(
values=tensor(0.2646),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5724e-04, 7.1055e-06, 7.1054e-06, 7.1012e-06]),
indices=tensor([  0, 771, 319, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 610 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2324e-04, 5.4539e-04, 9.5571e-03,
5.9934e-02, 1.2423e-01, 9.9086e-02,
        7.0381e-01, 4.1516e-05, 6.1850e-05, 7.1091e-04, 3.6787e-05, 3.3796e-06,
        1.0551e-04, 7.4143e-04, 5.6424e-04, 4.5506e-05, 1.0359e-05, 1.4244e-04,
        1.9082e-04, 4.8227e-05])

max probability is torch.return_types.max(
values=tensor(0.7038),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6920e-07, 1.4145e-08, 1.4144e-08, 1.4143e-08]),
indices=tensor([  0, 910, 316, 319]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 611 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.9713e-03, 2.5201e-03, 9.5109e-02,
1.5765e-01, 3.4787e-01, 2.2958e-01,
        1.2903e-01, 7.6681e-04, 1.3975e-03, 3.0046e-03, 8.0043e-04, 9.7892e-05,
        1.0058e-03, 3.0291e-03, 2.2494e-03, 4.9178e-04, 2.3836e-04, 2.3535e-03,
        5.5843e-03, 1.4507e-03])

max probability is torch.return_types.max(
values=tensor(0.3479),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0669e-04, 1.2254e-05, 1.2252e-05, 1.2246e-05]),
indices=tensor([  0, 319, 316,  85]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 612 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.1588e-04, 2.9050e-05, 9.7300e-04,
7.1462e-03, 3.8035e-03, 1.5742e-03,
        2.2728e-03, 4.3805e-04, 3.1975e-04, 7.7945e-03, 2.9901e-04, 4.6497e-04,
        6.0175e-04, 2.1888e-01, 4.1470e-01, 3.4257e-04, 3.0864e-03, 2.3146e-01,
        7.8956e-02, 2.5549e-02])

max probability is torch.return_types.max(
values=tensor(0.4147),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.4155e-05, 5.4885e-07, 5.4879e-07, 5.4875e-07]),
indices=tensor([  0, 748, 337, 687]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 613 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.2586e-06, 1.9171e-04, 8.9486e-07,
1.2315e-07, 3.1909e-06, 2.5252e-06,
        8.5906e-07, 4.6896e-05, 7.9052e-05, 5.4540e-06, 1.4951e-02, 5.9918e-05,
        1.5129e-03, 5.5449e-04, 5.7210e-06, 9.7496e-01, 7.1041e-03, 1.3719e-05,
        2.6863e-04, 2.2693e-04])
```

```
max probability is torch.return_types.max(
values=tensor(0.9750),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.5712e-08, 1.2120e-10, 1.2096e-10, 1.2089e-10]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 614 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.0690e-03, 5.3092e-04, 9.0682e-02,
2.0222e-02, 6.5823e-02, 3.7634e-01,
        2.8793e-01, 2.0648e-03, 2.0476e-02, 7.3244e-02, 4.7610e-03, 2.5806e-04,
        9.5725e-03, 2.4209e-02, 7.5720e-04, 2.1607e-03, 2.1952e-03, 2.6957e-03,
        4.8735e-03, 3.2852e-03])

max probability is torch.return_types.max(
values=tensor(0.3763),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1282e-05, 8.7165e-07, 8.7143e-07, 8.7130e-07]),
indices=tensor([  0, 771, 914, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 615 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7059e-05, 6.8144e-06, 9.8719e-06,
5.7937e-06, 2.9898e-05, 2.0756e-06,
        3.5988e-06, 7.1110e-05, 7.7736e-05, 1.5742e-05, 7.3624e-04, 5.2637e-06,
        3.1253e-05, 8.9123e-04, 6.2610e-08, 1.3108e-05, 1.8995e-02, 8.0608e-05,
        9.3389e-03, 9.6527e-01])

max probability is torch.return_types.max(
values=tensor(0.9653),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.4008e-03, 5.5228e-10, 5.5203e-10, 5.5197e-10]),
indices=tensor([  0, 655, 337, 691]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 616 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.0370e-05, 2.5336e-05, 7.1751e-01,
1.6023e-01, 1.1160e-01, 4.9688e-03,
        5.1904e-03, 9.9454e-06, 8.3137e-06, 2.4110e-04, 8.9047e-06, 3.5315e-08,
        2.6354e-05, 6.9367e-05, 3.4382e-06, 1.1434e-06, 6.3272e-06, 1.2570e-05,
        2.7737e-05, 1.8840e-05])

max probability is torch.return_types.max(
values=tensor(0.7175),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.1549e-07, 3.4682e-09, 3.4678e-09, 3.4675e-09]),
indices=tensor([  0, 910, 771, 466]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 617 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.0470e-06, 9.4322e-06, 5.8201e-08,
2.3603e-08, 2.4879e-07, 1.7578e-06,
        1.3514e-07, 6.8766e-07, 7.0823e-06, 1.2795e-06, 9.1289e-01, 5.6344e-05,
        1.3404e-03, 2.0886e-04, 2.2002e-07, 8.4496e-02, 6.6885e-04, 4.9243e-06,
        1.0219e-04, 2.1111e-04])

max probability is torch.return_types.max(
values=tensor(0.9129),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3661e-08, 5.4887e-12, 5.4782e-12, 5.4660e-12]),
indices=tensor([  0, 319, 771, 476]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 618 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.0606e-03, 1.7291e-03, 1.7397e-01,
3.0247e-01, 3.5966e-01, 1.1433e-01,
        4.3305e-02, 8.4210e-05, 7.5114e-05, 2.6381e-04, 9.8749e-05, 5.2942e-06,
        1.2995e-04, 3.5781e-04, 1.6397e-04, 2.6464e-05, 2.3923e-05, 1.2614e-04,
        6.1198e-04, 2.1304e-04])
```

```
max probability is torch.return_types.max(
values=tensor(0.3597),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9438e-05, 1.3396e-06, 1.3392e-06, 1.3382e-06]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 619 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.9748e-06, 7.7007e-06, 1.2583e-06,
6.1107e-07, 1.4544e-06, 2.0247e-06,
        1.3238e-06, 3.0256e-05, 3.2334e-06, 1.7648e-06, 5.5629e-04, 9.9693e-01,
        5.4011e-04, 7.0320e-05, 3.8341e-05, 4.9599e-06, 1.4934e-03, 2.2513e-05,
        2.1741e-04, 7.1694e-05])

max probability is torch.return_types.max(
values=tensor(0.9969),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1801e-07, 2.1809e-10, 2.1796e-10, 2.1791e-10]),
indices=tensor([  0, 691,  82, 382]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 620 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7803e-03, 1.2922e-03, 1.1247e-04,
1.3246e-05, 8.0347e-05, 4.3952e-04,
        3.0570e-04, 2.4362e-01, 7.3594e-01, 1.4934e-03, 2.7903e-04, 2.2512e-06,
        1.4218e-02, 6.4985e-05, 1.2929e-06, 3.1342e-04, 2.1835e-05, 8.9976e-07,
        7.0347e-06, 5.2243e-06])

max probability is torch.return_types.max(
values=tensor(0.7359),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5742e-08, 7.4641e-10, 7.4638e-10, 7.4548e-10]),
indices=tensor([  0, 319, 836, 794]))
End*****************************************************************************
```

```
**************************************************************************
Predicting Test Sample 621 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0129, 0.0024, 0.0146, 0.0077, 0.0219,
0.0194, 0.0078, 0.0260, 0.0605,
        0.0290, 0.1122, 0.0089, 0.1367, 0.1285, 0.0026, 0.0289, 0.0548, 0.0166,
        0.1644, 0.1167])

max probability is torch.return_types.max(
values=tensor(0.1644),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9095e-03, 2.6984e-05, 2.6981e-05, 2.6960e-05]),
indices=tensor([  0, 771, 914, 910]))
End*********************************************************************

**************************************************************************
Predicting Test Sample 622 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0043, 0.0011, 0.0166, 0.0150, 0.0336,
0.0286, 0.0174, 0.0031, 0.0084,
        0.0209, 0.0529, 0.0075, 0.0625, 0.2924, 0.0478, 0.0243, 0.0439, 0.1068,
        0.1260, 0.0686])

max probability is torch.return_types.max(
values=tensor(0.2924),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.8280e-04, 1.8628e-05, 1.8627e-05, 1.8618e-05]),
indices=tensor([  0, 771, 914, 323]))
End*********************************************************************

**************************************************************************
Predicting Test Sample 623 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0234, 0.0407, 0.0241, 0.0211, 0.0237,
0.0815, 0.0316, 0.0011, 0.0029,
        0.0050, 0.5023, 0.0084, 0.0340, 0.0522, 0.0011, 0.0272, 0.0120, 0.0087,
        0.0158, 0.0773])

max probability is torch.return_types.max(
values=tensor(0.5023),
indices=tensor(10))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.8573e-04, 5.5919e-06, 5.5892e-06, 5.5871e-06]),
indices=tensor([  0, 771, 319, 914]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 624 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.3669e-05, 1.2292e-04, 4.4060e-06,
1.9632e-06, 5.4959e-06, 2.3544e-05,
        8.0792e-06, 1.0000e-04, 2.3121e-05, 9.9607e-06, 8.4036e-03, 9.8134e-01,
        1.9381e-03, 1.8310e-04, 1.9404e-04, 2.4955e-04, 6.2425e-03, 9.2053e-05,
        7.1135e-04, 2.5402e-04])

max probability is torch.return_types.max(
values=tensor(0.9813),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1227e-06, 4.1970e-09, 4.1954e-09, 4.1930e-09]),
indices=tensor([  0, 323, 914, 382]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 625 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.0919e-02, 2.7038e-03, 4.5545e-04,
2.2837e-04, 1.5086e-03, 5.8207e-03,
        1.5693e-03, 2.6621e-03, 1.5843e-02, 8.2376e-03, 1.2571e-01, 7.7127e-02,
        7.0953e-02, 7.1433e-02, 3.3215e-03, 1.0454e-01, 4.2519e-01, 7.1939e-03,
        2.2890e-02, 2.2574e-02])

max probability is torch.return_types.max(
values=tensor(0.4252),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.2168e-04, 9.2754e-06, 9.2752e-06, 9.2723e-06]),
indices=tensor([  0, 771, 323, 910]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 626 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.5325e-04, 5.6909e-04, 1.1225e-01,
```

```
2.2072e-01, 5.1540e-01, 1.2196e-01,
        2.8114e-02, 1.2494e-05, 1.2285e-05, 3.4219e-05, 2.7443e-05, 9.4866e-07,
        2.9082e-05, 6.9047e-05, 5.4496e-05, 5.5951e-06, 2.6496e-06, 4.3105e-05,
        3.2487e-04, 4.3310e-05])

max probability is torch.return_types.max(
values=tensor(0.5154),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1621e-06, 8.8424e-08, 8.8379e-08, 8.8317e-08]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 627 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.1262e-03, 1.8103e-03, 1.9442e-03,
1.7284e-04, 1.0680e-03, 3.5211e-04,
        5.3814e-04, 4.5705e-02, 3.4133e-02, 3.8183e-03, 4.3496e-03, 7.1361e-02,
        4.6380e-02, 1.1573e-02, 8.5229e-04, 5.6677e-03, 7.4154e-01, 1.5673e-03,
        9.9969e-03, 7.2130e-03])

max probability is torch.return_types.max(
values=tensor(0.7415),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8307e-04, 1.7339e-06, 1.7335e-06, 1.7332e-06]),
indices=tensor([  0, 691, 230, 450]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 628 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.9969e-02, 2.4543e-01, 2.2213e-01,
1.6595e-01, 3.6694e-02, 1.4488e-01,
        8.1381e-02, 1.0427e-04, 9.7041e-05, 3.8130e-04, 3.3654e-04, 6.3537e-04,
        1.1172e-03, 1.8083e-04, 2.8201e-04, 8.3629e-06, 5.5535e-05, 1.7187e-04,
        8.7139e-05, 7.7680e-05])

max probability is torch.return_types.max(
values=tensor(0.2454),
indices=tensor(1))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([1.0316e-06, 3.4440e-08, 3.4438e-08, 3.4436e-08]),
indices=tensor([  0, 323, 159, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 629 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.4594e-04, 9.4039e-05, 2.7805e-03,
2.0234e-03, 7.9636e-03, 4.1172e-03,
        3.2422e-03, 1.0647e-03, 1.1116e-02, 7.2842e-03, 6.7126e-03, 1.6127e-05,
        7.6066e-03, 9.0572e-01, 2.0797e-05, 6.6380e-03, 1.7492e-02, 6.4509e-04,
        2.6167e-03, 1.1994e-02])

max probability is torch.return_types.max(
values=tensor(0.9057),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5193e-05, 8.7116e-08, 8.7097e-08, 8.7085e-08]),
indices=tensor([  0, 771, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 630 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0067, 0.0026, 0.0020, 0.0026, 0.0128,
0.1162, 0.0209, 0.0040, 0.0585,
        0.0193, 0.1964, 0.0026, 0.1053, 0.2278, 0.0025, 0.1080, 0.0326, 0.0065,
        0.0349, 0.0336])

max probability is torch.return_types.max(
values=tensor(0.2278),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0014e-04, 4.2600e-06, 4.2592e-06, 4.2550e-06]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 631 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.1487e-03, 3.0352e-04, 1.1792e-03,
1.2584e-03, 6.4664e-03, 6.3750e-03,
        1.6691e-03, 1.6569e-02, 6.7973e-02, 4.8983e-02, 1.9672e-02, 4.6490e-04,
```

```
        1.1426e-02, 4.0303e-02, 4.3697e-03, 2.6552e-02, 7.2190e-03, 2.6346e-02,
        6.6973e-01, 3.8088e-02])

max probability is torch.return_types.max(
values=tensor(0.6697),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9080e-04, 1.8219e-06, 1.8208e-06, 1.8207e-06]),
indices=tensor([  0, 771, 316, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 632 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.8497e-04, 1.6074e-03, 2.9338e-03,
4.7399e-04, 1.3450e-03, 7.8573e-04,
        5.4193e-04, 4.2382e-03, 9.1689e-03, 6.4402e-04, 5.6527e-02, 2.9225e-03,
        8.4953e-01, 1.4716e-02, 1.9768e-03, 3.5085e-02, 1.0629e-02, 1.1857e-03,
        2.6151e-03, 1.9789e-03])

max probability is torch.return_types.max(
values=tensor(0.8495),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8235e-05, 8.4719e-07, 8.4715e-07, 8.4681e-07]),
indices=tensor([  0, 910, 771,  44]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 633 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0035, 0.2229, 0.0032, 0.0015, 0.0092,
0.0055, 0.0036, 0.0673, 0.0479,
        0.0068, 0.0656, 0.0029, 0.0204, 0.0484, 0.0009, 0.1716, 0.0986, 0.0038,
        0.1514, 0.0559])

max probability is torch.return_types.max(
values=tensor(0.2229),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3624e-03, 8.1810e-06, 8.1781e-06, 8.1776e-06]),
indices=tensor([  0, 263, 910, 726]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 634 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.4550e-03, 3.0159e-05, 1.3627e-04,
4.9417e-05, 1.7530e-04, 8.0166e-05,
        5.7017e-05, 1.7795e-04, 1.9074e-04, 5.6364e-04, 7.5343e-03, 3.1518e-01,
        6.5052e-03, 1.0206e-01, 7.1204e-03, 1.3899e-03, 4.4463e-01, 3.8462e-02,
        4.8119e-02, 2.3996e-02])

max probability is torch.return_types.max(
values=tensor(0.4446),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.1344e-05, 5.1845e-08, 5.1840e-08, 5.1830e-08]),
indices=tensor([  0, 691,  82, 230]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 635 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.0407e-03, 3.8318e-04, 1.0869e-03,
1.0494e-03, 6.2323e-03, 2.3013e-03,
        9.4920e-04, 2.7447e-02, 3.7527e-02, 3.1917e-02, 3.4132e-03, 1.1805e-03,
        5.5457e-03, 1.8341e-02, 5.1128e-03, 1.3694e-02, 1.4450e-02, 1.8613e-02,
        7.8529e-01, 2.0129e-02])

max probability is torch.return_types.max(
values=tensor(0.7853),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0301e-04, 2.2028e-06, 2.2025e-06, 2.2023e-06]),
indices=tensor([  0, 263, 914, 197]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 636 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.4388e-04, 1.5170e-04, 3.2098e-01,
3.0289e-01, 3.4916e-01, 2.0846e-02,
        5.6589e-03, 5.9990e-06, 2.9638e-06, 2.0458e-05, 4.9149e-06, 8.3711e-08,
        1.0460e-05, 2.5998e-05, 7.4064e-06, 3.4111e-07, 1.2209e-06, 1.0547e-05,
        4.9681e-05, 2.0846e-05])
```

```
max probability is torch.return_types.max(
values=tensor(0.3492),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.4393e-07, 1.0648e-08, 1.0647e-08, 1.0643e-08]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 637 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.4792e-05, 5.9613e-06, 1.0067e-02,
8.9767e-01, 5.3284e-02, 1.9091e-02,
        1.8522e-02, 5.2392e-07, 2.5594e-07, 2.9616e-05, 2.1863e-06, 3.3731e-08,
        1.5085e-06, 1.1401e-03, 1.9849e-05, 2.1674e-07, 2.6532e-07, 1.5624e-05,
        3.5629e-05, 7.7710e-05])

max probability is torch.return_types.max(
values=tensor(0.8977),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1178e-07, 3.9233e-10, 3.9215e-10, 3.9214e-10]),
indices=tensor([  0, 771, 319, 726]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 638 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.9574e-06, 9.9964e-01, 2.4653e-05,
6.0920e-06, 2.4387e-05, 4.9043e-05,
        3.5980e-05, 9.5884e-06, 7.6243e-06, 1.4532e-07, 5.5571e-05, 7.6965e-07,
        1.1740e-04, 1.2647e-06, 6.2795e-08, 1.3076e-05, 3.9832e-06, 1.1171e-08,
        4.4475e-07, 1.2943e-07])

max probability is torch.return_types.max(
values=tensor(0.9996),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.4823e-10, 4.4513e-12, 4.4419e-12, 4.4403e-12]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************
```

```
**************************************************************************
Predicting Test Sample 639 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0064, 0.0092, 0.0061, 0.0036, 0.0109,
0.0167, 0.0112, 0.0847, 0.2060,
        0.0512, 0.1524, 0.0028, 0.0957, 0.0468, 0.0008, 0.1827, 0.0330, 0.0022,
        0.0258, 0.0228])

max probability is torch.return_types.max(
values=tensor(0.2060),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.0357e-04, 3.0058e-05, 3.0045e-05, 3.0037e-05]),
indices=tensor([  0, 319, 316, 771]))
End*************************************************************************

**************************************************************************
Predicting Test Sample 640 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.5352e-05, 5.1897e-05, 2.3652e-05,
2.0877e-06, 4.0807e-05, 1.0954e-06,
        7.9263e-07, 1.8859e-02, 2.4953e-03, 5.5355e-05, 3.7884e-04, 4.3875e-05,
        1.4726e-04, 4.9364e-04, 2.0410e-08, 3.5723e-05, 2.2510e-01, 6.2475e-06,
        6.7632e-02, 6.8249e-01])

max probability is torch.return_types.max(
values=tensor(0.6825),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1277e-03, 1.2940e-10, 1.2937e-10, 1.2933e-10]),
indices=tensor([  0, 655, 691, 450]))
End*************************************************************************

**************************************************************************
Predicting Test Sample 641 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.6234e-09, 1.2275e-08, 1.1222e-10,
2.3895e-11, 8.6261e-11, 5.9647e-10,
        2.4777e-10, 5.6813e-08, 1.1243e-09, 1.8113e-10, 1.6071e-06, 9.9997e-01,
        1.4584e-05, 1.4424e-08, 2.6579e-07, 1.2483e-09, 1.4829e-05, 5.7795e-09,
        1.3687e-07, 1.4378e-08])

max probability is torch.return_types.max(
```

```
values=tensor(1.0000),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1505e-12, 8.5050e-17, 8.4965e-17, 8.4949e-17]),
indices=tensor([  0, 691,  82, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 642 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.8808e-03, 8.1744e-05, 1.1142e-02,
1.1077e-02, 1.9398e-02, 9.2860e-03,
        4.6988e-03, 2.4558e-02, 5.5979e-02, 6.7503e-01, 2.5648e-03, 4.9859e-05,
        5.2221e-03, 4.1914e-02, 3.8713e-04, 2.7986e-03, 5.0423e-03, 1.8339e-03,
        1.1460e-01, 1.1005e-02])

max probability is torch.return_types.max(
values=tensor(0.6750),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.3270e-05, 4.3673e-07, 4.3671e-07, 4.3659e-07]),
indices=tensor([  0, 771, 896, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 643 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.4837e-02, 1.6963e-01, 2.2732e-03,
1.7340e-03, 2.4870e-03, 7.3872e-01,
        5.1308e-02, 4.0628e-04, 5.7517e-03, 5.5516e-04, 5.3569e-04, 5.5087e-05,
        1.1834e-03, 1.1204e-04, 7.1008e-06, 3.2418e-04, 1.2438e-05, 4.0906e-06,
        3.7605e-05, 1.4233e-05])

max probability is torch.return_types.max(
values=tensor(0.7387),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.7531e-08, 5.5670e-09, 5.5572e-09, 5.5567e-09]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 644 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.1327e-05, 2.4222e-05, 8.4690e-01,
1.0111e-01, 3.5605e-02, 3.5468e-03,
        1.1940e-02, 8.6726e-06, 8.8971e-06, 3.1279e-04, 1.2857e-05, 4.8827e-08,
        6.0753e-05, 2.7798e-04, 7.4521e-06, 2.0683e-06, 7.2304e-06, 3.3457e-05,
        1.3977e-05, 5.0878e-05])

max probability is torch.return_types.max(
values=tensor(0.8469),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9966e-07, 2.6046e-09, 2.6045e-09, 2.6044e-09]),
indices=tensor([  0, 910, 771, 572]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 645 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.6468e-03, 6.3037e-05, 1.4866e-02,
1.8569e-03, 7.6523e-03, 9.5116e-04,
        2.0938e-03, 3.5069e-04, 1.5219e-03, 3.6587e-03, 4.2707e-03, 4.7524e-05,
        3.5434e-03, 8.3235e-01, 1.4547e-04, 1.9748e-03, 4.9996e-02, 1.0509e-02,
        9.0774e-03, 5.3181e-02])

max probability is torch.return_types.max(
values=tensor(0.8323),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1925e-04, 1.3246e-07, 1.3243e-07, 1.3243e-07]),
indices=tensor([  0, 150, 748, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 646 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0067, 0.0041, 0.0029, 0.0020, 0.0096,
0.0347, 0.0127, 0.0090, 0.0818,
        0.0238, 0.2626, 0.0022, 0.1050, 0.1295, 0.0017, 0.2151, 0.0336, 0.0071,
        0.0208, 0.0229])

max probability is torch.return_types.max(
values=tensor(0.2626),
indices=tensor(10))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6370e-04, 1.2770e-05, 1.2762e-05, 1.2753e-05]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 647 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7132e-04, 3.4564e-05, 1.6705e-02,
1.3882e-02, 1.1667e-02, 1.4128e-03,
        2.2982e-03, 1.3892e-04, 3.5161e-04, 2.5270e-03, 1.1379e-02, 3.3631e-05,
        1.0457e-02, 9.0022e-01, 6.5646e-05, 8.3226e-04, 8.6166e-03, 7.3330e-04,
        2.1367e-03, 1.6273e-02])

max probability is torch.return_types.max(
values=tensor(0.9002),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6326e-05, 4.4146e-08, 4.4143e-08, 4.4142e-08]),
indices=tensor([  0, 914, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 648 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.6399e-05, 3.2208e-07, 4.7122e-04,
7.4359e-04, 1.0674e-03, 5.7609e-05,
        1.2029e-04, 6.9252e-06, 4.0875e-05, 6.8610e-04, 1.0184e-04, 4.5712e-07,
        1.7087e-04, 9.9075e-01, 1.4767e-05, 5.0009e-05, 1.9020e-03, 6.2707e-04,
        4.8038e-04, 2.6741e-03])

max probability is torch.return_types.max(
values=tensor(0.9907),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.9334e-07, 6.7475e-10, 6.7451e-10, 6.7448e-10]),
indices=tensor([  0, 323, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 649 :   Prediction is Correct?
No
```

```
first 20 classes probability: tensor([3.5083e-02, 6.9964e-04, 6.0404e-01,
1.5335e-01, 7.2793e-02, 3.1376e-02,
        1.9992e-02, 1.0052e-03, 2.1931e-03, 2.9721e-02, 2.9682e-03, 2.7140e-05,
        6.7233e-03, 3.1470e-02, 1.0980e-04, 4.2474e-04, 1.1384e-03, 7.5742e-04,
        1.1000e-03, 4.1973e-03])

max probability is torch.return_types.max(
values=tensor(0.6040),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.2167e-05, 8.4670e-07, 8.4641e-07, 8.4615e-07]),
indices=tensor([  0, 771, 910, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 650 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0188, 0.0023, 0.0199, 0.0178, 0.0323,
0.1107, 0.0366, 0.0426, 0.1568,
        0.2958, 0.0280, 0.0013, 0.0346, 0.0493, 0.0022, 0.0147, 0.0144, 0.0083,
        0.0506, 0.0371])

max probability is torch.return_types.max(
values=tensor(0.2958),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.9036e-04, 2.6685e-05, 2.6671e-05, 2.6669e-05]),
indices=tensor([  0, 771, 896, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 651 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.4102e-03, 6.3669e-03, 1.0829e-03,
8.7804e-04, 3.7944e-03, 2.0865e-02,
        6.2691e-03, 5.2138e-03, 4.4716e-02, 8.9991e-03, 4.5119e-01, 6.6775e-04,
        3.4334e-02, 6.4032e-02, 1.3165e-04, 3.1257e-01, 1.0927e-02, 9.2525e-04,
        6.7642e-03, 1.3234e-02])

max probability is torch.return_types.max(
values=tensor(0.4512),
indices=tensor(10))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([1.1172e-04, 3.7589e-06, 3.7527e-06, 3.7512e-06]),
indices=tensor([  0, 319, 771, 316]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 652 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.3033e-01, 1.4102e-03, 2.1114e-02,
7.9556e-03, 5.3254e-02, 4.8454e-03,
        5.5060e-03, 5.1319e-03, 1.5506e-02, 4.8876e-03, 9.8956e-03, 2.5596e-04,
        9.8746e-03, 2.4103e-01, 1.4415e-04, 1.8647e-03, 1.0797e-02, 1.9762e-02,
        7.4489e-02, 2.7964e-01])

max probability is torch.return_types.max(
values=tensor(0.2796),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2281e-03, 1.1219e-06, 1.1216e-06, 1.1214e-06]),
indices=tensor([  0, 914, 726, 910]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 653 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.8205e-03, 1.1337e-04, 1.0359e-03,
1.8356e-04, 2.4367e-03, 1.0432e-03,
        4.3742e-04, 5.2806e-04, 7.1435e-04, 3.2373e-04, 1.5301e-03, 1.6046e-02,
        2.4097e-02, 1.3954e-02, 6.1809e-02, 2.7527e-03, 2.5778e-02, 4.3224e-01,
        3.4666e-01, 6.1471e-02])

max probability is torch.return_types.max(
values=tensor(0.4322),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8234e-05, 1.1303e-08, 1.1296e-08, 1.1294e-08]),
indices=tensor([  0, 914, 748, 323]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 654 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0262, 0.0033, 0.0213, 0.0034, 0.0142,
0.0054, 0.0050, 0.0199, 0.0613,
```

```
        0.0110, 0.0657, 0.0093, 0.1919, 0.2060, 0.0020, 0.0164, 0.1538, 0.0294,
        0.0576, 0.0717])

max probability is torch.return_types.max(
values=tensor(0.2060),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7086e-03, 2.4582e-05, 2.4575e-05, 2.4574e-05]),
indices=tensor([  0, 914, 771, 323]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 655 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2421e-04, 5.6555e-05, 2.6398e-01,
6.5060e-01, 4.8468e-02, 2.5366e-02,
        1.1001e-02, 1.9244e-06, 6.2864e-07, 5.0496e-05, 2.3582e-06, 7.7826e-07,
        1.0800e-05, 3.9661e-05, 1.9800e-04, 4.8970e-08, 6.6200e-07, 6.9431e-05,
        1.8368e-05, 1.4801e-05])

max probability is torch.return_types.max(
values=tensor(0.6506),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.9745e-08, 1.3116e-09, 1.3114e-09, 1.3112e-09]),
indices=tensor([  0, 914, 159, 771]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 656 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0043, 0.0016, 0.0060, 0.0007, 0.0044,
0.0009, 0.0009, 0.0431, 0.0653,
        0.0063, 0.0244, 0.0094, 0.2704, 0.0426, 0.0020, 0.0982, 0.3560, 0.0038,
        0.0419, 0.0130])

max probability is torch.return_types.max(
values=tensor(0.3560),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2298e-04, 5.0535e-06, 5.0507e-06, 5.0506e-06]),
indices=tensor([  0, 910, 323, 316]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 657 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3774e-04, 1.6443e-04, 1.5718e-01,
4.5535e-01, 3.1785e-01, 5.8165e-02,
        1.1055e-02, 2.3474e-06, 1.5994e-06, 1.4773e-05, 2.6123e-06, 2.7189e-08,
        3.7679e-06, 2.4432e-05, 4.2536e-06, 3.0548e-07, 2.3488e-07, 3.5553e-06,
        2.4837e-05, 1.0215e-05])

max probability is torch.return_types.max(
values=tensor(0.4553),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3079e-07, 4.3501e-09, 4.3469e-09, 4.3426e-09]),
indices=tensor([  0, 319, 316,  85]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 658 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.5316e-03, 1.5363e-04, 1.8172e-04,
1.1078e-04, 1.8746e-04, 2.3491e-04,
        2.6951e-04, 4.3448e-03, 1.1879e-03, 2.3499e-03, 3.0717e-03, 9.2639e-01,
        4.4308e-03, 3.0040e-03, 2.1100e-03, 1.8601e-04, 2.6820e-02, 2.2267e-03,
        1.4423e-02, 3.1702e-03])

max probability is torch.return_types.max(
values=tensor(0.9264),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.0257e-05, 5.9645e-07, 5.9603e-07, 5.9597e-07]),
indices=tensor([  0, 691, 447, 382]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 659 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.3223e-02, 2.1554e-04, 1.3527e-03,
2.5130e-04, 2.0564e-03, 1.7168e-04,
        7.5529e-05, 1.4034e-01, 8.1640e-02, 4.3897e-02, 2.6431e-03, 5.1963e-04,
        8.5573e-03, 7.5383e-03, 5.4637e-04, 1.0349e-02, 4.7327e-02, 2.4206e-03,
        6.1754e-01, 9.2278e-03])
```

```
max probability is torch.return_types.max(
values=tensor(0.6175),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8012e-05, 9.1921e-08, 9.1921e-08, 9.1917e-08]),
indices=tensor([  0, 316, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 660 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0275e-03, 9.1056e-01, 1.9938e-03,
7.7017e-04, 1.8256e-03, 8.2710e-03,
        5.8560e-03, 2.0731e-03, 3.5722e-03, 1.2930e-04, 1.0612e-02, 5.1649e-04,
        4.7411e-02, 1.4964e-03, 9.3455e-05, 2.3952e-03, 9.0939e-04, 2.5417e-05,
        1.4576e-04, 1.2725e-04])

max probability is torch.return_types.max(
values=tensor(0.9106),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.6448e-06, 1.8998e-07, 1.8972e-07, 1.8968e-07]),
indices=tensor([  0, 319, 910,  44]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 661 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0965e-03, 6.2290e-04, 1.0395e-01,
1.8165e-01, 4.6486e-01, 4.0051e-02,
        3.5315e-02, 4.2778e-03, 5.8961e-03, 1.2723e-02, 1.1984e-03, 1.3576e-05,
        9.0387e-04, 4.6524e-02, 3.4156e-04, 5.4736e-04, 1.0996e-03, 3.7324e-03,
        6.7235e-02, 2.5049e-02])

max probability is torch.return_types.max(
values=tensor(0.4649),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.6664e-04, 2.4617e-06, 2.4609e-06, 2.4608e-06]),
indices=tensor([  0,  91, 771, 914]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 662 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.4440e-04, 4.7475e-03, 1.6039e-04,
1.8334e-05, 1.3916e-04, 7.7904e-04,
        6.9941e-04, 2.9428e-02, 1.1689e-01, 7.0392e-05, 1.1343e-02, 1.2686e-03,
        8.2975e-01, 9.2142e-04, 2.8280e-05, 2.3365e-03, 7.7505e-04, 7.7903e-06,
        4.2951e-05, 4.1968e-05])

max probability is torch.return_types.max(
values=tensor(0.8297),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.9242e-07, 1.7883e-08, 1.7875e-08, 1.7866e-08]),
indices=tensor([  0, 836, 319,  44]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 663 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0242, 0.0010, 0.0043, 0.0067, 0.0206,
0.0380, 0.0264, 0.0144, 0.0789,
        0.2792, 0.0346, 0.0037, 0.0326, 0.2124, 0.0030, 0.0198, 0.0693, 0.0113,
        0.0536, 0.0415])

max probability is torch.return_types.max(
values=tensor(0.2792),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.6290e-04, 2.4933e-05, 2.4922e-05, 2.4919e-05]),
indices=tensor([  0, 771, 914, 323]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 664 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.2131e-06, 1.3470e-06, 3.4257e-03,
9.7081e-01, 1.3739e-02, 7.5763e-03,
        4.3501e-03, 2.3903e-08, 8.0936e-09, 3.4227e-06, 1.0179e-07, 7.8278e-10,
        5.9287e-08, 7.7918e-05, 1.2871e-06, 4.7946e-09, 6.8947e-09, 5.4793e-07,
        1.7733e-06, 4.0694e-06])

max probability is torch.return_types.max(
```

```
values=tensor(0.9708),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4420e-09, 4.7062e-12, 4.7033e-12, 4.7025e-12]),
indices=tensor([  0, 771, 319, 914]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 665 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0213, 0.0023, 0.0043, 0.0074, 0.0337,
0.0630, 0.0313, 0.0041, 0.0190,
        0.0303, 0.0679, 0.0212, 0.0988, 0.2302, 0.0286, 0.0657, 0.0756, 0.0295,
        0.1084, 0.0378])

max probability is torch.return_types.max(
values=tensor(0.2302),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.4810e-04, 2.0108e-05, 2.0100e-05, 2.0099e-05]),
indices=tensor([  0, 771, 910, 323]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 666 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.6359e-01, 3.7936e-03, 5.9511e-03,
4.7394e-03, 1.4624e-03, 1.3796e-02,
        4.5701e-03, 8.0447e-05, 1.1300e-04, 1.1428e-03, 2.9009e-05, 2.0478e-04,
        1.8433e-04, 6.4157e-05, 3.7633e-05, 4.0107e-06, 4.4906e-05, 5.6575e-05,
        3.7912e-05, 8.9525e-05])

max probability is torch.return_types.max(
values=tensor(0.9636),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.3704e-07, 7.2664e-09, 7.2648e-09, 7.2630e-09]),
indices=tensor([  0, 271, 323, 914]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 667 :   Prediction is Correct?
```

```
Yes
first 20 classes probability: tensor([3.3456e-05, 1.9779e-04, 2.7579e-01,
2.7932e-01, 4.2104e-01, 1.9440e-02,
        4.1447e-03, 1.4784e-06, 7.6776e-07, 3.1018e-06, 1.0335e-06, 6.4871e-09,
        2.0879e-06, 4.0274e-06, 9.2582e-07, 7.0154e-08, 1.6956e-07, 1.2336e-06,
        1.1138e-05, 2.9284e-06])

max probability is torch.return_types.max(
values=tensor(0.4210),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0167e-07, 9.4517e-10, 9.4465e-10, 9.4393e-10]),
indices=tensor([  0, 319, 316, 720]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 668 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.6570e-04, 1.5597e-03, 2.2557e-01,
2.9313e-01, 3.5326e-01, 8.8173e-02,
        3.4799e-02, 7.7937e-05, 8.0134e-05, 2.8352e-04, 1.1271e-04, 4.8996e-06,
        1.7103e-04, 3.1452e-04, 1.3136e-04, 2.6244e-05, 1.7664e-05, 1.2619e-04,
        4.6306e-04, 1.4071e-04])

max probability is torch.return_types.max(
values=tensor(0.3533),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.8542e-06, 7.1922e-07, 7.1899e-07, 7.1846e-07]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 669 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7816e-01, 3.8695e-03, 4.5053e-02,
7.0017e-03, 9.5541e-03, 1.3025e-02,
        7.4840e-03, 1.6317e-01, 4.3411e-01, 1.0395e-01, 7.4944e-04, 2.2297e-05,
        2.8348e-02, 2.6766e-03, 6.7369e-05, 6.0649e-04, 8.1255e-04, 1.2643e-04,
        4.8211e-04, 5.8381e-04])

max probability is torch.return_types.max(
values=tensor(0.4341),
indices=tensor(8))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6886e-06, 1.4802e-07, 1.4802e-07, 1.4800e-07]),
indices=tensor([  0,  85, 319, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 670 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.5054e-03, 3.1668e-05, 7.4233e-03,
9.3682e-02, 5.6164e-02, 2.5120e-02,
        4.5856e-02, 3.9165e-04, 9.2456e-04, 2.8939e-02, 3.8591e-04, 1.3522e-05,
        4.2306e-04, 6.1211e-01, 2.7771e-02, 1.7592e-04, 3.3737e-04, 7.0661e-02,
        1.3946e-02, 1.3022e-02])

max probability is torch.return_types.max(
values=tensor(0.6121),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2159e-05, 1.0839e-07, 1.0838e-07, 1.0831e-07]),
indices=tensor([  0, 771, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 671 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.2619e-04, 1.6071e-03, 1.1758e-01,
2.4818e-01, 4.1419e-01, 1.5310e-01,
        6.1896e-02, 5.8267e-05, 6.4520e-05, 1.9823e-04, 8.9025e-05, 4.6003e-06,
        1.2482e-04, 3.1041e-04, 2.1927e-04, 3.9096e-05, 1.3890e-05, 1.3380e-04,
        7.0929e-04, 1.2943e-04])

max probability is torch.return_types.max(
values=tensor(0.4142),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.5724e-06, 6.5351e-07, 6.5323e-07, 6.5274e-07]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 672 :   Prediction is Correct?
Yes
```

```
first 20 classes probability: tensor([8.9296e-07, 1.9673e-07, 1.3172e-08,
2.8442e-09, 9.9250e-09, 1.3808e-08,
        1.2344e-08, 5.6629e-06, 1.7209e-07, 9.6777e-08, 2.1824e-05, 9.9919e-01,
        2.3259e-05, 6.2358e-07, 2.8279e-06, 3.1766e-07, 7.3041e-04, 9.2909e-07,
        1.7194e-05, 2.1475e-06])

max probability is torch.return_types.max(
values=tensor(0.9992),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2833e-09, 3.7369e-13, 3.7336e-13, 3.7331e-13]),
indices=tensor([  0, 691, 382, 230]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 673 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.7923e-05, 4.0873e-05, 4.9478e-05,
8.1698e-05, 1.5105e-04, 7.2107e-05,
        6.6076e-05, 4.1697e-05, 9.1900e-05, 1.2269e-04, 2.4376e-02, 2.3456e-05,
        2.1001e-04, 2.3237e-02, 2.5976e-07, 6.1719e-04, 6.1821e-02, 6.2083e-05,
        3.9560e-03, 8.8314e-01])

max probability is torch.return_types.max(
values=tensor(0.8831),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8160e-03, 3.8774e-09, 3.8756e-09, 3.8746e-09]),
indices=tensor([  0, 914, 771, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 674 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0283, 0.2389, 0.0037, 0.0021, 0.0056,
0.0020, 0.0082, 0.2800, 0.1091,
        0.0020, 0.0043, 0.0058, 0.2294, 0.0027, 0.0654, 0.0022, 0.0018, 0.0027,
        0.0012, 0.0004])

max probability is torch.return_types.max(
values=tensor(0.2800),
indices=tensor(7))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([1.8822e-05, 4.3087e-06, 4.3086e-06, 4.3080e-06]),
indices=tensor([  0, 316, 323, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 675 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.5189e-03, 6.4860e-04, 2.9120e-02,
1.2179e-01, 2.2732e-01, 4.2451e-01,
        1.6872e-01, 2.1571e-04, 5.4112e-04, 2.8850e-03, 4.0730e-04, 5.9369e-05,
        4.3659e-04, 4.0515e-03, 3.3105e-03, 2.6201e-04, 7.0892e-05, 2.4023e-03,
        5.1293e-03, 1.1363e-03])

max probability is torch.return_types.max(
values=tensor(0.4245),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8079e-05, 1.5232e-06, 1.5229e-06, 1.5221e-06]),
indices=tensor([  0, 319, 316, 144]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 676 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.1224e-03, 1.6038e-03, 2.2187e-04,
3.3632e-05, 3.5090e-04, 3.8180e-04,
        2.5425e-04, 1.3854e-02, 2.8964e-02, 1.4942e-03, 9.8292e-03, 3.9939e-02,
        3.6243e-02, 1.2367e-02, 1.8083e-04, 1.5820e-02, 8.0925e-01, 8.9151e-04,
        1.0514e-02, 1.2591e-02])

max probability is torch.return_types.max(
values=tensor(0.8093),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3719e-04, 8.9940e-07, 8.9930e-07, 8.9915e-07]),
indices=tensor([  0, 910, 323, 197]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 677 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.3257, 0.0014, 0.0537, 0.0171, 0.0193,
0.0207, 0.0246, 0.0224, 0.0508,
```

```
        0.3975, 0.0031, 0.0009, 0.0224, 0.0091, 0.0032, 0.0016, 0.0028, 0.0073,
        0.0062, 0.0027])

max probability is torch.return_types.max(
values=tensor(0.3975),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.5122e-05, 7.7564e-06, 7.7544e-06, 7.7543e-06]),
indices=tensor([  0, 771, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 678 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0130, 0.0022, 0.0039, 0.0015, 0.0102,
0.0100, 0.0061, 0.0499, 0.1329,
        0.0388, 0.0164, 0.0080, 0.0275, 0.1484, 0.0016, 0.0220, 0.3379, 0.0119,
        0.1132, 0.0363])

max probability is torch.return_types.max(
values=tensor(0.3379),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.0112e-04, 7.7134e-06, 7.7129e-06, 7.7123e-06]),
indices=tensor([  0, 323, 910, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 679 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.9347e-03, 2.1158e-03, 1.9153e-03,
1.1549e-03, 3.0145e-03, 2.4798e-03,
        1.3208e-03, 5.0300e-03, 9.6735e-03, 5.7700e-03, 3.6217e-02, 1.8067e-03,
        7.0179e-03, 6.6106e-02, 8.7680e-05, 8.6678e-03, 2.7492e-01, 2.9138e-03,
        5.6525e-02, 4.8890e-01])

max probability is torch.return_types.max(
values=tensor(0.4889),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4125e-02, 7.6512e-06, 7.6475e-06, 7.6449e-06]),
indices=tensor([  0, 914, 771, 655]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 680 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.7209e-06, 1.7203e-07, 1.4985e-07,
7.8364e-07, 1.7350e-06, 9.2714e-07,
        1.0675e-06, 4.0765e-06, 5.3436e-07, 4.3207e-06, 3.0005e-07, 6.6637e-05,
        1.6832e-05, 2.2181e-06, 9.9784e-01, 1.2054e-07, 9.1740e-07, 1.8449e-03,
        2.0807e-04, 1.8575e-06])

max probability is torch.return_types.max(
values=tensor(0.9978),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1115e-09, 4.7219e-11, 4.7204e-11, 4.7192e-11]),
indices=tensor([  0, 748, 337, 323]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 681 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0095, 0.0013, 0.0065, 0.0063, 0.0221,
0.1622, 0.0361, 0.0053, 0.0579,
        0.0623, 0.1149, 0.0035, 0.0437, 0.2572, 0.0015, 0.0328, 0.0494, 0.0105,
        0.0526, 0.0573])

max probability is torch.return_types.max(
values=tensor(0.2572),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.3576e-04, 7.1197e-06, 7.1157e-06, 7.1108e-06]),
indices=tensor([  0, 771, 914, 896]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 682 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.8322e-05, 1.0618e-05, 5.2689e-05,
2.8987e-05, 1.7888e-04, 2.3587e-05,
        1.1195e-05, 4.2234e-05, 9.5854e-05, 4.4504e-05, 3.9055e-03, 1.3048e-05,
        1.2409e-04, 2.6198e-02, 6.2566e-07, 1.7733e-04, 1.1221e-02, 1.1912e-03,
        1.7485e-02, 9.3780e-01])
```

```
max probability is torch.return_types.max(
values=tensor(0.9378),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3204e-03, 2.7839e-09, 2.7839e-09, 2.7838e-09]),
indices=tensor([  0, 337, 914, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 683 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.7246e-03, 2.1733e-04, 2.0963e-02,
5.2929e-02, 5.2466e-02, 1.2630e-02,
        1.4456e-02, 4.4258e-03, 6.8362e-03, 1.6802e-01, 1.8533e-02, 7.9603e-04,
        1.2262e-02, 3.2598e-01, 6.3264e-03, 5.6550e-03, 1.9005e-02, 3.6351e-02,
        1.4031e-01, 8.9850e-02])

max probability is torch.return_types.max(
values=tensor(0.3260),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.8821e-04, 7.9289e-06, 7.9279e-06, 7.9263e-06]),
indices=tensor([  0, 914, 771, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 684 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.0854e-10, 7.1721e-10, 7.9841e-13,
1.0260e-13, 6.9892e-13, 5.0743e-12,
        3.2606e-12, 9.5072e-09, 6.7993e-11, 3.7136e-12, 3.5807e-08, 9.9999e-01,
        6.3421e-07, 3.1097e-10, 1.4178e-08, 2.0603e-11, 9.5433e-06, 2.4795e-10,
        1.1550e-08, 1.3804e-09])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7281e-14, 1.5079e-19, 1.5057e-19, 1.5054e-19]),
indices=tensor([  0, 691, 382,  82]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 685 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.8448e-03, 2.0166e-03, 1.3463e-02,
4.4586e-02, 9.6023e-02, 4.1973e-01,
        2.8275e-01, 2.2599e-03, 1.0267e-02, 3.3072e-02, 5.7977e-03, 5.6919e-04,
        2.8833e-03, 4.5035e-02, 2.8745e-04, 2.3402e-03, 5.2394e-03, 1.0019e-03,
        1.1869e-02, 1.1842e-02])

max probability is torch.return_types.max(
values=tensor(0.4197),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0398e-04, 3.0658e-06, 3.0655e-06, 3.0648e-06]),
indices=tensor([  0, 914, 771, 896]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 686 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.8515e-03, 1.6005e-04, 1.6616e-04,
2.4675e-04, 5.4517e-04, 7.5123e-04,
        5.0236e-04, 6.8564e-04, 5.4919e-04, 1.9268e-03, 1.3989e-03, 6.4029e-02,
        5.6486e-03, 1.0810e-02, 7.5765e-01, 1.7721e-03, 2.1808e-02, 5.5849e-02,
        6.4892e-02, 6.0281e-03])

max probability is torch.return_types.max(
values=tensor(0.7576),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.8958e-05, 2.7891e-06, 2.7890e-06, 2.7887e-06]),
indices=tensor([  0, 748, 691, 230]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 687 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.9783e-04, 2.7140e-03, 5.8587e-03,
7.9510e-04, 3.9114e-03, 2.8714e-03,
        4.0171e-03, 2.9334e-01, 6.3489e-01, 4.1299e-03, 1.2127e-03, 9.0475e-05,
        3.6812e-02, 2.7744e-03, 3.4257e-05, 7.0186e-04, 3.0436e-03, 9.1461e-05,
        8.0092e-04, 8.7232e-04])

max probability is torch.return_types.max(
```

```
values=tensor(0.6349),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3074e-05, 1.3575e-07, 1.3575e-07, 1.3573e-07]),
indices=tensor([  0, 910,  85, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 688 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1381e-03, 9.4075e-01, 3.2140e-03,
8.7424e-04, 1.5976e-03, 8.4001e-03,
        1.7282e-02, 1.0262e-03, 3.1283e-03, 9.2500e-05, 1.0665e-03, 8.7135e-05,
        2.0573e-02, 2.8458e-04, 2.5337e-05, 1.9690e-04, 2.1940e-04, 5.5218e-06,
        1.0594e-05, 1.4637e-05])

max probability is torch.return_types.max(
values=tensor(0.9407),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.8411e-07, 1.8784e-08, 1.8761e-08, 1.8755e-08]),
indices=tensor([  0, 319, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 689 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.8083e-05, 9.9743e-01, 8.8455e-06,
3.7216e-06, 3.7368e-05, 5.0059e-05,
        3.8584e-05, 1.5560e-03, 3.6117e-04, 1.5054e-06, 2.3258e-05, 3.6232e-07,
        2.7998e-04, 1.2881e-06, 1.6799e-06, 1.8013e-04, 5.5464e-06, 6.9389e-08,
        2.5195e-06, 4.4966e-07])

max probability is torch.return_types.max(
values=tensor(0.9974),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0985e-09, 1.8690e-11, 1.8679e-11, 1.8632e-11]),
indices=tensor([  0, 319, 316, 836]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 690 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.2960e-05, 5.7912e-05, 4.8100e-01,
2.9616e-01, 2.1361e-01, 6.2790e-03,
        2.7347e-03, 2.9666e-06, 1.5781e-06, 2.1040e-05, 2.7634e-06, 1.4692e-08,
        1.0746e-05, 1.9574e-05, 4.0116e-06, 2.0208e-07, 6.7637e-07, 7.5314e-06,
        1.3146e-05, 9.2748e-06])

max probability is torch.return_types.max(
values=tensor(0.4810),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7054e-07, 1.7504e-09, 1.7502e-09, 1.7501e-09]),
indices=tensor([  0, 771, 910, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 691 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0555, 0.0902, 0.0665, 0.0072, 0.0185,
0.0143, 0.0104, 0.2463, 0.3002,
        0.0054, 0.0058, 0.0010, 0.1485, 0.0139, 0.0008, 0.0037, 0.0051, 0.0009,
        0.0021, 0.0018])

max probability is torch.return_types.max(
values=tensor(0.3002),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.0436e-05, 1.9950e-06, 1.9944e-06, 1.9938e-06]),
indices=tensor([  0, 910, 323, 726]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 692 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.2369e-04, 3.4349e-03, 5.5829e-05,
2.4623e-05, 7.2698e-05, 5.8110e-04,
        2.6005e-04, 2.0634e-04, 7.0226e-04, 1.4489e-04, 9.3710e-01, 2.6015e-03,
        3.8666e-02, 7.8125e-04, 4.4718e-06, 1.2951e-02, 1.9598e-03, 7.4415e-06,
        1.1929e-04, 1.9383e-04])

max probability is torch.return_types.max(
values=tensor(0.9371),
indices=tensor(10))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.0014e-07, 9.8742e-09, 9.8574e-09, 9.8477e-09]),
indices=tensor([  0, 319, 771,  44]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 693 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.9757e-04, 8.0338e-01, 3.0961e-03,
5.5263e-03, 1.0744e-02, 1.3847e-01,
        3.4244e-02, 1.2703e-04, 2.6724e-04, 2.2531e-05, 4.1737e-04, 1.5948e-05,
        7.7527e-04, 1.1866e-04, 4.3078e-05, 1.6556e-03, 1.2804e-05, 7.6941e-06,
        5.0152e-05, 1.3106e-05])

max probability is torch.return_types.max(
values=tensor(0.8034),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8320e-07, 1.2883e-08, 1.2859e-08, 1.2847e-08]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 694 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2818e-04, 1.0388e-03, 5.0190e-01,
4.5897e-01, 2.4038e-02, 1.3073e-02,
        8.4637e-04, 1.3456e-08, 9.5415e-09, 3.1586e-07, 1.8242e-08, 1.8973e-10,
        2.0746e-07, 4.5292e-08, 4.1586e-09, 2.3271e-10, 6.0787e-10, 2.7621e-09,
        1.0816e-08, 9.7609e-09])

max probability is torch.return_types.max(
values=tensor(0.5019),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8614e-11, 5.7364e-14, 5.7272e-14, 5.7271e-14]),
indices=tensor([  0, 319, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 695 :   Prediction is Correct?
No
```

```
first 20 classes probability: tensor([6.7279e-04, 1.0406e-03, 1.1583e-02,
8.2386e-02, 2.7471e-02, 7.5624e-01,
        1.2020e-01, 4.9148e-06, 1.3553e-05, 8.9062e-05, 1.6578e-05, 5.8064e-06,
        2.3523e-05, 7.5621e-05, 9.2449e-05, 5.6154e-06, 7.9621e-07, 1.9771e-05,
        4.8185e-05, 1.0852e-05])

max probability is torch.return_types.max(
values=tensor(0.7562),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.8477e-08, 3.1768e-09, 3.1759e-09, 3.1741e-09]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 696 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.5664e-05, 8.3407e-07, 8.1560e-07,
7.1315e-07, 9.1480e-06, 1.6229e-06,
        1.1104e-06, 7.0174e-06, 1.5539e-05, 1.8225e-05, 3.1901e-04, 2.3149e-04,
        7.4257e-05, 1.4709e-03, 4.1099e-02, 3.0985e-04, 3.9226e-04, 8.9723e-01,
        5.7136e-02, 1.6225e-03])

max probability is torch.return_types.max(
values=tensor(0.8972),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.1718e-07, 2.1173e-10, 2.1172e-10, 2.1159e-10]),
indices=tensor([  0, 337, 748, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 697 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.5589e-03, 9.4607e-05, 1.0183e-03,
2.2143e-04, 1.8392e-03, 1.1624e-04,
        7.6865e-05, 3.4137e-02, 2.2773e-02, 2.4474e-02, 1.5687e-03, 4.3506e-04,
        3.8636e-03, 1.2254e-02, 4.6408e-04, 7.0059e-03, 9.2514e-02, 6.7806e-03,
        7.4252e-01, 4.1914e-02])

max probability is torch.return_types.max(
values=tensor(0.7425),
indices=tensor(18))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9133e-04, 1.8836e-07, 1.8829e-07, 1.8826e-07]),
indices=tensor([  0, 197, 323, 914]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 698 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0913e-03, 9.8431e-01, 2.3644e-03,
4.8237e-04, 1.1667e-03, 1.9082e-03,
        1.7917e-03, 1.0368e-03, 1.0589e-03, 3.0226e-05, 2.0644e-04, 2.9867e-05,
        2.9360e-03, 1.1681e-04, 3.8685e-05, 1.1321e-03, 2.3140e-04, 7.8215e-06,
        3.1143e-05, 2.1556e-05])

max probability is torch.return_types.max(
values=tensor(0.9843),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.7545e-07, 1.3337e-08, 1.3331e-08, 1.3320e-08]),
indices=tensor([  0, 319, 316, 910]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 699 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.0163e-03, 6.8424e-05, 4.9083e-04,
7.2563e-04, 1.5924e-03, 1.1290e-03,
        6.9949e-04, 1.0994e-03, 1.2491e-03, 5.1840e-03, 2.2480e-03, 6.8065e-03,
        2.0292e-03, 1.6891e-02, 3.5126e-01, 2.4169e-03, 5.1122e-03, 3.0810e-01,
        2.7941e-01, 1.1188e-02])

max probability is torch.return_types.max(
values=tensor(0.3513),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.1897e-05, 1.2837e-06, 1.2835e-06, 1.2835e-06]),
indices=tensor([  0, 337, 748, 914]))
End*********************************************************************

currently at 700 current time is 21.863395929336548
*********************************************************************
Predicting Test Sample 700 :   Prediction is Correct?
Yes
```

```
first 20 classes probability: tensor([1.3990e-03, 8.1754e-04, 9.5920e-05,
4.8097e-05, 1.6295e-04, 1.8435e-04,
        8.7626e-05, 3.2992e-04, 4.1965e-04, 2.6793e-04, 2.7800e-01, 4.3545e-01,
        2.0700e-02, 1.7974e-02, 6.1682e-04, 1.5479e-02, 1.8413e-01, 5.3291e-03,
        1.6284e-02, 2.1744e-02])

max probability is torch.return_types.max(
values=tensor(0.4354),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3783e-04, 3.5974e-07, 3.5973e-07, 3.5946e-07]),
indices=tensor([  0, 914, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 701 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.1347e-03, 6.1641e-04, 7.5832e-04,
8.5477e-04, 4.2917e-03, 3.7564e-03,
        3.7124e-03, 3.2230e-03, 9.6400e-03, 1.1618e-02, 3.7466e-02, 8.4266e-03,
        6.5974e-03, 1.5916e-01, 3.1207e-04, 2.8410e-02, 4.3120e-01, 4.8886e-03,
        1.1599e-01, 1.6279e-01])

max probability is torch.return_types.max(
values=tensor(0.4312),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3350e-03, 2.9690e-06, 2.9673e-06, 2.9669e-06]),
indices=tensor([  0, 914, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 702 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.1219e-02, 2.1513e-04, 2.3825e-03,
5.8830e-04, 4.2311e-03, 5.8053e-04,
        6.1724e-04, 9.1373e-04, 1.1152e-03, 1.1068e-03, 3.8600e-03, 1.7779e-03,
        8.8048e-04, 2.4853e-02, 1.5412e-03, 1.8585e-03, 4.6583e-02, 2.1822e-01,
        4.0438e-01, 2.7139e-01])

max probability is torch.return_types.max(
values=tensor(0.4044),
indices=tensor(18))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3520e-03, 3.5011e-07, 3.5002e-07, 3.4995e-07]),
indices=tensor([  0, 158,  19, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 703 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4472e-06, 1.8648e-08, 3.0385e-05,
1.2168e-05, 3.5573e-05, 2.7877e-06,
        5.5662e-06, 1.4734e-02, 1.2816e-02, 9.6861e-01, 2.0894e-06, 4.9714e-09,
        1.6157e-05, 8.0182e-05, 1.1808e-07, 7.3013e-06, 4.8738e-05, 4.3236e-07,
        3.5803e-03, 1.6648e-05])

max probability is torch.return_types.max(
values=tensor(0.9686),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3598e-09, 3.4604e-13, 3.4541e-13, 3.4541e-13]),
indices=tensor([  0, 896, 771,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 704 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.3988e-05, 2.5914e-05, 5.3602e-06,
2.1676e-06, 4.1477e-06, 1.4919e-05,
        1.0977e-05, 8.6093e-05, 1.3862e-05, 9.0917e-06, 4.6350e-04, 9.9361e-01,
        2.1681e-03, 6.8620e-05, 3.2257e-04, 7.4209e-06, 2.8141e-03, 4.4799e-05,
        1.9016e-04, 5.0329e-05])

max probability is torch.return_types.max(
values=tensor(0.9936),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1578e-07, 1.2277e-09, 1.2270e-09, 1.2268e-09]),
indices=tensor([  0, 691,  82, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 705 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.6174e-01, 4.7364e-02, 8.5692e-03,
```

```
1.6567e-03, 4.5528e-03, 1.2828e-02,
        1.6651e-02, 1.7512e-01, 5.3214e-01, 5.9769e-03, 9.1737e-04, 1.1995e-04,
        3.0518e-02, 4.4984e-04, 1.7107e-04, 5.9523e-04, 1.2775e-04, 8.8870e-05,
        1.3098e-04, 7.9402e-05])

max probability is torch.return_types.max(
values=tensor(0.5321),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5606e-06, 2.0995e-07, 2.0981e-07, 2.0975e-07]),
indices=tensor([  0, 319, 836, 794]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 706 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.2187e-05, 2.8981e-06, 2.5984e-07,
4.9229e-08, 1.5750e-07, 5.8588e-07,
        6.5518e-07, 2.7633e-05, 1.8711e-06, 6.0070e-07, 2.3242e-05, 9.9684e-01,
        1.5892e-04, 3.6110e-06, 9.3064e-05, 3.4471e-07, 2.7357e-03, 1.1061e-05,
        5.0185e-05, 1.4232e-05])

max probability is torch.return_types.max(
values=tensor(0.9968),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9947e-08, 1.7316e-11, 1.7289e-11, 1.7288e-11]),
indices=tensor([  0, 691, 382, 450]))
End*********************************************************************************

*********************************************************************************
Predicting Test Sample 707 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.6076e-03, 6.5377e-03, 8.8629e-02,
1.1240e-01, 7.5265e-02, 3.0510e-01,
        3.9816e-01, 8.5857e-05, 1.9939e-04, 4.5196e-04, 1.3445e-04, 5.4016e-05,
        5.2722e-04, 7.4814e-04, 8.0329e-04, 3.7539e-05, 3.0430e-05, 7.4017e-04,
        2.0642e-04, 2.1001e-04])

max probability is torch.return_types.max(
values=tensor(0.3982),
indices=tensor(6))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([1.8290e-06, 6.7394e-08, 6.7384e-08, 6.7370e-08]),
indices=tensor([  0, 323, 910, 726]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 708 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.1921e-04, 1.5161e-04, 5.7036e-04,
7.0947e-04, 1.0861e-03, 1.2095e-03,
        5.4221e-04, 6.8466e-05, 4.0076e-04, 1.3045e-03, 5.0881e-01, 5.8153e-04,
        6.9142e-03, 4.0159e-01, 3.1383e-05, 1.1528e-02, 1.7850e-02, 1.6775e-03,
        3.6812e-03, 4.0257e-02])

max probability is torch.return_types.max(
values=tensor(0.5088),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.1625e-05, 7.1393e-08, 7.1317e-08, 7.1316e-08]),
indices=tensor([  0, 771, 323, 914]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 709 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3047e-04, 6.6126e-05, 3.6212e-04,
1.7488e-04, 1.6642e-03, 5.1375e-05,
        4.0861e-05, 5.3859e-03, 4.1491e-03, 4.4599e-04, 9.8476e-04, 1.3734e-05,
        6.9199e-04, 9.1734e-03, 3.0838e-05, 2.9812e-04, 2.9044e-03, 2.0828e-03,
        8.6549e-01, 1.0523e-01])

max probability is torch.return_types.max(
values=tensor(0.8655),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.9935e-04, 2.6480e-08, 2.6476e-08, 2.6473e-08]),
indices=tensor([  0, 197, 914, 337]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 710 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.5640e-03, 3.1183e-04, 4.3584e-01,
9.1450e-02, 5.4796e-02, 2.1309e-02,
```

```
        7.4606e-02, 4.1049e-03, 7.0281e-03, 2.9180e-01, 9.6653e-04, 2.5794e-05,
        3.6261e-03, 5.4976e-03, 2.7026e-04, 2.5601e-04, 1.1712e-03, 1.2158e-03,
        1.4798e-03, 1.0676e-03])

max probability is torch.return_types.max(
values=tensor(0.4358),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4200e-05, 6.2918e-07, 6.2907e-07, 6.2897e-07]),
indices=tensor([  0,  91, 771, 572]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 711 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.2786e-06, 5.0549e-08, 3.3420e-08,
1.3709e-07, 3.1301e-07, 1.4155e-07,
        1.6716e-07, 5.5344e-06, 5.3358e-07, 4.5135e-06, 8.7471e-08, 7.1040e-05,
        1.2316e-05, 1.6863e-06, 9.9758e-01, 5.5533e-08, 7.2369e-07, 2.1317e-03,
        1.8232e-04, 1.1348e-06])

max probability is torch.return_types.max(
values=tensor(0.9976),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6818e-10, 3.0885e-12, 3.0871e-12, 3.0871e-12]),
indices=tensor([  0, 748, 428, 323]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 712 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.2861e-04, 6.1879e-06, 1.7274e-04,
1.6021e-04, 9.3102e-04, 1.0237e-04,
        2.2166e-04, 4.0876e-04, 9.1248e-04, 1.6852e-03, 4.7069e-04, 2.1652e-04,
        2.9336e-04, 1.2998e-01, 2.7379e-03, 2.2464e-04, 1.0458e-02, 4.2591e-01,
        2.8066e-01, 1.4400e-01])

max probability is torch.return_types.max(
values=tensor(0.4259),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([1.8673e-04, 2.1733e-08, 2.1732e-08, 2.1729e-08]),
indices=tensor([  0, 655, 158,  19]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 713 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0226, 0.0029, 0.0621, 0.0132, 0.0320,
0.1561, 0.0781, 0.0254, 0.2474,
        0.1510, 0.0550, 0.0003, 0.0642, 0.0338, 0.0003, 0.0331, 0.0054, 0.0016,
        0.0058, 0.0062])

max probability is torch.return_types.max(
values=tensor(0.2474),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.6127e-05, 3.6175e-06, 3.6165e-06, 3.6159e-06]),
indices=tensor([  0, 319, 771, 896]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 714 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0276, 0.0033, 0.0929, 0.0252, 0.0724,
0.3738, 0.1617, 0.0067, 0.0662,
        0.0612, 0.0122, 0.0004, 0.0252, 0.0303, 0.0013, 0.0112, 0.0038, 0.0048,
        0.0076, 0.0060])

max probability is torch.return_types.max(
values=tensor(0.3738),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.3480e-05, 6.6191e-06, 6.6182e-06, 6.6178e-06]),
indices=tensor([  0, 771, 319, 896]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 715 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1922e-05, 1.2436e-06, 1.3385e-05,
9.0936e-06, 7.7606e-05, 3.6659e-06,
        2.3739e-06, 7.2874e-06, 7.7614e-06, 7.9296e-06, 1.0800e-04, 1.1678e-07,
        2.6357e-06, 5.6546e-04, 2.3231e-09, 1.2384e-06, 1.2449e-03, 1.8430e-05,
        1.9925e-03, 9.9465e-01])
```

```
max probability is torch.return_types.max(
values=tensor(0.9946),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2748e-03, 4.4707e-11, 4.4683e-11, 4.4656e-11]),
indices=tensor([  0, 655, 337, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 716 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.3808e-05, 1.7382e-05, 1.3600e-03,
2.5912e-03, 9.6550e-03, 1.3534e-03,
        6.3133e-04, 1.7263e-03, 1.4132e-03, 4.9291e-03, 6.3277e-04, 2.2260e-05,
        1.0517e-04, 5.1190e-03, 6.9780e-05, 8.2028e-05, 1.0113e-03, 1.2766e-03,
        8.3361e-01, 1.3348e-01])

max probability is torch.return_types.max(
values=tensor(0.8336),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.2293e-04, 4.5379e-08, 4.5379e-08, 4.5360e-08]),
indices=tensor([  0, 337, 914,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 717 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.2366e-06, 2.4942e-09, 8.6703e-08,
1.1849e-07, 5.0337e-07, 9.9131e-08,
        2.0658e-07, 1.4335e-07, 4.1519e-08, 4.9398e-07, 1.3633e-08, 5.4110e-06,
        1.3863e-06, 1.7930e-06, 9.9234e-01, 2.9447e-09, 2.2494e-07, 7.6156e-03,
        2.7710e-05, 2.4409e-07])

max probability is torch.return_types.max(
values=tensor(0.9923),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7456e-11, 2.9601e-13, 2.9587e-13, 2.9584e-13]),
indices=tensor([  0, 748, 428, 337]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 718 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.5406e-02, 4.5892e-01, 2.3867e-02,
4.7373e-03, 1.1737e-02, 8.0703e-03,
        1.6561e-02, 1.9171e-01, 2.1953e-01, 2.8811e-03, 8.6638e-04, 5.2554e-05,
        2.3049e-02, 8.4954e-04, 8.5436e-05, 6.9768e-04, 4.6325e-04, 4.7314e-05,
        1.4872e-04, 1.0261e-04])

max probability is torch.return_types.max(
values=tensor(0.4589),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.4378e-06, 2.2510e-07, 2.2485e-07, 2.2473e-07]),
indices=tensor([  0, 319, 316, 836]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 719 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.7151e-04, 8.6052e-04, 4.7238e-05,
1.9027e-05, 8.4099e-05, 4.4684e-04,
        2.3478e-04, 1.3035e-05, 1.1914e-04, 1.3248e-04, 9.7138e-01, 8.8971e-04,
        1.8524e-03, 8.9734e-03, 1.5933e-06, 8.1928e-03, 4.3665e-03, 1.3042e-04,
        3.3389e-04, 1.4401e-03])

max probability is torch.return_types.max(
values=tensor(0.9714),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1693e-06, 5.0877e-09, 5.0853e-09, 5.0803e-09]),
indices=tensor([  0, 771, 319, 323]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 720 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.7201e-04, 3.3037e-05, 3.9961e-05,
7.0710e-06, 3.3321e-05, 1.9590e-05,
        1.5321e-05, 1.7715e-04, 1.7049e-04, 1.5398e-04, 8.8226e-03, 2.5969e-01,
        6.3466e-03, 4.6773e-03, 7.3480e-05, 8.7129e-04, 7.0936e-01, 3.8429e-04,
        3.2030e-03, 5.2170e-03])
```

```
max probability is torch.return_types.max(
values=tensor(0.7094),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2942e-05, 1.3407e-08, 1.3405e-08, 1.3402e-08]),
indices=tensor([  0, 691, 914, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 721 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.3807e-02, 5.2132e-03, 2.2734e-02,
1.4127e-02, 3.9001e-02, 4.9946e-01,
        1.1290e-01, 3.2515e-03, 5.1579e-02, 3.3592e-02, 5.1502e-02, 7.5146e-04,
        3.5356e-02, 7.1609e-02, 3.1979e-04, 1.1416e-02, 5.8488e-03, 1.3841e-03,
        5.1631e-03, 7.5545e-03])

max probability is torch.return_types.max(
values=tensor(0.4995),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.4933e-05, 3.5591e-06, 3.5588e-06, 3.5571e-06]),
indices=tensor([  0, 319, 771, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 722 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.1722e-03, 1.0380e-03, 2.9534e-02,
1.2777e-01, 2.8393e-01, 4.1996e-01,
        1.2840e-01, 8.4004e-05, 2.0938e-04, 7.1229e-04, 2.8936e-04, 2.4566e-05,
        2.5532e-04, 1.0984e-03, 5.9272e-04, 1.4754e-04, 3.4297e-05, 3.8838e-04,
        1.8723e-03, 4.0029e-04])

max probability is torch.return_types.max(
values=tensor(0.4200),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3954e-05, 1.1389e-06, 1.1382e-06, 1.1374e-06]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 723 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0331e-04, 4.6418e-05, 2.1495e-06,
1.2599e-06, 6.6314e-06, 9.2946e-06,
        5.7082e-06, 2.0652e-06, 1.3166e-05, 1.4534e-05, 9.9565e-01, 1.1656e-04,
        2.9749e-04, 1.1630e-03, 4.2555e-08, 8.7639e-04, 7.0754e-04, 1.7384e-05,
        1.0296e-04, 8.6234e-04])

max probability is torch.return_types.max(
values=tensor(0.9957),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.4556e-07, 1.2383e-10, 1.2373e-10, 1.2355e-10]),
indices=tensor([  0, 771, 319, 323]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 724 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.9547e-06, 3.3069e-06, 2.8263e-02,
9.2255e-01, 4.1269e-02, 3.3657e-03,
        4.3467e-03, 5.7276e-07, 1.9118e-07, 2.7450e-05, 5.9835e-07, 9.5587e-10,
        9.7121e-07, 1.4368e-04, 7.5408e-07, 2.4077e-08, 1.0896e-07, 7.8804e-07,
        2.5566e-06, 1.2459e-05])

max probability is torch.return_types.max(
values=tensor(0.9226),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.2377e-08, 1.1905e-10, 1.1900e-10, 1.1898e-10]),
indices=tensor([  0, 771, 319, 910]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 725 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.8634e-02, 9.9166e-03, 1.2638e-02,
9.6354e-03, 2.8188e-02, 4.3563e-03,
        3.8816e-03, 1.3730e-02, 5.1323e-03, 6.9046e-03, 1.1590e-02, 7.0764e-04,
        2.8478e-03, 7.4829e-03, 6.3602e-05, 1.0849e-03, 2.4167e-02, 2.5795e-03,
        4.2127e-02, 7.5449e-01])

max probability is torch.return_types.max(
```

```
values=tensor(0.7545),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5893e-02, 4.1391e-06, 4.1384e-06, 4.1384e-06]),
indices=tensor([  0, 337, 691, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 726 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.8551e-04, 2.5779e-04, 8.5767e-02,
5.2411e-01, 2.1487e-01, 1.1398e-01,
        5.8011e-02, 2.4436e-05, 1.8045e-05, 2.7605e-04, 3.5758e-05, 2.4409e-06,
        4.2354e-05, 6.3619e-04, 4.6781e-04, 1.3443e-05, 6.0912e-06, 2.0546e-04,
        6.0347e-04, 2.1120e-04])

max probability is torch.return_types.max(
values=tensor(0.5241),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.3883e-06, 1.9245e-07, 1.9245e-07, 1.9238e-07]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 727 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.1621e-05, 7.6492e-06, 1.4124e-05,
1.3211e-05, 7.1187e-05, 1.1126e-05,
        5.4639e-06, 1.4283e-04, 2.3148e-04, 1.4003e-04, 9.8758e-04, 2.5422e-04,
        1.3129e-04, 1.1445e-02, 2.8673e-03, 9.3373e-04, 1.5773e-02, 8.2206e-02,
        7.6998e-01, 1.1443e-01])

max probability is torch.return_types.max(
values=tensor(0.7700),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9509e-04, 1.8460e-08, 1.8449e-08, 1.8444e-08]),
indices=tensor([  0, 337, 914, 230]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 728 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.5835e-04, 7.1684e-04, 4.1924e-02,
2.8670e-01, 3.4371e-01, 2.2110e-01,
        1.0152e-01, 4.3557e-05, 5.7072e-05, 3.5535e-04, 9.9061e-05, 3.8505e-06,
        7.7332e-05, 9.2004e-04, 3.1108e-04, 5.0661e-05, 1.2650e-05, 1.8753e-04,
        9.7805e-04, 2.8513e-04])

max probability is torch.return_types.max(
values=tensor(0.3437),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.2926e-06, 4.0165e-07, 4.0134e-07, 4.0102e-07]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 729 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.7567e-03, 6.3034e-04, 1.8688e-02,
2.3830e-02, 2.5269e-02, 8.1530e-03,
        7.6320e-03, 2.6807e-03, 5.8883e-03, 1.7113e-02, 4.3484e-02, 4.4820e-04,
        1.1964e-02, 4.1115e-01, 2.8054e-04, 6.9770e-03, 4.3551e-02, 5.8333e-03,
        5.1526e-02, 3.0127e-01])

max probability is torch.return_types.max(
values=tensor(0.4112),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.7306e-03, 7.4811e-06, 7.4784e-06, 7.4738e-06]),
indices=tensor([  0, 914, 771, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 730 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.8967e-05, 6.7536e-05, 2.6491e-06,
7.9385e-07, 1.9126e-06, 5.4516e-06,
        5.0305e-06, 1.2592e-04, 1.2199e-05, 2.9244e-06, 3.1622e-04, 9.9580e-01,
        1.2798e-03, 1.8906e-05, 1.0310e-04, 2.4208e-06, 2.1088e-03, 1.0499e-05,
        5.3232e-05, 1.7727e-05])

max probability is torch.return_types.max(
values=tensor(0.9958),
```

```
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.6712e-08, 4.1933e-10, 4.1908e-10, 4.1897e-10]),
indices=tensor([  0, 691, 382,  82]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 731 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.7696e-02, 2.8977e-03, 1.4460e-02,
9.4548e-03, 1.9825e-02, 2.0007e-02,
        1.2655e-02, 3.5928e-02, 6.8558e-02, 1.2440e-01, 4.1677e-02, 6.8814e-04,
        9.2757e-03, 2.7994e-02, 1.5667e-04, 1.3331e-02, 5.1901e-02, 4.5428e-03,
        1.0282e-01, 4.0052e-01])

max probability is torch.return_types.max(
values=tensor(0.4005),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.0242e-03, 1.3890e-05, 1.3885e-05, 1.3874e-05]),
indices=tensor([  0, 771, 914, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 732 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.1266e-04, 4.7786e-06, 3.5212e-04,
1.1057e-04, 1.0520e-03, 3.2517e-05,
        2.8213e-05, 1.8145e-04, 2.8521e-04, 3.2593e-04, 4.9840e-04, 3.6358e-05,
        1.1908e-04, 6.3631e-03, 2.1517e-03, 3.6673e-04, 8.5095e-04, 1.7618e-01,
        7.9245e-01, 1.8175e-02])

max probability is torch.return_types.max(
values=tensor(0.7924),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9770e-05, 5.2851e-09, 5.2831e-09, 5.2831e-09]),
indices=tensor([  0, 337, 748, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 733 :   Prediction is Correct?
```

```
No
first 20 classes probability: tensor([4.4927e-01, 2.2879e-01, 3.7920e-02,
2.8120e-02, 1.4915e-02, 2.8916e-02,
        4.5692e-02, 1.7815e-02, 2.6709e-02, 6.5981e-02, 2.5267e-03, 7.7329e-04,
        4.4255e-02, 2.8319e-03, 7.9511e-04, 5.9667e-04, 1.4571e-03, 3.5731e-04,
        3.3901e-04, 7.6251e-04])

max probability is torch.return_types.max(
values=tensor(0.4493),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3998e-05, 1.2350e-06, 1.2347e-06, 1.2345e-06]),
indices=tensor([  0, 319, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 734 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.0077e-04, 3.2465e-04, 7.5265e-02,
2.4186e-01, 4.8215e-01, 1.6876e-01,
        3.0925e-02, 6.0360e-06, 5.4219e-06, 2.2080e-05, 1.2905e-05, 5.4398e-07,
        1.1881e-05, 4.9130e-05, 4.8227e-05, 2.7395e-06, 1.0718e-06, 2.6086e-05,
        2.6404e-04, 2.9501e-05])

max probability is torch.return_types.max(
values=tensor(0.4822),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0309e-06, 3.6534e-08, 3.6519e-08, 3.6494e-08]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 735 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3516e-03, 1.6651e-05, 4.7605e-04,
8.7166e-05, 9.6280e-04, 9.7928e-05,
        6.5024e-05, 6.1968e-04, 1.9023e-03, 1.2897e-03, 1.7060e-03, 7.6406e-05,
        1.9441e-04, 1.6076e-02, 5.7372e-04, 1.0253e-03, 3.4428e-03, 1.2331e-01,
        8.2197e-01, 2.4692e-02])

max probability is torch.return_types.max(
values=tensor(0.8220),
indices=tensor(18))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.1626e-05, 1.6768e-08, 1.6764e-08, 1.6763e-08]),
indices=tensor([  0, 337, 158, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 736 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0271, 0.0044, 0.0025, 0.0007, 0.0051,
0.0027, 0.0020, 0.0375, 0.0690,
        0.0225, 0.0827, 0.0104, 0.1242, 0.0231, 0.0110, 0.3976, 0.0718, 0.0162,
        0.0564, 0.0130])

max probability is torch.return_types.max(
values=tensor(0.3976),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0625e-04, 2.0926e-05, 2.0921e-05, 2.0911e-05]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 737 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.9469e-03, 7.8587e-03, 1.0947e-03,
2.4783e-04, 2.1543e-03, 2.8723e-04,
        6.1514e-04, 2.4104e-02, 1.0141e-02, 5.0184e-04, 5.8575e-03, 2.2496e-02,
        6.0397e-03, 4.8516e-03, 6.8297e-03, 7.8027e-05, 1.9600e-02, 1.9549e-01,
        5.4003e-01, 1.4552e-01])

max probability is torch.return_types.max(
values=tensor(0.5400),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1108e-03, 1.4680e-07, 1.4677e-07, 1.4675e-07]),
indices=tensor([  0,  19, 158, 691]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 738 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.8964e-01, 1.3589e-04, 4.3497e-03,
```

```
6.3677e-04, 5.1512e-04, 1.8240e-03,
        7.2866e-04, 1.0190e-04, 4.9571e-04, 1.3719e-03, 4.8149e-06, 1.4848e-06,
        1.4297e-04, 2.0343e-05, 3.5380e-06, 1.6374e-06, 2.8207e-06, 1.2161e-05,
        5.2578e-06, 7.9715e-06])

max probability is torch.return_types.max(
values=tensor(0.9896),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4553e-08, 5.9238e-10, 5.9230e-10, 5.9227e-10]),
indices=tensor([  0, 319, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 739 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0333, 0.0038, 0.0036, 0.0011, 0.0065,
0.0027, 0.0038, 0.0159, 0.0232,
        0.0072, 0.0202, 0.0198, 0.0336, 0.0398, 0.2052, 0.0290, 0.0329, 0.3616,
        0.0972, 0.0134])

max probability is torch.return_types.max(
values=tensor(0.3616),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.8773e-04, 4.7640e-05, 4.7633e-05, 4.7610e-05]),
indices=tensor([  0, 748, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 740 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.7362e-03, 9.9037e-01, 1.1108e-03,
3.3354e-04, 4.6404e-04, 2.0136e-03,
        1.1544e-03, 1.0512e-04, 1.5807e-04, 1.6920e-05, 5.3746e-04, 3.4841e-05,
        7.8406e-04, 3.5192e-05, 6.2230e-07, 6.4160e-05, 6.9461e-05, 3.9857e-07,
        4.3377e-06, 5.1563e-06])

max probability is torch.return_types.max(
values=tensor(0.9904),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([8.7762e-08, 1.4006e-09, 1.3983e-09, 1.3976e-09]),
indices=tensor([  0, 319, 910, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 741 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0156, 0.0023, 0.0025, 0.0032, 0.0136,
0.0689, 0.0162, 0.0054, 0.0555,
        0.0354, 0.1323, 0.0055, 0.0591, 0.3010, 0.0037, 0.0794, 0.0560, 0.0168,
        0.0626, 0.0485])

max probability is torch.return_types.max(
values=tensor(0.3010),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.2647e-04, 1.6855e-05, 1.6849e-05, 1.6840e-05]),
indices=tensor([  0, 771, 319, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 742 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.6903e-03, 3.1638e-04, 7.3205e-02,
7.4689e-02, 9.8621e-02, 3.8296e-02,
        4.1912e-02, 2.0336e-03, 8.6183e-03, 6.4107e-02, 1.2580e-02, 8.8499e-05,
        7.8296e-03, 4.9199e-01, 3.2423e-04, 3.3470e-03, 8.1193e-03, 5.8645e-03,
        1.0505e-02, 4.5390e-02])

max probability is torch.return_types.max(
values=tensor(0.4920),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9554e-04, 3.3408e-06, 3.3391e-06, 3.3385e-06]),
indices=tensor([  0, 771, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 743 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.8982e-04, 3.9780e-04, 6.0206e-01,
2.7793e-01, 9.1299e-02, 1.7637e-02,
        9.3469e-03, 2.2524e-05, 1.9940e-05, 3.1473e-04, 2.4530e-05, 4.6526e-07,
        1.1133e-04, 7.8450e-05, 2.4926e-05, 3.5918e-06, 6.3069e-06, 3.6388e-05,
```

```
        3.5984e-05, 3.9244e-05])

max probability is torch.return_types.max(
values=tensor(0.6021),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.3944e-07, 2.1267e-08, 2.1263e-08, 2.1262e-08]),
indices=tensor([  0, 771, 466, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 744 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.8820e-02, 2.8404e-04, 2.2947e-02,
1.9049e-01, 6.4399e-02, 6.1138e-01,
        8.2134e-02, 4.4577e-05, 7.3564e-05, 1.4872e-03, 9.0684e-05, 1.2098e-04,
        1.2674e-04, 1.3644e-03, 2.2144e-03, 1.2600e-05, 1.9990e-05, 1.1066e-03,
        1.9385e-03, 8.0118e-04])

max probability is torch.return_types.max(
values=tensor(0.6114),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.7780e-06, 1.4119e-07, 1.4116e-07, 1.4113e-07]),
indices=tensor([  0, 914, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 745 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.0566e-03, 9.4576e-02, 1.0460e-03,
1.9687e-04, 1.6324e-03, 2.0932e-03,
        9.0471e-04, 7.5571e-02, 5.5464e-02, 2.1101e-03, 4.9224e-02, 9.0245e-04,
        5.1865e-02, 9.9772e-03, 2.6168e-04, 5.8479e-01, 4.0634e-02, 3.9470e-04,
        1.1917e-02, 1.3641e-02])

max probability is torch.return_types.max(
values=tensor(0.5848),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.5809e-05, 7.0254e-07, 7.0253e-07, 7.0194e-07]),
indices=tensor([  0, 319, 316, 910]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 746 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.9853e-02, 9.8321e-03, 4.8413e-03,
2.1645e-03, 7.4688e-03, 3.5272e-02,
        4.1682e-02, 2.2017e-01, 5.3142e-01, 4.6356e-02, 5.5644e-03, 2.1705e-03,
        1.9703e-02, 6.0105e-03, 1.9615e-04, 3.3522e-03, 7.9162e-03, 4.4862e-04,
        2.4512e-03, 1.5823e-03])

max probability is torch.return_types.max(
values=tensor(0.5314),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6772e-05, 1.5982e-06, 1.5982e-06, 1.5981e-06]),
indices=tensor([  0, 910, 316, 896]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 747 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.9867e-03, 3.1927e-04, 2.1836e-04,
4.4080e-04, 8.4561e-04, 3.6206e-03,
        1.5604e-03, 7.2718e-05, 6.0521e-04, 5.1677e-03, 9.4533e-01, 2.7093e-04,
        1.9657e-03, 2.4211e-02, 3.8201e-06, 5.3594e-03, 3.0342e-03, 1.6156e-04,
        9.3409e-04, 3.8170e-03])

max probability is torch.return_types.max(
values=tensor(0.9453),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.6812e-06, 6.2460e-08, 6.2446e-08, 6.2328e-08]),
indices=tensor([  0, 771, 319, 323]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 748 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.7202e-04, 8.5228e-03, 5.5733e-05,
1.8470e-05, 2.6422e-04, 1.4558e-04,
        5.2180e-05, 1.3890e-03, 2.0261e-03, 2.1787e-04, 3.3749e-01, 7.3615e-04,
        3.0517e-03, 2.6362e-02, 1.9800e-05, 2.9330e-01, 9.2824e-02, 1.8936e-03,
        9.0167e-02, 1.4073e-01])
```

```
max probability is torch.return_types.max(
values=tensor(0.3375),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2725e-04, 3.3184e-08, 3.3169e-08, 3.3161e-08]),
indices=tensor([  0, 771, 319, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 749 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0222, 0.0032, 0.0054, 0.0020, 0.0119,
0.0035, 0.0029, 0.1950, 0.2775,
        0.0347, 0.0186, 0.0015, 0.0248, 0.0420, 0.0009, 0.0207, 0.0217, 0.0093,
        0.2725, 0.0238])

max probability is torch.return_types.max(
values=tensor(0.2775),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.2237e-04, 5.7520e-06, 5.7519e-06, 5.7519e-06]),
indices=tensor([  0, 316, 896, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 750 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.1351, 0.0527, 0.0176, 0.0050, 0.0178,
0.0063, 0.0117, 0.4212, 0.2347,
        0.0089, 0.0063, 0.0030, 0.0312, 0.0038, 0.0018, 0.0039, 0.0043, 0.0033,
        0.0091, 0.0049])

max probability is torch.return_types.max(
values=tensor(0.4212),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7241e-04, 1.8092e-05, 1.8089e-05, 1.8085e-05]),
indices=tensor([  0, 316, 910, 323]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 751 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.7603e-04, 1.2832e-03, 2.4563e-04,
1.9934e-04, 1.9127e-04, 3.3168e-04,
        2.4673e-04, 1.7438e-03, 2.4066e-04, 2.4114e-04, 2.8165e-03, 9.6783e-01,
        5.9725e-03, 1.2590e-03, 3.8024e-03, 2.6051e-04, 7.6638e-03, 9.4104e-04,
        2.8799e-03, 9.1842e-04])

max probability is torch.return_types.max(
values=tensor(0.9678),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3844e-05, 2.6111e-07, 2.6097e-07, 2.6093e-07]),
indices=tensor([  0, 691,  82, 382]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 752 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0419, 0.0173, 0.0248, 0.0103, 0.0154,
0.0125, 0.0112, 0.0170, 0.0110,
        0.0231, 0.0588, 0.3955, 0.0692, 0.0181, 0.0131, 0.0122, 0.0791, 0.0196,
        0.0372, 0.0207])

max probability is torch.return_types.max(
values=tensor(0.3955),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3074e-03, 9.4634e-05, 9.4633e-05, 9.4627e-05]),
indices=tensor([  0, 158, 691,  82]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 753 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.6285e-03, 2.6875e-04, 3.5188e-02,
1.7111e-02, 5.4304e-02, 5.9001e-03,
        7.2008e-03, 1.2198e-02, 1.9963e-02, 2.3682e-02, 7.4426e-03, 9.3784e-05,
        4.5551e-03, 6.4390e-02, 1.9784e-04, 1.5452e-03, 1.8913e-02, 1.4672e-02,
        2.6204e-01, 4.4210e-01])

max probability is torch.return_types.max(
values=tensor(0.4421),
indices=tensor(19))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.3228e-03, 1.3429e-06, 1.3425e-06, 1.3423e-06]),
indices=tensor([  0, 914, 337,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 754 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0446, 0.0330, 0.0301, 0.0232, 0.0429,
0.0411, 0.0485, 0.0126, 0.0112,
        0.0110, 0.0093, 0.0236, 0.0081, 0.0227, 0.0246, 0.0040, 0.0356, 0.1447,
        0.1813, 0.1917])

max probability is torch.return_types.max(
values=tensor(0.1917),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.1127e-03, 5.1083e-05, 5.1079e-05, 5.1073e-05]),
indices=tensor([  0, 158, 691, 450]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 755 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.5524e-04, 5.7438e-04, 3.3917e-02,
2.0645e-01, 3.4971e-01, 3.2985e-01,
        7.7656e-02, 1.2297e-05, 1.7733e-05, 1.0424e-04, 3.5948e-05, 1.8851e-06,
        2.4520e-05, 1.9529e-04, 1.1784e-04, 1.6155e-05, 3.4821e-06, 5.5263e-05,
        5.3291e-04, 8.1724e-05])

max probability is torch.return_types.max(
values=tensor(0.3497),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1106e-06, 9.3562e-08, 9.3504e-08, 9.3433e-08]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 756 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.8157e-04, 1.0458e-05, 2.4137e-03,
```

```
1.4846e-03, 5.7733e-03, 2.8065e-04,
        2.4153e-04, 2.3816e-04, 4.5576e-04, 1.9536e-03, 3.5287e-03, 2.1751e-05,
        2.2426e-03, 3.9680e-01, 5.3101e-05, 8.0098e-04, 3.8483e-02, 9.6647e-03,
        4.6328e-02, 4.8827e-01])

max probability is torch.return_types.max(
values=tensor(0.4883),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.5212e-04, 2.8009e-08, 2.8000e-08, 2.7997e-08]),
indices=tensor([  0, 914, 771, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 757 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.6629e-04, 5.6429e-04, 1.9450e-01,
2.2056e-01, 5.2367e-01, 4.5928e-02,
        1.3819e-02, 1.5930e-05, 9.6547e-06, 2.2819e-05, 3.0660e-05, 1.1663e-06,
        3.9046e-05, 5.5887e-05, 5.8875e-05, 3.5122e-06, 3.9662e-06, 5.5640e-05,
        3.4089e-04, 4.5898e-05])

max probability is torch.return_types.max(
values=tensor(0.5237),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.0969e-06, 1.0495e-07, 1.0494e-07, 1.0488e-07]),
indices=tensor([  0, 319, 316, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 758 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0088, 0.0014, 0.0145, 0.0341, 0.0963,
0.1944, 0.1726, 0.0125, 0.0612,
        0.2020, 0.0116, 0.0008, 0.0104, 0.0851, 0.0015, 0.0080, 0.0137, 0.0078,
        0.0292, 0.0256])

max probability is torch.return_types.max(
values=tensor(0.2020),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([3.6371e-04, 8.5342e-06, 8.5309e-06, 8.5306e-06]),
indices=tensor([  0, 771, 319, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 759 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.6905e-03, 3.7620e-04, 1.6556e-03,
4.3160e-04, 1.9791e-03, 8.6843e-04,
        5.4191e-04, 9.9593e-04, 6.2737e-03, 5.7569e-03, 1.4602e-01, 1.1891e-03,
        4.0733e-02, 3.6725e-01, 6.4631e-05, 4.3512e-02, 3.2686e-01, 2.3402e-03,
        8.4150e-03, 3.7281e-02])

max probability is torch.return_types.max(
values=tensor(0.3672),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5664e-04, 6.5043e-07, 6.5013e-07, 6.4992e-07]),
indices=tensor([  0, 771, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 760 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.0190e-06, 1.4974e-09, 3.0831e-08,
4.0910e-07, 4.6966e-07, 4.7178e-07,
        4.6759e-07, 8.8542e-08, 1.2859e-08, 1.7500e-06, 1.2690e-08, 3.2138e-05,
        1.3799e-06, 6.8705e-07, 9.9939e-01, 2.3718e-09, 1.1798e-07, 5.4073e-04,
        3.2023e-05, 2.0610e-07])

max probability is torch.return_types.max(
values=tensor(0.9994),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1264e-11, 2.4183e-13, 2.4180e-13, 2.4167e-13]),
indices=tensor([  0, 337, 748, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 761 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0068, 0.0008, 0.0010, 0.0012, 0.0026,
0.0016, 0.0016, 0.0042, 0.0038,
        0.0054, 0.0060, 0.0302, 0.0151, 0.0230, 0.5885, 0.0056, 0.0140, 0.1304,
```

```
        0.1091, 0.0122])

max probability is torch.return_types.max(
values=tensor(0.5885),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5357e-04, 3.8019e-05, 3.8009e-05, 3.8006e-05]),
indices=tensor([  0, 748, 323, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 762 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.2178e-05, 3.7282e-05, 3.5567e-01,
4.5678e-01, 1.8069e-01, 5.5420e-03,
        1.2687e-03, 2.2595e-07, 8.4023e-08, 1.4974e-06, 2.1820e-07, 4.2299e-10,
        8.4983e-07, 2.2583e-06, 1.8598e-07, 6.2382e-09, 2.3295e-08, 2.1708e-07,
        8.7179e-07, 7.1974e-07])

max probability is torch.return_types.max(
values=tensor(0.4568),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.6824e-09, 4.7860e-11, 4.7828e-11, 4.7825e-11]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 763 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0145, 0.0010, 0.0109, 0.0158, 0.0445,
0.3155, 0.1272, 0.0029, 0.0383,
        0.1418, 0.0306, 0.0006, 0.0243, 0.1796, 0.0004, 0.0088, 0.0163, 0.0015,
        0.0084, 0.0141])

max probability is torch.return_types.max(
values=tensor(0.3155),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1472e-04, 3.1525e-06, 3.1504e-06, 3.1504e-06]),
indices=tensor([  0, 771, 319, 896]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 764 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.6883e-04, 3.7589e-03, 2.2109e-04,
6.6049e-05, 3.5271e-04, 5.7746e-04,
        2.0947e-04, 7.6909e-03, 9.4657e-03, 1.2726e-03, 1.5561e-01, 9.2899e-04,
        6.5572e-02, 2.4161e-03, 2.9507e-04, 7.3464e-01, 8.3239e-03, 2.1015e-04,
        5.8193e-03, 2.0949e-03])

max probability is torch.return_types.max(
values=tensor(0.7346),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.0549e-06, 2.1723e-07, 2.1700e-07, 2.1686e-07]),
indices=tensor([  0, 319, 316, 836]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 765 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.3107e-04, 2.2075e-04, 2.0084e-05,
8.4645e-06, 2.4208e-05, 4.7777e-05,
        3.2782e-05, 8.2155e-04, 2.2176e-04, 5.4888e-05, 6.3146e-03, 9.6622e-01,
        7.6420e-03, 3.9752e-04, 3.9026e-04, 4.3273e-04, 1.4526e-02, 2.0428e-04,
        1.4825e-03, 6.3178e-04])

max probability is torch.return_types.max(
values=tensor(0.9662),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.7639e-06, 7.0338e-08, 7.0338e-08, 7.0314e-08]),
indices=tensor([  0, 323, 914, 382]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 766 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.0651e-03, 2.3049e-03, 7.6396e-03,
2.7701e-03, 4.7268e-03, 5.6460e-03,
        3.6267e-03, 8.0144e-04, 3.7351e-03, 5.8736e-03, 6.0775e-01, 6.8809e-04,
        2.9637e-02, 1.9586e-01, 8.0287e-05, 8.5917e-02, 2.0230e-02, 1.1500e-03,
        2.6978e-03, 1.3749e-02])
```

```
max probability is torch.return_types.max(
values=tensor(0.6078),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.6368e-05, 1.0450e-06, 1.0447e-06, 1.0440e-06]),
indices=tensor([  0, 771, 319, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 767 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.3282, 0.0013, 0.1553, 0.0176, 0.0377,
0.0168, 0.0186, 0.0064, 0.0196,
        0.0788, 0.0225, 0.0068, 0.1218, 0.0681, 0.0013, 0.0048, 0.0366, 0.0136,
        0.0160, 0.0237])

max probability is torch.return_types.max(
values=tensor(0.3282),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7127e-04, 4.3557e-06, 4.3547e-06, 4.3545e-06]),
indices=tensor([  0, 914, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 768 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2859e-05, 1.2850e-07, 3.8652e-04,
9.7889e-04, 1.1718e-03, 3.7605e-05,
        5.2798e-05, 1.0192e-05, 4.2824e-05, 5.4568e-04, 6.1348e-05, 3.4032e-08,
        6.6430e-05, 9.8207e-01, 2.6013e-07, 1.5324e-05, 1.0054e-03, 7.4000e-05,
        4.2269e-04, 1.3048e-02])

max probability is torch.return_types.max(
values=tensor(0.9821),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3001e-06, 9.4383e-11, 9.4369e-11, 9.4342e-11]),
indices=tensor([  0, 914, 771, 323]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 769 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.3636e-06, 5.9732e-06, 4.6575e-06,
2.5793e-06, 2.8489e-05, 1.2871e-06,
        4.0419e-07, 2.9365e-04, 2.2095e-04, 2.9242e-05, 3.8992e-04, 6.4074e-06,
        4.7341e-05, 1.7429e-03, 3.9340e-06, 2.7153e-04, 1.0504e-02, 5.0164e-04,
        7.2896e-01, 2.5629e-01])

max probability is torch.return_types.max(
values=tensor(0.7290),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.8774e-04, 7.6150e-10, 7.6140e-10, 7.6129e-10]),
indices=tensor([  0, 337, 197, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 770 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0490e-05, 2.9183e-07, 3.9663e-04,
2.5876e-04, 5.4125e-04, 1.5558e-05,
        4.5680e-05, 6.6445e-06, 3.4745e-05, 4.6426e-04, 2.1691e-04, 2.9554e-07,
        4.0184e-04, 9.8493e-01, 9.9644e-07, 5.2944e-05, 9.5555e-03, 7.5460e-05,
        1.6854e-04, 2.8228e-03])

max probability is torch.return_types.max(
values=tensor(0.9849),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0487e-06, 2.0467e-10, 2.0466e-10, 2.0459e-10]),
indices=tensor([  0, 323, 910, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 771 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.8572e-03, 7.8401e-04, 1.4727e-01,
1.2857e-01, 4.4257e-02, 3.0575e-02,
        1.3011e-01, 3.8716e-03, 4.7747e-03, 4.7389e-01, 4.3272e-03, 1.2179e-04,
        9.1385e-03, 1.1699e-02, 1.0915e-03, 5.0355e-04, 1.6900e-03, 1.4778e-03,
        1.3091e-03, 1.3261e-03])

max probability is torch.return_types.max(
values=tensor(0.4739),
```

```
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8109e-05, 1.4114e-06, 1.4112e-06, 1.4112e-06]),
indices=tensor([  0, 771, 572,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 772 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.7557e-03, 1.0088e-04, 5.6578e-02,
1.5693e-01, 1.1107e-01, 2.0334e-02,
        4.1886e-02, 2.2580e-04, 6.9772e-04, 2.0723e-02, 1.6064e-03, 5.4129e-06,
        1.3754e-03, 5.6910e-01, 1.3355e-04, 4.0614e-04, 1.8473e-03, 1.1710e-03,
        1.4914e-03, 1.1371e-02])

max probability is torch.return_types.max(
values=tensor(0.5691),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.0045e-05, 1.7775e-07, 1.7772e-07, 1.7769e-07]),
indices=tensor([  0, 771, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 773 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.7279e-04, 3.2127e-04, 3.9711e-02,
8.7986e-03, 4.3834e-02, 1.7917e-01,
        7.2585e-01, 1.3910e-05, 8.8335e-05, 2.9030e-04, 6.5593e-05, 7.6740e-06,
        1.6321e-04, 6.7303e-04, 2.5408e-04, 8.2386e-05, 3.3219e-05, 1.8070e-04,
        1.5047e-04, 3.6679e-05])

max probability is torch.return_types.max(
values=tensor(0.7259),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7169e-07, 5.4797e-09, 5.4780e-09, 5.4779e-09]),
indices=tensor([  0, 910, 726, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 774 :   Prediction is Correct?
```

```
No
first 20 classes probability: tensor([0.0029, 0.0011, 0.0013, 0.0012, 0.0042,
0.0029, 0.0013, 0.0078, 0.0151,
        0.0138, 0.0301, 0.0113, 0.0266, 0.1244, 0.0057, 0.0466, 0.2595, 0.0208,
        0.2510, 0.1524])

max probability is torch.return_types.max(
values=tensor(0.2595),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8441e-03, 1.7957e-05, 1.7953e-05, 1.7949e-05]),
indices=tensor([  0, 914, 771, 323]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 775 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.1509e-04, 3.3631e-02, 5.0157e-05,
1.0640e-05, 1.0219e-04, 8.0172e-05,
        8.3846e-05, 8.0703e-01, 1.4893e-01, 7.3745e-05, 5.4790e-04, 1.1470e-04,
        8.2160e-03, 3.8041e-05, 1.5240e-05, 2.5428e-04, 1.0672e-04, 2.6052e-06,
        7.4151e-05, 1.7038e-05])

max probability is torch.return_types.max(
values=tensor(0.8070),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8102e-07, 6.2393e-09, 6.2378e-09, 6.2375e-09]),
indices=tensor([  0, 836, 316, 319]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 776 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.6398e-04, 1.1078e-03, 3.8551e-01,
3.4854e-01, 1.9151e-01, 4.6472e-02,
        2.3766e-02, 5.8908e-05, 5.9712e-05, 5.1963e-04, 1.7101e-04, 2.2295e-06,
        3.5704e-04, 4.8499e-04, 8.5728e-05, 3.8133e-05, 2.0702e-05, 8.0626e-05,
        2.0016e-04, 1.2930e-04])

max probability is torch.return_types.max(
values=tensor(0.3855),
indices=tensor(2))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.0827e-06, 3.3191e-07, 3.3182e-07, 3.3177e-07]),
indices=tensor([  0, 319, 771, 316]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 777 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0024, 0.0024, 0.0084, 0.0059, 0.0155,
0.0187, 0.0121, 0.0044, 0.0088,
        0.0083, 0.0527, 0.0537, 0.1004, 0.1515, 0.0938, 0.0422, 0.0992, 0.0786,
        0.1563, 0.0617])

max probability is torch.return_types.max(
values=tensor(0.1563),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.2265e-04, 2.3041e-05, 2.3034e-05, 2.3032e-05]),
indices=tensor([  0, 914, 323, 910]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 778 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.6241e-02, 1.5812e-03, 5.7342e-01,
1.2394e-01, 1.2728e-01, 4.8631e-02,
        2.6238e-02, 1.4964e-03, 5.8938e-03, 9.0767e-03, 2.0676e-03, 5.9959e-06,
        4.5391e-03, 2.5542e-02, 3.6353e-05, 4.5664e-04, 3.7370e-04, 4.2472e-04,
        4.4358e-04, 1.9962e-03])

max probability is torch.return_types.max(
values=tensor(0.5734),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3000e-05, 3.1267e-07, 3.1265e-07, 3.1256e-07]),
indices=tensor([  0, 771, 910, 319]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 779 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.2043e-07, 1.4092e-07, 4.6112e-07,
2.9460e-07, 1.6056e-06, 7.3765e-08,
```

```
        6.9332e-08, 2.4745e-06, 2.1380e-06, 1.7151e-06, 3.1686e-05, 1.3477e-08,
        4.1888e-07, 9.3789e-05, 2.6030e-10, 3.6846e-07, 2.0938e-03, 2.7102e-06,
        1.6899e-03, 9.9500e-01])

max probability is torch.return_types.max(
values=tensor(0.9950),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0744e-03, 1.4318e-12, 1.4314e-12, 1.4303e-12]),
indices=tensor([  0, 655, 337, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 780 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.5430e-04, 1.0211e-03, 2.4191e-01,
2.4763e-01, 4.2751e-01, 5.5328e-02,
        2.2831e-02, 7.7059e-05, 6.1435e-05, 1.8742e-04, 1.1195e-04, 4.4761e-06,
        1.6611e-04, 2.8002e-04, 1.6445e-04, 2.5334e-05, 2.4967e-05, 1.9237e-04,
        7.7218e-04, 1.9670e-04])

max probability is torch.return_types.max(
values=tensor(0.4275),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5991e-05, 8.7701e-07, 8.7691e-07, 8.7642e-07]),
indices=tensor([  0, 319, 316, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 781 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1032e-01, 2.2168e-03, 8.9381e-03,
3.7972e-03, 5.4136e-03, 2.1717e-02,
        4.8127e-03, 8.1219e-02, 1.5846e-01, 5.4672e-01, 1.6354e-02, 2.1477e-04,
        8.2701e-03, 4.7630e-03, 7.0497e-05, 5.1086e-03, 2.4385e-03, 3.4367e-04,
        1.3727e-02, 4.7712e-03])

max probability is torch.return_types.max(
values=tensor(0.5467),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([1.8342e-05, 3.2680e-07, 3.2679e-07, 3.2673e-07]),
indices=tensor([  0, 896, 771, 316]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 782 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.3558e-04, 6.6638e-05, 4.0920e-04,
2.0128e-04, 1.4220e-03, 3.5259e-05,
        2.9472e-05, 4.2053e-04, 1.6009e-04, 1.1415e-04, 1.5712e-03, 1.4391e-04,
        1.4811e-04, 3.3128e-03, 1.2911e-05, 4.7789e-05, 4.2889e-03, 3.2832e-03,
        2.6356e-01, 7.1690e-01])

max probability is torch.return_types.max(
values=tensor(0.7169),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5171e-03, 1.3673e-08, 1.3669e-08, 1.3669e-08]),
indices=tensor([  0, 655, 158, 691]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 783 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.7669e-02, 7.5680e-03, 3.7169e-03,
3.7242e-03, 9.5543e-03, 2.7372e-02,
        8.2173e-03, 6.7922e-03, 2.6957e-02, 1.8917e-02, 4.3828e-01, 5.3635e-03,
        2.2748e-02, 8.1894e-02, 2.1615e-04, 5.2915e-02, 4.0624e-02, 4.5812e-03,
        5.2501e-02, 1.5111e-01])

max probability is torch.return_types.max(
values=tensor(0.4383),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5344e-03, 1.7681e-05, 1.7666e-05, 1.7666e-05]),
indices=tensor([  0, 771, 914, 319]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 784 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.7285e-03, 1.1458e-01, 7.0246e-04,
6.6990e-05, 8.9625e-04, 8.6738e-04,
        9.6122e-04, 4.0890e-01, 2.1138e-01, 8.1236e-04, 8.6280e-03, 1.1808e-02,
```

```
        8.5186e-02, 3.9637e-03, 8.9733e-04, 5.2696e-02, 7.8159e-02, 8.9294e-04,
        6.7918e-03, 3.9549e-03])

max probability is torch.return_types.max(
values=tensor(0.4089),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.3308e-05, 1.1257e-06, 1.1253e-06, 1.1252e-06]),
indices=tensor([  0, 910, 316, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 785 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.6996e-03, 1.5458e-03, 3.0143e-02,
3.4147e-03, 1.1508e-02, 1.4244e-02,
        1.1748e-02, 1.5883e-01, 7.2680e-01, 1.4753e-02, 1.0845e-03, 5.5432e-06,
        1.7914e-02, 3.3178e-03, 2.1430e-05, 9.3099e-04, 5.1315e-04, 1.3510e-04,
        6.1323e-04, 6.6002e-04])

max probability is torch.return_types.max(
values=tensor(0.7268),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.5520e-06, 1.2757e-07, 1.2753e-07, 1.2752e-07]),
indices=tensor([  0, 896,  85, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 786 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.7045e-04, 3.0512e-04, 4.3481e-01,
3.1075e-01, 1.1783e-01, 1.0831e-01,
        2.7020e-02, 4.2933e-06, 8.5163e-06, 5.6748e-05, 1.8859e-05, 2.6876e-07,
        3.2084e-05, 1.8276e-04, 1.0176e-05, 2.6800e-06, 1.4026e-06, 1.9059e-05,
        2.9763e-05, 3.0103e-05])

max probability is torch.return_types.max(
values=tensor(0.4348),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1898e-07, 4.6213e-09, 4.6210e-09, 4.6196e-09]),
```

```
indices=tensor([  0, 771, 319, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 787 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.3498e-02, 3.7057e-02, 4.9615e-04,
6.0163e-05, 2.6412e-04, 2.4725e-03,
        1.6443e-03, 3.5106e-01, 5.5021e-01, 1.0863e-03, 1.0104e-03, 6.3486e-04,
        2.9587e-02, 1.6979e-04, 1.8332e-05, 1.9799e-04, 3.6698e-04, 1.0865e-05,
        8.3219e-05, 3.8451e-05])

max probability is torch.return_types.max(
values=tensor(0.5502),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.3641e-07, 3.3547e-08, 3.3542e-08, 3.3539e-08]),
indices=tensor([  0, 836, 319, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 788 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.5834e-05, 3.8636e-04, 2.4259e-06,
1.6311e-06, 6.8092e-06, 5.8623e-05,
        1.9454e-05, 2.1656e-06, 1.1673e-05, 9.7705e-06, 9.9389e-01, 2.9440e-04,
        3.1841e-04, 2.3279e-04, 1.7806e-07, 4.3602e-03, 2.5139e-04, 4.0044e-06,
        3.2252e-05, 7.1875e-05])

max probability is torch.return_types.max(
values=tensor(0.9939),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0405e-07, 3.9674e-10, 3.9612e-10, 3.9519e-10]),
indices=tensor([  0, 319, 771, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 789 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.5474e-05, 2.9550e-05, 6.9639e-01,
1.6881e-01, 1.1783e-01, 5.6321e-03,
        1.0320e-02, 2.0488e-05, 1.7178e-05, 3.9463e-04, 1.1917e-05, 4.4977e-08,
        4.1147e-05, 2.1773e-04, 9.1448e-06, 2.3233e-06, 8.9345e-06, 3.8940e-05,
```

```
        5.3359e-05, 7.5992e-05])

max probability is torch.return_types.max(
values=tensor(0.6964),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0401e-06, 9.9620e-09, 9.9614e-09, 9.9611e-09]),
indices=tensor([  0, 771, 910,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 790 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.1970e-04, 2.4157e-05, 2.7300e-04,
1.4162e-04, 6.9551e-04, 9.3152e-05,
        4.3871e-05, 9.9907e-04, 3.9333e-03, 1.4631e-03, 4.1969e-03, 6.2897e-05,
        1.0694e-02, 8.3374e-01, 6.5899e-05, 4.5250e-03, 6.9242e-02, 3.2838e-03,
        2.7340e-02, 3.8671e-02])

max probability is torch.return_types.max(
values=tensor(0.8337),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.6299e-05, 4.1609e-08, 4.1605e-08, 4.1605e-08]),
indices=tensor([  0, 771, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 791 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0223, 0.0016, 0.0108, 0.0269, 0.0827,
0.3305, 0.1951, 0.0044, 0.0277,
        0.1478, 0.0081, 0.0015, 0.0180, 0.0517, 0.0084, 0.0112, 0.0082, 0.0056,
        0.0206, 0.0066])

max probability is torch.return_types.max(
values=tensor(0.3305),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0350e-04, 1.0643e-05, 1.0642e-05, 1.0642e-05]),
indices=tensor([  0, 896, 319, 771]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 792 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.4673e-08, 4.6481e-09, 3.5794e-08,
6.6077e-08, 1.1401e-06, 1.8687e-08,
        4.8149e-09, 3.1879e-07, 2.2997e-07, 1.7571e-07, 3.5679e-05, 2.3843e-08,
        1.5736e-07, 9.0609e-04, 7.8178e-11, 1.8310e-07, 2.0378e-04, 2.8123e-06,
        1.4426e-02, 9.8437e-01])

max probability is torch.return_types.max(
values=tensor(0.9844),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.4226e-05, 1.8597e-14, 1.8592e-14, 1.8574e-14]),
indices=tensor([  0, 655, 337, 914]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 793 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.8365e-02, 4.4209e-04, 3.3660e-01,
7.0114e-02, 1.1435e-01, 9.2626e-03,
        1.3907e-02, 8.1344e-04, 1.1737e-03, 8.5300e-03, 3.4729e-03, 4.2757e-05,
        2.1164e-03, 7.1412e-02, 7.8922e-05, 4.6573e-04, 8.9026e-03, 8.9990e-03,
        1.6284e-02, 3.0216e-01])

max probability is torch.return_types.max(
values=tensor(0.3366),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7014e-03, 8.4227e-07, 8.4220e-07, 8.4210e-07]),
indices=tensor([  0, 150, 337, 914]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 794 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0812, 0.0076, 0.1856, 0.0408, 0.0438,
0.1858, 0.1774, 0.0230, 0.0951,
        0.1258, 0.0030, 0.0004, 0.0159, 0.0035, 0.0010, 0.0017, 0.0010, 0.0020,
        0.0014, 0.0008])

max probability is torch.return_types.max(
```

```
values=tensor(0.1858),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9394e-05, 3.3233e-06, 3.3232e-06, 3.3228e-06]),
indices=tensor([  0, 319, 896, 910]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 795 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.2487e-07, 1.2806e-07, 8.7513e-07,
2.4631e-07, 3.4753e-06, 3.4870e-08,
        2.7801e-08, 2.0551e-06, 7.6232e-07, 5.9198e-07, 2.7720e-05, 1.7752e-07,
        4.3663e-07, 5.3533e-04, 6.1656e-10, 9.9026e-07, 3.8265e-03, 9.0986e-06,
        7.8229e-03, 9.8721e-01])

max probability is torch.return_types.max(
values=tensor(0.9872),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.5701e-04, 1.1822e-12, 1.1809e-12, 1.1803e-12]),
indices=tensor([  0, 655, 691, 337]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 796 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0042, 0.0007, 0.0069, 0.0017, 0.0063,
0.0015, 0.0011, 0.0008, 0.0009,
        0.0012, 0.0056, 0.0023, 0.0007, 0.0131, 0.0023, 0.0022, 0.0398, 0.1625,
        0.4386, 0.3044])

max probability is torch.return_types.max(
values=tensor(0.4386),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3899e-03, 6.3468e-07, 6.3449e-07, 6.3445e-07]),
indices=tensor([  0, 158,  19, 230]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 797 :   Prediction is Correct?
```

```
Yes
first 20 classes probability: tensor([2.5266e-05, 3.7409e-07, 1.3868e-06,
3.4892e-06, 1.2353e-05, 7.7348e-06,
        9.8876e-06, 4.6645e-06, 2.3756e-06, 2.1705e-05, 4.5791e-06, 5.2229e-04,
        1.1382e-04, 5.0609e-05, 9.8767e-01, 3.3871e-06, 2.0143e-05, 1.0909e-02,
        6.0159e-04, 1.2958e-05])

max probability is torch.return_types.max(
values=tensor(0.9877),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.0624e-09, 4.8806e-10, 4.8803e-10, 4.8793e-10]),
indices=tensor([  0, 748, 337, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 798 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.8520e-04, 7.6338e-04, 1.3017e-01,
2.2014e-01, 4.9601e-01, 1.1354e-01,
        3.6344e-02, 5.2366e-05, 6.7032e-05, 2.2743e-04, 8.6039e-05, 1.8765e-06,
        7.3749e-05, 2.8355e-04, 1.1762e-04, 3.4473e-05, 1.0770e-05, 1.5467e-04,
        8.4340e-04, 1.7281e-04])

max probability is torch.return_types.max(
values=tensor(0.4960),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.4902e-06, 3.2714e-07, 3.2695e-07, 3.2672e-07]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 799 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.8658e-04, 3.9266e-04, 3.8062e-01,
3.1108e-01, 2.5145e-01, 2.4828e-02,
        2.6625e-02, 1.6755e-04, 1.0803e-04, 1.8414e-03, 1.3333e-04, 7.2348e-07,
        2.0281e-04, 6.2805e-04, 4.0367e-05, 3.1908e-05, 4.0832e-05, 8.0698e-05,
        4.4357e-04, 5.9091e-04])

max probability is torch.return_types.max(
values=tensor(0.3806),
indices=tensor(2))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9171e-05, 3.0630e-07, 3.0629e-07, 3.0628e-07]),
indices=tensor([  0,  91, 771, 319]))
End*****************************************************************************

currently at 800 current time is 25.014570474624634
*****************************************************************************
Predicting Test Sample 800 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.9442e-04, 1.1880e-03, 1.8610e-04,
1.1950e-05, 2.6338e-04, 1.5639e-05,
        1.6022e-05, 8.6861e-01, 1.2678e-01, 3.8346e-04, 2.2157e-05, 2.0345e-06,
        3.2267e-04, 1.3768e-05, 2.0630e-06, 2.7478e-05, 8.8755e-05, 5.8831e-06,
        1.3458e-03, 2.1586e-05])

max probability is torch.return_types.max(
values=tensor(0.8686),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1075e-07, 2.6494e-10, 2.6473e-10, 2.6472e-10]),
indices=tensor([  0, 316,  85, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 801 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.1386e-05, 3.8277e-05, 4.4254e-01,
4.0725e-01, 1.4190e-01, 5.5734e-03,
        2.6009e-03, 1.0775e-06, 5.3749e-07, 1.2015e-05, 1.2434e-06, 3.7876e-09,
        4.8822e-06, 1.5537e-05, 1.4460e-06, 8.3307e-08, 2.2822e-07, 2.9314e-06,
        4.1459e-06, 7.4657e-06])

max probability is torch.return_types.max(
values=tensor(0.4425),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.4013e-08, 4.6866e-10, 4.6860e-10, 4.6856e-10]),
indices=tensor([  0, 771, 319, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 802 :   Prediction is Correct?
```

```
Yes
first 20 classes probability: tensor([9.9986e-01, 9.2289e-07, 2.4434e-05,
7.4939e-06, 1.4916e-05, 4.3750e-05,
        1.5212e-05, 2.6534e-07, 1.2388e-06, 2.4614e-05, 1.1907e-07, 1.8128e-07,
        1.2816e-06, 2.8696e-06, 3.5230e-08, 9.0933e-09, 3.2266e-07, 1.7602e-06,
        3.5855e-07, 4.6789e-06])

max probability is torch.return_types.max(
values=tensor(0.9999),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7962e-10, 2.8142e-13, 2.8125e-13, 2.8116e-13]),
indices=tensor([  0, 323, 910, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 803 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.3674e-06, 2.1202e-06, 2.0290e-07,
6.2813e-08, 2.0339e-07, 6.8790e-07,
        4.6259e-07, 1.7137e-05, 1.5573e-06, 5.2473e-07, 7.7093e-05, 9.9792e-01,
        4.2349e-04, 6.8896e-06, 4.9106e-05, 1.0635e-06, 1.4264e-03, 5.9484e-06,
        4.8199e-05, 9.6286e-06])

max probability is torch.return_types.max(
values=tensor(0.9979),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6226e-08, 2.1723e-11, 2.1705e-11, 2.1703e-11]),
indices=tensor([  0, 691,  82, 382]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 804 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.8499e-03, 4.5134e-05, 7.9108e-03,
7.7710e-03, 6.1141e-03, 4.2664e-03,
        5.7241e-03, 4.6813e-03, 9.1838e-03, 9.0488e-01, 4.8679e-03, 9.1295e-05,
        4.6028e-03, 1.6206e-02, 1.0283e-04, 1.2067e-03, 6.6220e-03, 5.4126e-04,
        7.0779e-03, 5.8296e-03])

max probability is torch.return_types.max(
values=tensor(0.9049),
indices=tensor(9))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8963e-05, 4.1289e-07, 4.1265e-07, 4.1258e-07]),
indices=tensor([  0, 771, 914, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 805 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.5670e-04, 3.8451e-04, 2.7934e-01,
2.8004e-01, 4.0028e-01, 3.0679e-02,
        8.8232e-03, 9.6969e-06, 5.9056e-06, 2.9455e-05, 1.2019e-05, 1.8642e-07,
        1.9870e-05, 3.6799e-05, 1.4941e-05, 1.2829e-06, 2.2012e-06, 1.8529e-05,
        8.8478e-05, 2.7449e-05])

max probability is torch.return_types.max(
values=tensor(0.4003),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4429e-06, 3.4266e-08, 3.4254e-08, 3.4234e-08]),
indices=tensor([  0, 319, 316, 720]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 806 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.4367e-02, 1.5916e-03, 2.1797e-02,
5.1061e-03, 2.6346e-02, 1.4042e-02,
        1.2028e-02, 7.8740e-02, 4.0026e-01, 4.9402e-02, 2.2559e-02, 1.8311e-04,
        7.7117e-02, 1.8746e-01, 2.9417e-04, 1.5384e-02, 1.2271e-02, 6.0922e-03,
        1.4521e-02, 1.7885e-02])

max probability is torch.return_types.max(
values=tensor(0.4003),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2603e-04, 2.5640e-06, 2.5631e-06, 2.5624e-06]),
indices=tensor([  0, 771, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 807 :   Prediction is Correct?
Yes
```

```
first 20 classes probability: tensor([7.3248e-04, 9.9371e-04, 3.4628e-02,
6.7270e-02, 1.3814e-01, 1.7807e-01,
        5.7437e-01, 6.7501e-05, 1.2230e-04, 1.3948e-03, 7.7912e-05, 1.5843e-05,
        2.2016e-04, 8.6600e-04, 1.7014e-03, 1.2416e-04, 4.0261e-05, 4.2741e-04,
        5.2906e-04, 8.8049e-05])

max probability is torch.return_types.max(
values=tensor(0.5744),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3819e-06, 1.2141e-07, 1.2140e-07, 1.2138e-07]),
indices=tensor([  0, 910, 316, 726]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 808 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0137, 0.0043, 0.1196, 0.0491, 0.1027,
0.0449, 0.0268, 0.0204, 0.0389,
        0.0399, 0.0489, 0.0024, 0.1034, 0.1089, 0.0037, 0.0279, 0.0159, 0.0225,
        0.1210, 0.0603])

max probability is torch.return_types.max(
values=tensor(0.1210),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.7853e-04, 2.4968e-05, 2.4964e-05, 2.4958e-05]),
indices=tensor([  0, 771, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 809 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.9552e-04, 7.1952e-04, 1.2178e-01,
2.1163e-01, 5.0540e-01, 1.2413e-01,
        3.4303e-02, 2.8158e-05, 2.9432e-05, 7.4928e-05, 5.9944e-05, 2.3741e-06,
        5.8483e-05, 1.5029e-04, 9.4414e-05, 1.4016e-05, 7.6182e-06, 7.7418e-05,
        6.6035e-04, 9.4536e-05])

max probability is torch.return_types.max(
values=tensor(0.5054),
indices=tensor(4))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([6.0871e-06, 2.9901e-07, 2.9891e-07, 2.9871e-07]),
indices=tensor([  0, 319, 316, 794]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 810 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.0432e-05, 1.3198e-04, 2.8232e-01,
2.8516e-01, 4.0572e-01, 2.1322e-02,
        5.1733e-03, 2.3310e-06, 1.2564e-06, 9.1754e-06, 2.7601e-06, 3.9400e-08,
        5.4302e-06, 1.0638e-05, 5.2986e-06, 2.8497e-07, 4.4617e-07, 6.2056e-06,
        3.5332e-05, 8.5180e-06])

max probability is torch.return_types.max(
values=tensor(0.4057),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5818e-07, 3.9462e-09, 3.9452e-09, 3.9423e-09]),
indices=tensor([  0, 319, 316, 720]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 811 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.4277e-04, 4.1189e-04, 1.9196e-01,
3.7920e-01, 3.5234e-01, 5.9946e-02,
        1.5763e-02, 5.0142e-06, 3.3155e-06, 2.0410e-05, 8.5747e-06, 1.5949e-07,
        1.1898e-05, 4.2818e-05, 1.6150e-05, 1.1640e-06, 9.2547e-07, 1.3443e-05,
        6.8356e-05, 2.1204e-05])

max probability is torch.return_types.max(
values=tensor(0.3792),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.8399e-07, 2.2221e-08, 2.2211e-08, 2.2190e-08]),
indices=tensor([  0, 319, 316, 771]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 812 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.8481e-05, 8.2765e-06, 2.4209e-04,
2.2238e-04, 1.1911e-03, 1.6195e-04,
```

```
        7.1981e-05, 7.3301e-04, 1.1262e-03, 2.2870e-03, 9.7708e-04, 8.5283e-05,
        2.0194e-04, 5.9892e-03, 6.5004e-04, 8.3195e-04, 2.4775e-03, 8.1303e-03,
        9.5613e-01, 1.8257e-02])

max probability is torch.return_types.max(
values=tensor(0.9561),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.6541e-05, 6.8411e-08, 6.8408e-08, 6.8376e-08]),
indices=tensor([  0, 914, 337, 466]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 813 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.0642e-03, 5.7425e-04, 2.5722e-04,
8.9975e-05, 1.0492e-03, 9.9796e-05,
        5.2214e-05, 3.4745e-03, 3.5343e-03, 1.0012e-03, 6.9175e-02, 3.5839e-04,
        1.4028e-03, 5.3081e-03, 1.1717e-05, 6.3479e-03, 1.1522e-02, 3.9065e-03,
        6.2467e-01, 2.6142e-01])

max probability is torch.return_types.max(
values=tensor(0.6247),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.4984e-04, 3.5048e-08, 3.5042e-08, 3.5028e-08]),
indices=tensor([  0, 771, 914, 466]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 814 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.2630e-01, 7.7332e-04, 4.9524e-03,
3.9564e-03, 3.2906e-03, 2.9288e-02,
        9.4238e-03, 3.6223e-04, 1.2693e-03, 1.6050e-02, 2.0932e-04, 4.9905e-04,
        1.2039e-03, 6.2424e-04, 2.7514e-04, 5.4702e-05, 2.1279e-04, 4.6434e-04,
        3.1449e-04, 3.2508e-04])

max probability is torch.return_types.max(
values=tensor(0.9263),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([1.9225e-06, 1.6159e-07, 1.6158e-07, 1.6158e-07]),
indices=tensor([  0, 323, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 815 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.2727e-04, 9.5143e-01, 6.8335e-04,
1.1811e-04, 5.7955e-04, 5.8295e-04,
        7.9385e-04, 1.1520e-02, 9.5603e-03, 6.1530e-05, 1.2091e-03, 1.2504e-04,
        1.9968e-02, 1.2337e-04, 3.5302e-05, 1.6040e-03, 6.8668e-04, 4.7962e-06,
        3.6976e-05, 2.8298e-05])

max probability is torch.return_types.max(
values=tensor(0.9514),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.3691e-07, 2.3425e-08, 2.3402e-08, 2.3372e-08]),
indices=tensor([  0, 319, 316, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 816 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9947e-01, 4.2033e-07, 1.6576e-06,
2.3668e-07, 3.1510e-06, 3.3775e-07,
        7.3474e-07, 2.9977e-05, 7.8097e-05, 3.6880e-05, 3.3431e-07, 2.1977e-06,
        9.8318e-05, 1.3776e-06, 2.1093e-04, 5.7561e-07, 3.0814e-07, 4.6340e-05,
        1.3579e-05, 3.6091e-07])

max probability is torch.return_types.max(
values=tensor(0.9995),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6125e-10, 1.7444e-11, 1.7442e-11, 1.7435e-11]),
indices=tensor([  0, 323, 910,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 817 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.3886e-04, 3.0357e-04, 1.0578e-01,
2.3296e-01, 4.9605e-01, 1.3327e-01,
        3.0462e-02, 1.0779e-05, 9.8816e-06, 4.4781e-05, 2.3487e-05, 1.0466e-06,
```

```
        2.0930e-05, 8.5288e-05, 7.9789e-05, 5.0534e-06, 2.7486e-06, 5.7033e-05,
        4.4634e-04, 6.1090e-05])

max probability is torch.return_types.max(
values=tensor(0.4960),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3869e-06, 8.6061e-08, 8.6036e-08, 8.5969e-08]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 818 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.1584e-06, 3.1092e-07, 1.3808e-06,
1.6436e-06, 1.6234e-05, 2.0623e-06,
        6.1087e-07, 2.0720e-04, 1.3412e-04, 2.5390e-04, 1.4509e-04, 9.2763e-06,
        3.7091e-06, 1.5084e-04, 8.0935e-06, 1.6501e-05, 1.1444e-04, 4.3156e-04,
        9.8503e-01, 1.3454e-02])

max probability is torch.return_types.max(
values=tensor(0.9850),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1284e-05, 1.0034e-10, 1.0034e-10, 1.0027e-10]),
indices=tensor([  0, 914, 337,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 819 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.4618e-05, 6.9940e-06, 7.2213e-05,
7.9208e-05, 2.1581e-04, 4.1311e-05,
        2.1331e-05, 7.9537e-05, 2.3652e-04, 4.5909e-04, 1.0080e-02, 1.0303e-04,
        3.1727e-04, 1.8180e-01, 1.4716e-04, 1.4703e-03, 4.5030e-02, 3.7507e-02,
        2.3462e-01, 4.8714e-01])

max probability is torch.return_types.max(
values=tensor(0.4871),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.1852e-04, 1.0393e-08, 1.0393e-08, 1.0381e-08]),
```

```
indices=tensor([  0, 337, 914, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 820 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2034e-02, 1.1047e-03, 2.0179e-02,
1.1627e-02, 1.1780e-02, 4.2894e-02,
        5.5492e-02, 5.7423e-03, 2.6305e-02, 7.3652e-01, 2.2945e-02, 6.2521e-04,
        1.9307e-02, 1.2057e-02, 5.3279e-04, 4.1593e-03, 7.7798e-03, 1.8166e-03,
        2.6924e-03, 2.7628e-03])

max probability is torch.return_types.max(
values=tensor(0.7365),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4410e-05, 1.7152e-06, 1.7141e-06, 1.7138e-06]),
indices=tensor([  0, 771, 896, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 821 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.3101e-03, 2.7173e-03, 3.5147e-01,
2.5096e-01, 9.2335e-02, 4.1396e-02,
        2.1613e-01, 7.9162e-04, 6.9566e-04, 3.1432e-02, 7.1012e-04, 2.1685e-05,
        1.5413e-03, 3.6071e-03, 4.4796e-04, 5.8886e-05, 1.5288e-04, 9.9951e-04,
        3.1075e-04, 6.5595e-04])

max probability is torch.return_types.max(
values=tensor(0.3515),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.2555e-06, 2.6556e-07, 2.6552e-07, 2.6550e-07]),
indices=tensor([  0,  91, 771, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 822 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4855e-03, 3.2198e-01, 1.8629e-04,
3.8681e-05, 3.8322e-04, 7.6052e-03,
        2.8275e-03, 2.0219e-03, 1.9835e-02, 1.6257e-04, 9.6911e-02, 1.2061e-03,
        2.6854e-01, 1.7913e-03, 1.0195e-04, 2.7022e-01, 4.2343e-03, 2.6607e-05,
```

```
        1.6933e-04, 2.0190e-04])

max probability is torch.return_types.max(
values=tensor(0.3220),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5196e-06, 7.4222e-08, 7.3973e-08, 7.3970e-08]),
indices=tensor([  0, 319, 316, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 823 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.7234e-04, 7.8997e-04, 5.6869e-04,
1.2763e-04, 6.0300e-04, 8.6499e-04,
        1.1854e-03, 5.6511e-03, 2.5882e-02, 2.5615e-04, 1.0769e-02, 1.3669e-03,
        9.4452e-01, 3.1151e-03, 2.7777e-04, 2.0012e-03, 1.1881e-03, 1.2174e-04,
        1.4134e-04, 2.0637e-04])

max probability is torch.return_types.max(
values=tensor(0.9445),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7317e-06, 9.6474e-08, 9.6435e-08, 9.6408e-08]),
indices=tensor([  0,  44, 910, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 824 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3172e-02, 6.6761e-04, 1.2171e-02,
3.0491e-02, 4.1197e-02, 2.5687e-01,
        9.0773e-02, 3.7744e-03, 2.3849e-02, 4.5528e-01, 1.2291e-02, 2.9886e-04,
        7.1086e-03, 2.3224e-02, 1.0105e-03, 3.7400e-03, 2.0192e-03, 1.5973e-03,
        1.1677e-02, 5.3577e-03])

max probability is torch.return_types.max(
values=tensor(0.4553),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.2172e-05, 3.5825e-06, 3.5819e-06, 3.5815e-06]),
indices=tensor([  0, 771, 319, 896]))
```

```
End**********************************************************************

**********************************************************************
Predicting Test Sample 825 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.1487e-03, 3.5600e-03, 4.6652e-03,
2.0374e-03, 5.4686e-03, 1.0496e-02,
        7.4742e-03, 1.5759e-03, 8.0344e-03, 4.0073e-03, 8.2034e-01, 6.1250e-04,
        4.4446e-02, 1.8343e-02, 5.7337e-05, 5.7784e-02, 2.3481e-03, 4.9281e-04,
        1.8591e-03, 3.7221e-03])

max probability is torch.return_types.max(
values=tensor(0.8203),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0006e-05, 5.5136e-07, 5.5078e-07, 5.5008e-07]),
indices=tensor([  0, 319, 771, 316]))
End**********************************************************************

**********************************************************************
Predicting Test Sample 826 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0335, 0.2471, 0.0152, 0.0037, 0.0121,
0.0402, 0.0259, 0.0382, 0.1109,
        0.0074, 0.0677, 0.0105, 0.1520, 0.0448, 0.0028, 0.0370, 0.0375, 0.0040,
        0.0089, 0.0115])

max probability is torch.return_types.max(
values=tensor(0.2471),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.3819e-04, 9.2461e-05, 9.2441e-05, 9.2440e-05]),
indices=tensor([  0, 910,  44, 323]))
End**********************************************************************

**********************************************************************
Predicting Test Sample 827 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.0712e-03, 1.3007e-03, 4.2533e-03,
3.4647e-03, 2.4535e-02, 6.6304e-03,
        1.8609e-03, 3.7010e-02, 1.9058e-02, 1.4912e-02, 1.5030e-02, 1.5484e-03,
        1.8719e-03, 1.1057e-02, 8.3078e-05, 3.1582e-03, 1.1374e-02, 3.6856e-03,
        5.6804e-01, 2.6049e-01])
```

```
max probability is torch.return_types.max(
values=tensor(0.5680),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6375e-03, 9.7570e-07, 9.7538e-07, 9.7530e-07]),
indices=tensor([  0, 914, 337, 197]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 828 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0024, 0.0011, 0.0182, 0.0063, 0.0601,
0.0631, 0.1024, 0.0053, 0.0323,
        0.0273, 0.0243, 0.0031, 0.0253, 0.3044, 0.0012, 0.0228, 0.1799, 0.0075,
        0.0736, 0.0360])

max probability is torch.return_types.max(
values=tensor(0.3044),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.3200e-04, 3.0570e-06, 3.0561e-06, 3.0560e-06]),
indices=tensor([  0, 914, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 829 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0202, 0.0438, 0.0088, 0.0021, 0.0072,
0.0193, 0.0216, 0.0269, 0.0853,
        0.0064, 0.0481, 0.0314, 0.5561, 0.0232, 0.0076, 0.0262, 0.0291, 0.0033,
        0.0021, 0.0023])

max probability is torch.return_types.max(
values=tensor(0.5561),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1916e-04, 3.0411e-05, 3.0404e-05, 3.0396e-05]),
indices=tensor([  0, 910, 323, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 830 :   Prediction is Correct?
```

```
Yes
first 20 classes probability: tensor([7.6281e-02, 5.0587e-04, 9.0787e-03,
6.1868e-03, 1.7830e-02, 1.5401e-02,
        1.2099e-02, 2.1007e-03, 5.3900e-03, 3.1244e-02, 2.9828e-03, 6.1060e-04,
        1.8891e-03, 1.3682e-02, 1.0533e-01, 1.5995e-03, 1.7383e-03, 6.2574e-01,
        5.4811e-02, 1.2281e-02])

max probability is torch.return_types.max(
values=tensor(0.6257),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.4850e-05, 3.2886e-06, 3.2881e-06, 3.2880e-06]),
indices=tensor([  0, 323, 337, 771]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 831 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.6198e-05, 8.4562e-04, 1.6339e-05,
9.3306e-06, 6.0496e-05, 3.4328e-04,
        1.0646e-04, 6.2292e-05, 5.6331e-04, 7.6693e-05, 7.4156e-01, 1.0309e-04,
        2.7958e-03, 1.0607e-02, 2.6025e-06, 2.3819e-01, 2.6145e-03, 4.7276e-05,
        3.5392e-04, 1.5914e-03])

max probability is torch.return_types.max(
values=tensor(0.7416),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8168e-06, 8.0824e-09, 8.0662e-09, 8.0544e-09]),
indices=tensor([  0, 319, 771, 316]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 832 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.9491e-04, 4.8141e-05, 1.7864e-05,
1.0177e-05, 3.7140e-05, 2.0660e-05,
        2.8523e-05, 3.5384e-05, 3.4442e-05, 2.6722e-05, 4.1827e-04, 5.7236e-05,
        2.4883e-05, 1.4573e-03, 9.4675e-08, 2.1061e-05, 5.5728e-02, 8.2782e-05,
        1.8072e-03, 9.3735e-01])

max probability is torch.return_types.max(
values=tensor(0.9373),
indices=tensor(19))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6025e-03, 8.9179e-10, 8.9137e-10, 8.9047e-10]),
indices=tensor([  0, 655, 691, 450]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 833 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0114e-04, 1.6008e-05, 7.4840e-04,
4.8236e-04, 1.2106e-03, 2.1976e-04,
        2.0095e-04, 1.1498e-04, 6.5461e-04, 1.3992e-03, 4.5052e-03, 1.9010e-05,
        3.3574e-03, 8.5964e-01, 5.2609e-06, 9.3931e-04, 9.5191e-02, 3.1917e-04,
        2.1578e-03, 2.8657e-02])

max probability is torch.return_types.max(
values=tensor(0.8596),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.9484e-05, 1.4213e-08, 1.4209e-08, 1.4208e-08]),
indices=tensor([  0, 914, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 834 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.8660e-03, 4.7996e-02, 2.2884e-04,
6.2423e-05, 3.9377e-04, 9.6014e-04,
        1.2636e-03, 5.6098e-01, 3.6613e-01, 5.4308e-04, 1.1300e-03, 1.0712e-04,
        1.6933e-02, 1.5115e-04, 3.2567e-05, 8.7903e-04, 1.2306e-04, 1.2358e-05,
        1.1185e-04, 3.6643e-05])

max probability is torch.return_types.max(
values=tensor(0.5610),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.9458e-07, 5.7548e-08, 5.7533e-08, 5.7517e-08]),
indices=tensor([  0, 836, 319, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 835 :   Prediction is Correct?
Yes
```

```
first 20 classes probability: tensor([3.9121e-04, 2.5290e-04, 7.5843e-05,
3.7448e-05, 1.6218e-04, 4.0258e-04,
        1.5633e-04, 1.2237e-04, 1.3266e-03, 4.9396e-04, 9.0739e-01, 4.4477e-04,
        1.9603e-02, 8.3488e-03, 1.1065e-05, 5.7515e-02, 2.0120e-03, 1.1628e-04,
        5.6099e-04, 5.5013e-04])

max probability is torch.return_types.max(
values=tensor(0.9074),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2589e-06, 3.3630e-08, 3.3580e-08, 3.3530e-08]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 836 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0153, 0.0050, 0.0099, 0.0408, 0.0914,
0.5604, 0.2052, 0.0011, 0.0043,
        0.0111, 0.0039, 0.0015, 0.0047, 0.0088, 0.0054, 0.0046, 0.0011, 0.0018,
        0.0096, 0.0018])

max probability is torch.return_types.max(
values=tensor(0.5604),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.0620e-05, 1.2869e-05, 1.2868e-05, 1.2863e-05]),
indices=tensor([  0, 319, 316, 144]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 837 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0116, 0.0034, 0.0020, 0.0014, 0.0061,
0.0096, 0.0045, 0.3029, 0.3296,
        0.0999, 0.0083, 0.0057, 0.0182, 0.0219, 0.0018, 0.0170, 0.0324, 0.0052,
        0.1015, 0.0127])

max probability is torch.return_types.max(
values=tensor(0.3296),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([1.9069e-04, 4.1723e-06, 4.1721e-06, 4.1719e-06]),
indices=tensor([  0, 323, 910, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 838 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.8965e-04, 2.1621e-03, 3.7525e-04,
4.2692e-05, 2.6269e-04, 4.8161e-04,
        1.0830e-03, 3.1141e-01, 6.7049e-01, 3.4035e-04, 2.2879e-04, 9.7672e-06,
        1.1877e-02, 9.1864e-05, 4.1241e-06, 7.6087e-05, 3.7426e-05, 3.5195e-06,
        1.6806e-05, 9.2387e-06])

max probability is torch.return_types.max(
values=tensor(0.6705),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1188e-07, 4.0989e-09, 4.0975e-09, 4.0954e-09]),
indices=tensor([  0, 836, 319, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 839 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.4990e-04, 7.1409e-07, 5.3444e-05,
2.2043e-05, 2.3883e-04, 4.9705e-06,
        3.5551e-06, 2.0880e-03, 2.2096e-03, 4.3596e-03, 3.3965e-05, 1.6923e-06,
        6.0056e-05, 4.2611e-03, 6.3279e-05, 1.4012e-04, 7.9414e-04, 1.6382e-03,
        9.7852e-01, 5.2544e-03])

max probability is torch.return_types.max(
values=tensor(0.9785),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.5527e-06, 8.7623e-10, 8.7589e-10, 8.7579e-10]),
indices=tensor([  0, 197, 263,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 840 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.2311e-04, 7.8834e-05, 1.7784e-03,
2.1726e-03, 2.9200e-03, 4.0211e-04,
        4.1335e-04, 3.9019e-04, 5.8468e-04, 1.6104e-03, 1.1884e-02, 5.8314e-05,
```

```
        1.0820e-03, 1.0173e-01, 1.1240e-05, 6.6821e-04, 4.3055e-02, 1.7381e-03,
        2.4490e-02, 7.9915e-01])

max probability is torch.return_types.max(
values=tensor(0.7992),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.3609e-03, 2.1350e-07, 2.1346e-07, 2.1341e-07]),
indices=tensor([  0, 914, 337, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 841 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0137, 0.0021, 0.0068, 0.0104, 0.0438,
0.3140, 0.1343, 0.0030, 0.0335,
        0.1048, 0.0808, 0.0009, 0.0228, 0.1163, 0.0013, 0.0480, 0.0147, 0.0051,
        0.0174, 0.0221])

max probability is torch.return_types.max(
values=tensor(0.3140),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3389e-04, 4.0859e-06, 4.0852e-06, 4.0824e-06]),
indices=tensor([  0, 319, 771, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 842 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.0697e-05, 9.5160e-06, 1.5728e-04,
1.9210e-04, 5.4002e-04, 3.3787e-04,
        3.5012e-04, 6.9901e-02, 9.1282e-02, 8.1525e-01, 2.9846e-04, 1.1113e-06,
        2.7473e-04, 1.9802e-03, 2.5571e-06, 2.4999e-04, 7.0100e-04, 2.4147e-05,
        1.7423e-02, 9.8560e-04])

max probability is torch.return_types.max(
values=tensor(0.8152),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2129e-06, 1.2907e-09, 1.2895e-09, 1.2890e-09]),
indices=tensor([  0, 896, 771, 363]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 843 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4647e-04, 5.0659e-05, 8.0475e-01,
9.6986e-02, 6.2243e-02, 1.5857e-02,
        1.8987e-02, 1.2782e-05, 1.9649e-05, 2.2732e-04, 2.6724e-05, 1.9389e-07,
        5.6675e-05, 3.2314e-04, 1.6434e-05, 7.6721e-06, 1.1403e-05, 7.4004e-05,
        7.1990e-05, 1.1554e-04])

max probability is torch.return_types.max(
values=tensor(0.8048),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0680e-06, 1.4069e-08, 1.4069e-08, 1.4067e-08]),
indices=tensor([  0, 910, 771, 466]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 844 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.6572e-05, 6.1169e-04, 1.4641e-05,
4.7517e-06, 5.6637e-05, 5.3647e-06,
        3.1593e-06, 5.7642e-04, 3.1942e-04, 8.0405e-06, 4.4937e-04, 4.1683e-06,
        8.7465e-05, 1.4001e-03, 7.8531e-08, 1.7214e-05, 1.4030e-02, 3.8749e-05,
        2.1538e-02, 9.5545e-01])

max probability is torch.return_types.max(
values=tensor(0.9555),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.3571e-03, 7.1151e-10, 7.1107e-10, 7.1077e-10]),
indices=tensor([  0, 655, 337, 691]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 845 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.2604e-03, 1.3227e-01, 1.0641e-02,
1.2714e-02, 1.6245e-02, 4.6785e-01,
        7.6267e-02, 1.7129e-04, 1.5481e-03, 1.4582e-03, 2.3270e-01, 4.1146e-04,
        2.0109e-02, 7.2166e-03, 8.1748e-05, 1.3233e-02, 6.0927e-04, 6.7181e-05,
        5.7477e-04, 4.1265e-04])
```

```
max probability is torch.return_types.max(
values=tensor(0.4679),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.4598e-06, 1.6241e-07, 1.6207e-07, 1.6195e-07]),
indices=tensor([  0, 319, 771, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 846 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0474, 0.0041, 0.0687, 0.0876, 0.0824,
0.2535, 0.3010, 0.0019, 0.0061,
        0.0395, 0.0037, 0.0009, 0.0049, 0.0413, 0.0049, 0.0009, 0.0022, 0.0204,
        0.0085, 0.0165])

max probability is torch.return_types.max(
values=tensor(0.3010),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3838e-04, 3.5910e-06, 3.5894e-06, 3.5894e-06]),
indices=tensor([  0, 914, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 847 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.8007e-03, 4.4544e-05, 3.0038e-05,
1.1064e-04, 2.0728e-04, 4.6189e-04,
        3.0874e-04, 8.8711e-05, 4.3876e-05, 1.7661e-04, 8.9600e-05, 4.2702e-03,
        8.9454e-04, 4.6269e-04, 9.5646e-01, 2.2410e-05, 1.1538e-04, 3.1640e-02,
        2.4162e-03, 2.6206e-04])

max probability is torch.return_types.max(
values=tensor(0.9565),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.8096e-07, 9.5336e-08, 9.5336e-08, 9.5322e-08]),
indices=tensor([  0, 323, 337, 897]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 848 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.1581e-04, 1.0911e-04, 2.1552e-02,
1.1664e-01, 9.4625e-02, 1.2406e-02,
        2.9604e-02, 2.6948e-03, 4.5498e-03, 7.2663e-02, 3.7641e-03, 5.7421e-05,
        5.0283e-03, 5.6475e-01, 8.0712e-04, 1.8175e-03, 6.8054e-03, 3.0937e-03,
        2.7016e-02, 2.8954e-02])

max probability is torch.return_types.max(
values=tensor(0.5647),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6098e-04, 2.3986e-06, 2.3985e-06, 2.3985e-06]),
indices=tensor([  0,  91, 914, 771]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 849 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.0621e-02, 3.4710e-04, 1.3998e-02,
3.0122e-02, 5.1596e-02, 1.2745e-01,
        5.8243e-02, 2.5875e-03, 1.1368e-02, 1.1813e-01, 1.5879e-02, 9.1861e-04,
        8.1998e-03, 3.6052e-01, 6.8050e-04, 3.5075e-03, 2.4905e-02, 5.6390e-03,
        5.4500e-02, 9.7356e-02])

max probability is torch.return_types.max(
values=tensor(0.3605),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.2859e-04, 3.0594e-06, 3.0586e-06, 3.0564e-06]),
indices=tensor([  0, 914, 771, 323]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 850 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.6312e-02, 2.7692e-02, 2.2910e-03,
8.2292e-05, 1.0874e-03, 5.6463e-04,
        4.4033e-04, 7.7668e-03, 3.0872e-02, 4.9749e-04, 2.2487e-02, 2.6748e-03,
        4.8205e-01, 3.0832e-02, 1.1823e-04, 3.6605e-02, 3.2410e-01, 1.7496e-04,
        6.4973e-04, 2.5436e-03])

max probability is torch.return_types.max(
```

```
values=tensor(0.4820),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6880e-05, 1.5308e-07, 1.5299e-07, 1.5281e-07]),
indices=tensor([  0, 910, 323, 726]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 851 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0175, 0.0024, 0.0156, 0.0083, 0.0262,
0.0195, 0.0114, 0.0069, 0.0286,
        0.0240, 0.1368, 0.0015, 0.0312, 0.3434, 0.0025, 0.0471, 0.0239, 0.0517,
        0.0725, 0.1089])

max probability is torch.return_types.max(
values=tensor(0.3434),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1567e-03, 1.9835e-05, 1.9818e-05, 1.9818e-05]),
indices=tensor([  0, 771, 914, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 852 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.9674e-06, 1.4405e-03, 5.8655e-07,
6.7681e-08, 1.7914e-06, 4.4394e-06,
        1.7122e-06, 7.9631e-05, 1.2526e-04, 2.5106e-06, 1.2070e-02, 3.4105e-05,
        8.3713e-04, 9.8265e-05, 2.5133e-06, 9.8372e-01, 1.3295e-03, 5.6587e-06,
        1.5356e-04, 8.8628e-05])

max probability is torch.return_types.max(
values=tensor(0.9837),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5107e-08, 6.9449e-11, 6.9335e-11, 6.9250e-11]),
indices=tensor([  0, 319, 316, 476]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 853 :   Prediction is Correct?
```

```
No
first 20 classes probability: tensor([8.6231e-05, 1.2792e-04, 3.4681e-01,
4.0053e-01, 2.2940e-01, 1.8624e-02,
        4.3844e-03, 1.5266e-06, 8.2480e-07, 9.1291e-06, 1.4891e-06, 1.7017e-08,
        3.9990e-06, 7.8638e-06, 1.8428e-06, 6.7370e-08, 2.3045e-07, 2.0697e-06,
        7.1661e-06, 4.2009e-06])

max probability is torch.return_types.max(
values=tensor(0.4005),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1140e-07, 1.5981e-09, 1.5974e-09, 1.5967e-09]),
indices=tensor([  0, 319, 316, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 854 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9504e-01, 6.3819e-05, 4.1376e-04,
3.9261e-05, 1.1117e-04, 7.5218e-05,
        1.0568e-04, 1.0001e-03, 2.2732e-03, 2.6068e-04, 3.2850e-06, 3.6129e-05,
        5.0870e-04, 1.8213e-05, 4.0654e-06, 2.5596e-07, 1.0294e-05, 1.4931e-05,
        8.7338e-06, 9.6290e-06])

max probability is torch.return_types.max(
values=tensor(0.9950),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4508e-08, 1.6221e-10, 1.6219e-10, 1.6214e-10]),
indices=tensor([  0, 391, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 855 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.7286e-06, 3.3712e-05, 8.3771e-08,
1.8051e-08, 6.6437e-07, 1.1769e-06,
        2.5711e-07, 7.8123e-06, 6.0382e-05, 2.2048e-06, 1.5183e-01, 3.9996e-04,
        1.1000e-03, 2.2233e-03, 1.2179e-06, 7.8941e-01, 5.3318e-02, 2.6894e-05,
        3.9268e-04, 1.1911e-03])

max probability is torch.return_types.max(
values=tensor(0.7894),
indices=tensor(15))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3630e-07, 3.2717e-11, 3.2637e-11, 3.2635e-11]),
indices=tensor([  0, 319, 771, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 856 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0914, 0.0103, 0.0032, 0.0006, 0.0040,
0.0093, 0.0030, 0.0314, 0.1924,
        0.0116, 0.0420, 0.0030, 0.2434, 0.1333, 0.0007, 0.0279, 0.1274, 0.0042,
        0.0128, 0.0405])

max probability is torch.return_types.max(
values=tensor(0.2434),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.8774e-04, 7.6174e-06, 7.6173e-06, 7.6153e-06]),
indices=tensor([  0, 771, 910, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 857 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.9068e-04, 2.3732e-01, 3.2064e-05,
8.1707e-06, 1.0019e-04, 6.8536e-05,
        9.4541e-05, 6.7343e-01, 8.4975e-02, 5.9881e-05, 1.6412e-04, 1.0507e-05,
        2.6294e-03, 1.8306e-05, 9.8132e-06, 5.1672e-04, 6.0049e-05, 1.7006e-06,
        8.6963e-05, 1.7903e-05])

max probability is torch.return_types.max(
values=tensor(0.6734),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7775e-07, 1.9263e-09, 1.9251e-09, 1.9250e-09]),
indices=tensor([  0, 316, 319, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 858 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.3598e-07, 1.6629e-09, 1.7477e-08,
```

```
5.4289e-08, 2.0684e-07, 6.8904e-08,
        7.4488e-08, 3.9138e-07, 5.5157e-08, 1.3693e-06, 2.1759e-08, 2.4730e-05,
        1.1581e-06, 5.2712e-07, 9.9801e-01, 6.3674e-09, 4.4360e-07, 1.6400e-03,
        3.2171e-04, 5.5583e-07])

max probability is torch.return_types.max(
values=tensor(0.9980),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.8758e-11, 5.1627e-13, 5.1627e-13, 5.1609e-13]),
indices=tensor([  0, 748, 337, 230]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 859 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.9223e-03, 2.3876e-03, 2.6566e-01,
3.3514e-01, 1.6727e-01, 1.0249e-01,
        1.0479e-01, 3.5606e-04, 4.0354e-04, 3.9011e-03, 5.9781e-04, 3.0701e-05,
        1.1355e-03, 3.5574e-03, 9.7468e-04, 2.0528e-04, 1.4567e-04, 6.4491e-04,
        8.0225e-04, 7.0113e-04])

max probability is torch.return_types.max(
values=tensor(0.3351),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6682e-05, 5.0826e-06, 5.0825e-06, 5.0816e-06]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 860 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.7782e-04, 4.3078e-04, 1.1698e-02,
9.8823e-02, 2.0844e-01, 5.7783e-01,
        1.0089e-01, 1.0293e-05, 3.2289e-05, 1.1238e-04, 3.4718e-05, 2.0084e-06,
        2.7507e-05, 1.5662e-04, 7.7146e-05, 1.5397e-05, 2.1934e-06, 3.5294e-05,
        3.0724e-04, 5.3281e-05])

max probability is torch.return_types.max(
values=tensor(0.5778),
indices=tensor(5))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([8.5952e-07, 3.8247e-08, 3.8213e-08, 3.8195e-08]),
indices=tensor([  0, 319, 316, 794]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 861 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.4831e-01, 9.5493e-04, 8.2813e-04,
1.9792e-04, 9.1950e-04, 7.8360e-04,
        5.4703e-04, 9.9493e-04, 2.6193e-03, 3.7595e-03, 2.4711e-02, 7.1953e-04,
        8.2411e-04, 6.6950e-04, 5.1821e-05, 3.3911e-03, 6.7836e-04, 1.8788e-03,
        4.8707e-03, 2.1916e-03])

max probability is torch.return_types.max(
values=tensor(0.9483),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.8037e-06, 9.0411e-08, 9.0361e-08, 9.0313e-08]),
indices=tensor([  0, 319, 771, 323]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 862 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.6526e-03, 8.8077e-04, 6.2257e-02,
4.3946e-02, 7.2659e-02, 4.7728e-01,
        3.3608e-01, 6.3294e-05, 3.8395e-04, 1.4752e-03, 2.5611e-04, 2.3107e-05,
        5.5217e-04, 1.2704e-03, 2.3764e-04, 1.6564e-04, 6.2392e-05, 1.5103e-04,
        3.5226e-04, 1.3747e-04])

max probability is torch.return_types.max(
values=tensor(0.4773),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7855e-06, 1.1777e-07, 1.1775e-07, 1.1775e-07]),
indices=tensor([  0, 910, 896, 316]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 863 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2389e-02, 4.4142e-04, 1.5281e-03,
2.8932e-04, 2.7759e-03, 4.9991e-04,
```

```
        4.0705e-04, 1.7884e-03, 4.5814e-03, 2.0961e-03, 1.6492e-02, 4.0463e-03,
        5.1413e-03, 7.3258e-02, 6.8927e-03, 9.6542e-03, 4.5444e-02, 4.3738e-01,
        3.0298e-01, 6.9590e-02])

max probability is torch.return_types.max(
values=tensor(0.4374),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.1616e-04, 1.8841e-06, 1.8841e-06, 1.8841e-06]),
indices=tensor([  0, 748, 337, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 864 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3914e-05, 5.9554e-03, 2.5816e-06,
5.9593e-07, 1.0208e-05, 1.3089e-05,
        4.4137e-06, 7.5792e-04, 4.6153e-04, 8.5009e-06, 2.9540e-02, 3.8539e-05,
        2.4573e-03, 5.5398e-04, 7.8622e-06, 9.5743e-01, 1.4406e-03, 1.3415e-05,
        6.2802e-04, 6.6443e-04])

max probability is torch.return_types.max(
values=tensor(0.9574),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9950e-07, 6.7445e-10, 6.7366e-10, 6.7285e-10]),
indices=tensor([  0, 319, 316, 476]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 865 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.1616e-05, 9.4376e-06, 2.8296e-05,
2.5347e-05, 2.0039e-04, 3.9286e-05,
        1.3559e-05, 3.4005e-03, 1.9567e-03, 2.1127e-03, 6.3246e-04, 1.8141e-03,
        5.2267e-04, 1.7024e-03, 1.4353e-03, 1.3146e-03, 6.6589e-03, 3.0614e-03,
        9.6977e-01, 5.1999e-03])

max probability is torch.return_types.max(
values=tensor(0.9698),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([1.4209e-05, 1.5494e-08, 1.5489e-08, 1.5485e-08]),
indices=tensor([  0, 914, 197, 323]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 866 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.3415e-05, 2.9985e-04, 8.7872e-06,
1.6757e-06, 4.1781e-05, 3.9956e-05,
        1.3147e-05, 4.2268e-04, 2.3846e-03, 9.7360e-05, 3.1634e-02, 5.1517e-05,
        4.4613e-03, 1.8449e-02, 1.3657e-05, 8.9904e-01, 3.6101e-02, 2.3836e-04,
        2.6059e-03, 4.0302e-03])

max probability is torch.return_types.max(
values=tensor(0.8990),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4810e-06, 4.2041e-09, 4.1980e-09, 4.1963e-09]),
indices=tensor([  0, 319, 771, 316]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 867 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.6896e-03, 2.2374e-03, 4.6953e-03,
3.4385e-03, 1.3318e-02, 8.3336e-01,
        8.4835e-02, 6.2968e-04, 2.1378e-02, 4.6137e-03, 4.2207e-03, 1.2122e-04,
        8.6296e-03, 6.3393e-03, 2.5624e-04, 3.4972e-03, 4.1429e-04, 5.3505e-04,
        1.4824e-03, 1.0223e-03])

max probability is torch.return_types.max(
values=tensor(0.8334),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.2641e-06, 2.9403e-07, 2.9379e-07, 2.9374e-07]),
indices=tensor([  0, 319, 771, 896]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 868 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.4219e-05, 9.9008e-05, 1.4054e-03,
2.5871e-04, 2.4455e-03, 1.1790e-04,
        9.4795e-05, 2.1226e-01, 1.0493e-01, 2.0223e-02, 4.7185e-04, 7.3550e-06,
```

```
        7.7712e-04, 6.3645e-03, 4.4137e-06, 3.1761e-04, 2.3986e-02, 1.7951e-04,
        5.8066e-01, 4.5046e-02])

max probability is torch.return_types.max(
values=tensor(0.5807),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4323e-04, 9.3913e-09, 9.3862e-09, 9.3856e-09]),
indices=tensor([  0, 197,  91, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 869 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4634e-03, 1.5523e-03, 3.4842e-01,
2.4518e-01, 2.5523e-01, 7.8588e-02,
        5.7511e-02, 2.3454e-04, 3.4162e-04, 1.3305e-03, 6.2040e-04, 2.1487e-05,
        1.2068e-03, 1.7937e-03, 5.8169e-04, 2.0814e-04, 1.1845e-04, 7.3052e-04,
        9.9653e-04, 5.5294e-04])

max probability is torch.return_types.max(
values=tensor(0.3484),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.2531e-05, 3.4489e-06, 3.4483e-06, 3.4481e-06]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 870 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.5667e-09, 2.3798e-08, 1.2675e-07,
1.1374e-07, 8.2057e-07, 1.4685e-08,
        1.3886e-08, 4.2216e-06, 1.3752e-06, 1.0024e-06, 1.6617e-06, 5.2140e-10,
        4.7860e-08, 1.3769e-05, 4.2886e-11, 6.5225e-08, 8.7299e-04, 2.5356e-07,
        6.0110e-03, 9.9227e-01])

max probability is torch.return_types.max(
values=tensor(0.9923),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.2401e-04, 4.9024e-14, 4.9004e-14, 4.8942e-14]),
```

```
indices=tensor([  0, 655, 337, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 871 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9998e-01, 4.4588e-08, 9.0583e-06,
4.2008e-07, 1.3689e-06, 1.3967e-06,
        1.2203e-06, 5.3100e-08, 4.0336e-07, 3.3493e-06, 2.7379e-09, 1.3054e-08,
        2.0815e-07, 1.7242e-08, 7.2312e-09, 4.3367e-10, 8.5423e-09, 2.0014e-07,
        1.8968e-08, 3.4063e-08])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9380e-12, 5.4161e-15, 5.4145e-15, 5.4122e-15]),
indices=tensor([  0, 323, 910, 271]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 872 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.4950e-03, 1.6186e-03, 8.9713e-02,
1.7477e-01, 4.0885e-01, 2.3097e-01,
        7.8849e-02, 2.7487e-04, 6.2079e-04, 1.7762e-03, 5.5073e-04, 1.7975e-05,
        4.2536e-04, 1.8986e-03, 3.9122e-04, 3.1170e-04, 8.2451e-05, 6.0311e-04,
        2.7879e-03, 8.0338e-04])

max probability is torch.return_types.max(
values=tensor(0.4089),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.3455e-05, 2.2804e-06, 2.2789e-06, 2.2771e-06]),
indices=tensor([  0, 319, 316, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 873 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0043, 0.0017, 0.0662, 0.0171, 0.0672,
0.0235, 0.0190, 0.0254, 0.0400,
        0.0744, 0.0272, 0.0019, 0.0896, 0.1399, 0.0031, 0.0569, 0.0942, 0.0071,
        0.1837, 0.0479])
```

```
max probability is torch.return_types.max(
values=tensor(0.1837),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.8336e-04, 9.7471e-06, 9.7463e-06, 9.7454e-06]),
indices=tensor([  0, 771, 910, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 874 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.0563e-04, 5.0783e-05, 8.6844e-04,
9.5968e-04, 1.1375e-02, 5.4191e-04,
        1.9592e-04, 5.0560e-03, 3.7448e-03, 4.0815e-03, 6.1954e-04, 1.1984e-05,
        3.2587e-04, 1.1406e-02, 6.4136e-05, 5.6775e-04, 2.5760e-03, 1.6389e-03,
        8.4568e-01, 1.0937e-01])

max probability is torch.return_types.max(
values=tensor(0.8457),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.0411e-04, 6.4039e-08, 6.4028e-08, 6.4024e-08]),
indices=tensor([  0, 197,  91, 420]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 875 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4493e-04, 5.1690e-06, 3.0144e-04,
1.2455e-04, 6.8071e-04, 2.2039e-05,
        1.5926e-05, 5.6059e-03, 5.9068e-03, 1.4928e-02, 5.6297e-04, 1.3201e-05,
        8.0330e-04, 8.0196e-02, 2.6200e-05, 1.6079e-03, 5.4901e-02, 8.1355e-04,
        7.9534e-01, 3.7930e-02])

max probability is torch.return_types.max(
values=tensor(0.7953),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.9056e-05, 8.0655e-09, 8.0650e-09, 8.0622e-09]),
indices=tensor([  0, 914, 197, 263]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 876 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4052e-03, 6.1589e-05, 3.7769e-03,
3.4922e-03, 9.4848e-03, 3.0379e-03,
        2.6098e-03, 1.8598e-02, 6.1710e-02, 3.5545e-01, 3.7946e-03, 1.6570e-04,
        8.2267e-03, 2.3307e-01, 3.2055e-04, 8.3785e-03, 8.8476e-02, 2.0605e-03,
        1.6648e-01, 2.8429e-02])

max probability is torch.return_types.max(
values=tensor(0.3555),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6246e-04, 8.4421e-07, 8.4407e-07, 8.4393e-07]),
indices=tensor([  0, 914, 771, 896]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 877 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.9623e-02, 7.1242e-04, 4.6200e-03,
8.3802e-03, 1.0163e-02, 2.8329e-02,
        2.2700e-02, 1.3402e-02, 2.4739e-02, 8.3393e-01, 6.1430e-03, 1.4897e-04,
        1.7707e-03, 7.0053e-03, 1.7349e-04, 9.9743e-04, 1.5154e-03, 9.9389e-04,
        1.0410e-02, 3.6330e-03])

max probability is torch.return_types.max(
values=tensor(0.8339),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4291e-05, 6.1793e-07, 6.1769e-07, 6.1769e-07]),
indices=tensor([  0, 771, 896, 319]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 878 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.5428e-04, 6.0137e-04, 2.5971e-04,
5.3998e-05, 3.1504e-04, 1.4049e-04,
        5.4462e-05, 1.8800e-03, 4.5249e-03, 3.6512e-04, 1.6608e-01, 2.3016e-03,
        4.1583e-01, 1.3396e-02, 4.6883e-04, 3.5419e-01, 2.9786e-02, 5.5755e-04,
        5.6592e-03, 3.1532e-03])
```

```
max probability is torch.return_types.max(
values=tensor(0.4158),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.7116e-06, 1.2782e-07, 1.2775e-07, 1.2772e-07]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 879 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.4982e-03, 3.6117e-03, 2.1183e-03,
1.1022e-03, 4.8200e-03, 5.7931e-02,
        2.4988e-02, 5.5498e-02, 7.2258e-01, 3.7141e-02, 2.7447e-02, 8.1155e-05,
        2.4508e-02, 1.7329e-02, 2.0835e-05, 8.0556e-03, 1.2435e-03, 1.6785e-04,
        1.0879e-03, 1.4863e-03])

max probability is torch.return_types.max(
values=tensor(0.7226),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.8959e-06, 2.9734e-07, 2.9698e-07, 2.9696e-07]),
indices=tensor([  0, 319, 836, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 880 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.6104e-05, 1.0797e-04, 2.9021e-04,
2.7276e-04, 1.6314e-03, 2.8883e-04,
        9.7477e-05, 1.7145e-03, 1.9578e-03, 8.6650e-04, 5.0694e-03, 2.0808e-04,
        1.2066e-03, 7.2390e-03, 1.0907e-03, 5.9984e-03, 4.0561e-03, 1.1820e-02,
        9.0640e-01, 4.9194e-02])

max probability is torch.return_types.max(
values=tensor(0.9064),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5301e-04, 1.5534e-07, 1.5532e-07, 1.5529e-07]),
indices=tensor([  0, 466, 914, 337]))
End*****************************************************************************
```

```
**************************************************************************
Predicting Test Sample 881 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.2718e-04, 6.8658e-04, 1.3097e-02,
5.6168e-02, 1.9264e-01, 1.4768e-01,
        5.7137e-01, 1.7548e-04, 4.6384e-04, 3.2836e-03, 5.1385e-04, 3.0111e-05,
        7.7338e-04, 8.4653e-03, 1.1104e-03, 5.4505e-04, 1.4592e-04, 5.1655e-04,
        1.1239e-03, 4.1166e-04])

max probability is torch.return_types.max(
values=tensor(0.5714),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.9883e-06, 2.7687e-07, 2.7682e-07, 2.7679e-07]),
indices=tensor([  0, 910, 319, 316]))
End*************************************************************************

**************************************************************************
Predicting Test Sample 882 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.8583e-04, 5.9399e-05, 5.9690e-05,
1.2981e-04, 4.3301e-04, 5.6930e-04,
        7.3613e-04, 1.7989e-04, 1.6213e-04, 5.7031e-04, 1.6008e-04, 2.3185e-03,
        1.6351e-03, 1.6931e-03, 9.6109e-01, 6.2121e-04, 7.5279e-04, 2.3049e-02,
        4.5552e-03, 2.7781e-04])

max probability is torch.return_types.max(
values=tensor(0.9611),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6390e-06, 2.7245e-07, 2.7245e-07, 2.7237e-07]),
indices=tensor([  0, 323, 748, 910]))
End*************************************************************************

**************************************************************************
Predicting Test Sample 883 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.5562e-04, 1.4612e-05, 3.9576e-05,
4.1036e-05, 2.1646e-04, 4.3408e-05,
        3.0954e-05, 1.0373e-04, 1.9144e-04, 4.1367e-04, 3.7653e-03, 3.0812e-04,
        8.8899e-05, 2.3891e-02, 1.7649e-04, 4.6865e-04, 2.9998e-02, 9.2926e-02,
        2.7126e-01, 5.7468e-01])

max probability is torch.return_types.max(
```

```
values=tensor(0.5747),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0693e-03, 1.6225e-08, 1.6224e-08, 1.6218e-08]),
indices=tensor([  0, 158, 337, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 884 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0383, 0.0070, 0.0396, 0.0473, 0.0668,
0.3381, 0.1640, 0.0075, 0.0298,
        0.1039, 0.0167, 0.0049, 0.0353, 0.0296, 0.0053, 0.0094, 0.0066, 0.0054,
        0.0119, 0.0054])

max probability is torch.return_types.max(
values=tensor(0.3381),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6562e-04, 2.8101e-05, 2.8095e-05, 2.8095e-05]),
indices=tensor([  0, 771, 910, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 885 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.1583e-04, 9.8430e-05, 5.2507e-01,
3.1345e-01, 1.4197e-01, 1.2789e-02,
        6.2564e-03, 4.3046e-06, 2.7297e-06, 4.3565e-05, 4.3132e-06, 7.0756e-08,
        1.5458e-05, 3.0968e-05, 5.3919e-06, 3.1650e-07, 1.3149e-06, 9.1941e-06,
        1.4740e-05, 1.5769e-05])

max probability is torch.return_types.max(
values=tensor(0.5251),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.1400e-07, 4.6914e-09, 4.6910e-09, 4.6904e-09]),
indices=tensor([  0, 771, 910, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 886 :   Prediction is Correct?
```

```
Yes
first 20 classes probability: tensor([5.2042e-06, 4.5937e-07, 5.3888e-07,
2.1927e-06, 4.9696e-06, 3.2404e-06,
        2.7068e-06, 5.7895e-06, 1.2910e-06, 1.1010e-05, 3.4544e-06, 5.9488e-04,
        8.9303e-05, 2.3023e-05, 9.9660e-01, 4.7455e-06, 1.9401e-05, 1.6224e-03,
        9.9359e-04, 1.0674e-05])

max probability is torch.return_types.max(
values=tensor(0.9966),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.8381e-09, 4.3192e-10, 4.3172e-10, 4.3171e-10]),
indices=tensor([  0, 748, 255, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 887 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.2935e-04, 2.2785e-05, 3.1723e-01,
3.2219e-01, 3.0586e-01, 1.6056e-02,
        1.7851e-02, 5.4513e-05, 9.3392e-05, 2.0870e-03, 4.5389e-05, 1.2840e-07,
        9.3527e-05, 1.5006e-02, 2.3190e-05, 1.4565e-05, 3.5840e-05, 3.5142e-04,
        6.2769e-04, 1.6068e-03])

max probability is torch.return_types.max(
values=tensor(0.3222),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.6403e-06, 2.1289e-08, 2.1281e-08, 2.1279e-08]),
indices=tensor([  0, 771, 910, 726]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 888 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4125e-03, 6.7288e-04, 1.6936e-04,
4.4924e-05, 1.4900e-04, 3.0704e-04,
        1.6444e-04, 1.9044e-03, 1.9987e-03, 8.8288e-04, 1.2589e-01, 4.3286e-01,
        2.0354e-02, 1.2009e-02, 8.6429e-05, 7.6300e-03, 3.8083e-01, 5.2004e-04,
        4.9216e-03, 6.6879e-03])

max probability is torch.return_types.max(
values=tensor(0.4329),
indices=tensor(11))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.2795e-05, 4.3846e-07, 4.3829e-07, 4.3815e-07]),
indices=tensor([  0, 323, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 889 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.6034e-06, 2.7782e-05, 3.8477e-07,
2.0516e-08, 8.5907e-07, 7.3762e-07,
        2.0800e-06, 5.4999e-01, 4.4961e-01, 5.6434e-07, 9.2789e-07, 3.1782e-08,
        3.6605e-04, 1.8133e-07, 3.4729e-09, 2.0378e-07, 1.6777e-07, 4.0620e-09,
        1.1362e-07, 3.6656e-08])

max probability is torch.return_types.max(
values=tensor(0.5500),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2204e-11, 1.7055e-14, 1.7055e-14, 1.7054e-14]),
indices=tensor([  0, 836, 957, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 890 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.7138e-04, 1.9984e-04, 6.4830e-03,
1.7845e-01, 1.7240e-01, 4.6326e-01,
        1.7735e-01, 1.0730e-05, 2.4590e-05, 3.3875e-04, 2.3132e-05, 1.0242e-06,
        1.8547e-05, 4.5289e-04, 1.2905e-04, 1.5353e-05, 2.4037e-06, 3.2510e-05,
        3.3627e-04, 8.1756e-05])

max probability is torch.return_types.max(
values=tensor(0.4633),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.3162e-07, 2.3817e-08, 2.3799e-08, 2.3778e-08]),
indices=tensor([  0, 319, 316, 144]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 891 :   Prediction is Correct?
Yes
```

```
first 20 classes probability: tensor([9.7961e-01, 2.0104e-04, 2.0392e-04,
1.3600e-05, 1.9931e-04, 1.0498e-04,
        1.9540e-04, 1.7481e-04, 1.6435e-03, 1.9811e-04, 2.3867e-04, 1.8763e-04,
        1.5632e-02, 3.7422e-04, 2.0826e-04, 8.8502e-05, 9.3124e-05, 4.7608e-04,
        7.1725e-05, 8.2938e-05])

max probability is torch.return_types.max(
values=tensor(0.9796),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3296e-07, 6.7921e-09, 6.7921e-09, 6.7892e-09]),
indices=tensor([  0, 910, 323, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 892 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0215, 0.0224, 0.0335, 0.0042, 0.0340,
0.0112, 0.0111, 0.0119, 0.0303,
        0.0074, 0.0407, 0.0039, 0.0180, 0.0497, 0.0038, 0.1483, 0.1264, 0.0789,
        0.2484, 0.0813])

max probability is torch.return_types.max(
values=tensor(0.2484),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4016e-03, 1.2248e-05, 1.2248e-05, 1.2247e-05]),
indices=tensor([  0, 197, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 893 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.6139e-04, 1.5659e-06, 6.5561e-04,
8.7828e-04, 1.3277e-03, 1.3368e-03,
        5.9259e-04, 2.1121e-03, 6.7302e-03, 9.7714e-01, 6.6594e-05, 4.5170e-07,
        1.6754e-04, 1.9166e-03, 1.1873e-05, 8.7333e-05, 1.6637e-04, 4.5607e-05,
        5.5664e-03, 6.2937e-04])

max probability is torch.return_types.max(
values=tensor(0.9771),
indices=tensor(9))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([4.1712e-07, 1.8436e-09, 1.8436e-09, 1.8422e-09]),
indices=tensor([  0, 896, 771, 363]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 894 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0011, 0.0005, 0.0058, 0.0053, 0.0111,
0.0029, 0.0033, 0.0101, 0.0076,
        0.0285, 0.0773, 0.0551, 0.0845, 0.1054, 0.0311, 0.0141, 0.1338, 0.0305,
        0.3261, 0.0580])

max probability is torch.return_types.max(
values=tensor(0.3261),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.0746e-04, 7.6302e-06, 7.6261e-06, 7.6249e-06]),
indices=tensor([  0, 914, 323, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 895 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.0982e-10, 2.1966e-09, 4.2751e-12,
7.0114e-13, 3.1209e-12, 2.8058e-11,
        8.7507e-12, 1.0310e-08, 1.2644e-10, 1.7295e-11, 3.1713e-07, 9.9999e-01,
        2.8126e-06, 8.1544e-10, 2.6954e-08, 2.1846e-10, 5.8978e-06, 3.9713e-10,
        1.4879e-08, 1.0247e-09])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7357e-14, 8.0957e-19, 8.0886e-19, 8.0880e-19]),
indices=tensor([  0, 691,  82, 382]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 896 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.3356e-05, 1.2603e-05, 2.0680e-04,
5.5777e-05, 2.3848e-04, 2.1103e-05,
        2.2958e-05, 1.6363e-04, 2.1047e-04, 1.5431e-04, 1.1503e-03, 2.8438e-04,
```

```
        4.8224e-04, 3.0146e-02, 1.3119e-04, 7.8952e-04, 2.9529e-01, 9.6459e-03,
        1.2471e-01, 5.3387e-01])

max probability is torch.return_types.max(
values=tensor(0.5339),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3586e-03, 2.3223e-08, 2.3216e-08, 2.3213e-08]),
indices=tensor([  0, 230, 691, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 897 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.8138e-02, 2.9874e-01, 2.9222e-03,
9.1973e-04, 3.1024e-03, 6.4710e-03,
        4.4597e-03, 3.8519e-01, 2.3266e-01, 3.5373e-03, 4.5708e-03, 3.5910e-04,
        2.3203e-02, 9.6058e-04, 1.8880e-04, 1.1181e-02, 1.0393e-03, 9.0152e-05,
        8.2502e-04, 3.9150e-04])

max probability is torch.return_types.max(
values=tensor(0.3852),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2437e-05, 1.1153e-06, 1.1152e-06, 1.1151e-06]),
indices=tensor([  0, 319, 316, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 898 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0089, 0.0045, 0.1308, 0.0661, 0.2130,
0.0243, 0.0215, 0.0325, 0.0273,
        0.0242, 0.0037, 0.0003, 0.0060, 0.0347, 0.0027, 0.0027, 0.0125, 0.0196,
        0.2524, 0.0847])

max probability is torch.return_types.max(
values=tensor(0.2524),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.8332e-03, 2.4680e-05, 2.4678e-05, 2.4675e-05]),
indices=tensor([  0, 197,  91, 914]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 899 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0096, 0.0011, 0.0059, 0.0171, 0.0513,
0.5565, 0.2011, 0.0010, 0.0083,
        0.0705, 0.0093, 0.0013, 0.0121, 0.0227, 0.0031, 0.0087, 0.0049, 0.0016,
        0.0097, 0.0028])

max probability is torch.return_types.max(
values=tensor(0.5565),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2019e-05, 1.4873e-06, 1.4871e-06, 1.4871e-06]),
indices=tensor([  0, 319, 771, 896]))
End**************************************************************************

currently at 900 current time is 28.076111793518066
**************************************************************************
Predicting Test Sample 900 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.5025e-12, 4.2886e-11, 4.0444e-10,
3.5638e-10, 4.6601e-09, 2.2188e-11,
        1.9756e-11, 8.7322e-09, 4.1833e-09, 1.9774e-09, 3.1577e-08, 1.5012e-13,
        1.1031e-10, 9.2799e-07, 1.2039e-15, 2.5639e-10, 1.1615e-04, 4.9880e-10,
        1.1933e-04, 9.9969e-01])

max probability is torch.return_types.max(
values=tensor(0.9997),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.3745e-05, 1.1399e-18, 1.1391e-18, 1.1368e-18]),
indices=tensor([  0, 655, 337, 691]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 901 :   Prediction is Correct?
No
first 20 classes probability: tensor([4.4620e-04, 1.7159e-05, 8.4752e-05,
8.1080e-05, 4.8979e-04, 1.1614e-04,
        6.3001e-05, 1.3818e-03, 2.2328e-03, 3.3129e-03, 1.0242e-03, 4.4585e-04,
        9.4131e-04, 6.8266e-03, 5.4919e-02, 1.9986e-03, 2.4876e-03, 1.0340e-01,
        8.1229e-01, 7.2446e-03])
```

```
max probability is torch.return_types.max(
values=tensor(0.8123),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6871e-05, 1.8229e-07, 1.8226e-07, 1.8222e-07]),
indices=tensor([  0, 914, 337, 263]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 902 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1633e-01, 2.6876e-03, 2.2106e-01,
2.0902e-01, 1.0178e-01, 1.9083e-01,
        5.9182e-02, 2.1003e-03, 6.0638e-03, 7.1130e-02, 1.8166e-03, 3.8052e-05,
        2.3926e-03, 7.9584e-03, 2.4726e-04, 4.2919e-04, 3.2209e-04, 8.6158e-04,
        1.1968e-03, 2.5007e-03])

max probability is torch.return_types.max(
values=tensor(0.2211),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.3547e-05, 2.1441e-06, 2.1440e-06, 2.1422e-06]),
indices=tensor([  0, 319, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 903 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.5238e-03, 2.5105e-04, 8.6375e-03,
6.9606e-03, 1.4646e-02, 4.7882e-03,
        4.0123e-03, 1.4202e-03, 5.3673e-03, 1.1533e-02, 1.0606e-02, 1.7723e-04,
        8.5891e-03, 7.0649e-01, 5.1482e-04, 2.8818e-03, 4.2152e-02, 1.5476e-02,
        1.7724e-02, 1.2939e-01])

max probability is torch.return_types.max(
values=tensor(0.7065),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.5907e-04, 2.1909e-06, 2.1909e-06, 2.1902e-06]),
indices=tensor([  0, 914, 771, 323]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 904 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.4013e-03, 4.4792e-03, 2.0211e-01,
2.2472e-01, 1.7217e-01, 2.1367e-01,
        1.7074e-01, 1.8100e-04, 3.1073e-04, 1.3748e-03, 3.2241e-04, 3.2522e-05,
        5.4991e-04, 1.2609e-03, 6.7689e-04, 2.0868e-04, 8.4252e-05, 4.5301e-04,
        7.6014e-04, 3.5308e-04])

max probability is torch.return_types.max(
values=tensor(0.2247),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8668e-05, 2.2326e-06, 2.2319e-06, 2.2312e-06]),
indices=tensor([  0, 319, 316, 910]))
End********************************************************************************

**********************************************************************************
Predicting Test Sample 905 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.7877e-04, 1.1381e-05, 6.0652e-04,
8.6455e-04, 2.5188e-03, 1.4476e-04,
        8.9102e-05, 8.7685e-04, 4.5889e-04, 2.6592e-03, 2.2412e-03, 3.1920e-04,
        7.0696e-04, 2.4134e-02, 1.4671e-04, 2.7489e-04, 6.0411e-03, 6.6063e-03,
        6.0248e-01, 3.4756e-01])

max probability is torch.return_types.max(
values=tensor(0.6025),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.3024e-04, 5.3080e-08, 5.3070e-08, 5.3057e-08]),
indices=tensor([  0, 337, 914, 158]))
End********************************************************************************

**********************************************************************************
Predicting Test Sample 906 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.1996e-03, 5.6746e-04, 5.8703e-01,
7.3736e-02, 9.1138e-02, 2.3939e-02,
        1.0554e-01, 1.2468e-03, 2.9372e-03, 1.1937e-02, 5.4556e-03, 7.4872e-05,
        6.5848e-03, 6.6064e-02, 3.9289e-04, 1.0826e-03, 5.1106e-03, 3.7184e-03,
        3.4124e-03, 7.4881e-03])
```

```
max probability is torch.return_types.max(
values=tensor(0.5870),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.8782e-05, 1.3115e-06, 1.3111e-06, 1.3111e-06]),
indices=tensor([  0, 150, 726, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 907 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.2033e-03, 6.0712e-04, 2.2028e-02,
1.4297e-01, 2.3891e-01, 4.4590e-01,
        1.4296e-01, 6.4086e-05, 1.5039e-04, 8.6507e-04, 1.7242e-04, 1.1517e-05,
        1.3274e-04, 8.0232e-04, 6.5125e-04, 1.0486e-04, 1.7009e-05, 3.0734e-04,
        1.3977e-03, 2.7325e-04])

max probability is torch.return_types.max(
values=tensor(0.4459),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.1989e-06, 4.8192e-07, 4.8160e-07, 4.8125e-07]),
indices=tensor([  0, 319, 316, 144]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 908 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.9585e-03, 2.5937e-01, 7.6521e-04,
2.0657e-04, 1.4111e-03, 8.0476e-03,
        5.1643e-03, 1.6851e-01, 2.8065e-01, 1.9438e-03, 3.4825e-02, 5.0060e-03,
        1.3058e-01, 4.3418e-03, 5.8114e-04, 7.2254e-02, 9.3546e-03, 2.8391e-04,
        2.3560e-03, 1.6470e-03])

max probability is torch.return_types.max(
values=tensor(0.2806),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.1097e-05, 4.9895e-06, 4.9894e-06, 4.9883e-06]),
indices=tensor([  0, 319, 836, 316]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 909 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.3161e-04, 7.7355e-05, 6.7682e-04,
1.6276e-04, 1.1405e-03, 2.0478e-05,
        1.3457e-05, 9.3442e-03, 3.1691e-03, 6.6628e-04, 2.2492e-03, 2.5942e-04,
        1.1798e-02, 2.4268e-02, 8.8850e-04, 7.9182e-03, 9.8382e-02, 3.2496e-03,
        7.8446e-01, 5.0788e-02])

max probability is torch.return_types.max(
values=tensor(0.7845),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4978e-04, 8.9250e-08, 8.9245e-08, 8.9229e-08]),
indices=tensor([  0, 197, 420, 263]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 910 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.4672e-03, 1.3454e-03, 6.7033e-02,
6.4778e-02, 1.7228e-01, 2.2020e-01,
        2.3540e-01, 6.4609e-03, 2.1144e-02, 1.0122e-01, 4.7314e-03, 1.8632e-04,
        5.3233e-03, 2.3668e-02, 2.4470e-03, 4.9386e-03, 3.3689e-03, 6.0707e-03,
        3.1767e-02, 1.4166e-02])

max probability is torch.return_types.max(
values=tensor(0.2354),
indices=tensor(6))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.9492e-04, 1.0206e-05, 1.0205e-05, 1.0202e-05]),
indices=tensor([  0, 896, 771, 316]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 911 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7648e-03, 2.8487e-04, 2.6408e-02,
4.5074e-02, 5.6113e-02, 3.1229e-02,
        8.9197e-02, 1.0399e-02, 3.0491e-02, 5.5814e-01, 1.8433e-02, 2.2920e-04,
        2.1506e-02, 7.8618e-02, 7.1939e-04, 2.3907e-03, 5.3644e-03, 3.3190e-03,
        8.4545e-03, 1.0273e-02])

max probability is torch.return_types.max(
```

```
values=tensor(0.5581),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.1101e-05, 1.6040e-06, 1.6031e-06, 1.6030e-06]),
indices=tensor([  0, 771, 323, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 912 :   Prediction is Correct?
No
first 20 classes probability: tensor([6.7749e-04, 9.2969e-05, 9.6896e-03,
3.1035e-01, 1.3861e-01, 8.9557e-02,
        2.3430e-01, 4.7576e-04, 6.8737e-04, 5.2005e-02, 3.3075e-04, 2.6987e-05,
        5.9617e-04, 1.3499e-01, 9.5804e-03, 1.1057e-04, 2.2815e-04, 4.1684e-03,
        9.2656e-03, 4.0435e-03])

max probability is torch.return_types.max(
values=tensor(0.3104),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2273e-05, 2.1462e-07, 2.1460e-07, 2.1457e-07]),
indices=tensor([  0, 323, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 913 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.5997e-06, 9.3149e-05, 6.7216e-08,
9.4178e-09, 4.4891e-07, 7.9078e-07,
        2.1121e-07, 2.5134e-04, 4.0901e-04, 2.4778e-06, 8.4260e-03, 4.7049e-06,
        7.4171e-04, 9.1317e-05, 4.7988e-07, 9.8908e-01, 7.6282e-04, 1.6473e-06,
        7.3116e-05, 5.6879e-05])

max probability is torch.return_types.max(
values=tensor(0.9891),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.0051e-09, 7.1746e-12, 7.1623e-12, 7.1568e-12]),
indices=tensor([  0, 319, 316, 836]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 914 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.7519e-04, 3.6195e-04, 1.0907e-02,
1.0953e-01, 2.0765e-01, 5.6471e-01,
        1.0523e-01, 8.8230e-06, 2.4821e-05, 9.8411e-05, 2.7220e-05, 1.8023e-06,
        2.2661e-05, 1.5450e-04, 7.7841e-05, 1.1248e-05, 1.8736e-06, 2.9766e-05,
        2.9507e-04, 5.0448e-05])

max probability is torch.return_types.max(
values=tensor(0.5647),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.4825e-07, 3.0043e-08, 3.0022e-08, 3.0004e-08]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 915 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.6907e-03, 6.1812e-01, 6.2096e-05,
1.5134e-05, 1.2899e-04, 2.5118e-04,
        2.1154e-04, 3.0035e-01, 6.7840e-02, 5.2336e-05, 4.7749e-04, 2.9602e-04,
        8.6267e-03, 3.6644e-05, 4.1353e-05, 4.9036e-04, 2.2649e-04, 3.0237e-06,
        6.3537e-05, 1.3623e-05])

max probability is torch.return_types.max(
values=tensor(0.6181),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5602e-07, 6.4388e-09, 6.4380e-09, 6.4374e-09]),
indices=tensor([  0, 319, 316, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 916 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.4122e-04, 2.7013e-03, 1.8002e-05,
5.3213e-06, 4.1789e-05, 3.6954e-05,
        6.0863e-05, 8.1764e-01, 1.4562e-01, 4.6097e-04, 1.9335e-03, 1.0300e-02,
        1.7275e-02, 1.2207e-04, 3.5432e-05, 1.0328e-04, 1.2046e-03, 2.1038e-05,
        1.2679e-03, 1.9317e-04])

max probability is torch.return_types.max(
values=tensor(0.8176),
```

```
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0276e-06, 1.1014e-08, 1.1013e-08, 1.1011e-08]),
indices=tensor([  0, 316, 294, 181]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 917 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0013, 0.0004, 0.0011, 0.0010, 0.0032,
0.0022, 0.0010, 0.0057, 0.0082,
        0.0063, 0.0408, 0.0212, 0.0128, 0.1188, 0.0014, 0.0119, 0.1977, 0.0147,
        0.3186, 0.2222])

max probability is torch.return_types.max(
values=tensor(0.3186),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5475e-03, 5.9230e-06, 5.9177e-06, 5.9168e-06]),
indices=tensor([  0, 914, 771, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 918 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.8257e-04, 8.4907e-04, 3.4648e-03,
2.5634e-03, 1.3567e-02, 2.1204e-03,
        9.7312e-04, 3.2930e-02, 2.7210e-02, 1.0171e-02, 8.3423e-03, 5.2453e-04,
        6.0688e-03, 1.4387e-02, 1.5910e-03, 8.5421e-03, 8.0305e-03, 1.0291e-02,
        8.0859e-01, 3.6797e-02])

max probability is torch.return_types.max(
values=tensor(0.8086),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.8206e-04, 1.7122e-06, 1.7121e-06, 1.7121e-06]),
indices=tensor([  0, 466, 914, 197]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 919 :   Prediction is Correct?
No
```

```
first 20 classes probability: tensor([1.3778e-03, 6.7141e-04, 1.5816e-02,
1.1401e-01, 2.1627e-01, 5.2263e-01,
        1.2727e-01, 2.5169e-05, 6.7060e-05, 2.4334e-04, 7.0540e-05, 6.2897e-06,
        6.7597e-05, 3.3722e-04, 1.8928e-04, 3.6534e-05, 6.7667e-06, 8.0308e-05,
        5.7764e-04, 1.1195e-04])

max probability is torch.return_types.max(
values=tensor(0.5226),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4992e-06, 1.4753e-07, 1.4744e-07, 1.4735e-07]),
indices=tensor([  0, 319, 316, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 920 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.1983e-02, 1.6764e-02, 3.5586e-03,
7.8501e-04, 4.5288e-03, 5.5231e-03,
        1.6457e-02, 4.0629e-01, 5.0381e-01, 3.0096e-03, 1.9279e-03, 1.1625e-03,
        1.5508e-02, 1.5155e-03, 1.2944e-04, 1.3868e-03, 2.9232e-03, 2.8993e-04,
        1.0292e-03, 5.5411e-04])

max probability is torch.return_types.max(
values=tensor(0.5038),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2386e-05, 8.8927e-07, 8.8901e-07, 8.8889e-07]),
indices=tensor([  0, 957, 910, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 921 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.6062e-03, 4.4318e-04, 1.6185e-03,
2.5672e-03, 7.9871e-03, 1.9426e-02,
        6.0987e-03, 3.8587e-02, 1.6822e-01, 2.3937e-01, 4.9130e-02, 3.7872e-04,
        9.1166e-03, 1.4205e-01, 1.5805e-04, 1.6104e-02, 2.7097e-02, 2.8712e-03,
        1.6680e-01, 9.7236e-02])

max probability is torch.return_types.max(
values=tensor(0.2394),
indices=tensor(9))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.5178e-04, 1.6729e-06, 1.6719e-06, 1.6716e-06]),
indices=tensor([  0, 771, 914, 896]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 922 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0126, 0.0030, 0.0044, 0.0040, 0.0158,
0.0641, 0.0273, 0.0045, 0.0359,
        0.0267, 0.1738, 0.0117, 0.1271, 0.2764, 0.0065, 0.0668, 0.0484, 0.0189,
        0.0275, 0.0277])

max probability is torch.return_types.max(
values=tensor(0.2764),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.2187e-04, 1.7550e-05, 1.7538e-05, 1.7538e-05]),
indices=tensor([  0, 771, 323, 319]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 923 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.3431e-02, 3.5222e-03, 6.1282e-01,
7.8170e-02, 6.2232e-02, 4.7870e-02,
        7.7255e-02, 3.2920e-03, 9.7158e-03, 2.2890e-02, 1.8608e-03, 7.4552e-05,
        1.2938e-02, 4.6960e-03, 1.0013e-03, 7.9374e-04, 3.3909e-04, 2.7543e-03,
        7.8675e-04, 1.1646e-03])

max probability is torch.return_types.max(
values=tensor(0.6128),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0229e-05, 2.5080e-06, 2.5076e-06, 2.5065e-06]),
indices=tensor([  0, 771, 910, 572]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 924 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7592e-05, 4.7477e-08, 4.8796e-07,
4.5537e-07, 7.1169e-06, 7.6880e-07,
```

```
        6.4404e-07, 1.4343e-06, 2.1621e-06, 4.9453e-06, 1.5381e-05, 3.3338e-05,
        2.7538e-06, 6.9766e-05, 2.6915e-02, 5.3648e-06, 2.0190e-05, 9.0595e-01,
        6.6221e-02, 7.3065e-04])

max probability is torch.return_types.max(
values=tensor(0.9060),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.5754e-08, 7.4619e-12, 7.4604e-12, 7.4531e-12]),
indices=tensor([  0, 337, 748, 158]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 925 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.6461e-03, 1.0467e-03, 2.0918e-03,
4.0605e-04, 1.3873e-03, 6.1816e-04,
        3.3086e-04, 3.1426e-03, 3.5135e-03, 1.3586e-03, 5.3289e-02, 6.2898e-02,
        3.6291e-02, 7.3666e-02, 4.5185e-03, 3.3835e-02, 4.9387e-01, 2.5589e-02,
        9.3000e-02, 1.0117e-01])

max probability is torch.return_types.max(
values=tensor(0.4939),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2249e-03, 4.3082e-06, 4.3082e-06, 4.3079e-06]),
indices=tensor([  0, 323, 914, 230]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 926 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0918, 0.0018, 0.0352, 0.0213, 0.0617,
0.0754, 0.0633, 0.0027, 0.0196,
        0.0613, 0.0337, 0.0008, 0.0185, 0.4037, 0.0015, 0.0115, 0.0121, 0.0263,
        0.0169, 0.0344])

max probability is torch.return_types.max(
values=tensor(0.4037),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2658e-04, 6.8038e-06, 6.7999e-06, 6.7995e-06]),
```

```
indices=tensor([  0, 771, 323, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 927 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2851e-03, 2.6697e-04, 4.2155e-01,
2.5605e-01, 2.4629e-01, 3.7719e-02,
        2.3819e-02, 4.8719e-04, 4.4508e-04, 8.2374e-03, 1.1310e-04, 4.6734e-06,
        2.3198e-04, 8.2520e-04, 1.3113e-04, 2.2158e-05, 6.7637e-05, 3.4504e-04,
        1.2147e-03, 4.1326e-04])

max probability is torch.return_types.max(
values=tensor(0.4216),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4187e-05, 4.8740e-07, 4.8731e-07, 4.8719e-07]),
indices=tensor([  0, 771,  91, 466]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 928 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.9689e-03, 7.6024e-01, 1.0297e-03,
4.4447e-04, 2.2661e-03, 7.5598e-04,
        9.2191e-04, 2.0628e-02, 4.6411e-03, 4.4111e-04, 1.4673e-02, 1.8957e-03,
        1.7336e-02, 2.5117e-03, 2.2699e-03, 1.4832e-01, 1.2604e-02, 5.5094e-04,
        3.3900e-03, 1.4637e-03])

max probability is torch.return_types.max(
values=tensor(0.7602),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9608e-05, 6.7165e-07, 6.7128e-07, 6.7124e-07]),
indices=tensor([  0, 316, 910, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 929 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.7206e-04, 8.6653e-05, 4.9018e-04,
1.8581e-04, 2.1908e-03, 1.2770e-04,
        6.6451e-05, 8.0160e-02, 2.0480e-02, 7.9742e-03, 5.0130e-04, 1.9267e-04,
        1.1242e-03, 2.7633e-03, 4.0793e-04, 8.2441e-04, 7.6012e-03, 1.9588e-03,
```

```
        8.5910e-01, 1.3306e-02])

max probability is torch.return_types.max(
values=tensor(0.8591),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.6789e-05, 3.2955e-08, 3.2931e-08, 3.2926e-08]),
indices=tensor([  0, 197, 263,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 930 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0132, 0.0013, 0.0010, 0.0018, 0.0082,
0.1803, 0.0171, 0.0033, 0.0654,
        0.0481, 0.2861, 0.0032, 0.0790, 0.1654, 0.0005, 0.0405, 0.0227, 0.0029,
        0.0296, 0.0290])

max probability is torch.return_types.max(
values=tensor(0.2861),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0636e-04, 1.4939e-06, 1.4935e-06, 1.4922e-06]),
indices=tensor([  0, 771, 319, 144]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 931 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1331e-03, 1.1780e-03, 2.2562e-04,
1.0141e-04, 7.8397e-04, 1.9841e-03,
        6.3957e-04, 2.3822e-03, 1.9742e-02, 3.3005e-03, 2.3574e-01, 4.8873e-04,
        3.3818e-02, 4.2184e-02, 8.7911e-05, 6.0021e-01, 4.0533e-02, 6.5493e-04,
        5.1973e-03, 9.1962e-03])

max probability is torch.return_types.max(
values=tensor(0.6002),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.1009e-05, 4.1613e-07, 4.1553e-07, 4.1532e-07]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 932 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.2849e-03, 8.2319e-04, 1.0861e-03,
3.6542e-04, 2.6103e-03, 5.1711e-04,
        1.9252e-04, 6.1398e-02, 7.2854e-02, 2.0938e-02, 1.5638e-02, 8.5037e-04,
        1.9292e-02, 2.5538e-02, 1.2895e-03, 1.2033e-01, 8.0053e-02, 5.0331e-03,
        5.3665e-01, 2.4585e-02])

max probability is torch.return_types.max(
values=tensor(0.5367),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7967e-04, 1.5810e-06, 1.5801e-06, 1.5801e-06]),
indices=tensor([  0, 316, 263, 896]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 933 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0095, 0.0036, 0.0112, 0.0054, 0.0298,
0.3337, 0.1355, 0.0156, 0.2772,
        0.0268, 0.0198, 0.0004, 0.0362, 0.0605, 0.0009, 0.0087, 0.0037, 0.0044,
        0.0065, 0.0073])

max probability is torch.return_types.max(
values=tensor(0.3337),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.0318e-05, 3.4243e-06, 3.4237e-06, 3.4226e-06]),
indices=tensor([  0, 319, 771, 896]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 934 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.3045e-04, 8.6074e-05, 6.6100e-03,
8.0593e-03, 9.8817e-03, 2.7734e-03,
        1.6348e-03, 3.5135e-03, 2.5686e-03, 2.8740e-02, 6.6449e-03, 4.0051e-03,
        1.5863e-02, 7.1327e-02, 1.0069e-01, 8.6981e-03, 2.4503e-02, 3.4697e-02,
        6.2822e-01, 3.8228e-02])

max probability is torch.return_types.max(
```

```
values=tensor(0.6282),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4729e-04, 2.9239e-06, 2.9234e-06, 2.9220e-06]),
indices=tensor([  0, 914, 337, 197]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 935 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4871e-06, 4.9576e-10, 4.0226e-10,
6.4708e-09, 3.4688e-08, 4.3866e-08,
        2.7680e-08, 1.6425e-08, 3.4077e-09, 1.7397e-07, 4.1526e-09, 1.0708e-05,
        7.7408e-07, 5.4207e-08, 9.9978e-01, 9.5178e-09, 2.1018e-08, 1.8080e-04,
        2.1049e-05, 2.1798e-08])

max probability is torch.return_types.max(
values=tensor(0.9998),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.9190e-13, 1.3320e-14, 1.3316e-14, 1.3315e-14]),
indices=tensor([  0, 255, 323, 812]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 936 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0854, 0.0008, 0.0142, 0.0065, 0.0187,
0.0886, 0.0913, 0.0092, 0.1096,
        0.4623, 0.0092, 0.0008, 0.0450, 0.0363, 0.0007, 0.0042, 0.0090, 0.0026,
        0.0023, 0.0020])

max probability is torch.return_types.max(
values=tensor(0.4623),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.3875e-05, 1.4444e-06, 1.4438e-06, 1.4437e-06]),
indices=tensor([  0, 771, 319, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 937 :   Prediction is Correct?
```

```
No
first 20 classes probability: tensor([9.9145e-04, 3.3540e-02, 1.4129e-03,
3.0243e-04, 9.9924e-04, 1.1211e-03,
        1.6375e-03, 2.4541e-02, 2.8858e-02, 1.6541e-04, 3.2148e-02, 6.9182e-03,
        8.6141e-01, 1.4738e-03, 1.7721e-04, 1.7477e-03, 1.6267e-03, 8.9429e-05,
        2.0801e-04, 3.3574e-04])

max probability is torch.return_types.max(
values=tensor(0.8614),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.5834e-06, 3.0451e-07, 3.0436e-07, 3.0423e-07]),
indices=tensor([  0,  44, 319, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 938 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0691e-06, 2.9229e-06, 7.9472e-08,
3.1357e-08, 1.4200e-07, 2.9137e-07,
        2.4644e-07, 1.9155e-05, 1.9048e-06, 4.8917e-07, 2.2730e-04, 9.9828e-01,
        2.9307e-04, 4.7070e-06, 1.7169e-05, 4.5280e-06, 1.0780e-03, 3.9224e-06,
        5.8745e-05, 9.2569e-06])

max probability is torch.return_types.max(
values=tensor(0.9983),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7369e-08, 1.8600e-11, 1.8600e-11, 1.8598e-11]),
indices=tensor([  0, 914, 323, 382]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 939 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0617e-03, 8.7259e-04, 6.9102e-02,
2.7201e-01, 3.5899e-01, 2.3347e-01,
        6.1451e-02, 3.8766e-05, 4.6049e-05, 2.8481e-04, 7.2046e-05, 4.6985e-06,
        6.2053e-05, 4.0062e-04, 2.6005e-04, 3.5256e-05, 1.0436e-05, 1.6260e-04,
        1.0436e-03, 2.1251e-04])

max probability is torch.return_types.max(
values=tensor(0.3590),
indices=tensor(4))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.7314e-06, 4.3161e-07, 4.3141e-07, 4.3110e-07]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 940 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.7678e-03, 3.8070e-03, 1.5436e-01,
2.1092e-01, 3.3124e-01, 1.6088e-01,
        1.0621e-01, 4.9111e-04, 7.1865e-04, 1.6237e-03, 8.1782e-04, 8.4976e-05,
        1.2626e-03, 2.3094e-03, 1.3725e-03, 4.8268e-04, 2.2678e-04, 1.0660e-03,
        3.0913e-03, 9.3867e-04])

max probability is torch.return_types.max(
values=tensor(0.3312),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0667e-04, 1.5936e-05, 1.5933e-05, 1.5923e-05]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 941 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.0989e-04, 1.6314e-05, 6.0868e-04,
7.3346e-04, 4.1116e-03, 1.1638e-03,
        2.3261e-04, 4.6128e-02, 1.0453e-01, 1.7058e-01, 6.7266e-04, 8.5793e-07,
        4.6267e-04, 1.4424e-02, 1.0134e-05, 9.8045e-04, 1.5675e-03, 2.2482e-04,
        6.2986e-01, 2.3456e-02])

max probability is torch.return_types.max(
values=tensor(0.6299),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3408e-05, 5.2939e-09, 5.2911e-09, 5.2878e-09]),
indices=tensor([  0, 896, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 942 :   Prediction is Correct?
Yes
```

```
first 20 classes probability: tensor([6.9557e-02, 1.9123e-06, 2.2596e-04,
1.2114e-05, 4.2395e-04, 1.7270e-05,
        2.9455e-05, 1.6829e-05, 6.8774e-05, 7.2117e-05, 7.0134e-05, 1.2968e-04,
        1.7384e-04, 2.7874e-04, 5.2179e-03, 1.1906e-05, 9.0006e-05, 9.0844e-01,
        1.4396e-02, 7.6490e-04])

max probability is torch.return_types.max(
values=tensor(0.9084),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5834e-07, 1.9019e-10, 1.9004e-10, 1.9001e-10]),
indices=tensor([  0, 748, 158, 428]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 943 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0233, 0.0075, 0.0284, 0.0246, 0.0506,
0.0779, 0.0745, 0.0397, 0.1006,
        0.1140, 0.0355, 0.0080, 0.0630, 0.0988, 0.0032, 0.0193, 0.0646, 0.0117,
        0.0285, 0.0456])

max probability is torch.return_types.max(
values=tensor(0.1140),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8104e-03, 8.2366e-05, 8.2360e-05, 8.2354e-05]),
indices=tensor([  0, 771, 914, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 944 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0014, 0.0022, 0.1372, 0.1617, 0.1083,
0.0318, 0.0426, 0.0015, 0.0020,
        0.0095, 0.0265, 0.0008, 0.0702, 0.3391, 0.0041, 0.0066, 0.0148, 0.0040,
        0.0131, 0.0194])

max probability is torch.return_types.max(
values=tensor(0.3391),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([1.8456e-04, 2.9854e-06, 2.9848e-06, 2.9841e-06]),
indices=tensor([  0, 910, 323, 914]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 945 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0088, 0.0667, 0.2141, 0.1579, 0.2017,
0.0689, 0.0638, 0.0040, 0.0040,
        0.0047, 0.0175, 0.0012, 0.0240, 0.0092, 0.0039, 0.0082, 0.0028, 0.0029,
        0.0060, 0.0029])

max probability is torch.return_types.max(
values=tensor(0.2141),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([0.0005, 0.0001, 0.0001, 0.0001]),
indices=tensor([  0, 319, 316, 910]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 946 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.0470e-03, 1.1572e-02, 1.0410e-03,
3.3035e-04, 1.6455e-03, 1.1061e-03,
        7.5860e-04, 8.8931e-03, 1.2294e-02, 2.0247e-03, 1.8475e-01, 2.5303e-03,
        4.7262e-02, 2.6219e-02, 6.7373e-04, 6.1715e-01, 3.9626e-02, 1.8339e-03,
        1.9421e-02, 1.5575e-02])

max probability is torch.return_types.max(
values=tensor(0.6171),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4638e-04, 3.3043e-06, 3.3020e-06, 3.3016e-06]),
indices=tensor([  0, 319, 316, 771]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 947 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3495e-04, 1.1422e-05, 1.2695e-04,
1.3915e-04, 7.1077e-04, 3.7978e-05,
        2.3506e-05, 2.5821e-04, 1.6715e-04, 1.3488e-04, 1.1935e-03, 1.0526e-04,
        2.7400e-04, 5.4217e-03, 3.9454e-05, 4.9409e-05, 1.5980e-03, 9.9042e-03,
```

```
        4.2650e-01, 5.5226e-01])

max probability is torch.return_types.max(
values=tensor(0.5523),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.0263e-04, 4.5385e-09, 4.5381e-09, 4.5369e-09]),
indices=tensor([  0, 337, 158, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 948 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4474e-03, 8.9258e-03, 1.2689e-03,
4.3668e-04, 2.2163e-03, 1.2297e-03,
        9.5103e-04, 1.4438e-01, 1.2782e-01, 7.1391e-03, 9.8473e-02, 4.1696e-04,
        1.0959e-01, 1.2810e-02, 3.1044e-04, 4.4978e-01, 1.2896e-02, 5.2867e-04,
        9.3608e-03, 8.0733e-03])

max probability is torch.return_types.max(
values=tensor(0.4498),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.6513e-05, 2.0043e-06, 2.0035e-06, 2.0035e-06]),
indices=tensor([  0, 319, 836, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 949 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.6164e-04, 2.6454e-03, 5.7586e-04,
2.0125e-04, 9.0504e-04, 5.5393e-04,
        2.4347e-04, 3.0483e-03, 5.2065e-03, 9.7494e-04, 1.6148e-01, 4.6371e-04,
        6.1824e-02, 1.5223e-02, 2.5052e-04, 6.9662e-01, 2.0397e-02, 7.0250e-04,
        1.0866e-02, 1.6827e-02])

max probability is torch.return_types.max(
values=tensor(0.6966),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.5652e-05, 6.1227e-07, 6.1156e-07, 6.1140e-07]),
indices=tensor([  0, 319, 771, 316]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 950 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.8798e-04, 2.3345e-05, 8.7304e-04,
7.6623e-04, 2.1207e-03, 6.7978e-04,
        7.5494e-04, 4.0528e-02, 8.1281e-02, 6.9920e-01, 9.6102e-04, 3.6258e-05,
        1.3631e-03, 1.4381e-02, 1.7336e-04, 1.4464e-03, 5.2623e-03, 1.1092e-03,
        1.4458e-01, 3.4010e-03])

max probability is torch.return_types.max(
values=tensor(0.6992),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0293e-05, 6.0570e-08, 6.0523e-08, 6.0521e-08]),
indices=tensor([  0, 896, 771, 323]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 951 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1250e-03, 8.7383e-04, 1.5717e-03,
3.0667e-04, 1.0459e-03, 3.2222e-04,
        3.0637e-04, 9.4092e-04, 4.4427e-03, 7.9048e-04, 3.7287e-01, 5.3205e-04,
        2.7911e-01, 1.7077e-01, 1.9944e-04, 1.4121e-01, 1.5470e-02, 1.4279e-03,
        1.4499e-03, 5.0531e-03])

max probability is torch.return_types.max(
values=tensor(0.3729),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0819e-05, 1.9341e-07, 1.9339e-07, 1.9323e-07]),
indices=tensor([  0, 319, 771, 910]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 952 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.8598e-04, 6.0247e-05, 5.9193e-04,
3.2185e-04, 8.4501e-04, 1.3586e-04,
        7.6599e-05, 2.0533e-04, 5.6185e-04, 4.2076e-04, 9.8776e-03, 6.2322e-05,
        1.9488e-03, 3.4243e-01, 7.1503e-06, 1.0787e-03, 1.2035e-01, 1.4054e-03,
        2.0722e-02, 4.9698e-01])
```

```
max probability is torch.return_types.max(
values=tensor(0.4970),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6670e-03, 5.6026e-08, 5.5974e-08, 5.5974e-08]),
indices=tensor([  0, 914, 337, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 953 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.1330e-03, 1.5980e-02, 8.3554e-03,
4.7715e-03, 6.1959e-03, 8.7125e-01,
        7.7359e-02, 1.2641e-04, 2.1767e-03, 4.1696e-04, 8.9189e-04, 1.0482e-04,
        3.9286e-03, 4.1728e-04, 5.5896e-05, 4.6656e-04, 5.9505e-05, 2.9824e-05,
        1.2931e-04, 8.4515e-05])

max probability is torch.return_types.max(
values=tensor(0.8713),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0123e-06, 6.5299e-08, 6.5238e-08, 6.5231e-08]),
indices=tensor([  0, 319, 316,  44]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 954 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0085, 0.0032, 0.0164, 0.0034, 0.0133,
0.0191, 0.0234, 0.0583, 0.3484,
        0.0261, 0.0674, 0.0020, 0.2940, 0.0455, 0.0010, 0.0255, 0.0174, 0.0053,
        0.0076, 0.0092])

max probability is torch.return_types.max(
values=tensor(0.3484),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2385e-04, 5.3175e-06, 5.3158e-06, 5.3158e-06]),
indices=tensor([  0, 771, 910, 319]))
End*****************************************************************************
```

```
**************************************************************************
Predicting Test Sample 955 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.3399e-01, 1.0138e-01, 6.5773e-03,
1.5360e-03, 3.9182e-03, 4.8944e-03,
        1.0146e-02, 3.9704e-01, 3.3459e-01, 1.0710e-03, 1.1438e-04, 3.7760e-05,
        4.4638e-03, 6.5915e-05, 3.8431e-05, 3.4200e-05, 2.4116e-05, 1.6027e-05,
        3.7472e-05, 1.5714e-05])

max probability is torch.return_types.max(
values=tensor(0.3970),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7470e-07, 6.2377e-09, 6.2330e-09, 6.2319e-09]),
indices=tensor([  0, 319, 316, 836]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 956 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.1305e-05, 5.4536e-05, 7.7335e-04,
4.2134e-04, 1.0579e-03, 1.0601e-04,
        1.0252e-04, 6.6445e-04, 1.1821e-03, 8.6221e-04, 3.1478e-03, 1.2721e-05,
        7.3876e-04, 1.3671e-01, 2.9656e-06, 6.5748e-04, 8.0261e-02, 6.6773e-04,
        3.6614e-02, 7.3098e-01])

max probability is torch.return_types.max(
values=tensor(0.7310),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.8272e-03, 6.6202e-08, 6.6196e-08, 6.6191e-08]),
indices=tensor([  0, 914, 655, 337]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 957 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.1622e-05, 2.8533e-04, 3.6362e-01,
2.3375e-01, 3.7184e-01, 1.8448e-02,
        1.1156e-02, 1.8710e-05, 1.2121e-05, 6.5173e-05, 3.3911e-05, 5.1546e-07,
        1.0562e-04, 1.0955e-04, 6.9068e-05, 7.2374e-06, 6.9716e-06, 7.5956e-05,
        2.0250e-04, 4.6189e-05])

max probability is torch.return_types.max(
```

```
values=tensor(0.3718),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6559e-06, 5.2950e-08, 5.2950e-08, 5.2949e-08]),
indices=tensor([  0, 316, 466, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 958 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.3341e-04, 1.6540e-05, 7.5082e-04,
4.5857e-04, 1.9082e-03, 1.6929e-04,
        1.6240e-04, 1.8446e-02, 1.6294e-02, 6.3587e-02, 1.0492e-03, 2.5778e-05,
        5.9330e-04, 1.4610e-02, 9.3775e-05, 1.0677e-03, 1.4835e-02, 1.1574e-03,
        8.2802e-01, 3.6400e-02])

max probability is torch.return_types.max(
values=tensor(0.8280),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6066e-04, 5.7724e-08, 5.7723e-08, 5.7721e-08]),
indices=tensor([  0, 914, 197,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 959 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0029, 0.0041, 0.0042, 0.0018, 0.0070,
0.0282, 0.0110, 0.0025, 0.0209,
        0.0068, 0.3685, 0.0033, 0.2080, 0.0828, 0.0010, 0.1920, 0.0353, 0.0018,
        0.0062, 0.0080])

max probability is torch.return_types.max(
values=tensor(0.3685),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.8512e-05, 3.7850e-06, 3.7826e-06, 3.7791e-06]),
indices=tensor([  0, 319, 771, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 960 :   Prediction is Correct?
```

```
Yes
first 20 classes probability: tensor([0.0492, 0.0481, 0.0046, 0.0015, 0.0076,
0.0818, 0.0224, 0.0161, 0.1495,
        0.0064, 0.1445, 0.0117, 0.2661, 0.0869, 0.0013, 0.0406, 0.0239, 0.0032,
        0.0073, 0.0112])

max probability is torch.return_types.max(
values=tensor(0.2661),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.6423e-04, 1.6811e-05, 1.6804e-05, 1.6804e-05]),
indices=tensor([  0, 319, 771,  44]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 961 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.5113e-05, 5.0057e-06, 7.4130e-01,
1.7938e-01, 7.3928e-02, 2.3367e-03,
        2.7046e-03, 4.7372e-06, 3.7697e-06, 2.5284e-04, 2.1286e-06, 2.3408e-09,
        1.1608e-05, 2.7698e-05, 5.6828e-07, 1.5506e-07, 7.6854e-07, 2.5276e-06,
        5.5205e-06, 8.3012e-06])

max probability is torch.return_types.max(
values=tensor(0.7413),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.7934e-08, 2.4261e-10, 2.4250e-10, 2.4250e-10]),
indices=tensor([  0, 771, 910,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 962 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.7924e-03, 6.9117e-01, 1.8271e-02,
5.5229e-03, 1.7958e-02, 2.1812e-02,
        2.6598e-02, 3.6076e-03, 7.0012e-03, 1.0762e-03, 5.8643e-02, 7.5739e-04,
        4.8838e-02, 7.9608e-03, 5.0533e-04, 7.4733e-02, 3.0858e-03, 5.0984e-04,
        1.4816e-03, 1.4335e-03])

max probability is torch.return_types.max(
values=tensor(0.6912),
indices=tensor(1))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.1095e-05, 3.4186e-06, 3.4123e-06, 3.4118e-06]),
indices=tensor([  0, 319, 910, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 963 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.1676, 0.0026, 0.0055, 0.0028, 0.0120,
0.0202, 0.0139, 0.0152, 0.0800,
        0.0866, 0.0351, 0.0189, 0.1095, 0.1302, 0.0058, 0.0255, 0.1605, 0.0211,
        0.0286, 0.0331])

max probability is torch.return_types.max(
values=tensor(0.1676),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.3410e-04, 2.5945e-05, 2.5945e-05, 2.5940e-05]),
indices=tensor([  0, 771, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 964 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9998e-01, 8.9227e-07, 5.4782e-06,
5.2653e-07, 1.5953e-06, 4.9546e-06,
        2.2686e-06, 1.0055e-07, 7.7690e-07, 1.8799e-06, 2.7566e-08, 1.8239e-08,
        5.5193e-07, 9.4891e-08, 2.8711e-09, 4.0048e-09, 1.2214e-08, 4.9254e-08,
        1.3870e-08, 6.0470e-08])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5055e-12, 1.2089e-14, 1.2087e-14, 1.2076e-14]),
indices=tensor([  0, 910, 323, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 965 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.8817e-03, 7.2099e-04, 6.3351e-02,
3.4346e-02, 1.1929e-01, 2.6190e-01,
```

```
        1.6001e-01, 5.6448e-03, 4.3215e-02, 1.0456e-01, 1.1094e-02, 2.0644e-04,
        1.8802e-02, 1.1099e-01, 1.0856e-03, 9.2815e-03, 7.5204e-03, 3.4919e-03,
        2.3430e-02, 1.3864e-02])

max probability is torch.return_types.max(
values=tensor(0.2619),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2676e-04, 3.3748e-06, 3.3738e-06, 3.3729e-06]),
indices=tensor([  0, 771, 896, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 966 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9871e-01, 9.5995e-07, 5.4876e-05,
1.6810e-06, 4.5140e-05, 6.4743e-06,
        1.1778e-05, 6.0651e-06, 9.3917e-05, 1.3618e-04, 6.4486e-06, 3.6863e-06,
        1.8589e-04, 4.1279e-05, 1.9073e-05, 9.1848e-06, 1.7199e-05, 5.7876e-04,
        5.2389e-05, 1.4468e-05])

max probability is torch.return_types.max(
values=tensor(0.9987),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.0227e-09, 9.6133e-11, 9.6115e-11, 9.6063e-11]),
indices=tensor([  0, 323, 910, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 967 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7233e-03, 1.4472e-03, 9.1369e-02,
2.1692e-01, 3.9249e-01, 2.0186e-01,
        8.0764e-02, 1.8916e-04, 2.5022e-04, 7.5891e-04, 4.0397e-04, 3.3565e-05,
        3.7153e-04, 1.4652e-03, 7.6316e-04, 1.6158e-04, 7.5933e-05, 5.7926e-04,
        2.9319e-03, 7.2209e-04])

max probability is torch.return_types.max(
values=tensor(0.3925),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([5.4312e-05, 4.9032e-06, 4.9020e-06, 4.8987e-06]),
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 968 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0039, 0.0013, 0.0096, 0.0384, 0.1048,
0.4184, 0.3108, 0.0007, 0.0037,
        0.0315, 0.0055, 0.0009, 0.0115, 0.0352, 0.0045, 0.0042, 0.0029, 0.0017,
        0.0075, 0.0021])

max probability is torch.return_types.max(
values=tensor(0.4184),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6889e-05, 9.4722e-07, 9.4721e-07, 9.4718e-07]),
indices=tensor([  0, 910, 771, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 969 :   Prediction is Correct?
No
first 20 classes probability: tensor([5.7191e-03, 3.5764e-03, 2.8275e-03,
8.6644e-04, 3.5425e-03, 2.2411e-03,
        1.8240e-03, 3.8118e-01, 3.4434e-01, 3.0971e-02, 2.4778e-02, 2.4300e-03,
        2.2146e-02, 6.2617e-03, 2.4033e-04, 1.8867e-02, 2.9887e-02, 1.5480e-03,
        9.0832e-02, 2.0488e-02])

max probability is torch.return_types.max(
values=tensor(0.3812),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.6594e-04, 5.1448e-06, 5.1442e-06, 5.1439e-06]),
indices=tensor([  0, 316, 896,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 970 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.3947e-07, 9.4487e-09, 5.5674e-09,
3.9208e-08, 1.2762e-07, 8.0919e-08,
        6.9651e-08, 1.9240e-07, 2.6027e-08, 1.9994e-07, 2.8099e-08, 9.0842e-06,
        1.3808e-06, 2.2697e-07, 9.9849e-01, 2.2500e-08, 5.7408e-08, 1.4199e-03,
```

```
        8.0181e-05, 2.1508e-07])

max probability is torch.return_types.max(
values=tensor(0.9985),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.0749e-11, 7.2859e-13, 7.2853e-13, 7.2826e-13]),
indices=tensor([  0, 748, 337, 255]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 971 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.7920e-04, 5.5635e-04, 1.6409e-01,
2.3010e-01, 5.1634e-01, 6.8692e-02,
        1.9245e-02, 1.3659e-05, 1.0504e-05, 3.2713e-05, 3.2099e-05, 7.8052e-07,
        3.1749e-05, 7.1094e-05, 4.8218e-05, 6.0996e-06, 3.5762e-06, 4.2562e-05,
        3.4730e-04, 4.7681e-05])

max probability is torch.return_types.max(
values=tensor(0.5163),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8481e-06, 1.1023e-07, 1.1019e-07, 1.1010e-07]),
indices=tensor([  0, 319, 316, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 972 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.4893e-04, 2.7432e-04, 3.0753e-01,
2.2683e-01, 4.2597e-01, 2.5656e-02,
        1.2789e-02, 2.5323e-05, 1.7702e-05, 5.8886e-05, 3.2099e-05, 8.3086e-07,
        8.0143e-05, 9.8368e-05, 5.5131e-05, 4.0278e-06, 6.2591e-06, 6.3867e-05,
        2.1779e-04, 4.4197e-05])

max probability is torch.return_types.max(
values=tensor(0.4260),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5505e-06, 9.7127e-08, 9.7123e-08, 9.7075e-08]),
indices=tensor([  0, 316, 319, 910]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 973 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.8830e-03, 5.3672e-04, 2.7984e-04,
7.6701e-05, 2.8599e-04, 2.3275e-04,
        1.2596e-04, 1.1791e-03, 7.3418e-04, 3.9131e-04, 2.9475e-02, 4.0903e-01,
        4.5161e-02, 1.7307e-02, 4.3858e-03, 3.1374e-03, 4.0775e-01, 8.0545e-03,
        2.1702e-02, 4.7295e-02])

max probability is torch.return_types.max(
values=tensor(0.4090),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.6724e-04, 6.4383e-07, 6.4380e-07, 6.4370e-07]),
indices=tensor([  0, 691, 914,  82]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 974 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9277e-01, 6.6221e-05, 2.7898e-03,
9.7143e-04, 5.0177e-04, 2.3429e-03,
        4.1137e-04, 2.8898e-06, 8.7367e-06, 1.1301e-04, 3.5955e-07, 5.5117e-07,
        3.2488e-06, 1.3051e-06, 1.0651e-06, 4.3094e-08, 3.3504e-07, 8.3414e-06,
        2.4585e-06, 5.5306e-06])

max probability is torch.return_types.max(
values=tensor(0.9928),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5519e-09, 2.2329e-11, 2.2328e-11, 2.2316e-11]),
indices=tensor([  0, 323, 271, 291]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 975 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0104, 0.0031, 0.1245, 0.0238, 0.0402,
0.0191, 0.0397, 0.0129, 0.0316,
        0.0834, 0.0427, 0.0024, 0.0661, 0.1781, 0.0007, 0.0139, 0.2278, 0.0049,
        0.0160, 0.0408])
```

```
max probability is torch.return_types.max(
values=tensor(0.2278),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1026e-03, 1.7791e-05, 1.7790e-05, 1.7788e-05]),
indices=tensor([  0, 323, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 976 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.3179e-03, 2.1880e-04, 7.6093e-05,
5.3375e-05, 3.0525e-04, 1.2388e-04,
        1.0162e-04, 1.2830e-03, 8.8104e-04, 3.5848e-04, 2.1885e-02, 1.4475e-01,
        3.7384e-03, 4.4206e-03, 6.3974e-02, 4.6527e-04, 4.9421e-03, 4.0874e-01,
        2.9993e-01, 3.9035e-02])

max probability is torch.return_types.max(
values=tensor(0.4087),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.5729e-04, 2.5231e-07, 2.5227e-07, 2.5226e-07]),
indices=tensor([  0, 230, 158, 691]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 977 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.4016e-04, 5.3204e-03, 1.3110e-04,
2.2602e-05, 2.0118e-04, 5.6810e-05,
        5.8222e-05, 8.1648e-01, 1.7496e-01, 1.9371e-04, 1.3064e-04, 1.0102e-05,
        1.4171e-03, 5.7021e-05, 4.6094e-06, 8.0781e-05, 1.2945e-04, 5.5663e-06,
        2.3903e-04, 4.9372e-05])

max probability is torch.return_types.max(
values=tensor(0.8165),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.4435e-07, 4.8518e-09, 4.8497e-09, 4.8476e-09]),
indices=tensor([  0, 316, 957,  85]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 978 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.4367e-03, 1.8398e-04, 2.9564e-02,
2.9553e-02, 3.5655e-02, 4.9827e-03,
        7.3955e-03, 1.1418e-03, 2.6982e-03, 1.1143e-02, 6.3315e-03, 1.0060e-04,
        1.1830e-02, 7.6995e-01, 4.9243e-04, 2.3366e-03, 2.4198e-02, 5.5312e-03,
        8.8387e-03, 4.4987e-02])

max probability is torch.return_types.max(
values=tensor(0.7699),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.8797e-04, 1.4279e-06, 1.4278e-06, 1.4278e-06]),
indices=tensor([  0, 910, 323, 914]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 979 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.1964e-06, 1.8124e-06, 1.3355e-05,
4.0306e-06, 2.1667e-05, 8.5009e-07,
        4.6294e-07, 1.3140e-05, 1.9263e-05, 1.0094e-05, 3.7315e-04, 8.7049e-07,
        3.2435e-05, 2.9464e-02, 1.5268e-08, 5.3143e-05, 1.0798e-01, 4.6882e-05,
        3.6572e-03, 8.5750e-01])

max probability is torch.return_types.max(
values=tensor(0.8575),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.0298e-04, 9.5838e-11, 9.5769e-11, 9.5757e-11]),
indices=tensor([  0, 655, 691, 337]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 980 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0251, 0.0037, 0.2099, 0.1106, 0.1598,
0.1414, 0.1467, 0.0047, 0.0086,
        0.0366, 0.0026, 0.0006, 0.0049, 0.0135, 0.0160, 0.0020, 0.0030, 0.0550,
        0.0176, 0.0144])

max probability is torch.return_types.max(
values=tensor(0.2099),
```

```
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.8769e-04, 2.4073e-05, 2.4073e-05, 2.4071e-05]),
indices=tensor([  0, 914, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 981 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9999e-01, 8.4872e-08, 4.8369e-07,
1.7551e-08, 2.1826e-07, 2.7385e-07,
        2.9586e-07, 3.3380e-07, 2.9303e-06, 1.1960e-06, 3.4000e-08, 6.7115e-07,
        6.0437e-06, 1.5605e-07, 1.7830e-08, 6.4253e-09, 1.1251e-07, 1.3938e-07,
        2.9164e-08, 3.8214e-08])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7476e-12, 1.7031e-14, 1.7026e-14, 1.7009e-14]),
indices=tensor([  0, 910, 323, 391]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 982 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2820e-03, 4.4928e-02, 5.7098e-04,
4.3404e-05, 3.7029e-04, 1.3914e-04,
        2.0602e-04, 7.1682e-01, 2.2847e-01, 1.2320e-04, 1.7395e-04, 1.8368e-05,
        6.3542e-03, 5.2490e-05, 1.3423e-05, 1.6900e-04, 1.7227e-04, 3.4885e-06,
        6.4489e-05, 1.5017e-05])

max probability is torch.return_types.max(
values=tensor(0.7168),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5560e-07, 5.0718e-09, 5.0710e-09, 5.0709e-09]),
indices=tensor([  0, 316, 319, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 983 :   Prediction is Correct?
```

```
Yes
first 20 classes probability: tensor([0.0140, 0.0011, 0.0124, 0.0451, 0.0935,
0.1989, 0.1937, 0.0034, 0.0198,
        0.2316, 0.0260, 0.0004, 0.0226, 0.1011, 0.0013, 0.0061, 0.0046, 0.0021,
        0.0077, 0.0096])

max probability is torch.return_types.max(
values=tensor(0.2316),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.4376e-05, 5.1929e-06, 5.1929e-06, 5.1899e-06]),
indices=tensor([  0, 319, 771, 896]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 984 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.2344e-03, 3.1667e-04, 8.8954e-05,
1.2407e-04, 3.5799e-04, 3.2100e-03,
        4.9800e-04, 3.0952e-04, 2.7108e-03, 6.1052e-03, 9.3298e-01, 1.3950e-04,
        1.4872e-03, 6.1984e-03, 3.9877e-06, 3.1733e-02, 8.8721e-04, 1.2959e-04,
        2.3546e-03, 4.0956e-03])

max probability is torch.return_types.max(
values=tensor(0.9330),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.8138e-06, 3.1210e-08, 3.1171e-08, 3.1132e-08]),
indices=tensor([  0, 319, 771, 144]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 985 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0065, 0.0098, 0.0029, 0.0008, 0.0057,
0.0161, 0.0118, 0.0305, 0.2252,
        0.0091, 0.1873, 0.0005, 0.1195, 0.0387, 0.0003, 0.3154, 0.0059, 0.0014,
        0.0042, 0.0059])

max probability is torch.return_types.max(
values=tensor(0.3154),
indices=tensor(15))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([4.4302e-05, 2.4453e-06, 2.4420e-06, 2.4409e-06]),
indices=tensor([  0, 319, 836, 316]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 986 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0240, 0.0026, 0.0015, 0.0009, 0.0018,
0.0027, 0.0022, 0.0051, 0.0049,
        0.0089, 0.0657, 0.5755, 0.0367, 0.0300, 0.0026, 0.0118, 0.1676, 0.0074,
        0.0174, 0.0159])

max probability is torch.return_types.max(
values=tensor(0.5755),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.3353e-04, 1.4802e-05, 1.4801e-05, 1.4797e-05]),
indices=tensor([  0, 323, 914,  82]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 987 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([7.8031e-03, 6.7896e-01, 3.8154e-02,
1.0181e-02, 2.0896e-02, 5.8737e-02,
        5.6505e-02, 1.9415e-02, 3.9880e-02, 6.1078e-03, 6.2994e-03, 7.9902e-05,
        2.7768e-02, 3.0012e-03, 4.3814e-04, 2.1835e-02, 1.5070e-03, 2.2549e-04,
        6.2522e-04, 5.0206e-04])

max probability is torch.return_types.max(
values=tensor(0.6790),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.2095e-05, 1.1412e-06, 1.1396e-06, 1.1392e-06]),
indices=tensor([  0, 319, 316, 896]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 988 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.9408e-06, 1.1864e-07, 4.5646e-07,
5.0668e-07, 9.7233e-06, 6.1365e-07,
        9.5584e-08, 8.9068e-05, 1.2191e-05, 1.1759e-05, 1.9926e-04, 4.3960e-04,
```

```
        1.0252e-05, 1.7643e-04, 2.3767e-06, 7.4477e-06, 7.3701e-04, 1.4139e-04,
        9.4456e-01, 5.3587e-02])

max probability is torch.return_types.max(
values=tensor(0.9446),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.1973e-06, 2.9771e-12, 2.9756e-12, 2.9734e-12]),
indices=tensor([  0, 914, 337, 197]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 989 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.5228e-02, 4.1581e-01, 3.4017e-02,
9.1672e-03, 2.3741e-02, 1.8728e-01,
        1.5233e-01, 7.1607e-04, 4.1440e-03, 3.3123e-03, 8.9884e-02, 1.1749e-03,
        1.8422e-02, 8.5482e-03, 1.8549e-04, 1.1674e-02, 1.8234e-03, 3.9065e-04,
        8.2906e-04, 6.5061e-04])

max probability is torch.return_types.max(
values=tensor(0.4158),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0580e-05, 7.0836e-07, 7.0714e-07, 7.0687e-07]),
indices=tensor([  0, 319, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 990 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.9605e-04, 2.8804e-05, 1.4316e-06,
2.8957e-07, 5.9187e-07, 4.3691e-06,
        2.0019e-06, 1.7442e-05, 3.5113e-06, 1.3586e-06, 1.4096e-03, 9.9648e-01,
        1.2290e-03, 1.0212e-05, 1.2504e-05, 8.0157e-06, 5.6345e-04, 4.1917e-06,
        2.5004e-05, 6.0894e-06])

max probability is torch.return_types.max(
values=tensor(0.9965),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.8697e-09, 2.7857e-11, 2.7851e-11, 2.7850e-11]),
```

```
indices=tensor([  0, 323, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 991 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0177, 0.0096, 0.0291, 0.0198, 0.0336,
0.6626, 0.0880, 0.0016, 0.0055,
        0.0059, 0.0312, 0.0266, 0.0142, 0.0091, 0.0021, 0.0085, 0.0057, 0.0022,
        0.0150, 0.0050])

max probability is torch.return_types.max(
values=tensor(0.6626),
indices=tensor(5))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0997e-04, 7.2857e-06, 7.2824e-06, 7.2824e-06]),
indices=tensor([  0, 914, 771, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 992 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.7906e-03, 4.1681e-03, 4.7810e-03,
4.6716e-03, 1.2586e-02, 1.0144e-02,
        5.7702e-03, 2.2315e-02, 3.9320e-02, 2.2648e-02, 3.9899e-02, 1.4227e-03,
        2.0416e-02, 1.6826e-01, 2.2004e-04, 1.4336e-02, 1.2848e-01, 3.1669e-03,
        5.4147e-02, 3.9404e-01])

max probability is torch.return_types.max(
values=tensor(0.3940),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1766e-02, 2.9975e-05, 2.9968e-05, 2.9951e-05]),
indices=tensor([  0, 771, 914, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 993 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.1938e-09, 1.0000e+00, 5.5081e-09,
1.3475e-09, 2.6242e-08, 2.7522e-08,
        2.8485e-08, 2.3809e-07, 3.4754e-08, 1.1284e-11, 1.4582e-09, 1.0808e-11,
        6.3313e-08, 4.0357e-11, 3.7383e-12, 1.6273e-09, 8.3282e-10, 9.2192e-14,
        4.8292e-11, 6.7132e-12])
```

```
max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.8987e-15, 1.4838e-19, 1.4829e-19, 1.4770e-19]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 994 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.6185e-07, 8.0029e-09, 3.3926e-09,
4.5854e-08, 1.7146e-07, 2.6841e-07,
        1.5599e-07, 2.2145e-07, 5.7454e-08, 1.0489e-06, 7.3941e-08, 2.1279e-05,
        3.4116e-06, 9.3968e-07, 9.9917e-01, 4.0646e-07, 3.7155e-07, 5.8167e-04,
        2.1720e-04, 2.8800e-07])

max probability is torch.return_types.max(
values=tensor(0.9992),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.5048e-11, 1.2541e-12, 1.2541e-12, 1.2539e-12]),
indices=tensor([  0, 255, 812, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 995 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0119, 0.0023, 0.0078, 0.0044, 0.0226,
0.0054, 0.0029, 0.0052, 0.0090,
        0.0050, 0.0450, 0.0020, 0.0072, 0.0346, 0.0033, 0.0158, 0.0118, 0.1064,
        0.5187, 0.1707])

max probability is torch.return_types.max(
values=tensor(0.5187),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9996e-03, 6.2494e-06, 6.2484e-06, 6.2482e-06]),
indices=tensor([  0, 914, 337, 466]))
End*****************************************************************************
```

```
**************************************************************************
Predicting Test Sample 996 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([8.8730e-04, 8.1548e-04, 4.6374e-02,
3.4911e-01, 2.3140e-01, 1.1569e-01,
        2.3709e-01, 2.9280e-04, 2.7555e-04, 4.7678e-03, 1.9508e-04, 1.2762e-05,
        3.1145e-04, 5.1135e-03, 1.9231e-03, 1.9265e-04, 8.3456e-05, 7.4951e-04,
        1.9655e-03, 8.0446e-04])

max probability is torch.return_types.max(
values=tensor(0.3491),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5895e-05, 2.0127e-06, 2.0119e-06, 2.0116e-06]),
indices=tensor([  0, 319, 316, 910]))
End*************************************************************************

**************************************************************************
Predicting Test Sample 997 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1322e-04, 2.2242e-05, 1.1231e-04,
7.2503e-05, 3.4040e-04, 1.8148e-04,
        7.6685e-05, 1.0962e-04, 9.3857e-04, 5.3153e-04, 2.5477e-02, 2.2419e-04,
        1.0665e-02, 7.1629e-01, 2.8157e-05, 4.0507e-03, 1.7803e-01, 1.3509e-03,
        6.5508e-03, 5.4762e-02])

max probability is torch.return_types.max(
values=tensor(0.7163),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([6.1479e-05, 1.5413e-08, 1.5413e-08, 1.5406e-08]),
indices=tensor([  0, 771, 914, 323]))
End*************************************************************************

**************************************************************************
Predicting Test Sample 998 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.8655e-07, 9.9999e-01, 1.1601e-07,
3.2564e-08, 3.7326e-07, 1.8330e-06,
        1.0369e-06, 1.1966e-06, 6.4253e-07, 6.2764e-10, 4.8439e-08, 4.6230e-10,
        8.2026e-07, 1.9707e-09, 1.7727e-10, 1.6396e-07, 1.3506e-08, 1.1095e-11,
        1.8086e-09, 3.3366e-10])

max probability is torch.return_types.max(
```

```
values=tensor(1.0000),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5825e-13, 1.4297e-16, 1.4283e-16, 1.4238e-16]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 999 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0449, 0.0066, 0.0072, 0.0077, 0.0107,
0.0310, 0.0248, 0.0048, 0.0100,
        0.0627, 0.2420, 0.2125, 0.1029, 0.0687, 0.0051, 0.0140, 0.0849, 0.0085,
        0.0136, 0.0120])

max probability is torch.return_types.max(
values=tensor(0.2420),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.0406e-04, 2.6415e-05, 2.6412e-05, 2.6407e-05]),
indices=tensor([  0, 323, 914, 771]))
End*****************************************************************************

currently at 1000 current time is 31.172511339187622
*****************************************************************************
Predicting Test Sample 1000 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.8496e-04, 3.3795e-04, 4.1457e-01,
2.5437e-01, 2.4968e-01, 3.6141e-02,
        3.3529e-02, 2.2013e-04, 2.4358e-04, 2.9421e-03, 1.7125e-04, 4.9513e-06,
        4.1249e-04, 1.7092e-03, 6.0519e-04, 7.0679e-05, 8.1151e-05, 1.0756e-03,
        1.1890e-03, 5.4149e-04])

max probability is torch.return_types.max(
values=tensor(0.4146),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9376e-05, 1.1574e-06, 1.1571e-06, 1.1569e-06]),
indices=tensor([  0, 771, 910, 466]))
End*****************************************************************************

*****************************************************************************
```

```
Predicting Test Sample 1001 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.4044e-03, 3.1914e-04, 3.3457e-03,
6.9229e-04, 4.0036e-03, 6.3602e-04,
        6.3434e-04, 5.2745e-03, 4.8729e-03, 5.5712e-03, 6.0307e-03, 5.2504e-03,
        5.8807e-03, 3.1196e-02, 6.5111e-04, 5.5013e-03, 6.1796e-01, 1.2678e-02,
        1.5620e-01, 1.2647e-01])

max probability is torch.return_types.max(
values=tensor(0.6180),
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.9639e-03, 1.5362e-06, 1.5362e-06, 1.5360e-06]),
indices=tensor([  0, 691, 230, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1002 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9348e-05, 9.9316e-01, 7.0907e-05,
2.5861e-05, 1.7212e-04, 1.8420e-04,
        3.1880e-04, 1.5213e-03, 6.8150e-04, 3.2902e-06, 2.2091e-04, 5.1679e-05,
        3.0479e-03, 4.4946e-05, 7.0292e-06, 1.2097e-04, 2.2747e-04, 1.3860e-06,
        1.6303e-05, 1.8495e-05])

max probability is torch.return_types.max(
values=tensor(0.9932),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.3877e-07, 1.5939e-09, 1.5938e-09, 1.5923e-09]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1003 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.1020, 0.0129, 0.0035, 0.0011, 0.0042,
0.0080, 0.0046, 0.0013, 0.0031,
        0.0011, 0.0887, 0.1700, 0.1592, 0.2633, 0.0024, 0.0089, 0.1095, 0.0084,
        0.0073, 0.0389])

max probability is torch.return_types.max(
values=tensor(0.2633),
indices=tensor(13))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.0186e-04, 1.5287e-06, 1.5284e-06, 1.5280e-06]),
indices=tensor([  0, 323, 910, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1004 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.4529e-04, 1.8499e-01, 1.3306e-04,
2.5798e-05, 1.9005e-04, 2.1597e-04,
        1.1355e-04, 5.1573e-03, 1.9810e-03, 1.1591e-04, 5.9085e-02, 1.7102e-03,
        2.2900e-02, 1.3498e-03, 4.0246e-04, 7.0949e-01, 8.6140e-03, 1.2011e-04,
        1.5509e-03, 8.2646e-04])

max probability is torch.return_types.max(
values=tensor(0.7095),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.4082e-06, 8.6760e-08, 8.6686e-08, 8.6649e-08]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1005 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0104, 0.0070, 0.0034, 0.0014, 0.0053,
0.0063, 0.0036, 0.0146, 0.0553,
        0.0143, 0.3676, 0.0018, 0.1405, 0.0501, 0.0005, 0.2425, 0.0339, 0.0022,
        0.0100, 0.0186])

max probability is torch.return_types.max(
values=tensor(0.3676),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.4728e-04, 1.0900e-05, 1.0891e-05, 1.0886e-05]),
indices=tensor([  0, 319, 771, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1006 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0104, 0.0054, 0.0071, 0.0029, 0.0062,
```

```
0.0030, 0.0051, 0.0032, 0.0053,
        0.0039, 0.5019, 0.0326, 0.2534, 0.0807, 0.0025, 0.0071, 0.0208, 0.0157,
        0.0092, 0.0187])

max probability is torch.return_types.max(
values=tensor(0.5019),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2735e-04, 4.9286e-06, 4.9267e-06, 4.9266e-06]),
indices=tensor([  0, 771, 323, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1007 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4561e-05, 8.6394e-03, 2.7915e-07,
1.9009e-08, 6.4442e-07, 4.3717e-07,
        4.7524e-07, 9.1733e-01, 7.3629e-02, 3.7111e-07, 1.9183e-06, 9.3666e-07,
        3.7008e-04, 2.8019e-07, 5.7966e-08, 5.1653e-07, 4.9788e-06, 5.9794e-09,
        9.6887e-07, 1.2533e-07])

max probability is torch.return_types.max(
values=tensor(0.9173),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.4855e-10, 3.1801e-13, 3.1766e-13, 3.1762e-13]),
indices=tensor([  0, 316, 319, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1008 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0032, 0.0012, 0.0209, 0.0278, 0.0217,
0.0090, 0.0133, 0.0040, 0.0053,
        0.0291, 0.0908, 0.0079, 0.0709, 0.5178, 0.0007, 0.0039, 0.0912, 0.0032,
        0.0141, 0.0581])

max probability is torch.return_types.max(
values=tensor(0.5178),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.5671e-04, 5.3155e-06, 5.3132e-06, 5.3119e-06]),
```

```
indices=tensor([  0, 914, 323, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1009 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1995e-02, 6.1984e-02, 5.1660e-03,
1.7667e-03, 5.2872e-03, 6.5374e-02,
        4.4073e-02, 1.3491e-01, 6.2204e-01, 9.8364e-03, 5.1881e-03, 3.8021e-04,
        2.6167e-02, 1.3444e-03, 8.0171e-05, 1.8551e-03, 6.9490e-04, 9.2990e-05,
        4.5621e-04, 2.3602e-04])

max probability is torch.return_types.max(
values=tensor(0.6220),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1084e-05, 1.1294e-06, 1.1289e-06, 1.1289e-06]),
indices=tensor([  0, 319, 836, 794]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1010 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0044, 0.0006, 0.0013, 0.0010, 0.0042,
0.0041, 0.0017, 0.0039, 0.0082,
        0.0066, 0.0557, 0.0262, 0.0134, 0.0901, 0.0073, 0.0078, 0.0436, 0.1019,
        0.4294, 0.1773])

max probability is torch.return_types.max(
values=tensor(0.4294),
indices=tensor(18))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2712e-03, 9.5611e-06, 9.5527e-06, 9.5515e-06]),
indices=tensor([  0, 914, 771, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1011 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.1536, 0.0080, 0.0793, 0.0269, 0.0561,
0.0930, 0.1325, 0.0129, 0.0516,
        0.1460, 0.0439, 0.0013, 0.0308, 0.0528, 0.0018, 0.0243, 0.0131, 0.0104,
        0.0091, 0.0189])
```

```
max probability is torch.return_types.max(
values=tensor(0.1536),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.8666e-04, 3.5113e-05, 3.5104e-05, 3.5096e-05]),
indices=tensor([  0, 771, 319, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1012 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.3003e-11, 6.7338e-10, 3.8856e-12,
2.7867e-12, 1.9450e-11, 1.8886e-10,
        2.7793e-11, 3.5574e-13, 2.4680e-11, 9.9899e-11, 9.9999e-01, 5.1217e-09,
        3.3116e-07, 4.4760e-07, 1.2959e-14, 6.0946e-06, 5.4976e-08, 1.9403e-11,
        4.7299e-10, 3.2526e-09])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6129e-15, 1.6389e-20, 1.6369e-20, 1.6280e-20]),
indices=tensor([  0, 319, 771, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1013 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.7897e-02, 3.4660e-03, 6.1622e-03,
1.8725e-03, 8.9733e-03, 8.2948e-02,
        4.8649e-02, 3.6878e-02, 5.8719e-01, 4.8632e-02, 1.8364e-02, 5.8818e-04,
        8.0002e-02, 1.7166e-02, 2.5384e-04, 1.0251e-02, 3.9125e-03, 7.4194e-04,
        1.4958e-03, 1.6125e-03])

max probability is torch.return_types.max(
values=tensor(0.5872),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7007e-05, 3.1036e-06, 3.1014e-06, 3.1007e-06]),
indices=tensor([  0, 319, 836, 896]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 1014 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.6804e-08, 3.2296e-08, 4.3055e-07,
4.0733e-07, 1.2233e-06, 2.9493e-08,
        5.2849e-08, 1.4074e-07, 1.2822e-07, 1.8059e-07, 2.4518e-06, 3.2066e-10,
        3.9936e-08, 8.2453e-05, 6.0006e-12, 2.9470e-08, 1.7586e-03, 1.8907e-07,
        8.0859e-05, 9.9758e-01])

max probability is torch.return_types.max(
values=tensor(0.9976),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.9558e-04, 3.4399e-14, 3.4357e-14, 3.4351e-14]),
indices=tensor([  0, 655, 337, 691]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 1015 :   Prediction is Correct?
No
first 20 classes probability: tensor([8.9291e-02, 1.4590e-03, 2.7081e-02,
1.4419e-02, 6.3737e-02, 6.0052e-02,
        7.2816e-02, 1.9967e-03, 9.7676e-03, 2.5043e-02, 2.5727e-03, 4.1080e-04,
        2.3370e-03, 7.9292e-03, 4.3107e-02, 1.6281e-03, 1.1148e-03, 5.2925e-01,
        3.1880e-02, 1.1058e-02])

max probability is torch.return_types.max(
values=tensor(0.5293),
indices=tensor(17))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.1938e-05, 3.1089e-06, 3.1088e-06, 3.1083e-06]),
indices=tensor([  0, 771, 323, 337]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 1016 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.7660e-03, 3.6220e-05, 3.1750e-04,
7.5580e-04, 3.2031e-03, 1.6858e-03,
        1.2426e-03, 2.4922e-04, 2.3531e-03, 5.6972e-03, 6.2292e-03, 9.3798e-05,
        3.3902e-03, 9.1775e-01, 2.9777e-04, 3.7980e-03, 1.3262e-02, 4.8041e-03,
        1.2684e-02, 2.0220e-02])

max probability is torch.return_types.max(
```

```
values=tensor(0.9177),
indices=tensor(13))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([3.3629e-05, 1.4292e-07, 1.4290e-07, 1.4286e-07]),
indices=tensor([  0, 771, 323, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1017 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0048, 0.0007, 0.0093, 0.0094, 0.0189,
0.0388, 0.0143, 0.0039, 0.0122,
        0.0389, 0.1672, 0.0261, 0.3050, 0.1519, 0.0030, 0.0154, 0.0975, 0.0048,
        0.0429, 0.0331])

max probability is torch.return_types.max(
values=tensor(0.3050),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6225e-04, 2.0001e-06, 1.9998e-06, 1.9983e-06]),
indices=tensor([  0, 771, 914, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1018 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.0821e-04, 3.9896e-03, 2.4183e-03,
4.7968e-04, 2.5082e-03, 7.8201e-04,
        1.0686e-03, 4.6879e-01, 4.2978e-01, 6.7814e-03, 3.8469e-03, 1.1931e-04,
        2.7654e-02, 6.9018e-03, 1.2499e-05, 5.3675e-03, 2.4179e-02, 6.5179e-05,
        8.9814e-03, 5.7223e-03])

max probability is torch.return_types.max(
values=tensor(0.4688),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.1185e-05, 1.6819e-07, 1.6817e-07, 1.6817e-07]),
indices=tensor([  0, 896,  85, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1019 :   Prediction is Correct?
```

```
Yes
first 20 classes probability: tensor([9.9895e-01, 4.8282e-08, 3.4326e-04,
6.5694e-06, 4.5438e-05, 1.0164e-06,
        3.6682e-06, 2.9829e-06, 1.6992e-05, 2.9989e-04, 5.0126e-07, 1.8664e-07,
        4.8476e-05, 1.8183e-04, 7.4954e-07, 1.9160e-07, 4.3983e-06, 7.5569e-05,
        1.4215e-05, 8.0904e-06])

max probability is torch.return_types.max(
values=tensor(0.9989),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.8666e-10, 1.5153e-12, 1.5151e-12, 1.5150e-12]),
indices=tensor([  0, 323, 910, 748]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1020 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.5781e-02, 3.6748e-02, 8.0038e-04,
4.9371e-04, 5.1469e-03, 2.3751e-01,
        6.3005e-02, 3.9630e-02, 5.4661e-01, 2.9798e-03, 1.2460e-02, 6.8937e-04,
        2.7588e-02, 3.9705e-03, 5.0315e-05, 1.6861e-03, 1.2124e-03, 2.5517e-04,
        1.5526e-03, 1.6380e-03])

max probability is torch.return_types.max(
values=tensor(0.5466),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.6594e-05, 1.8213e-07, 1.8202e-07, 1.8196e-07]),
indices=tensor([  0, 319,  44, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1021 :   Prediction is Correct?
No
first 20 classes probability: tensor([9.8998e-05, 2.8362e-06, 1.2076e-05,
7.3321e-06, 7.5139e-05, 3.3171e-06,
        2.4289e-06, 7.7021e-05, 6.1085e-05, 3.6018e-05, 1.6827e-03, 5.5753e-04,
        2.6006e-04, 7.6679e-03, 3.9323e-04, 9.6844e-05, 3.6104e-03, 1.6267e-01,
        5.6246e-01, 2.6011e-01])

max probability is torch.return_types.max(
values=tensor(0.5625),
indices=tensor(18))
```

```
the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1285e-04, 7.3107e-10, 7.3095e-10, 7.3056e-10]),
indices=tensor([  0, 158, 337, 230]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1022 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.7015e-06, 1.8007e-05, 3.6566e-06,
3.2886e-07, 2.2916e-06, 5.5098e-06,
        5.2801e-06, 3.1391e-04, 1.8073e-03, 3.7674e-07, 1.6796e-03, 5.7977e-05,
        9.9599e-01, 5.8019e-05, 1.6973e-06, 4.5922e-05, 9.8983e-06, 4.0576e-07,
        6.0053e-07, 1.1974e-06])

max probability is torch.return_types.max(
values=tensor(0.9960),
indices=tensor(12))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0813e-09, 1.5155e-11, 1.5141e-11, 1.5138e-11]),
indices=tensor([  0,  44, 319, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1023 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.5744e-03, 9.1640e-04, 1.4696e-03,
1.4036e-03, 4.4501e-03, 5.9080e-03,
        3.9340e-03, 4.6466e-03, 2.2827e-02, 1.8843e-02, 7.6634e-02, 7.0142e-04,
        1.1796e-02, 2.6117e-01, 6.4361e-05, 3.6786e-02, 2.0721e-01, 2.1626e-03,
        2.8524e-02, 3.0526e-01])

max probability is torch.return_types.max(
values=tensor(0.3053),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1080e-03, 1.6978e-06, 1.6970e-06, 1.6965e-06]),
indices=tensor([  0, 771, 914, 323]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1024 :   Prediction is Correct?
No
```

```
first 20 classes probability: tensor([7.8469e-04, 3.2962e-04, 3.7435e-01,
2.4541e-01, 3.1105e-01, 3.8443e-02,
        2.3930e-02, 1.1922e-04, 1.1130e-04, 9.4816e-04, 1.5092e-04, 5.7113e-06,
        3.1800e-04, 7.6131e-04, 3.3311e-04, 5.0160e-05, 5.7871e-05, 5.0458e-04,
        1.0482e-03, 3.8115e-04])

max probability is torch.return_types.max(
values=tensor(0.3744),
indices=tensor(2))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.7017e-05, 9.3901e-07, 9.3889e-07, 9.3884e-07]),
indices=tensor([  0, 771, 910, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1025 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0225, 0.0065, 0.0115, 0.0036, 0.0152,
0.0799, 0.0503, 0.0614, 0.5313,
        0.0607, 0.0193, 0.0006, 0.0609, 0.0250, 0.0008, 0.0224, 0.0063, 0.0025,
        0.0049, 0.0059])

max probability is torch.return_types.max(
values=tensor(0.5313),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.1242e-04, 8.9660e-06, 8.9624e-06, 8.9609e-06]),
indices=tensor([  0, 319, 896, 771]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1026 :   Prediction is Correct?
No
first 20 classes probability: tensor([2.0361e-03, 1.0655e-03, 1.8145e-03,
1.1022e-03, 2.3825e-03, 1.7761e-03,
        1.0229e-03, 1.5178e-03, 4.8319e-03, 5.8519e-03, 5.3519e-01, 6.3266e-03,
        1.0795e-01, 1.3504e-01, 3.7359e-04, 5.0132e-02, 8.5333e-02, 3.4579e-03,
        1.1437e-02, 3.8915e-02])

max probability is torch.return_types.max(
values=tensor(0.5352),
indices=tensor(10))

the max probability of the rest of 980 dim is
```

```
torch.return_types.topk(
values=tensor([2.7149e-04, 2.3084e-06, 2.3062e-06, 2.3057e-06]),
indices=tensor([  0, 771, 323, 914]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 1027 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2032e-06, 1.8869e-08, 9.7046e-09,
1.2001e-07, 3.6959e-07, 7.2419e-07,
        4.1698e-07, 2.8643e-07, 6.7947e-08, 9.6107e-07, 3.4756e-08, 5.7642e-06,
        1.9178e-06, 7.5053e-07, 9.9878e-01, 1.7000e-07, 1.0198e-07, 1.1283e-03,
        8.1023e-05, 1.8223e-07])

max probability is torch.return_types.max(
values=tensor(0.9988),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.5213e-11, 1.5923e-12, 1.5923e-12, 1.5922e-12]),
indices=tensor([  0, 323, 812, 748]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 1028 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.3553e-05, 3.9648e-04, 1.7041e-06,
3.0135e-07, 1.4742e-06, 1.3483e-06,
        1.5267e-06, 1.0165e-03, 8.7221e-05, 2.7362e-06, 1.4296e-03, 9.8301e-01,
        3.1507e-03, 3.5265e-05, 5.3820e-05, 1.5760e-04, 1.0472e-02, 7.3539e-06,
        7.2883e-05, 3.3550e-05])

max probability is torch.return_types.max(
values=tensor(0.9830),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1101e-07, 1.0628e-09, 1.0625e-09, 1.0625e-09]),
indices=tensor([  0, 323, 382, 910]))
End********************************************************************************

********************************************************************************
Predicting Test Sample 1029 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.3569e-05, 8.2918e-07, 2.4989e-06,
6.8509e-06, 2.1078e-05, 1.0125e-05,
```

```
        8.8631e-06, 2.5906e-05, 8.8429e-06, 4.9343e-05, 1.3182e-05, 3.1981e-04,
        1.5467e-04, 7.3021e-05, 9.8221e-01, 9.7866e-06, 1.7759e-05, 1.0870e-02,
        6.1327e-03, 5.1923e-05])

max probability is torch.return_types.max(
values=tensor(0.9822),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.1475e-08, 2.9189e-09, 2.9189e-09, 2.9176e-09]),
indices=tensor([  0, 337, 748, 255]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1030 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.8302e-05, 1.8662e-06, 2.1254e-06,
8.9280e-06, 9.9382e-06, 1.4969e-05,
        1.4030e-05, 8.2019e-06, 1.9845e-06, 1.9669e-05, 6.4610e-06, 2.5235e-03,
        1.4175e-04, 3.3211e-04, 9.8628e-01, 6.1751e-06, 1.3812e-04, 7.9498e-03,
        2.3873e-03, 1.0292e-04])

max probability is torch.return_types.max(
values=tensor(0.9863),
indices=tensor(14))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.6317e-08, 1.2868e-09, 1.2867e-09, 1.2860e-09]),
indices=tensor([  0, 691, 748, 337]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1031 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([5.1119e-09, 1.0000e+00, 1.8984e-08,
3.3763e-09, 4.5408e-08, 7.1403e-08,
        4.8290e-08, 1.2583e-07, 2.5322e-08, 2.8768e-11, 4.7545e-08, 2.5515e-11,
        6.1118e-07, 6.1089e-11, 3.1652e-11, 2.5795e-07, 4.4629e-10, 3.8799e-13,
        1.0759e-10, 8.6435e-12])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
```

```
values=tensor([1.4666e-15, 8.2740e-19, 8.2468e-19, 8.2132e-19]),
indices=tensor([  0, 319, 316, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1032 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.0152e-08, 9.9998e-01, 1.0103e-08,
6.8924e-10, 3.4212e-08, 8.3471e-09,
        1.4023e-08, 1.5733e-05, 1.1586e-06, 1.7332e-11, 2.5219e-09, 9.5603e-11,
        9.7661e-07, 3.2092e-10, 3.8024e-11, 1.0952e-09, 1.3747e-08, 5.9356e-13,
        2.5002e-10, 3.5639e-11])

max probability is torch.return_types.max(
values=tensor(1.0000),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.1414e-14, 8.8861e-19, 8.8718e-19, 8.8515e-19]),
indices=tensor([  0, 316, 319, 910]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1033 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.5630e-05, 7.0919e-06, 2.1526e-02,
9.2480e-01, 3.9485e-02, 6.0422e-03,
        6.9831e-03, 7.8695e-07, 4.0076e-07, 4.1562e-05, 3.3045e-06, 8.7375e-09,
        3.5478e-06, 1.0201e-03, 6.3554e-06, 2.9797e-07, 3.3416e-07, 7.3481e-06,
        1.1901e-05, 4.9135e-05])

max probability is torch.return_types.max(
values=tensor(0.9248),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.1099e-08, 3.7197e-10, 3.7182e-10, 3.7175e-10]),
indices=tensor([  0, 771, 910, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1034 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.8752e-01, 8.1926e-03, 2.1828e-02,
6.0600e-03, 1.5657e-02, 1.4096e-02,
        2.3636e-02, 1.9471e-01, 4.4151e-01, 3.7831e-02, 3.6565e-03, 3.3420e-04,
```

```
        2.6736e-02, 7.6432e-03, 2.1746e-04, 9.4851e-04, 1.8244e-03, 9.1658e-04,
        2.2206e-03, 1.7560e-03])

max probability is torch.return_types.max(
values=tensor(0.4415),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([4.5202e-05, 2.8134e-06, 2.8127e-06, 2.8124e-06]),
indices=tensor([  0, 319,  85, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1035 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9921e-01, 2.5259e-06, 1.0071e-04,
5.7026e-06, 5.2353e-05, 3.2160e-05,
        4.4531e-05, 7.7085e-06, 1.3387e-04, 6.4753e-05, 4.7777e-06, 2.2529e-06,
        2.8779e-04, 2.3756e-05, 2.0330e-06, 1.3476e-06, 2.0844e-06, 1.8785e-05,
        2.5564e-06, 3.5311e-06])

max probability is torch.return_types.max(
values=tensor(0.9992),
indices=tensor(0))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2787e-09, 8.4579e-11, 8.4545e-11, 8.4523e-11]),
indices=tensor([  0, 910, 323, 319]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1036 :   Prediction is Correct?
No
first 20 classes probability: tensor([7.5103e-04, 9.5299e-04, 1.5094e-01,
2.8036e-01, 3.8430e-01, 1.3075e-01,
        4.9779e-02, 3.7762e-05, 3.4473e-05, 1.8419e-04, 6.4551e-05, 4.8662e-06,
        7.8050e-05, 2.5838e-04, 1.7350e-04, 2.1621e-05, 1.1401e-05, 1.1280e-04,
        6.0340e-04, 1.1877e-04])

max probability is torch.return_types.max(
values=tensor(0.3843),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.0208e-06, 4.8005e-07, 4.8002e-07, 4.7959e-07]),
```

```
indices=tensor([  0, 319, 316,  85]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1037 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4364e-02, 4.8644e-05, 3.0275e-02,
1.5442e-02, 6.8922e-03, 1.0016e-02,
        6.6066e-03, 3.4080e-03, 5.2929e-03, 8.9171e-01, 5.2156e-04, 2.3615e-04,
        2.7566e-03, 2.8855e-03, 3.9406e-04, 1.3467e-04, 1.6722e-03, 6.6860e-04,
        4.9186e-03, 1.4120e-03])

max probability is torch.return_types.max(
values=tensor(0.8917),
indices=tensor(9))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([8.5630e-06, 3.5876e-07, 3.5870e-07, 3.5859e-07]),
indices=tensor([  0, 914, 771,  91]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1038 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.1509e-07, 2.2768e-07, 6.9147e-07,
6.0793e-07, 4.3044e-06, 1.6106e-07,
        1.5928e-07, 1.8910e-05, 1.2900e-05, 1.6777e-06, 9.7021e-06, 1.2071e-08,
        6.9334e-07, 2.5892e-05, 4.5489e-10, 1.4981e-07, 1.2207e-03, 1.3029e-06,
        6.7084e-03, 9.8929e-01])

max probability is torch.return_types.max(
values=tensor(0.9893),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.7012e-03, 1.9369e-12, 1.9365e-12, 1.9349e-12]),
indices=tensor([  0, 655, 337, 914]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1039 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([4.5269e-04, 1.2462e-03, 1.6435e-01,
2.3379e-01, 4.7974e-01, 9.0191e-02,
        2.8030e-02, 4.1735e-05, 3.3246e-05, 8.7140e-05, 8.3381e-05, 3.9400e-06,
        9.4721e-05, 1.7530e-04, 1.4612e-04, 1.9058e-05, 1.1997e-05, 1.2092e-04,
```

```
        7.1812e-04, 1.1536e-04])

max probability is torch.return_types.max(
values=tensor(0.4797),
indices=tensor(4))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.5463e-06, 5.6954e-07, 5.6938e-07, 5.6896e-07]),
indices=tensor([  0, 319, 316, 957]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1040 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.4509e-02, 6.7837e-02, 2.6274e-03,
6.8609e-04, 2.3705e-03, 2.3610e-03,
        2.2477e-03, 5.3811e-01, 2.7591e-01, 3.3529e-03, 5.9669e-03, 3.0515e-03,
        6.5606e-02, 2.2219e-03, 6.7224e-04, 1.3863e-03, 4.6446e-03, 2.8480e-04,
        1.4877e-03, 7.9575e-04])

max probability is torch.return_types.max(
values=tensor(0.5381),
indices=tensor(7))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.6286e-05, 4.0263e-06, 4.0261e-06, 4.0256e-06]),
indices=tensor([  0, 316, 319, 836]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1041 :   Prediction is Correct?
No
first 20 classes probability: tensor([1.7556e-04, 4.3181e-04, 2.5613e-02,
3.4618e-01, 1.8132e-01, 1.0241e-01,
        3.4100e-01, 3.4364e-05, 4.0003e-05, 6.7461e-04, 4.2842e-05, 9.5395e-07,
        5.6380e-05, 1.5306e-03, 1.4390e-04, 2.1261e-05, 6.6340e-06, 5.4320e-05,
        1.3764e-04, 9.6762e-05])

max probability is torch.return_types.max(
values=tensor(0.3462),
indices=tensor(3))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0803e-06, 3.4790e-08, 3.4767e-08, 3.4767e-08]),
indices=tensor([  0, 319, 316, 910]))
```

```
End**************************************************************************

**************************************************************************
Predicting Test Sample 1042 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([1.2368e-06, 2.2409e-06, 2.6522e-08,
4.3847e-08, 1.4926e-07, 1.2041e-06,
        1.3234e-07, 6.9552e-08, 4.4692e-07, 1.5476e-06, 9.9649e-01, 2.5130e-05,
        3.0007e-05, 2.9518e-05, 6.4186e-09, 3.3498e-03, 3.3584e-05, 3.9575e-07,
        1.2982e-05, 2.3316e-05])

max probability is torch.return_types.max(
values=tensor(0.9965),
indices=tensor(10))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.3992e-09, 1.2058e-12, 1.2042e-12, 1.2003e-12]),
indices=tensor([  0, 319, 771, 476]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 1043 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([3.4131e-04, 9.9814e-01, 9.2108e-05,
2.8982e-05, 7.2111e-05, 2.2029e-04,
        1.8186e-04, 1.7140e-04, 6.1890e-05, 2.4191e-06, 2.0674e-04, 1.2753e-04,
        2.8078e-04, 3.4132e-06, 7.9274e-07, 3.9268e-05, 2.1373e-05, 1.5989e-07,
        2.8941e-06, 8.8257e-07])

max probability is torch.return_types.max(
values=tensor(0.9981),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0276e-08, 1.5840e-10, 1.5834e-10, 1.5834e-10]),
indices=tensor([  0, 319, 323, 910]))
End**************************************************************************

**************************************************************************
Predicting Test Sample 1044 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.1914e-04, 9.9709e-01, 8.9186e-05,
5.9420e-05, 8.8670e-05, 9.4455e-04,
        3.4451e-04, 3.6734e-05, 2.8321e-05, 1.8419e-06, 7.5339e-05, 4.1410e-05,
        2.3877e-04, 4.4676e-06, 1.4682e-06, 2.2491e-05, 6.5031e-06, 1.8077e-07,
        1.7233e-06, 6.6202e-07])
```

```
max probability is torch.return_types.max(
values=tensor(0.9971),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([5.5921e-09, 1.2058e-10, 1.2043e-10, 1.2043e-10]),
indices=tensor([  0, 319, 323, 316]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1045 :   Prediction is Correct?
No
first 20 classes probability: tensor([3.3954e-03, 1.3999e-03, 1.3778e-02,
5.3924e-03, 2.3614e-02, 7.7022e-03,
        6.4307e-03, 2.1777e-02, 2.6201e-02, 1.0672e-02, 1.0174e-02, 4.2968e-04,
        4.0638e-03, 4.5470e-02, 1.6158e-04, 3.6659e-03, 1.0090e-01, 8.2527e-03,
        1.5911e-01, 5.3369e-01])

max probability is torch.return_types.max(
values=tensor(0.5337),
indices=tensor(19))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.0485e-02, 3.3979e-06, 3.3964e-06, 3.3964e-06]),
indices=tensor([  0, 914, 337, 655]))
End*****************************************************************************

*****************************************************************************
Predicting Test Sample 1046 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([0.0185, 0.2963, 0.0327, 0.0136, 0.0287,
0.0196, 0.0225, 0.0294, 0.0197,
        0.0027, 0.0462, 0.0646, 0.1229, 0.0188, 0.0093, 0.0059, 0.0467, 0.0059,
        0.0112, 0.0115])

max probability is torch.return_types.max(
values=tensor(0.2963),
indices=tensor(1))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([0.0022, 0.0002, 0.0002, 0.0002]),
indices=tensor([  0, 323, 910,  44]))
End*****************************************************************************
```

```
**********************************************************************************
Predicting Test Sample 1047 :   Prediction is Correct?
No
first 20 classes probability: tensor([0.0074, 0.0016, 0.0022, 0.0021, 0.0050,
0.0042, 0.0038, 0.0049, 0.0051,
        0.0132, 0.0289, 0.2260, 0.0357, 0.0548, 0.1494, 0.0240, 0.1490, 0.0783,
        0.1126, 0.0291])

max probability is torch.return_types.max(
values=tensor(0.2260),
indices=tensor(11))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([9.5365e-04, 6.3927e-05, 6.3924e-05, 6.3910e-05]),
indices=tensor([  0, 323, 914, 230]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 1048 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([2.1269e-03, 6.7172e-03, 7.6824e-04,
3.0508e-04, 1.7758e-03, 4.4124e-03,
        2.1466e-03, 5.6518e-03, 3.0770e-02, 2.5781e-03, 1.6711e-01, 3.0586e-04,
        4.0866e-02, 2.3173e-02, 1.2313e-04, 6.9639e-01, 7.0574e-03, 4.3751e-04,
        2.2113e-03, 4.1233e-03])

max probability is torch.return_types.max(
values=tensor(0.6964),
indices=tensor(15))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([2.2154e-05, 9.9845e-07, 9.9667e-07, 9.9637e-07]),
indices=tensor([  0, 319, 836, 316]))
End*********************************************************************************

**********************************************************************************
Predicting Test Sample 1049 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([9.9949e-03, 1.1772e-02, 4.3555e-03,
4.9650e-04, 3.3239e-03, 3.1029e-03,
        1.6473e-03, 4.9988e-02, 1.5334e-01, 6.9713e-03, 2.5256e-02, 2.5635e-03,
        1.4503e-01, 7.1481e-02, 1.7629e-04, 6.3438e-02, 4.2500e-01, 7.3804e-04,
        5.3678e-03, 1.4856e-02])

max probability is torch.return_types.max(
values=tensor(0.4250),
```

```
indices=tensor(16))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([1.4541e-04, 1.0177e-06, 1.0173e-06, 1.0166e-06]),
indices=tensor([  0, 910, 323, 771]))
End*********************************************************************

*********************************************************************
Predicting Test Sample 1050 :   Prediction is Correct?
Yes
first 20 classes probability: tensor([6.0040e-03, 8.3900e-03, 4.3249e-03,
7.2000e-04, 2.4153e-03, 8.9865e-03,
        6.3053e-03, 1.2230e-01, 5.9262e-01, 4.0811e-03, 8.2534e-03, 3.7077e-04,
        2.2140e-01, 5.9335e-03, 2.3760e-04, 4.6717e-03, 1.1841e-03, 1.9241e-04,
        3.2363e-04, 4.0397e-04])

max probability is torch.return_types.max(
values=tensor(0.5926),
indices=tensor(8))

the max probability of the rest of 980 dim is
torch.return_types.topk(
values=tensor([7.9799e-06, 9.3112e-07, 9.3088e-07, 9.3035e-07]),
indices=tensor([  0, 319, 836, 794]))
End*********************************************************************

the overall testing error is
0.5657142857142857
```

[9]:
```
# Save the weight as the output
torch.save(model.state_dict(), 'model_weights_alexnet_replication.pth')
```

[ ]:
```

```