[HW4_prob2]_VGG16_Quantization_aware_train

October 25, 2021

[1]:
```
import argparse
import os
import time
import shutil

import torch
import torch.nn as nn
import torch.optim as optim
import torch.nn.functional as F
import torch.backends.cudnn as cudnn


import torchvision
import torchvision.transforms as transforms

from models import *


global best_prec
use_gpu = torch.cuda.is_available()
print('=> Building model...')



batch_size = 128
model_name = "VGG16_quant"
model = VGG16_quant()
print(model)

normalize = transforms.Normalize(mean=[0.491, 0.482, 0.447], std=[0.247, 0.243,␣
↪0.262])


train_dataset = torchvision.datasets.CIFAR10(
    root='./data',
    train=True,
    download=True,
```

```
    transform=transforms.Compose([
        transforms.RandomCrop(32, padding=4),
        transforms.RandomHorizontalFlip(),
        transforms.ToTensor(),
        normalize,
    ]))
trainloader = torch.utils.data.DataLoader(train_dataset, batch_size=batch_size,␣
 ↪shuffle=True, num_workers=2)


test_dataset = torchvision.datasets.CIFAR10(
    root='./data',
    train=False,
    download=True,
    transform=transforms.Compose([
        transforms.ToTensor(),
        normalize,
    ]))

testloader = torch.utils.data.DataLoader(test_dataset, batch_size=batch_size,␣
 ↪shuffle=False, num_workers=2)


print_freq = 100 # every 100 batches, accuracy printed. Here, each batch␣
 ↪includes "batch_size" data points
# CIFAR10 has 50,000 training data, and 10,000 validation data.

def train(trainloader, model, criterion, optimizer, epoch):
    batch_time = AverageMeter()
    data_time = AverageMeter()
    losses = AverageMeter()
    top1 = AverageMeter()

    model.train()

    end = time.time()
    for i, (input, target) in enumerate(trainloader):
        # measure data loading time
        data_time.update(time.time() - end)

        input, target = input.cuda(), target.cuda()

        # compute output
        output = model(input)
        loss = criterion(output, target)

        # measure accuracy and record loss
```

```
        prec = accuracy(output, target)[0]
        losses.update(loss.item(), input.size(0))
        top1.update(prec.item(), input.size(0))

        # compute gradient and do SGD step
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()

        # measure elapsed time
        batch_time.update(time.time() - end)
        end = time.time()


        if i % print_freq == 0:
            print('Epoch: [{0}][{1}/{2}]\t'
                  'Time {batch_time.val:.3f} ({batch_time.avg:.3f})\t'
                  'Data {data_time.val:.3f} ({data_time.avg:.3f})\t'
                  'Loss {loss.val:.4f} ({loss.avg:.4f})\t'
                  'Prec {top1.val:.3f}% ({top1.avg:.3f}%)'.format(
                   epoch, i, len(trainloader), batch_time=batch_time,
                   data_time=data_time, loss=losses, top1=top1))



def validate(val_loader, model, criterion ):
    batch_time = AverageMeter()
    losses = AverageMeter()
    top1 = AverageMeter()

    # switch to evaluate mode
    model.eval()

    end = time.time()
    with torch.no_grad():
        for i, (input, target) in enumerate(val_loader):

            input, target = input.cuda(), target.cuda()

            # compute output
            output = model(input)
            loss = criterion(output, target)

            # measure accuracy and record loss
            prec = accuracy(output, target)[0]
            losses.update(loss.item(), input.size(0))
            top1.update(prec.item(), input.size(0))
```

```
            # measure elapsed time
            batch_time.update(time.time() - end)
            end = time.time()

            if i % print_freq == 0:  # This line shows how frequently print out␣
↪the status. e.g., i%5 => every 5 batch, prints out
                print('Test: [{0}/{1}]\t'
                  'Time {batch_time.val:.3f} ({batch_time.avg:.3f})\t'
                  'Loss {loss.val:.4f} ({loss.avg:.4f})\t'
                  'Prec {top1.val:.3f}% ({top1.avg:.3f}%)'.format(
                   i, len(val_loader), batch_time=batch_time, loss=losses,
                   top1=top1))

    print(' * Prec {top1.avg:.3f}% '.format(top1=top1))
    return top1.avg


def accuracy(output, target, topk=(1,)):
    """Computes the precision@k for the specified values of k"""
    maxk = max(topk)
    batch_size = target.size(0)

    _, pred = output.topk(maxk, 1, True, True)
    pred = pred.t()
    correct = pred.eq(target.view(1, -1).expand_as(pred))

    res = []
    for k in topk:
        correct_k = correct[:k].view(-1).float().sum(0)
        res.append(correct_k.mul_(100.0 / batch_size))
    return res


class AverageMeter(object):
    """Computes and stores the average and current value"""
    def __init__(self):
        self.reset()

    def reset(self):
        self.val = 0
        self.avg = 0
        self.sum = 0
        self.count = 0

    def update(self, val, n=1):
        self.val = val
```

```
        self.sum += val * n
        self.count += n
        self.avg = self.sum / self.count


def save_checkpoint(state, is_best, fdir):
    filepath = os.path.join(fdir, 'checkpoint.pth')
    torch.save(state, filepath)
    if is_best:
        shutil.copyfile(filepath, os.path.join(fdir, 'model_best.pth.tar'))


def adjust_learning_rate(optimizer, epoch):
    """For resnet, the lr starts from 0.1, and is divided by 10 at 80 and 120␣
 ↪epochs"""
    adjust_list = [150, 225]
    if epoch in adjust_list:
        for param_group in optimizer.param_groups:
            param_group['lr'] = param_group['lr'] * 0.1

#model = nn.DataParallel(model).cuda()
#all_params = checkpoint['state_dict']
#model.load_state_dict(all_params, strict=False)
#criterion = nn.CrossEntropyLoss().cuda()
#validate(testloader, model, criterion)
```

```
=> Building model…
VGG_quant(
  (features): Sequential(
    (0): QuantConv2d(
      3, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
      (weight_quant): weight_quantize_fn()
    )
    (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (2): ReLU(inplace=True)
    (3): QuantConv2d(
      64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
      (weight_quant): weight_quantize_fn()
    )
    (4): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (5): ReLU(inplace=True)
    (6): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
    (7): QuantConv2d(
      64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
```

```
      (weight_quant): weight_quantize_fn()
    )
    (8): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (9): ReLU(inplace=True)
    (10): QuantConv2d(
      128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
      (weight_quant): weight_quantize_fn()
    )
    (11): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (12): ReLU(inplace=True)
    (13): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
    (14): QuantConv2d(
      128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
      (weight_quant): weight_quantize_fn()
    )
    (15): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (16): ReLU(inplace=True)
    (17): QuantConv2d(
      256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
      (weight_quant): weight_quantize_fn()
    )
    (18): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (19): ReLU(inplace=True)
    (20): QuantConv2d(
      256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
      (weight_quant): weight_quantize_fn()
    )
    (21): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (22): ReLU(inplace=True)
    (23): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
    (24): QuantConv2d(
      256, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
      (weight_quant): weight_quantize_fn()
    )
    (25): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (26): ReLU(inplace=True)
    (27): QuantConv2d(
      512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
      (weight_quant): weight_quantize_fn()
    )
```

```
    (28): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (29): ReLU(inplace=True)
    (30): QuantConv2d(
      512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
      (weight_quant): weight_quantize_fn()
    )
    (31): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (32): ReLU(inplace=True)
    (33): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
    (34): QuantConv2d(
      512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
      (weight_quant): weight_quantize_fn()
    )
    (35): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (36): ReLU(inplace=True)
    (37): QuantConv2d(
      512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
      (weight_quant): weight_quantize_fn()
    )
    (38): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (39): ReLU(inplace=True)
    (40): QuantConv2d(
      512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
      (weight_quant): weight_quantize_fn()
    )
    (41): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    (42): ReLU(inplace=True)
    (43): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
    (44): AvgPool2d(kernel_size=1, stride=1, padding=0)
  )
  (classifier): Linear(in_features=512, out_features=10, bias=True)
)
Files already downloaded and verified
Files already downloaded and verified
```

[ ]:
```
# This cell won't be given, but students will complete the training

lr = 4e-2
weight_decay = 1e-4
epochs = 500
```

```
best_prec = 0

#model = nn.DataParallel(model).cuda()
model.cuda()
criterion = nn.CrossEntropyLoss().cuda()
optimizer = torch.optim.SGD(model.parameters(), lr=lr, momentum=0.9,␣
 ↪weight_decay=weight_decay)
#cudnn.benchmark = True

if not os.path.exists('result'):
    os.makedirs('result')
fdir = 'result/'+str(model_name)
if not os.path.exists(fdir):
    os.makedirs(fdir)


for epoch in range(0, epochs):
    adjust_learning_rate(optimizer, epoch)

    train(trainloader, model, criterion, optimizer, epoch)

    # evaluate on test set
    print("Validation starts")
    prec = validate(testloader, model, criterion)

    # remember best precision and save checkpoint
    is_best = prec > best_prec
    best_prec = max(prec,best_prec)
    print('best acc: {:1f}'.format(best_prec))
    save_checkpoint({
        'epoch': epoch + 1,
        'state_dict': model.state_dict(),
        'best_prec': best_prec,
        'optimizer': optimizer.state_dict(),
    }, is_best, fdir)
```

[ ]:
```
# HW

#  1. Train with 4 bits for both weight and activation to achieve >90% accuracy
#  2. Find x_int and w_int for the 2nd convolution layer
#  3. Check the recovered psum has similar value to the un-quantized original␣
 ↪psum
#     (such as example 1 in W3S2)
```

[5]:
```
class SaveOutput:
    def __init__(self):
        self.outputs = []
```

```
    def __call__(self, module, module_in):
        self.outputs.append(module_in)
    def clear(self):
        self.outputs = []

######### Save inputs from selected layer ##########
save_output = SaveOutput()

for layer in model.modules():
    if isinstance(layer, torch.nn.Conv2d):
        print("prehooked")
        layer.register_forward_pre_hook(save_output) ## Input for the module␣
 ↪will be grapped

#####################################################

use_gpu = torch.cuda.is_available()
device = torch.device("cuda" if use_gpu else "cpu")

dataiter = iter(trainloader)
images, labels = dataiter.next()
images = images.to(device)
out = model(images)
```

```
prehooked
prehooked
prehooked
prehooked
prehooked
prehooked
prehooked
prehooked
prehooked
prehooked
prehooked
prehooked
prehooked
```

[2]:
```
PATH = "result/VGG16_quant/model_best.pth.tar"
checkpoint = torch.load(PATH)
model.load_state_dict(checkpoint['state_dict'])
device = torch.device("cuda")

model.cuda()
model.eval()

test_loss = 0
```

```
correct = 0

with torch.no_grad():
    for data, target in testloader:
        data, target = data.to(device), target.to(device) # loading to GPU
        output = model(data)
        pred = output.argmax(dim=1, keepdim=True)
        correct += pred.eq(target.view_as(pred)).sum().item()

test_loss /= len(testloader.dataset)

print('\nTest set: Accuracy: {}/{} ({:.0f}%)\n'.format(
        correct, len(testloader.dataset),
        100. * correct / len(testloader.dataset)))
```

```
/opt/conda/lib/python3.9/site-packages/torch/nn/functional.py:718: UserWarning:
Named tensors and all their associated APIs are an experimental feature and
subject to change. Please do not use them for anything important until they are
released as stable. (Triggered internally at
/pytorch/c10/core/TensorImpl.h:1156.)
  return torch.max_pool2d(input, kernel_size, stride, padding, dilation,
ceil_mode)


Test set: Accuracy: 9053/10000 (91%)
```

[ ]:
```
#send an input and grap the value by using prehook like HW3
```

[6]:
```
w_bit = 4
weight_q = model.features[3].weight_q # quantized value is stored during the␣
 ↪training
w_alpha = model.features[3].weight_quant.wgt_alpha
w_delta = w_alpha/(2**(w_bit-1)-1)
weight_int = weight_q/w_delta
print(weight_int) # you should see clean integer numbers
```

```
tensor([[[[ 1.0000,  1.0000,  1.0000],
          [ 1.0000,  1.0000,  1.0000],
          [-0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  1.0000],
          [ 1.0000,  0.0000,  1.0000],
          [ 1.0000,  0.0000, -0.0000]],

         [[ 0.0000,  1.0000,  1.0000],
          [ 0.0000, -0.0000,  0.0000],
          [ 0.0000,  0.0000,  1.0000]],
```

```
 …,

 [[ 0.0000,  1.0000,  1.0000],
  [ 1.0000,  0.0000,  0.0000],
  [ 1.0000,  0.0000,  1.0000]],

 [[ 2.0000,  2.0000,  2.0000],
  [ 2.0000,  2.0000,  2.0000],
  [ 1.0000,  1.0000,  2.0000]],

 [[ 1.0000,  1.0000,  1.0000],
  [ 1.0000, -0.0000,  1.0000],
  [ 0.0000, -1.0000, -0.0000]]],


[[[-1.0000, -2.0000, -3.0000],
  [ 0.0000,  2.0000,  1.0000],
  [ 3.0000,  4.0000,  4.0000]],

 [[-2.0000,  1.0000,  3.0000],
  [-1.0000, -2.0000, -1.0000],
  [ 3.0000,  2.0000,  1.0000]],

 [[ 0.0000,  1.0000, -0.0000],
  [-2.0000, -2.0000, -3.0000],
  [ 1.0000,  1.0000,  1.0000]],

 …,

 [[ 3.0000,  2.0000,  2.0000],
  [-3.0000, -5.0000, -7.0000],
  [ 2.0000,  4.0000,  1.0000]],

 [[-0.0000, -0.0000, -1.0000],
  [ 1.0000,  1.0000,  0.0000],
  [ 1.0000,  1.0000,  1.0000]],

 [[ 2.0000,  1.0000,  3.0000],
  [ 1.0000, -1.0000, -2.0000],
  [-1.0000, -1.0000, -0.0000]]],


[[[ 1.0000,  1.0000,  1.0000],
  [ 0.0000,  2.0000,  1.0000],
  [ 1.0000,  0.0000,  0.0000]],

 [[ 3.0000,  0.0000,  0.0000],
```

```
  [ 1.0000, -4.0000, -3.0000],
  [ 1.0000, -2.0000,  0.0000]],

 [[-7.0000, -3.0000, -3.0000],
  [ 1.0000,  3.0000,  3.0000],
  [ 6.0000,  6.0000,  6.0000]],

 …,

 [[-3.0000, -3.0000,  1.0000],
  [-2.0000, -2.0000,  3.0000],
  [ 3.0000,  3.0000,  1.0000]],

 [[-1.0000, -1.0000, -2.0000],
  [-1.0000, -2.0000, -2.0000],
  [-2.0000, -2.0000, -1.0000]],

 [[-4.0000, -2.0000, -2.0000],
  [-0.0000,  1.0000,  4.0000],
  [ 3.0000,  2.0000,  0.0000]]],


…,


[[[ 2.0000,  1.0000, -0.0000],
  [-0.0000,  1.0000,  0.0000],
  [-2.0000, -0.0000, -0.0000]],

 [[ 3.0000, -3.0000, -4.0000],
  [ 3.0000,  0.0000, -2.0000],
  [-0.0000,  2.0000,  0.0000]],

 [[-1.0000,  2.0000,  5.0000],
  [-3.0000, -3.0000, -1.0000],
  [-3.0000, -4.0000, -5.0000]],

 …,

 [[ 1.0000, -1.0000, -4.0000],
  [ 2.0000,  3.0000,  0.0000],
  [-3.0000,  2.0000,  4.0000]],

 [[ 1.0000,  1.0000,  1.0000],
  [ 1.0000,  1.0000,  1.0000],
  [-0.0000,  0.0000,  0.0000]],

 [[-2.0000,  1.0000,  7.0000],
```

```
  [-3.0000, -6.0000,  1.0000],
  [ 3.0000, -4.0000, -4.0000]]],


[[[ 1.0000,  0.0000, -0.0000],
  [ 0.0000, -1.0000, -1.0000],
  [ 0.0000,  1.0000, -0.0000]],

 [[-1.0000, -4.0000,  4.0000],
  [ 2.0000, -7.0000,  7.0000],
  [ 2.0000, -7.0000,  6.0000]],

 [[ 0.0000,  2.0000,  5.0000],
  [-3.0000, -1.0000,  3.0000],
  [-2.0000, -2.0000,  1.0000]],

 …,

 [[-2.0000,  2.0000, -1.0000],
  [-2.0000,  1.0000,  1.0000],
  [-1.0000, -2.0000,  1.0000]],

 [[ 1.0000,  0.0000, -1.0000],
  [ 1.0000,  0.0000, -1.0000],
  [ 0.0000,  0.0000, -0.0000]],

 [[-1.0000,  1.0000, -5.0000],
  [-2.0000, -2.0000,  5.0000],
  [ 6.0000, -7.0000,  7.0000]]],


[[[ 1.0000, -1.0000, -1.0000],
  [-0.0000, -1.0000, -2.0000],
  [ 0.0000, -2.0000, -2.0000]],

 [[-6.0000,  3.0000,  7.0000],
  [-5.0000,  2.0000,  6.0000],
  [-4.0000,  2.0000,  3.0000]],

 [[ 2.0000, -1.0000, -3.0000],
  [ 3.0000, -0.0000, -2.0000],
  [ 4.0000,  0.0000, -2.0000]],

 …,

 [[ 5.0000,  4.0000, -5.0000],
  [ 3.0000,  3.0000, -6.0000],
  [ 3.0000,  2.0000, -4.0000]],
```

```
          [[ 1.0000,  1.0000,  1.0000],
           [ 0.0000, -0.0000,  0.0000],
           [-0.0000, -1.0000, -0.0000]],

          [[-4.0000,  2.0000,  2.0000],
           [-5.0000,  0.0000,  3.0000],
           [-3.0000,  1.0000,  1.0000]]]], device='cuda:0',
       grad_fn=<DivBackward0>)
```

[7]:
```
x_bit = 4
x = save_output.outputs[1][0]  # input of the 2nd conv layer
x_alpha  = model.features[3].act_alpha
x_delta = x_alpha/(2**x_bit-1)

act_quant_fn = act_quantization(x_bit) # define the quantization function
x_q = act_quant_fn(x, x_alpha)         # create the quantized value for x

x_int = x_q/x_delta
print(x_int) # you should see clean integer numbers
```

```
tensor([[[[ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 2.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 2.0000,  0.0000,  0.0000,  …,  1.0000,  1.0000,  1.0000],
          …,
          [ 1.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 2.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 6.0000,  3.0000,  3.0000,  …,  3.0000,  3.0000,  3.0000]],

         [[12.0000, 10.0000, 10.0000,  …, 10.0000, 10.0000,  0.0000],
          [ 3.0000,  4.0000,  4.0000,  …,  2.0000,  2.0000,  0.0000],
          [ 0.0000,  5.0000,  3.0000,  …,  4.0000,  2.0000,  0.0000],
          …,
          [ 4.0000,  6.0000,  4.0000,  …,  8.0000,  9.0000,  0.0000],
          [ 0.0000,  5.0000,  3.0000,  …,  7.0000,  9.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]],

         [[ 4.0000,  3.0000,  3.0000,  …,  3.0000,  3.0000,  1.0000],
          [ 3.0000,  1.0000,  1.0000,  …,  1.0000,  2.0000,  1.0000],
          [ 3.0000,  1.0000,  2.0000,  …,  3.0000,  3.0000,  2.0000],
          …,
          [ 4.0000,  4.0000,  6.0000,  …, 12.0000, 12.0000,  9.0000],
          [ 3.0000,  2.0000,  3.0000,  …, 11.0000, 11.0000,  9.0000],
          [ 4.0000,  4.0000,  4.0000,  …, 11.0000, 10.0000,  8.0000]],

         …,

         [[ 0.0000,  2.0000,  2.0000,  …,  2.0000,  2.0000,  8.0000],
```

```
  [ 0.0000,  2.0000,  1.0000,  …,  0.0000,  0.0000,  3.0000],
  [ 0.0000,  4.0000,  3.0000,  …,  0.0000,  0.0000,  5.0000],
  …,
  [ 0.0000,  4.0000,  2.0000,  …,  5.0000,  4.0000,  9.0000],
  [ 0.0000,  6.0000,  4.0000,  …,  6.0000,  4.0000,  8.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]],

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

 [[ 0.0000,  4.0000,  4.0000,  …,  4.0000,  4.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [ 0.0000,  1.0000,  4.0000,  …,  4.0000,  2.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  5.0000,  3.0000,  0.0000],
  [ 0.0000,  7.0000,  4.0000,  …,  8.0000,  9.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 2.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 2.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 2.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 6.0000,  3.0000,  3.0000,  …,  3.0000,  3.0000,  2.0000]],

 [[ 0.0000,  8.0000,  9.0000,  …, 10.0000,  9.0000,  0.0000],
  [ 0.0000,  6.0000,  6.0000,  …,  6.0000,  5.0000,  0.0000],
  [ 0.0000,  6.0000,  6.0000,  …,  6.0000,  5.0000,  0.0000],
  …,
  [10.0000, 14.0000, 12.0000,  …, 11.0000, 11.0000,  0.0000],
  [ 4.0000,  5.0000,  5.0000,  …,  5.0000,  5.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]],

 [[ 3.0000,  2.0000,  2.0000,  …,  3.0000,  3.0000,  2.0000],
  [ 2.0000,  0.0000,  0.0000,  …,  1.0000,  1.0000,  1.0000],
  [ 2.0000,  0.0000,  0.0000,  …,  1.0000,  1.0000,  1.0000],
  …,
  [ 4.0000,  2.0000,  3.0000,  …,  4.0000,  4.0000,  2.0000],
  [ 3.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 4.0000,  3.0000,  3.0000,  …,  3.0000,  3.0000,  2.0000]],
```

```
 …,

 [[ 0.0000,  8.0000,  2.0000,  …,  2.0000,  2.0000,  8.0000],
  [ 0.0000, 12.0000,  4.0000,  …,  4.0000,  4.0000,  8.0000],
  [ 0.0000, 12.0000,  4.0000,  …,  4.0000,  4.0000,  8.0000],
  …,
  [ 0.0000,  5.0000,  2.0000,  …,  1.0000,  1.0000,  8.0000],
  [ 0.0000,  3.0000,  3.0000,  …,  3.0000,  3.0000,  9.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]],

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

 [[ 0.0000,  8.0000,  5.0000,  …,  4.0000,  4.0000,  0.0000],
  [ 0.0000,  5.0000,  3.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  5.0000,  2.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [ 0.0000,  0.0000,  1.0000,  …,  7.0000,  7.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  4.0000,  4.0000,  …,  4.0000,  4.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 2.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 2.0000,  1.0000,  8.0000,  …,  9.0000,  9.0000,  7.0000],
  …,
  [ 2.0000,  0.0000,  0.0000,  …,  1.0000,  1.0000,  2.0000],
  [ 2.0000,  0.0000,  0.0000,  …,  2.0000,  2.0000,  2.0000],
  [ 6.0000,  2.0000,  0.0000,  …,  1.0000,  1.0000,  1.0000]],

 [[12.0000, 10.0000, 10.0000,  …, 10.0000, 10.0000,  0.0000],
  [ 4.0000,  5.0000,  5.0000,  …,  5.0000,  5.0000,  0.0000],
  [ 4.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [ 4.0000,  0.0000, 11.0000,  …,  6.0000,  6.0000, 11.0000],
  [ 4.0000,  0.0000, 10.0000,  …,  6.0000,  4.0000, 15.0000],
  [ 0.0000,  0.0000, 14.0000,  …, 11.0000, 10.0000, 15.0000]],

 [[ 4.0000,  3.0000,  3.0000,  …,  3.0000,  3.0000,  1.0000],
  [ 3.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 3.0000,  2.0000,  3.0000,  …,  5.0000,  5.0000,  3.0000],
  …,
  [ 3.0000,  0.0000,  0.0000,  …,  1.0000,  1.0000,  1.0000],
```

```
  [ 3.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  1.0000],
  [ 4.0000,  1.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]],

 …,

 [[ 0.0000,  2.0000,  2.0000,  …,  2.0000,  2.0000,  8.0000],
  [ 0.0000,  3.0000,  3.0000,  …,  3.0000,  3.0000,  9.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [ 0.0000, 10.0000, 15.0000,  …,  5.0000,  5.0000,  6.0000],
  [ 0.0000,  8.0000, 15.0000,  …,  3.0000,  2.0000,  4.0000],
  [ 0.0000,  0.0000, 15.0000,  …, 15.0000, 15.0000, 13.0000]],

 [[ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [ 0.0000,  0.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  1.0000,  …,  0.0000,  1.0000,  0.0000],
  [ 0.0000,  0.0000,  1.0000,  …,  1.0000,  1.0000,  1.0000]],

 [[ 0.0000,  4.0000,  4.0000,  …,  4.0000,  4.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000, 11.0000,  5.0000,  …, 11.0000, 11.0000,  0.0000],
  …,
  [ 0.0000,  0.0000, 15.0000,  …, 10.0000, 12.0000, 10.0000],
  [ 0.0000,  0.0000, 15.0000,  …, 10.0000, 10.0000, 12.0000],
  [ 0.0000,  0.0000, 15.0000,  …,  9.0000,  9.0000, 15.0000]]],


…,


[[[ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 2.0000,  1.0000,  1.0000,  …,  0.0000,  0.0000,  1.0000],
  [ 2.0000,  2.0000,  2.0000,  …,  0.0000,  0.0000,  1.0000],
  …,
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 2.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 6.0000,  3.0000,  3.0000,  …,  3.0000,  3.0000,  2.0000]],

 [[12.0000, 10.0000,  7.0000,  …,  9.0000, 15.0000,  0.0000],
  [ 6.0000,  9.0000,  2.0000,  …,  3.0000, 15.0000,  0.0000],
  [ 7.0000, 12.0000,  6.0000,  …,  3.0000, 15.0000,  0.0000],
  …,
  [ 8.0000,  8.0000,  8.0000,  …,  8.0000,  4.0000,  0.0000],
  [ 4.0000,  5.0000,  5.0000,  …,  5.0000,  5.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]],
```

```
[[ 3.0000,  4.0000,  3.0000,  …,  3.0000,  3.0000,  2.0000],
 [ 4.0000,  4.0000,  4.0000,  …,  1.0000,  2.0000,  2.0000],
 [ 4.0000,  5.0000,  4.0000,  …,  2.0000,  2.0000,  2.0000],
 …,
 [ 3.0000,  2.0000,  2.0000,  …,  1.0000,  0.0000,  0.0000],
 [ 3.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
 [ 4.0000,  3.0000,  3.0000,  …,  3.0000,  3.0000,  2.0000]],

…,

[[ 0.0000,  2.0000,  3.0000,  …,  1.0000,  2.0000,  3.0000],
 [ 0.0000,  1.0000,  4.0000,  …,  2.0000,  5.0000,  2.0000],
 [ 0.0000,  2.0000,  4.0000,  …,  1.0000,  6.0000,  2.0000],
 …,
 [ 0.0000,  2.0000,  2.0000,  …,  2.0000,  3.0000,  7.0000],
 [ 0.0000,  3.0000,  3.0000,  …,  3.0000,  3.0000,  9.0000],
 [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]],

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

[[ 0.0000,  0.0000,  2.0000,  …,  2.0000,  6.0000,  0.0000],
 [ 0.0000,  1.0000,  3.0000,  …,  1.0000,  6.0000,  0.0000],
 [ 4.0000,  4.0000,  0.0000,  …,  0.0000,  9.0000,  0.0000],
 …,
 [ 0.0000,  1.0000,  1.0000,  …,  1.0000,  2.0000,  0.0000],
 [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
 [ 0.0000,  4.0000,  4.0000,  …,  4.0000,  4.0000,  0.0000]]],


[[[ 3.0000,  4.0000,  4.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 1.0000,  2.0000,  2.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 1.0000,  2.0000,  2.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [ 0.0000,  1.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  3.0000,  3.0000,  2.0000]],

 [[ 6.0000,  6.0000,  6.0000,  …, 10.0000, 10.0000,  0.0000],
  [ 9.0000,  8.0000,  8.0000,  …,  5.0000,  5.0000,  0.0000],
  [ 9.0000,  9.0000, 10.0000,  …,  5.0000,  5.0000,  0.0000],
  …,
```

```
  [ 8.0000, 10.0000, 10.0000,  …,  5.0000,  5.0000,  0.0000],
  [15.0000, 15.0000, 15.0000,  …,  5.0000,  5.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]],

 [[ 0.0000,  0.0000,  0.0000,  …,  3.0000,  3.0000,  1.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [15.0000, 15.0000, 15.0000,  …,  0.0000,  0.0000,  0.0000],
  [13.0000, 15.0000, 15.0000,  …,  0.0000,  0.0000,  0.0000],
  [11.0000, 15.0000, 15.0000,  …,  3.0000,  3.0000,  2.0000]],

 …,

 [[15.0000,  7.0000,  7.0000,  …,  2.0000,  2.0000,  8.0000],
  [15.0000,  5.0000,  5.0000,  …,  3.0000,  3.0000,  9.0000],
  [15.0000,  7.0000,  7.0000,  …,  3.0000,  3.0000,  9.0000],
  …,
  [ 9.0000,  9.0000,  9.0000,  …,  3.0000,  3.0000,  9.0000],
  [15.0000, 15.0000, 15.0000,  …,  3.0000,  3.0000,  9.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]],

 [[ 1.0000,  1.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 2.0000,  2.0000,  2.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 2.0000,  2.0000,  2.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]],

 [[14.0000, 11.0000, 12.0000,  …,  4.0000,  4.0000,  0.0000],
  [15.0000, 11.0000, 12.0000,  …,  0.0000,  0.0000,  0.0000],
  [15.0000, 11.0000, 11.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [10.0000, 10.0000, 10.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 9.0000,  7.0000,  7.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000, 12.0000, 12.0000,  …,  4.0000,  4.0000,  0.0000]]],


[[[ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 9.0000,  9.0000,  9.0000,  …,  3.0000,  0.0000,  0.0000],
  [ 0.0000,  2.0000,  2.0000,  …,  4.0000,  0.0000,  0.0000],
  …,
  [ 0.0000,  0.0000,  0.0000,  …,  3.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  3.0000,  0.0000,  0.0000],
  [ 1.0000,  0.0000,  0.0000,  …,  7.0000,  3.0000,  2.0000]],

 [[ 0.0000,  0.0000,  0.0000,  …, 10.0000, 10.0000,  0.0000],
```

```
          [ 1.0000,  4.0000,  4.0000,  …,  8.0000,  5.0000,  0.0000],
          [11.0000, 11.0000, 12.0000,  …,  4.0000,  5.0000,  0.0000],
          …,
          [15.0000,  2.0000, 10.0000,  …,  3.0000,  5.0000,  0.0000],
          [ 8.0000,  7.0000,  8.0000,  …,  3.0000,  5.0000,  0.0000],
          [ 7.0000,  8.0000,  6.0000,  …,  0.0000,  0.0000,  0.0000]],

         [[ 4.0000,  4.0000,  4.0000,  …,  4.0000,  3.0000,  1.0000],
          [ 0.0000,  0.0000,  0.0000,  …,  1.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  …,  2.0000,  0.0000,  0.0000],
          …,
          [ 0.0000,  0.0000,  0.0000,  …,  1.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  …,  1.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  …,  3.0000,  3.0000,  2.0000]],

         …,

         [[ 0.0000,  0.0000,  0.0000,  …,  0.0000,  2.0000,  8.0000],
          [ 6.0000,  8.0000,  6.0000,  …,  0.0000,  3.0000,  9.0000],
          [ 9.0000,  9.0000, 10.0000,  …,  0.0000,  3.0000,  9.0000],
          …,
          [ 9.0000, 10.0000, 14.0000,  …,  0.0000,  3.0000,  9.0000],
          [ 5.0000, 10.0000, 15.0000,  …,  0.0000,  3.0000,  9.0000],
          [ 3.0000,  4.0000,  7.0000,  …,  0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 0.0000,  1.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 1.0000,  1.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
          …,
          [ 1.0000,  0.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]],

         [[ 0.0000,  8.0000,  8.0000,  …,  6.0000,  4.0000,  0.0000],
          [ 5.0000,  3.0000,  3.0000,  …,  0.0000,  0.0000,  0.0000],
          [11.0000,  5.0000,  6.0000,  …,  0.0000,  0.0000,  0.0000],
          …,
          [13.0000,  3.0000, 15.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 9.0000,  5.0000, 11.0000,  …,  0.0000,  0.0000,  0.0000],
          [10.0000,  9.0000,  8.0000,  …,  0.0000,  4.0000,  0.0000]]]],
       device='cuda:0', grad_fn=<DivBackward0>)
```

[8]:
```
conv_int = torch.nn.Conv2d(in_channels = 64, out_channels=64, kernel_size = 3,␣
 ↪bias = False)
conv_int.weight = torch.nn.parameter.Parameter(weight_int)

output_int = conv_int(x_int)
```

```
output_recovered = output_int*w_delta*x_delta
print(output_recovered)
```

```
tensor([[[[ 3.5523e+01,  3.5433e+01,  3.4400e+01,  …,  3.3816e+01,
            3.3816e+01,  3.2918e+01],
          [ 3.1346e+01,  3.2918e+01,  3.1885e+01,  …,  3.2199e+01,
            3.2604e+01,  3.2783e+01],
          [ 3.1840e+01,  3.3322e+01,  3.1705e+01,  …,  3.2244e+01,
            3.1077e+01,  3.1571e+01],
          …,
          [ 3.8352e+01,  4.0957e+01,  4.1181e+01,  …,  2.9101e+01,
            2.8382e+01,  2.8292e+01],
          [ 3.2289e+01,  3.0358e+01,  2.9640e+01,  …,  3.0044e+01,
            2.8831e+01,  2.8966e+01],
          [ 2.8158e+01,  2.7170e+01,  2.6541e+01,  …,  2.7035e+01,
            2.6900e+01,  2.5688e+01]],

         [[-2.2454e+00,  1.0643e+01,  1.2529e+01,  …,  2.6496e+01,
            3.1616e+01,  3.3277e+01],
          [-1.8862e+00, -1.5718e+00, -3.8621e+00,  …,  2.2275e+01,
            2.2230e+01,  1.8368e+01],
          [-1.0284e+01, -5.5237e+00, -4.5807e+00,  …,  2.3802e+00,
           -7.8141e+00, -1.8278e+01],
          …,
          [-7.3740e+01, -7.3381e+01, -5.6944e+01,  …, -8.2183e+00,
           -9.0266e+00, -1.1137e+01],
          [ 1.0374e+01,  2.2903e+01,  2.2724e+01,  …, -1.4371e+01,
           -1.3877e+01, -9.1164e+00],
          [-8.3530e+00, -1.1766e+01, -1.5314e+01,  …, -2.4520e+01,
           -2.5643e+01, -1.6706e+01]],

         [[-1.6347e+01, -1.2889e+01, -1.2080e+01,  …, -7.9937e+00,
           -8.6224e+00, -7.4997e+00],
          [-2.0972e+01, -1.5583e+01, -1.8592e+01,  …, -1.6257e+01,
           -1.3967e+01, -1.0239e+01],
          [-2.3757e+01, -2.5463e+01, -2.6721e+01,  …, -2.0568e+01,
           -1.8502e+01, -1.1452e+01],
          …,
          [-1.0823e+01, -8.7123e+00, -1.2754e+01,  …, -1.8412e+00,
           -2.6945e+00, -1.2574e+00],
          [-2.0388e+01, -1.0733e+01, -1.6077e+01,  …,  1.9311e+00,
            1.3922e+00, -2.1556e+00],
          [-1.9445e+01, -1.4236e+01, -1.4461e+01,  …,  1.3023e+00,
            6.7363e-01, -7.5446e+00]],

         …,
```

```
 [[-6.6465e+00, -7.3201e+00, -9.5206e+00,  …, -1.4775e+01,
   -1.0239e+01, -1.3562e+01],
  [-5.9279e+00, -3.3232e+00, -5.6585e+00,  …, -1.2440e+01,
   -9.4308e+00, -3.9070e+00],
  [-4.7154e+00, -1.0329e+00, -2.2005e+00,  …, -3.1436e+00,
   -1.0778e+01, -2.4251e+00],
  …,
  [-1.7155e+01, -1.9221e+01, -1.3832e+01,  …, -1.9535e+01,
   -1.5987e+01, -9.9697e+00],
  [-1.6212e+01, -1.6796e+01, -1.6571e+01,  …, -2.2050e+01,
   -2.0029e+01, -1.2709e+01],
  [-1.1407e+01, -8.9817e+00, -1.7290e+01,  …, -2.3846e+01,
   -2.4251e+01, -2.4116e+01]],

 [[-1.8188e+01,  2.2005e+00, -1.0509e+01,  …, -5.9728e+00,
   -5.1645e+00, -3.5074e+01],
  [-2.4969e+01,  1.4281e+01, -3.6825e+00,  …, -6.4668e+00,
   -4.6705e+00, -1.7829e+01],
  [-2.6945e+01,  1.2170e+01,  1.6167e+00,  …,  5.0747e+00,
   -1.5359e+01, -2.0838e+01],
  …,
  [-1.5269e+00, -1.1182e+01, -5.3890e+00,  …,  7.6345e-01,
   -7.5895e+00, -3.8217e+01],
  [-1.3338e+01, -1.2125e+00, -8.2183e+00,  …,  7.1854e+00,
   -6.8710e+00, -4.1361e+01],
  [-2.1915e+01,  6.8710e+00, -9.1164e+00,  …,  6.2872e-01,
   -3.0987e+00, -4.5537e+01]],

 [[-1.9311e+00,  2.7843e+00,  3.5478e+00,  …,  1.0329e+00,
   -3.1885e+00, -1.9670e+01],
  [ 2.3802e+00,  5.0298e+00,  1.0104e+01,  …,  9.5655e+00,
   -1.8412e+00, -1.6571e+01],
  [ 7.2752e+00,  5.8381e-01,  1.2844e+01,  …,  1.5044e+01,
    2.9191e+00, -1.9805e+01],
  …,
  [ 1.5359e+01,  1.2934e+01,  9.2062e+00,  …,  1.3473e-01,
    4.8501e+00, -2.6137e+01],
  [ 1.0554e+01,  8.9817e+00,  1.4820e+00,  …, -1.7963e-01,
    6.1525e+00, -2.7394e+01],
  [-1.4820e+00,  6.7363e-01,  7.1854e-01,  …, -5.8381e+00,
    3.9070e+00, -2.9415e+01]]],


[[[ 3.2514e+01,  3.3412e+01,  3.1301e+01,  …,  3.4535e+01,
    3.5029e+01,  3.5702e+01],
  [ 3.0987e+01,  3.1840e+01,  2.9595e+01,  …,  3.1032e+01,
    3.1660e+01,  3.2469e+01],
  [ 3.0897e+01,  3.2020e+01,  2.9595e+01,  …,  3.1032e+01,
```

```
   3.1660e+01,  3.2469e+01],
 …,
 [ 4.1361e+01,  4.3427e+01,  4.3741e+01,  …,  4.9444e+01,
   5.0747e+01,  5.3441e+01],
 [ 3.5837e+01,  3.3771e+01,  3.3726e+01,  …,  4.3876e+01,
   4.3516e+01,  4.2843e+01],
 [ 3.1301e+01,  2.9280e+01,  3.0179e+01,  …,  3.5388e+01,
   3.5523e+01,  3.4939e+01]],

[[-2.2454e+01, -9.9697e+00, -1.2080e+01,  …, -1.1811e+01,
  -1.2305e+01, -1.1901e+01],
 [-1.3248e+01, -5.2543e+00, -6.7812e+00,  …, -3.1436e+00,
  -4.2663e+00, -2.5598e+00],
 [-1.1137e+01, -3.9519e+00, -5.6136e+00,  …, -3.1436e+00,
  -4.2663e+00, -2.5598e+00],
 …,
 [-8.4024e+01, -9.6508e+01, -9.6104e+01,  …, -1.4155e+02,
  -1.3823e+02, -1.2889e+02],
 [-4.5358e+00,  4.4908e-02,  2.2454e-01,  …, -7.9488e+00,
  -7.6344e+00, -4.3112e+00],
 [-7.5446e+00, -1.3203e+01, -1.2889e+01,  …, -1.0239e+01,
  -1.0104e+01, -1.2305e+01]],

[[-6.8710e+00, -2.6047e+00, -3.4580e+00,  …, -1.0104e+01,
  -5.5237e+00, -6.2872e+00],
 [-1.6032e+01, -1.3607e+01, -1.4461e+01,  …, -1.8233e+01,
  -1.3832e+01, -1.8772e+01],
 [-1.5628e+01, -1.4191e+01, -1.3877e+01,  …, -1.8233e+01,
  -1.3832e+01, -1.8772e+01],
 …,
 [-3.1391e+01, -3.7184e+01, -3.6286e+01,  …,  4.5358e+00,
  -2.7843e+00, -7.2303e+00],
 [-4.4010e+01, -3.2379e+01, -2.9235e+01,  …, -9.7901e+00,
  -1.3158e+01, -1.3023e+01],
 [-2.3083e+01, -1.8368e+01, -1.7290e+01,  …, -1.4281e+01,
  -1.4461e+01, -1.5808e+01]],

…,

[[-1.4056e+01, -1.4236e+01, -1.8008e+01,  …, -6.6016e+00,
  -3.3232e+00, -6.6465e+00],
 [-6.7363e+00, -6.4668e+00, -1.2934e+01,  …, -4.8052e+00,
  -2.9191e+00, -1.9760e+00],
 [-8.8919e+00, -5.7932e+00, -1.1227e+01,  …, -4.8052e+00,
  -2.9191e+00, -1.9760e+00],
 …,
 [ 3.3232e+00,  5.0298e+00,  7.9488e+00,  …, -1.2574e+01,
  -1.0104e+01, -5.6585e+00],
```

```
  [-1.4461e+01, -6.0627e+00, -9.4757e+00,  …, -2.6810e+01,
   -2.5912e+01, -2.2050e+01],
  [-1.3787e+01, -6.8710e+00, -7.7692e+00,  …, -6.6465e+00,
   -6.1525e+00, -1.3742e+01]],

 [[-1.8772e+01, -2.6496e+00, -8.8470e+00,  …, -2.8741e+00,
   -6.6016e+00, -3.6780e+01],
  [-1.8008e+01,  1.7065e+00, -6.3321e+00,  …,  3.5927e-01,
   -3.0538e+00, -3.3861e+01],
  [-2.1421e+01,  1.3023e+00, -2.8741e+00,  …,  3.5927e-01,
   -3.0538e+00, -3.3861e+01],
  …,
  [-3.2334e+00, -1.1766e+01, -9.6553e+00,  …, -1.6032e+01,
   -1.1497e+01, -1.6212e+01],
  [-2.1152e+01, -1.0643e+01, -1.2035e+01,  …, -1.5898e+01,
   -1.8008e+01, -4.4864e+01],
  [-1.9086e+01, -8.3081e+00, -6.5117e+00,  …, -8.9368e+00,
   -8.3081e+00, -4.4010e+01]],

 [[ 2.1736e+01,  3.8621e+00, -3.0089e+00,  …, -3.0089e+00,
   -3.4130e+00, -1.5359e+01],
  [ 2.3577e+01,  1.1497e+01,  2.8741e+00,  …,  9.8799e-01,
    4.4909e-01, -7.2752e+00],
  [ 2.3442e+01,  1.0104e+01,  6.7812e+00,  …,  9.8799e-01,
    4.4909e-01, -7.2752e+00],
  …,
  [ 2.3936e+01,  2.7349e+01,  9.7901e+00,  …, -3.0089e+00,
    4.1316e+00,  1.3203e+01],
  [ 1.5987e+01,  2.0972e+01,  1.0015e+01,  …,  7.3201e+00,
    5.7483e+00, -4.1316e+00],
  [-2.5149e+00,  5.8381e-01, -1.8412e+00,  …, -5.7483e+00,
   -5.3890e+00, -2.3936e+01]]],


[[[ 3.7050e+01,  3.6196e+01,  3.3098e+01,  …,  3.2110e+01,
    3.2110e+01,  3.1391e+01],
  [ 4.0148e+01,  3.7903e+01,  3.9250e+01,  …,  3.7139e+01,
    3.7139e+01,  3.4624e+01],
  [ 5.1959e+01,  6.1525e+01,  6.7677e+01,  …,  6.7093e+01,
    6.7093e+01,  6.6959e+01],
  …,
  [ 4.2708e+01,  3.8981e+01,  3.6466e+01,  …,  2.9415e+01,
    2.7484e+01,  2.6990e+01],
  [ 4.2663e+01,  3.9385e+01,  3.5882e+01,  …,  2.4296e+01,
    2.3802e+01,  2.4475e+01],
  [ 3.8352e+01,  3.5747e+01,  3.5657e+01,  …,  3.1301e+01,
    3.1660e+01,  3.2379e+01]],
```

```
[[-9.5206e+00, -1.1407e+01, -8.3530e+00,  …, -2.7843e+00,
  -2.7843e+00, -1.7963e+00],
 [ 5.7438e+01,  1.0113e+02,  1.2242e+02,  …,  1.2790e+02,
   1.2790e+02,  1.2377e+02],
 [-1.3607e+01, -3.4265e+01, -2.7080e+01,  …, -3.1975e+01,
  -3.1975e+01, -2.8158e+01],
 …,
 [ 6.7363e-01, -1.7739e+01, -1.2125e+00,  …, -1.8368e+01,
  -2.2993e+01, -2.1601e+01],
 [-6.6465e+00, -3.9025e+01, -2.7304e+01,  …, -6.5566e+00,
  -6.9608e+00, -5.2992e+00],
 [ 2.9191e+00, -1.1227e+01,  8.3979e+00,  …,  1.6437e+01,
   1.8323e+01,  1.3607e+01]],

[[-1.1946e+01, -1.5673e+01, -1.1182e+01,  …, -1.1272e+01,
  -1.1272e+01, -1.4999e+01],
 [-1.2350e+01,  2.8292e+00,  5.3890e-01,  …, -1.3922e+00,
  -1.3922e+00, -1.2574e+00],
 [-6.5117e+00, -5.7483e+00, -1.2709e+01,  …, -1.7829e+01,
  -1.7829e+01, -1.4999e+01],
 …,
 [-3.5478e+01, -4.6256e+00,  9.0266e+00,  …, -1.2080e+01,
  -1.6616e+00,  6.6914e+00],
 [-3.3816e+01, -4.8052e+00,  4.1765e+00,  …,  1.3697e+01,
   1.6751e+01,  1.8368e+01],
 [-3.3502e+01,  2.8292e+00,  1.1182e+01,  …,  2.2993e+01,
   1.5359e+01,  1.9670e+01]],

…,

[[-9.7002e+00, -1.6796e+01, -1.5179e+01,  …, -1.3697e+01,
  -1.3697e+01, -1.7649e+01],
 [-2.3802e+00, -2.0613e+01, -2.4026e+01,  …, -2.3218e+01,
  -2.3218e+01, -3.0807e+01],
 [ 1.3248e+01,  7.0955e+00,  3.9519e+00,  …,  1.5269e+00,
   1.5269e+00, -7.4997e+00],
 …,
 [-1.2260e+01, -2.1960e+01, -1.6077e+01,  …, -1.1991e+01,
  -1.8368e+01, -1.5763e+01],
 [-9.4757e+00, -2.1287e+01, -1.5089e+01,  …, -2.6047e+01,
  -2.9370e+01, -3.3726e+01],
 [-1.3697e+01, -2.3218e+01, -1.6975e+01,  …, -2.1870e+01,
  -1.8862e+01, -2.7529e+01]],

[[-5.0298e+00, -8.0386e+00, -8.0386e+00,  …, -6.0626e+00,
  -6.0626e+00, -3.8846e+01],
 [ 3.8532e+01, -1.8592e+01,  4.1316e+00,  …, -2.2454e-01,
  -2.2454e-01, -9.2062e+00],
```

```
  [ 6.3501e+01, -2.1107e+01,  2.4251e+00,  …,  1.0329e+00,
    1.0329e+00,  2.8921e+01],
  …,
  [ 3.9879e+01, -9.5655e+00, -8.3530e+00,  …, -2.2005e+00,
   -2.4251e+00,  4.1316e+00],
  [ 3.8307e+01, -7.9039e+00, -4.6705e+00,  …, -6.5566e+00,
   -6.4219e+00,  1.0015e+01],
  [ 3.3816e+01, -8.8470e+00, -2.7394e+00,  …, -1.8862e+00,
   -7.9937e+00,  1.7110e+01]],

 [[-5.2992e+00, -2.9640e+00, -4.1316e+00,  …, -7.1405e+00,
   -7.1405e+00, -2.7394e+01],
  [-8.0835e+00,  1.0823e+01, -6.8710e+00,  …, -1.0284e+01,
   -1.0284e+01, -2.0119e+01],
  [-1.2260e+01,  3.1930e+01, -8.2183e+00,  …, -1.0329e+01,
   -1.0329e+01,  1.2574e+00],
  …,
  [ 6.3321e+00,  5.5193e+01,  8.9818e-02,  …,  1.8323e+01,
    1.9984e+01,  8.9817e-01],
  [ 4.3561e+00,  5.4699e+01,  1.9311e+00,  …,  6.2423e+00,
    7.0506e+00, -5.7034e+00],
  [ 2.2903e+00,  5.9998e+01,  6.6914e+00,  …, -4.9399e-01,
    8.9818e-02, -1.7963e+00]]],


…,


[[[ 2.5373e+01,  2.6182e+01,  2.3263e+01,  …,  2.9460e+01,
    2.9505e+01,  2.7259e+01],
  [ 2.6765e+01,  2.9146e+01,  3.0762e+01,  …,  3.3232e+01,
    3.1481e+01,  2.7170e+01],
  [ 2.9325e+01,  2.7484e+01,  3.1436e+01,  …,  3.0897e+01,
    2.9640e+01,  2.7753e+01],
  …,
  [ 3.5568e+01,  3.6825e+01,  3.7229e+01,  …,  3.6825e+01,
    3.9160e+01,  3.7139e+01],
  [ 3.2648e+01,  3.1930e+01,  3.2199e+01,  …,  3.0583e+01,
    3.1167e+01,  3.1975e+01],
  [ 3.0717e+01,  2.9370e+01,  2.9415e+01,  …,  2.8517e+01,
    2.8786e+01,  2.8921e+01]],

 [[ 4.0867e+00, -3.7274e+00, -7.3650e+00,  …,  4.5807e+00,
   -8.5775e+00, -1.9041e+01],
  [ 2.2454e+00, -6.8710e+00, -8.1733e+00,  …, -2.6316e+01,
   -1.4371e+01, -1.8682e+01],
  [-4.4370e+01, -3.5433e+01, -2.2409e+01,  …, -2.8382e+01,
   -1.1946e+01, -1.6167e+01],
```

```
 …,
 [-8.6808e+01, -8.5596e+01, -8.7572e+01,  …, -7.8410e+01,
  -7.1494e+01, -4.6660e+01],
 [ 2.3352e+00,  5.8830e+00,  5.9728e+00,  …,  6.4219e+00,
    8.8470e+00, -5.8381e-01],
 [-4.4459e+00, -4.2663e+00, -4.3112e+00,  …, -4.0867e+00,
  -4.9848e+00, -2.2454e+00]],

[[-2.1062e+01, -1.1362e+01, -1.7065e+00,  …, -1.7963e-01,
  -4.4909e+00, -2.2454e-01],
 [-7.8141e+00, -2.4251e+00,  1.4730e+01,  …,  9.2961e+00,
  -4.4909e-02,  2.4251e+00],
 [ 1.3383e+01,  1.3877e+01,  3.4445e+01,  …, -7.1854e-01,
  -4.6256e+00,  2.5149e+00],
 …,
 [-1.7200e+01, -2.1691e+01, -2.6092e+01,  …, -3.1660e+01,
  -3.1795e+01, -2.7170e+01],
 [-2.6002e+01, -2.6496e+01, -2.5912e+01,  …, -2.6990e+01,
  -2.4475e+01, -1.8682e+01],
 [-2.0658e+01, -1.9670e+01, -1.8008e+01,  …, -1.6526e+01,
  -1.6212e+01, -2.5014e+01]],

…,

[[-9.1164e+00, -1.3877e+01, -3.0268e+01,  …, -1.0553e+01,
  -1.2619e+01, -1.6616e+00],
 [-1.0464e+01, -2.4700e+00, -6.1974e+00,  …, -1.4281e+01,
  -1.3967e+01, -9.3410e+00],
 [-1.9400e+01, -1.4955e+01, -4.7603e+00,  …, -1.6661e+01,
  -2.0748e+01, -1.3832e+01],
 …,
 [-5.9279e+00, -3.9969e+00, -3.2783e+00,  …,  2.2005e+00,
    6.6465e+00, -7.1854e-01],
 [-1.0733e+01, -9.3859e+00, -9.4308e+00,  …, -7.7692e+00,
  -2.8292e+00,  2.2903e+00],
 [-9.1164e+00, -8.0835e+00, -7.3650e+00,  …, -9.0715e+00,
  -7.0057e+00, -6.1076e+00]],

[[-2.7709e+01,  3.2918e+01, -1.4865e+01,  …, -1.8143e+01,
    2.5239e+01, -7.8231e+01],
 [-2.2140e+01,  8.5326e-01,  2.2948e+01,  …, -2.6047e+01,
    2.3352e+01, -7.4997e+01],
 [ 3.0538e+00, -3.7903e+01,  2.3891e+01,  …, -2.1601e+01,
    2.9191e+00, -6.5028e+01],
 …,
 [-1.7559e+01, -9.8799e+00, -6.9159e+00,  …, -5.5237e+00,
    4.2663e+00, -5.1869e+01],
 [-1.8906e+01, -7.4997e+00, -7.3201e+00,  …, -5.3890e+00,
```

```
   -7.8590e+00, -3.6062e+01],
  [-1.7425e+01, -5.1196e+00, -5.4339e+00,  …, -4.4010e+00,
   -5.7932e+00, -3.8397e+01]],

 [[-1.9400e+01,  7.3650e+00,  2.0972e+01,  …, -1.8143e+01,
   -1.0733e+01, -1.1721e+01],
  [-2.6990e+01, -7.5446e+00,  1.0688e+01,  …, -2.0927e+01,
   -1.3293e+01, -1.1227e+01],
  [-3.3502e+01, -2.2320e+01, -1.2305e+01,  …, -1.9805e+01,
   -1.1991e+01, -1.4640e+01],
  …,
  [-1.8862e+00,  1.3473e-01,  7.4099e+00,  …,  4.3112e+00,
    2.2050e+01, -4.7603e+00],
  [ 4.5358e+00,  8.7572e+00,  1.1811e+01,  …,  1.0374e+01,
    1.5044e+01, -6.2423e+00],
  [-5.1645e+00, -1.0778e+00, -1.6616e+00,  …, -7.6344e-01,
   -1.4371e+00, -1.6212e+01]]],


[[[ 4.3292e+01,  4.2798e+01,  4.2304e+01,  …,  4.0777e+01,
    3.6241e+01,  3.5882e+01],
  [ 4.0507e+01,  3.9834e+01,  3.9430e+01,  …,  4.0373e+01,
    3.3142e+01,  3.2020e+01],
  [ 4.0507e+01,  3.9295e+01,  3.8891e+01,  …,  4.0822e+01,
    3.3232e+01,  3.2020e+01],
  …,
  [ 3.3951e+01,  3.4175e+01,  3.4310e+01,  …,  3.7678e+01,
    3.2828e+01,  3.2020e+01],
  [ 3.8352e+01,  4.0373e+01,  3.9609e+01,  …,  4.0912e+01,
    3.4580e+01,  3.2020e+01],
  [ 3.6466e+01,  3.8711e+01,  3.8352e+01,  …,  3.8217e+01,
    3.2738e+01,  2.8831e+01]],

 [[-1.3023e+00, -1.6167e+00, -2.4251e+00,  …,  1.1227e+00,
   -1.3922e+01, -1.1631e+01],
  [-1.0149e+01, -1.0329e+01, -8.8470e+00,  …,  2.5149e+00,
   -3.5029e+00, -1.3023e+00],
  [-8.1733e+00, -9.8350e+00, -9.7002e+00,  …,  4.8501e+00,
   -2.6047e+00, -1.3023e+00],
  …,
  [-1.6616e+01, -1.8906e+01, -2.0972e+01,  …,  2.5598e+00,
   -3.9969e+00, -1.3023e+00],
  [-1.6167e+01, -1.6975e+01, -1.4820e+01,  …, -4.9848e+00,
   -3.9070e+00, -1.3023e+00],
  [-1.1640e+02, -1.1667e+02, -1.1842e+02,  …, -1.8188e+01,
   -2.1107e+00,  1.2125e+00]],

 [[-6.9159e+00, -8.9817e+00, -1.2170e+01,  …, -5.4339e+00,
```

```
  -1.5269e+01, -1.4236e+01],
 [-2.5598e+00, -4.7603e+00, -4.0418e+00,  …, -1.3787e+01,
  -2.4116e+01, -2.3891e+01],
 [-2.5598e+00, -5.2094e+00, -1.7065e+00,  …, -1.4056e+01,
  -2.4475e+01, -2.3891e+01],
 …,
 [ 8.7123e+00,  1.1047e+01,  1.2529e+01,  …, -2.7394e+00,
  -2.2993e+01, -2.3891e+01],
 [ 1.5853e+01,  1.7874e+01,  1.9356e+01,  …, -6.9608e+00,
  -2.4969e+01, -2.3891e+01],
 [-4.8501e+00,  1.6275e-06,  3.7274e+00,  …, -1.9625e+01,
  -2.9460e+01, -2.8337e+01]],

…,

[[ 4.9399e-01, -6.5566e+00, -8.0386e+00,  …, -2.2275e+01,
  -7.0057e+00, -7.6794e+00],
 [-6.4668e+00, -5.1196e+00, -8.7123e+00,  …, -1.6661e+01,
  -5.3441e+00, -3.9519e+00],
 [-1.3607e+01, -9.4308e+00, -7.8590e+00,  …, -1.5449e+01,
  -5.2992e+00, -3.9519e+00],
 …,
 [-2.2140e+01, -1.9760e+01, -1.1721e+01,  …, -1.6437e+01,
  -7.9488e+00, -3.9519e+00],
 [-2.3397e+01, -2.4745e+01, -2.1826e+01,  …, -1.2844e+01,
  -8.2632e+00, -3.9519e+00],
 [-3.1885e+01, -3.1930e+01, -3.2738e+01,  …, -1.3787e+01,
  -9.3859e+00, -9.3410e+00]],

[[-3.9070e+00,  8.5326e-01, -7.3650e+00,  …,  1.3922e+00,
  -1.7559e+01, -3.8711e+01],
 [-4.3112e+00, -3.1436e-01, -3.7723e+00,  …,  3.0089e+00,
  -1.4011e+01, -3.5523e+01],
 [-6.0177e+00, -2.8741e+00,  3.5927e-01,  …,  3.3232e+00,
  -1.4281e+01, -3.5523e+01],
 …,
 [-2.2903e+00, -4.4909e-01, -7.1854e-01,  …, -1.0554e+01,
  -1.6392e+01, -3.5523e+01],
 [-6.5117e+00,  4.0418e-01, -5.7034e+00,  …, -2.0119e+01,
  -1.3652e+01, -3.5523e+01],
 [-1.6257e+01, -6.7363e+00, -8.1733e+00,  …, -2.9954e+01,
  -1.4461e+01, -3.9385e+01]],

[[ 1.0598e+01,  7.5895e+00,  4.4459e+00,  …, -6.2064e+01,
  -3.7274e+00, -1.8772e+01],
 [ 1.0598e+01,  4.8950e+00,  4.0418e+00,  …, -5.8067e+01,
  -4.4909e-01, -1.3248e+01],
 [ 1.3248e+01,  2.2454e-01,  2.0658e+00,  …, -5.8695e+01,
```

```
   -8.5326e-01, -1.3248e+01],
  …,
  [-8.9817e-01,  8.0835e-01, -1.0778e+00,  …, -3.8038e+01,
    8.1733e+00, -1.3248e+01],
  [ 4.9399e-01,  3.7274e+00,  6.7363e-01,  …, -3.7454e+01,
    6.7812e+00, -1.3248e+01],
  [-2.5598e+00, -3.0538e+00, -5.9728e+00,  …, -3.5702e+01,
   -1.3922e+00, -1.5987e+01]]],


[[[ 4.8097e+01,  4.8726e+01,  4.9444e+01,  …,  4.3606e+01,
    4.0957e+01,  3.6196e+01],
  [ 4.5896e+01,  4.4594e+01,  4.7783e+01,  …,  4.7738e+01,
    4.1630e+01,  3.4490e+01],
  [ 2.6721e+01,  2.7574e+01,  3.2783e+01,  …,  4.3157e+01,
    4.0777e+01,  3.4804e+01],
  …,
  [ 4.0642e+01,  3.7364e+01,  3.4804e+01,  …,  4.0463e+01,
    4.0597e+01,  3.4041e+01],
  [ 3.5343e+01,  3.6690e+01,  3.2918e+01,  …,  4.0867e+01,
    4.1136e+01,  3.4220e+01],
  [ 2.7394e+01,  3.1975e+01,  3.2469e+01,  …,  3.8981e+01,
    3.9564e+01,  3.1436e+01]],

 [[-1.6302e+01, -7.1405e+00,  1.0374e+01,  …, -7.4997e+00,
   -2.3802e+00, -1.2844e+01],
  [-3.1616e+01, -4.1361e+01, -5.3127e+01,  …, -1.9625e+01,
   -4.5358e+00, -1.0598e+01],
  [ 1.8412e+00, -2.5014e+01, -5.7977e+01,  …, -2.2275e+01,
   -2.4700e+00, -5.9728e+00],
  …,
  [-3.7499e+01, -4.1316e+01, -2.9056e+01,  …, -2.1915e+01,
    2.8292e+00, -3.7723e+00],
  [-1.9625e+01, -1.9176e+01, -3.1885e+00,  …, -1.5493e+01,
    5.6585e+00, -3.9969e+00],
  [-3.6421e+01, -5.0387e+01, -4.9175e+01,  …, -4.2933e+01,
   -1.0329e+00,  1.2574e+00]],

 [[-2.3263e+01, -2.6271e+01, -2.7798e+01,  …, -1.7110e+01,
   -4.9848e+00, -1.2979e+01],
  [-5.7034e+01, -5.5731e+01, -4.6031e+01,  …, -4.3112e+00,
   -6.1974e+00, -2.5283e+01],
  [-3.2244e+01, -2.7259e+01, -2.8472e+01,  …,  1.3832e+01,
   -5.7034e+00, -2.3532e+01],
  …,
  [-3.5029e+00, -2.3802e+00, -8.0386e+00,  …,  1.2125e+00,
   -1.1092e+01, -2.5553e+01],
  [-9.2512e+00, -3.7723e+00, -6.1076e+00,  …, -2.2454e-01,
```

```
    -1.0823e+01, -2.4520e+01],
   [ 4.7154e+00,  5.6585e+00,  1.2215e+01,  …,  1.8412e+00,
    -1.4461e+01, -2.7170e+01]],

  …,

  [[ 1.5763e+01,  1.1003e+01,  6.9159e+00,  …, -2.1736e+01,
    -1.1317e+01, -8.8021e+00],
   [ 2.5688e+01,  2.4026e+01,  2.1691e+01,  …, -6.7363e-01,
    -8.6224e+00, -2.9191e+00],
   [-6.8261e+00,  2.5149e+00,  7.6344e-01,  …,  5.9279e+00,
    -7.6345e-01, -2.6047e+00],
   …,
   [ 2.7394e+00, -8.8919e+00, -2.5239e+01,  …, -6.8261e+00,
    -6.5566e+00, -3.1436e+00],
   [-4.5807e+00, -5.8381e-01, -2.0074e+01,  …, -3.2334e+00,
    -5.2543e+00, -3.3681e+00],
   [-1.9850e+01, -7.1405e+00, -1.3068e+01,  …, -8.9817e-02,
    -6.3321e+00, -8.8470e+00]],

  [[-1.1227e+00, -1.3338e+01,  1.4999e+01,  …, -6.8081e+01,
     6.0177e+00, -4.9220e+01],
   [-2.7843e+00, -1.9356e+01,  3.0134e+01,  …, -6.3725e+01,
    -3.5927e-01, -4.8052e+01],
   [-1.4146e+01, -3.2783e+01,  2.4340e+01,  …, -5.9324e+01,
    -2.4700e+00, -5.0522e+01],
   …,
   [ 5.6899e+01, -6.6465e+00, -5.5956e+01,  …, -7.9264e+01,
     8.0835e-01, -4.9804e+01],
   [ 1.7829e+01,  1.8592e+01, -2.8292e+01,  …, -8.0207e+01,
     9.8799e-01, -4.9489e+01],
   [-1.4416e+01,  1.8278e+01,  1.9760e+00,  …, -6.9024e+01,
    -7.6344e+00, -5.2094e+01]],

  [[-1.0643e+01, -7.2752e+00, -1.4101e+01,  …, -3.7723e+00,
    -4.3516e+01, -2.0164e+01],
   [ 1.4820e+00,  6.4668e+00,  4.5358e+00,  …,  8.7572e+00,
    -4.9399e+01, -1.3967e+01],
   [ 5.4339e+00,  3.5927e-01, -1.9311e+00,  …, -4.0418e-01,
    -4.7468e+01, -1.3967e+01],
   …,
   [ 2.9640e+00,  4.2394e+01, -2.7080e+01,  …,  1.2574e+01,
    -5.0792e+01, -1.5359e+01],
   [-1.1452e+01,  4.8007e+01,  2.6496e+00,  …,  2.1960e+01,
    -5.0118e+01, -1.4910e+01],
   [-2.2903e+01,  2.4834e+01,  2.0433e+01,  …,  2.6047e+01,
    -5.2947e+01, -1.9400e+01]]]], device='cuda:0',
grad_fn=<MulBackward0>)
```

[9]:
```
conv_ref = torch.nn.Conv2d(in_channels = 64, out_channels=64, kernel_size = 3,␣
↪bias = False)
conv_ref.weight = model.features[3].weight_q

output_ref = conv_ref(x)
print(output_ref)
```

```
tensor([[[[ 3.7574e+01,  3.5596e+01,  3.4591e+01,  …,  3.4172e+01,
            3.4110e+01,  3.4823e+01],
          [ 3.4715e+01,  3.3057e+01,  3.1997e+01,  …,  3.2323e+01,
            3.3063e+01,  3.3822e+01],
          [ 3.5386e+01,  3.3531e+01,  3.1652e+01,  …,  3.2016e+01,
            3.1414e+01,  3.2671e+01],
          …,
          [ 4.8401e+01,  4.6134e+01,  4.5257e+01,  …,  2.9082e+01,
            2.8506e+01,  2.9465e+01],
          [ 3.6501e+01,  3.0779e+01,  2.9704e+01,  …,  3.0195e+01,
            2.9111e+01,  3.0249e+01],
          [ 3.3386e+01,  2.9133e+01,  2.8211e+01,  …,  2.8034e+01,
            2.8137e+01,  2.7297e+01]],

         [[-3.8149e+00,  1.5426e+01,  1.6959e+01,  …,  3.1023e+01,
            3.5726e+01,  3.6240e+01],
          [-4.9415e+00, -2.3626e+00, -4.4896e+00,  …,  2.2486e+01,
            2.2290e+01,  1.7581e+01],
          [-1.3813e+01, -5.1999e+00, -3.6968e+00,  …,  2.3966e+00,
           -8.5527e+00, -1.9398e+01],
          …,
          [-8.5223e+01, -8.5492e+01, -6.7132e+01,  …, -7.3215e+00,
           -8.8111e+00, -1.0079e+01],
          [ 1.0631e+01,  2.9586e+01,  2.9661e+01,  …, -1.3711e+01,
           -1.2912e+01, -8.0916e+00],
          [-2.6263e+01, -1.8167e+01, -2.1890e+01,  …, -2.9625e+01,
           -3.0759e+01, -2.2590e+01]],

         [[-1.1755e+01, -1.1677e+01, -1.1223e+01,  …, -7.9397e+00,
           -8.4850e+00, -1.1660e+01],
          [-1.6725e+01, -1.5585e+01, -1.8619e+01,  …, -1.7809e+01,
           -1.4351e+01, -1.3569e+01],
          [-1.9797e+01, -2.5979e+01, -2.7274e+01,  …, -1.9660e+01,
           -1.7860e+01, -1.4954e+01],
          …,
          [-1.8242e+01, -1.6738e+01, -2.0195e+01,  …, -1.7295e+00,
           -2.8496e+00, -3.0794e+00],
          [-1.7588e+01, -1.2339e+01, -1.7286e+01,  …,  9.3641e-01,
            5.5521e-01, -3.2349e+00],
          [-1.5112e+01, -1.6382e+01, -1.7292e+01,  …,  1.5393e+00,
```

```
  -1.4173e-01, -9.3086e+00]],

 …,

 [[-1.5415e+01, -1.0155e+01, -1.2702e+01,  …, -1.7062e+01,
   -1.3513e+01, -7.9224e+00],
  [-1.3860e+01, -2.6629e+00, -6.1452e+00,  …, -1.0919e+01,
   -9.2000e+00, -1.4336e+00],
  [-1.3332e+01, -1.3587e+00, -2.6777e+00,  …, -3.6098e+00,
   -1.0626e+01, -6.0631e-01],
  …,
  [-2.3620e+01, -1.7294e+01, -1.1856e+01,  …, -2.0397e+01,
   -1.6029e+01, -8.2189e+00],
  [-2.7708e+01, -1.9903e+01, -1.8923e+01,  …, -2.1961e+01,
   -2.0424e+01, -1.1799e+01],
  [-2.1663e+01, -8.2925e+00, -1.7069e+01,  …, -2.3679e+01,
   -2.4989e+01, -2.2507e+01]],

 [[-2.9996e+01,  1.6686e+00, -1.0783e+01,  …, -5.7538e+00,
   -5.7149e+00, -5.5862e+01],
  [-3.5692e+01,  1.3999e+01, -3.3789e+00,  …, -6.7188e+00,
   -4.6093e+00, -3.0251e+01],
  [-3.7302e+01,  1.2595e+01,  1.2391e+00,  …,  4.1575e+00,
   -1.5928e+01, -3.2709e+01],
  …,
  [-4.2341e+00, -1.4897e+01, -6.4697e+00,  …,  1.1247e+00,
   -7.8849e+00, -5.2035e+01],
  [-2.1250e+01, -1.9231e+00, -7.9816e+00,  …,  7.1797e+00,
   -7.1453e+00, -5.7010e+01],
  [-3.3156e+01,  5.2041e+00, -1.1999e+01,  …,  1.0608e-01,
   -4.0448e+00, -6.2240e+01]],

 [[-2.7983e+00,  2.9724e+00,  4.1569e+00,  …,  2.1082e+00,
   -1.9641e+00, -2.1512e+01],
  [ 2.8801e+00,  4.1995e+00,  1.0171e+01,  …,  9.6309e+00,
   -1.5388e+00, -1.8190e+01],
  [ 8.3302e+00,  4.7789e-01,  1.3140e+01,  …,  1.5191e+01,
    2.6189e+00, -2.0880e+01],
  …,
  [ 3.1235e+01,  2.1301e+01,  1.8094e+01,  …,  9.0099e-01,
    5.5464e+00, -2.8194e+01],
  [ 1.2274e+01,  1.2506e+01,  3.9691e+00,  …, -1.6236e-01,
    6.8805e+00, -2.9644e+01],
  [-1.0723e+00,  1.6690e+00,  1.3475e+00,  …, -5.3470e+00,
    4.5006e+00, -3.1848e+01]]],


[[[ 3.5497e+01,  3.3586e+01,  3.1377e+01,  …,  3.4976e+01,
```

```
   3.5157e+01,  3.8170e+01],
 [ 3.5019e+01,  3.1882e+01,  2.9860e+01,  …,  3.1546e+01,
   3.1670e+01,  3.4160e+01],
 [ 3.4536e+01,  3.1846e+01,  2.9830e+01,  …,  3.1546e+01,
   3.1670e+01,  3.4160e+01],
 …,
 [ 4.9897e+01,  4.8228e+01,  4.8806e+01,  …,  6.3270e+01,
   6.5068e+01,  7.1066e+01],
 [ 4.3582e+01,  3.7426e+01,  3.6713e+01,  …,  5.3115e+01,
   5.2610e+01,  5.2304e+01],
 [ 3.6521e+01,  3.2046e+01,  3.2971e+01,  …,  4.1197e+01,
   4.1358e+01,  4.1752e+01]],

[[-2.3129e+01, -6.4682e+00, -8.5747e+00,  …, -7.3567e+00,
  -7.9155e+00, -8.6777e+00],
 [-1.5636e+01, -4.6583e+00, -6.4283e+00,  …, -3.7280e+00,
  -4.4674e+00, -1.4245e+00],
 [-1.3984e+01, -3.3889e+00, -5.0614e+00,  …, -3.7280e+00,
  -4.4674e+00, -1.4245e+00],
 …,
 [-1.1937e+02, -1.3546e+02, -1.3876e+02,  …, -2.5179e+02,
  -2.4927e+02, -2.2439e+02],
 [-1.4322e+00,  7.2299e+00,  6.0545e+00,  …,  6.9208e-01,
  -8.6558e-02,  8.0017e+00],
 [-3.0393e+01, -2.4892e+01, -2.2885e+01,  …, -1.8570e+01,
  -1.8504e+01, -2.3378e+01]],

[[-3.0994e+00, -3.4276e+00, -4.5720e+00,  …, -1.0890e+01,
  -6.3209e+00, -9.9105e+00],
 [-1.2640e+01, -1.4092e+01, -1.4571e+01,  …, -1.8448e+01,
  -1.4510e+01, -2.3302e+01],
 [-1.0977e+01, -1.3968e+01, -1.3711e+01,  …, -1.8448e+01,
  -1.4510e+01, -2.3302e+01],
 …,
 [-3.3824e+01, -4.9546e+01, -4.7235e+01,  …, -1.5528e+01,
  -2.2865e+01, -1.6578e+01],
 [-4.8670e+01, -4.5656e+01, -3.9812e+01,  …, -2.8514e+01,
  -2.9526e+01, -2.6727e+01],
 [-2.1037e+01, -2.1429e+01, -1.9691e+01,  …, -1.5725e+01,
  -1.6065e+01, -2.0521e+01]],

…,

[[-2.2018e+01, -1.6334e+01, -1.9574e+01,  …, -8.3187e+00,
  -5.2219e+00, -2.7343e+00],
 [-1.4749e+01, -5.8483e+00, -1.3099e+01,  …, -4.6204e+00,
  -1.7352e+00,  2.6885e-01],
 [-1.7114e+01, -4.8474e+00, -1.1208e+01,  …, -4.6204e+00,
```

```
  -1.7352e+00,  2.6885e-01],
 …,
 [ 1.9084e+00,  1.6868e+01,  2.0950e+01,  …,  4.8817e+00,
   7.9481e+00,  4.0146e+00],
 [-2.8871e+01, -1.0425e+01, -1.3038e+01,  …, -4.0736e+01,
  -4.0571e+01, -2.8435e+01],
 [-2.6205e+01, -1.0679e+01, -1.2010e+01,  …, -1.0696e+01,
  -1.0424e+01, -8.7038e+00]],

[[-2.9247e+01, -3.3879e+00, -8.4980e+00,  …, -3.5531e+00,
  -7.0380e+00, -6.0654e+01],
 [-2.8525e+01,  1.1327e+00, -6.6125e+00,  …,  5.3217e-02,
  -3.1639e+00, -5.6849e+01],
 [-3.1789e+01,  1.5937e+00, -3.6937e+00,  …,  5.3217e-02,
  -3.1639e+00, -5.6849e+01],
 …,
 [ 4.2506e+00, -1.8585e+01, -8.1504e+00,  …, -1.5922e+01,
  -9.7621e+00, -2.9163e+01],
 [-2.3758e+01, -1.3629e+01, -1.2863e+01,  …, -1.9745e+01,
  -2.2461e+01, -8.3035e+01],
 [-3.1277e+01, -1.0648e+01, -9.1299e+00,  …, -1.4149e+01,
  -1.3946e+01, -7.2885e+01]],

[[ 2.6271e+01,  5.0073e+00, -2.2489e+00,  …, -2.0211e+00,
  -2.5746e+00, -1.7703e+01],
 [ 2.8548e+01,  1.1100e+01,  3.3648e+00,  …,  7.7989e-01,
   5.0338e-01, -1.1996e+01],
 [ 2.8512e+01,  9.8936e+00,  7.1181e+00,  …,  7.7989e-01,
   5.0338e-01, -1.1996e+01],
 …,
 [ 3.9388e+01,  4.0637e+01,  9.5987e+00,  …, -1.0020e+00,
   4.0035e+00,  2.0070e+01],
 [ 2.1806e+01,  3.4133e+01,  1.7064e+01,  …,  2.7753e+01,
   2.4113e+01,  8.4956e+00],
 [-4.4630e+00,  1.3724e+00, -1.0145e+00,  …, -4.0042e+00,
  -4.0132e+00, -2.4908e+01]]],


[[[ 4.4032e+01,  4.2789e+01,  4.0851e+01,  …,  3.9524e+01,
    3.9524e+01,  3.9337e+01],
  [ 6.1214e+01,  5.2126e+01,  5.6984e+01,  …,  5.0169e+01,
    5.0169e+01,  4.6427e+01],
  [ 8.4146e+01,  9.3893e+01,  1.1184e+02,  …,  1.0193e+02,
    1.0193e+02,  1.0282e+02],
  …,
  [ 5.7198e+01,  4.8367e+01,  4.2823e+01,  …,  3.0764e+01,
    2.7688e+01,  2.6991e+01],
  [ 5.6405e+01,  4.8595e+01,  4.2388e+01,  …,  2.4600e+01,
```

```
   2.4041e+01,  2.4590e+01],
 [ 5.2025e+01,  4.4397e+01,  4.1316e+01,  …,  3.1703e+01,
   3.1868e+01,  3.2817e+01]],

[[-2.1976e+01, -3.3823e+01, -3.4328e+01,  …, -2.5695e+01,
  -2.5695e+01, -2.6116e+01],
 [ 1.3913e+02,  2.4578e+02,  2.9191e+02,  …,  2.8620e+02,
   2.8620e+02,  2.7870e+02],
 [-5.6406e+01, -1.1621e+02, -1.1602e+02,  …, -1.1162e+02,
  -1.1162e+02, -1.1256e+02],
 …,
 [-2.3945e-01, -1.9250e+01,  2.0779e+00,  …, -1.8279e+01,
  -2.2441e+01, -2.1262e+01],
 [-1.2449e+01, -4.6650e+01, -2.7414e+01,  …, -6.7393e+00,
  -6.9458e+00, -4.1944e+00],
 [-1.1467e+01, -1.0092e+01,  1.5810e+01,  …,  1.7480e+01,
   1.8548e+01,  1.4337e+01]],

[[-5.4257e+00, -1.5139e+01, -9.8056e+00,  …, -1.0374e+01,
  -1.0374e+01, -2.1525e+01],
 [-1.1452e-01,  1.5348e+01,  2.9529e+01,  …,  1.9203e+01,
   1.9203e+01,  1.6341e+01],
 [-6.4100e+00, -7.6922e+00, -8.1035e+00,  …, -1.9656e+01,
  -1.9656e+01, -1.5132e+01],
 …,
 [-4.8682e+01, -7.9613e+00,  1.1411e+01,  …, -1.3138e+01,
   6.1898e-01,  6.9250e+00],
 [-4.5404e+01, -9.1168e+00,  3.9256e+00,  …,  1.4776e+01,
   1.6619e+01,  1.8321e+01],
 [-4.7207e+01,  2.4821e+00,  1.6918e+01,  …,  2.2479e+01,
   1.6509e+01,  2.0448e+01]],

…,

[[-1.7121e+01, -1.2736e+01, -2.0469e+01,  …, -1.7480e+01,
  -1.7480e+01, -2.1045e+01],
 [-1.5966e+01, -3.7832e+01, -4.9463e+01,  …, -4.2025e+01,
  -4.2025e+01, -6.3248e+01],
 [ 1.3794e+01,  8.7233e+00, -5.9123e+00,  …, -1.9956e+01,
  -1.9956e+01, -3.0689e+01],
 …,
 [-2.1077e+01, -3.1896e+01, -1.7934e+01,  …, -1.3246e+01,
  -2.0224e+01, -1.5652e+01],
 [-1.7558e+01, -2.9969e+01, -1.7780e+01,  …, -2.6443e+01,
  -2.9246e+01, -3.4177e+01],
 [-2.5025e+01, -3.4328e+01, -2.0945e+01,  …, -2.1183e+01,
  -1.8136e+01, -2.7680e+01]],
```

```
 [[-1.2962e+01, -1.4161e+01, -1.7304e+01,  …, -1.4239e+01,
   -1.4239e+01, -7.6306e+01],
  [ 1.1520e+02, -7.4808e+01,  1.5301e+01,  …,  2.4026e-01,
    2.4026e-01, -2.2397e+01],
  [ 1.6077e+02, -4.9910e+01, -3.6189e+00,  …, -9.9715e-01,
   -9.9715e-01,  7.9389e+01],
  …,
  [ 6.8438e+01, -1.4944e+01, -1.3454e+01,  …, -8.9813e-01,
   -1.2609e+00,  3.6759e+00],
  [ 6.3487e+01, -1.0052e+01, -1.1302e+01,  …, -6.5326e+00,
   -6.2640e+00,  9.8745e+00],
  [ 6.0168e+01, -1.8789e+01, -5.7630e+00,  …, -1.3508e+00,
   -9.4151e+00,  1.7503e+01]],

 [[-1.1768e+01, -7.7736e+00, -4.5978e+00,  …, -8.4359e+00,
   -8.4359e+00, -3.0824e+01],
  [-2.4051e+01,  2.6993e+01, -1.1031e+01,  …, -2.0616e+01,
   -2.0616e+01, -2.0845e+01],
  [-4.3312e+01,  8.0368e+01, -9.4370e+00,  …, -2.1324e+01,
   -2.1324e+01,  2.4709e+01],
  …,
  [-9.8112e+00,  9.6693e+01,  5.5739e+00,  …,  2.0961e+01,
    2.1046e+01,  2.1457e-01],
  [-1.1620e+01,  9.5313e+01,  7.3666e+00,  …,  5.4936e+00,
    8.1006e+00, -6.1741e+00],
  [-1.3258e+01,  9.7094e+01,  1.1748e+01,  …, -1.8957e+00,
    3.8622e-01, -1.9030e+00]]],


…,


[[[ 2.5864e+01,  2.6185e+01,  2.3074e+01,  …,  3.0280e+01,
    3.0301e+01,  3.0064e+01],
  [ 2.8632e+01,  2.9291e+01,  3.0943e+01,  …,  3.3823e+01,
    3.2256e+01,  2.9427e+01],
  [ 3.0336e+01,  2.8148e+01,  3.2846e+01,  …,  3.0889e+01,
    3.0449e+01,  3.0054e+01],
  …,
  [ 3.6965e+01,  3.7860e+01,  3.8016e+01,  …,  3.7750e+01,
    4.0600e+01,  4.0006e+01],
  [ 3.4706e+01,  3.2888e+01,  3.2907e+01,  …,  3.1127e+01,
    3.1677e+01,  3.3977e+01],
  [ 3.5827e+01,  3.2015e+01,  3.2038e+01,  …,  3.1286e+01,
    3.1481e+01,  3.2270e+01]],

 [[ 1.2213e+00, -6.6107e+00, -9.0398e+00,  …,  3.1323e+00,
   -5.3554e+00, -1.7190e+01],
```

```
 [ 4.1176e+00, -6.5549e+00, -8.2191e+00,  …, -2.6817e+01,
  -1.4926e+01, -1.9232e+01],
 [-4.6005e+01, -3.4873e+01, -2.1065e+01,  …, -3.0584e+01,
  -1.2421e+01, -1.5913e+01],
 …,
 [-1.0196e+02, -1.0211e+02, -1.0526e+02,  …, -8.7233e+01,
  -7.6561e+01, -5.0122e+01],
 [ 8.1777e-01,  8.0825e+00,  7.7370e+00,  …,  7.7741e+00,
   1.0284e+01,  1.5119e+00],
 [-2.5236e+01, -1.4129e+01, -1.4620e+01,  …, -1.4577e+01,
  -1.6228e+01, -1.4423e+01]],

[[-2.2509e+01, -1.2506e+01, -1.9582e+00,  …,  8.3093e-01,
  -4.4484e+00,  9.7434e-01],
 [-7.6593e+00, -1.8976e+00,  1.5941e+01,  …,  1.0952e+01,
  -1.4894e+00,  2.6293e+00],
 [ 1.6450e+01,  1.4087e+01,  3.3728e+01,  …,  7.8632e-01,
  -5.2178e+00,  2.8837e+00],
 …,
 [-1.8336e+01, -2.4510e+01, -2.8454e+01,  …, -3.3509e+01,
  -3.3050e+01, -2.6983e+01],
 [-2.5597e+01, -2.8744e+01, -2.9264e+01,  …, -2.7211e+01,
  -2.4136e+01, -2.1127e+01],
 [-1.7475e+01, -2.1298e+01, -2.0276e+01,  …, -1.9142e+01,
  -1.8596e+01, -3.1950e+01]],

…,

[[-1.2945e+01, -1.4092e+01, -3.1354e+01,  …, -1.1045e+01,
  -1.5987e+01,  2.1360e+00],
 [-1.3646e+01, -3.7258e+00, -6.5502e+00,  …, -1.2843e+01,
  -1.5095e+01, -7.8689e+00],
 [-2.1720e+01, -1.5115e+01, -2.2327e+00,  …, -1.6175e+01,
  -2.1738e+01, -1.1562e+01],
 …,
 [-3.2871e+00,  7.0145e-01,  1.9448e+00,  …,  4.6849e+00,
   6.4720e+00,  2.7754e+00],
 [-1.6398e+01, -1.2155e+01, -1.2131e+01,  …, -8.9475e+00,
  -3.9827e+00,  6.6850e+00],
 [-1.9474e+01, -9.9270e+00, -9.1442e+00,  …, -9.9007e+00,
  -7.4689e+00, -4.3471e+00]],

[[-2.8556e+01,  3.4330e+01, -1.4565e+01,  …, -1.6528e+01,
   3.1437e+01, -1.1218e+02],
 [-2.0948e+01,  9.8370e-01,  2.3286e+01,  …, -2.4642e+01,
   2.8215e+01, -1.0494e+02],
 [ 3.7546e+00, -3.9517e+01,  2.7187e+01,  …, -2.2171e+01,
   7.2439e+00, -9.3917e+01],
```

```
  …,
   [-2.3493e+01, -9.2038e+00, -6.9605e+00,  …, -5.6332e+00,
     5.4664e+00, -7.7854e+01],
   [-2.6755e+01, -8.4343e+00, -7.9921e+00,  …, -6.3046e+00,
    -8.1543e+00, -6.0705e+01],
   [-2.7668e+01, -7.7890e+00, -8.0993e+00,  …, -7.2818e+00,
    -8.7186e+00, -6.4823e+01]],

  [[-1.9865e+01,  7.7563e+00,  2.2805e+01,  …, -2.0284e+01,
    -4.1947e+00, -1.2304e+01],
   [-2.8156e+01, -7.8273e+00,  1.0129e+01,  …, -2.4868e+01,
    -9.5650e+00, -1.2014e+01],
   [-3.4534e+01, -2.2975e+01, -1.5712e+01,  …, -2.3240e+01,
    -8.9771e+00, -1.5284e+01],
   …,
   [-2.5387e+00,  3.0179e-01,  8.1551e+00,  …,  4.0526e+00,
     2.4576e+01, -3.6053e+00],
   [ 3.7749e+00,  1.0673e+01,  1.4476e+01,  …,  1.1084e+01,
     1.6237e+01, -6.5256e+00],
   [-5.5410e+00, -1.1648e+00, -1.2319e+00,  …, -1.6517e+00,
    -1.6393e+00, -2.0568e+01]]],


 [[[ 4.3279e+01,  4.2365e+01,  4.1717e+01,  …,  5.2630e+01,
     3.8585e+01,  3.8432e+01],
   [ 4.0443e+01,  3.9213e+01,  3.8741e+01,  …,  5.3843e+01,
     3.7426e+01,  3.4206e+01],
   [ 4.0553e+01,  3.8862e+01,  3.8585e+01,  …,  5.5143e+01,
     3.7937e+01,  3.4206e+01],
   …,
   [ 3.5298e+01,  3.5826e+01,  3.5847e+01,  …,  4.9290e+01,
     3.7170e+01,  3.4206e+01],
   [ 4.5053e+01,  4.9262e+01,  4.8495e+01,  …,  5.4250e+01,
     3.9361e+01,  3.4206e+01],
   [ 4.5768e+01,  4.8592e+01,  4.8297e+01,  …,  4.8449e+01,
     3.9688e+01,  3.2418e+01]],

  [[-1.4461e+00, -1.1158e+00, -1.5851e+00,  …,  8.3318e+00,
    -1.6905e+01, -8.4475e+00],
   [-1.0165e+01, -9.9224e+00, -7.9748e+00,  …,  1.2319e+01,
    -7.8140e+00, -2.7141e-01],
   [-7.7374e+00, -1.0565e+01, -1.0693e+01,  …,  1.5642e+01,
    -7.7729e+00, -2.7141e-01],
   …,
   [-1.7928e+01, -1.9936e+01, -2.2601e+01,  …,  8.6124e+00,
    -6.5537e+00, -2.7141e-01],
   [-1.2282e+01, -1.3701e+01, -1.1120e+01,  …, -1.2719e+00,
    -1.0273e+01, -2.7141e-01],
```

```
 [-1.9118e+02, -1.8834e+02, -1.8942e+02,  …, -3.7713e+01,
  -1.6525e+01, -1.1356e+01]],

[[-6.9536e+00, -1.0834e+01, -1.2820e+01,  …,  1.2636e+00,
  -1.0420e+01, -1.6620e+01],
 [-3.3131e+00, -4.9640e+00, -3.9228e+00,  …, -8.5584e+00,
  -1.8728e+01, -2.7247e+01],
 [-2.3871e+00, -4.9599e+00, -2.2999e+00,  …, -8.9740e+00,
  -1.8973e+01, -2.7247e+01],
 …,
 [ 1.4649e+01,  1.6362e+01,  1.8406e+01,  …,  2.1525e+00,
  -1.8682e+01, -2.7247e+01],
 [ 2.4737e+01,  2.4446e+01,  2.5914e+01,  …, -2.0870e+00,
  -2.0002e+01, -2.7247e+01],
 [-1.4479e+01, -1.2865e+01, -1.0513e+01,  …, -2.4163e+01,
  -2.5844e+01, -3.5097e+01]],

…,

[[ 5.1622e-02, -5.4441e+00, -8.0419e+00,  …, -3.1869e+01,
  -1.7388e+01, -3.1821e+00],
 [-6.3459e+00, -5.8704e+00, -9.2180e+00,  …, -2.5154e+01,
  -1.6093e+01, -2.3037e+00],
 [-1.3595e+01, -9.0477e+00, -7.9977e+00,  …, -2.3929e+01,
  -1.6684e+01, -2.3037e+00],
 …,
 [-2.5045e+01, -2.5559e+01, -1.8217e+01,  …, -2.1046e+01,
  -1.6793e+01, -2.3037e+00],
 [-1.2729e+01, -1.4546e+01, -1.1701e+01,  …, -1.9645e+01,
  -1.5608e+01, -2.3037e+00],
 [-3.1192e+01, -2.9993e+01, -3.0051e+01,  …, -1.6647e+01,
  -2.1453e+01, -8.4177e+00]],

[[-3.5370e+00,  4.8800e-02, -7.2864e+00,  …,  3.0600e+01,
  -3.1760e+01, -6.4392e+01],
 [-4.5001e+00, -4.2328e-01, -2.8520e+00,  …,  3.3183e+01,
  -2.8089e+01, -6.0521e+01],
 [-6.3432e+00, -2.2246e+00,  5.2496e-01,  …,  3.4527e+01,
  -2.8794e+01, -6.0521e+01],
 …,
 [-7.0434e-02,  2.3668e+00,  2.6784e-01,  …,  1.5165e+01,
  -2.8033e+01, -6.0521e+01],
 [-8.3407e+00, -1.5149e+00, -8.5569e+00,  …,  1.5819e+00,
  -2.1418e+01, -6.0521e+01],
 [-1.9802e+01, -4.6883e+00, -8.1568e+00,  …, -3.2126e+01,
  -1.6963e+01, -6.6446e+01]],

[[ 1.0499e+01,  7.5362e+00,  4.5120e+00,  …, -1.2095e+02,
```

```
   -6.8899e+00, -2.1434e+01],
  [ 1.1464e+01,  4.8587e+00,  3.6463e+00,  …, -1.2688e+02,
   -2.9890e+00, -1.7820e+01],
  [ 1.4352e+01,  2.2764e-01,  2.2633e+00,  …, -1.3110e+02,
   -3.3750e+00, -1.7820e+01],
  …,
  [-3.5609e+00, -1.7723e-01, -8.3912e-01,  …, -9.1411e+01,
    1.0305e+01, -1.7820e+01],
  [-2.5659e+00,  8.0480e-01, -5.8498e-01,  …, -8.7054e+01,
    6.4086e+00, -1.7820e+01],
  [-4.4898e+00, -3.9078e+00, -5.5840e+00,  …, -7.3619e+01,
   -1.4189e-01, -2.1036e+01]]],


[[[ 6.3458e+01,  6.3068e+01,  6.4235e+01,  …,  5.5559e+01,
    5.4354e+01,  3.9204e+01],
  [ 5.0410e+01,  4.8587e+01,  5.2952e+01,  …,  5.7462e+01,
    5.5390e+01,  3.8737e+01],
  [ 2.7040e+01,  2.8139e+01,  3.4742e+01,  …,  4.8100e+01,
    5.4676e+01,  4.0765e+01],
  …,
  [ 4.0683e+01,  3.7476e+01,  3.5010e+01,  …,  4.4546e+01,
    5.3457e+01,  3.9986e+01],
  [ 3.5293e+01,  3.6845e+01,  3.3207e+01,  …,  4.4803e+01,
    5.4569e+01,  4.0258e+01],
  [ 2.7502e+01,  3.2050e+01,  3.2553e+01,  …,  4.3998e+01,
    5.2491e+01,  3.9313e+01]],

 [[-6.1236e+01, -4.7717e+01, -2.4899e+01,  …, -3.5964e+01,
   -4.6837e+00, -1.3574e+01],
  [-3.7093e+01, -4.7721e+01, -6.4568e+01,  …, -2.2207e+01,
   -2.6406e+00, -1.3851e+01],
  [ 2.0613e+00, -2.6510e+01, -6.2307e+01,  …, -2.3947e+01,
    5.0152e+00, -8.8036e+00],
  …,
  [-3.7221e+01, -4.1347e+01, -2.9357e+01,  …, -2.2269e+01,
    1.0605e+01, -6.2447e+00],
  [-1.9342e+01, -1.8912e+01, -2.9577e+00,  …, -1.5910e+01,
    1.2243e+01, -7.8422e+00],
  [-3.6796e+01, -5.0361e+01, -4.9514e+01,  …, -5.3232e+01,
   -1.3115e+01, -2.1989e+01]],

 [[-2.9494e+01, -3.2471e+01, -3.3522e+01,  …, -2.3873e+01,
    2.1137e+00, -1.2792e+01],
  [-6.3604e+01, -6.3499e+01, -5.2480e+01,  …, -1.4456e+01,
    2.9814e+00, -2.4456e+01],
  [-3.3227e+01, -2.8118e+01, -3.0180e+01,  …,  6.5679e+00,
   -1.7958e-02, -2.1280e+01],
```

```
 …,
 [-3.3326e+00, -2.9271e+00, -7.6673e+00,  …, -2.6412e+00,
  -4.9737e+00, -2.4197e+01],
 [-9.0029e+00, -3.3322e+00, -6.0629e+00,  …, -3.5931e+00,
  -4.1861e+00, -2.3372e+01],
 [ 5.1670e+00,  5.2946e+00,  1.0967e+01,  …,  2.0810e+00,
  -9.5006e+00, -2.7218e+01]],

…,

[[ 1.1458e+01,  3.9721e+00, -6.7121e-01,  …, -4.9043e+01,
  -3.0877e+01, -4.9224e+00],
 [ 3.3152e+01,  3.2519e+01,  3.1770e+01,  …, -1.6882e+01,
  -1.9470e+01, -8.5533e+00],
 [-7.5828e+00,  3.4183e+00,  1.8861e+00,  …, -4.0005e+00,
  -8.1293e+00, -1.1649e+01],
 …,
 [ 1.9131e+00, -9.7531e+00, -2.4642e+01,  …, -1.5053e+01,
  -1.4925e+01, -1.2030e+01],
 [-4.7699e+00, -2.0149e+00, -2.0758e+01,  …, -1.0971e+01,
  -1.3047e+01, -1.1444e+01],
 [-2.0926e+01, -7.9046e+00, -1.2594e+01,  …, -9.3406e+00,
  -8.6811e+00, -2.0202e+01]],

[[-3.0101e+00, -1.6500e+01,  1.6828e+01,  …, -1.3903e+02,
   4.1480e+01, -8.7850e+01],
 [-2.2227e+00, -2.3857e+01,  3.4474e+01,  …, -1.4060e+02,
   3.1385e+01, -8.7011e+01],
 [-1.1115e+01, -3.8562e+01,  2.8728e+01,  …, -1.3086e+02,
   2.4253e+01, -8.9239e+01],
 …,
 [ 5.7656e+01, -6.5689e+00, -5.8145e+01,  …, -1.4404e+02,
   2.7527e+01, -8.7017e+01],
 [ 1.7227e+01,  1.9517e+01, -2.9320e+01,  …, -1.4599e+02,
   2.8625e+01, -8.6774e+01],
 [-1.4572e+01,  1.8537e+01,  1.9882e+00,  …, -1.1989e+02,
   1.2790e+01, -8.8645e+01]],

[[-1.1828e+01, -8.4513e+00, -1.4800e+01,  …,  9.8326e+00,
  -8.8125e+01, -2.7345e+01],
 [ 7.9357e+00,  1.1356e+01,  8.0939e+00,  …,  4.0442e+01,
  -1.1140e+02, -2.2632e+01],
 [ 9.8853e+00,  1.1442e+00, -4.3349e+00,  …,  3.5553e+01,
  -1.1865e+02, -2.1689e+01],
 …,
 [ 3.3123e+00,  4.3334e+01, -2.7494e+01,  …,  4.3585e+01,
  -1.1314e+02, -2.2244e+01],
 [-1.1915e+01,  4.8733e+01,  2.7541e+00,  …,  5.4652e+01,
```

```
          -1.1253e+02, -2.2358e+01],
         [-2.2583e+01,  2.4901e+01,  2.0519e+01,  …,  5.5135e+01,
          -1.1066e+02, -2.5653e+01]]]], device='cuda:0',
       grad_fn=<CudnnConvolutionBackward>)
```

[10]:
```
difference = abs( output_ref - output_recovered )
print(difference.mean())  ## It should be small, e.g.,2.3 in my trainned model
```

```
tensor(4.1960, device='cuda:0', grad_fn=<MeanBackward0>)
```

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: