[HW4_prob2]_Resnet_Quantization_aware_train_4bits

October 30, 2022

[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...')
#device = torch.device("cuda" if use_gpu else "cpu")


batch_size = 128
model_name = "Resnet_quant"
model = resnet20_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):
```

```python
        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…
ResNet_Cifar(
  (conv1): Conv2d(3, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1),
bias=False)
  (bn1): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (relu): ReLU(inplace=True)
  (layer1): Sequential(
    (0): BasicBlock(
      (conv1): QuantConv2d(
        16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
```

```
      (bn2): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
    (1): BasicBlock(
      (conv1): QuantConv2d(
        16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
    (2): BasicBlock(
      (conv1): QuantConv2d(
        16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
  )
  (layer2): Sequential(
    (0): BasicBlock(
      (conv1): QuantConv2d(
        16, 32, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
```

```
track_running_stats=True)
      (downsample): Sequential(
        (0): QuantConv2d(
          16, 32, kernel_size=(1, 1), stride=(2, 2), bias=False
          (weight_quant): weight_quantize_fn()
        )
        (1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      )
    )
    (1): BasicBlock(
      (conv1): QuantConv2d(
        32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
    (2): BasicBlock(
      (conv1): QuantConv2d(
        32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
  )
  (layer3): Sequential(
    (0): BasicBlock(
      (conv1): QuantConv2d(
        32, 64, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
```

```
        64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (downsample): Sequential(
        (0): QuantConv2d(
          32, 64, kernel_size=(1, 1), stride=(2, 2), bias=False
          (weight_quant): weight_quantize_fn()
        )
        (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      )
    )
    (1): BasicBlock(
      (conv1): QuantConv2d(
        64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
    (2): BasicBlock(
      (conv1): QuantConv2d(
        64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
  )
```

```
  (avgpool): AvgPool2d(kernel_size=8, stride=1, padding=0)
  (fc): Linear(in_features=64, out_features=10, bias=True)
)
Files already downloaded and verified
Files already downloaded and verified
```

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

lr = 4.4e-2
weight_decay = 1e-4
epochs = 180
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)
```

[2]:
```
PATH = "result/Resnet_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)))
```

```
Test set: Accuracy: 9178/10000 (92%)
```

[3]:
```
class SaveOutput:
    def __init__(self):
        self.outputs = []
    def __call__(self,module,module_in):
        self.outputs.append(module_in)
    def clear(self):
        self.outputs = []

save_output = SaveOutput()

for layer in model.modules():
```

```
    if isinstance(layer, nn.Conv2d):
        layer.register_forward_pre_hook(save_output)

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

[4]:
```
w_bit = 4
weight_q = model.layer1[0].conv1.weight_q # quantized value is stored during␣
↪the training
w_alpha = model.layer1[0].conv1.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([[[[-0.0000,  1.0000,  3.0000],
          [-2.0000, -1.0000,  2.0000],
          [-1.0000,  0.0000,  2.0000]],

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

         [[ 5.0000,  4.0000, -3.0000],
          [ 7.0000,  7.0000,  3.0000],
          [-0.0000,  1.0000, -2.0000]],

         …,

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

         [[-4.0000,  2.0000,  2.0000],
          [-4.0000,  4.0000,  7.0000],
          [ 0.0000,  4.0000, -0.0000]],

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


        [[[ 2.0000,  4.0000,  6.0000],
          [-1.0000,  3.0000,  4.0000],
          [-4.0000, -0.0000, -0.0000]],
```

```
 [[ 2.0000, -1.0000,  1.0000],
  [-3.0000,  1.0000, -2.0000],
  [ 4.0000,  6.0000,  3.0000]],

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

 …,

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

 [[ 1.0000,  6.0000, -3.0000],
  [ 4.0000,  7.0000,  4.0000],
  [-2.0000,  1.0000, -1.0000]],

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


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

 [[ 0.0000,  4.0000, -0.0000],
  [ 0.0000, -3.0000,  7.0000],
  [ 6.0000,  7.0000, -1.0000]],

 [[ 0.0000, -1.0000,  0.0000],
  [ 7.0000, -3.0000,  0.0000],
  [-0.0000,  3.0000, -1.0000]],

 …,

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

 [[ 3.0000,  0.0000,  0.0000],
  [-3.0000,  3.0000,  0.0000],
  [-2.0000, -7.0000, -7.0000]],

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

```
…,


[[[ 6.0000,  5.0000,  5.0000],
  [ 1.0000, -1.0000,  0.0000],
  [-4.0000, -6.0000, -3.0000]],

 [[-1.0000, -7.0000, -7.0000],
  [-1.0000,  0.0000, -7.0000],
  [ 3.0000, -4.0000,  4.0000]],

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

 …,

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

 [[ 5.0000,  1.0000, -3.0000],
  [ 1.0000, -6.0000, -1.0000],
  [-7.0000, -2.0000,  7.0000]],

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


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

 [[-1.0000,  2.0000, -0.0000],
  [-0.0000, -1.0000,  5.0000],
  [-1.0000, -7.0000,  2.0000]],

 [[-1.0000, -3.0000,  0.0000],
  [ 7.0000, -6.0000, -2.0000],
  [ 7.0000, -6.0000,  2.0000]],

 …,

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

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

         [[-2.0000, -1.0000,  3.0000],
          [ 2.0000,  2.0000,  1.0000],
          [ 2.0000, -2.0000, -6.0000]],

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


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

         [[-3.0000,  1.0000, -0.0000],
          [ 1.0000,  7.0000, -4.0000],
          [ 1.0000, -7.0000,  3.0000]],

         [[-2.0000,  4.0000, -0.0000],
          [-7.0000,  7.0000, -1.0000],
          [-0.0000, -1.0000, -0.0000]],

         …,

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

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

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

[5]:

```
print(model)
x_bit = 4
x = save_output.outputs[1][0]  # input of the 2nd conv layer
x_alpha  = model.layer1[0].conv1.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
```

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

```
ResNet_Cifar(
  (conv1): Conv2d(3, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1),
bias=False)
  (bn1): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (relu): ReLU(inplace=True)
  (layer1): Sequential(
    (0): BasicBlock(
      (conv1): QuantConv2d(
        16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
    (1): BasicBlock(
      (conv1): QuantConv2d(
        16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
    (2): BasicBlock(
      (conv1): QuantConv2d(
        16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
```

```
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
  )
  (layer2): Sequential(
    (0): BasicBlock(
      (conv1): QuantConv2d(
        16, 32, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (downsample): Sequential(
        (0): QuantConv2d(
          16, 32, kernel_size=(1, 1), stride=(2, 2), bias=False
          (weight_quant): weight_quantize_fn()
        )
        (1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      )
    )
    (1): BasicBlock(
      (conv1): QuantConv2d(
        32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
```

```
    (2): BasicBlock(
      (conv1): QuantConv2d(
        32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
  )
  (layer3): Sequential(
    (0): BasicBlock(
      (conv1): QuantConv2d(
        32, 64, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (downsample): Sequential(
        (0): QuantConv2d(
          32, 64, kernel_size=(1, 1), stride=(2, 2), bias=False
          (weight_quant): weight_quantize_fn()
        )
        (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      )
    )
    (1): BasicBlock(
      (conv1): QuantConv2d(
        64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
```

```
      )
      (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
    (2): BasicBlock(
      (conv1): QuantConv2d(
        64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (conv2): QuantConv2d(
        64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False
        (weight_quant): weight_quantize_fn()
      )
      (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
      (relu): ReLU(inplace=True)
      (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
    )
  )
  (avgpool): AvgPool2d(kernel_size=8, stride=1, padding=0)
  (fc): Linear(in_features=64, out_features=10, bias=True)
)
tensor([[[[ 3.0000,  2.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 4.0000,  3.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 4.0000,  3.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
          …,
          [ 4.0000,  3.0000,  0.0000,  …,  1.0000,  1.0000,  0.0000],
          [ 4.0000,  3.0000,  0.0000,  …,  1.0000,  1.0000,  0.0000],
          [ 3.0000,  2.0000,  2.0000,  …,  2.0000,  2.0000,  0.0000]],

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

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

```
 [ 6.0000,  5.0000,  4.0000,  …,  4.0000,  4.0000,  0.0000]],

…,

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

[[11.0000, 10.0000, 15.0000,  …,  0.0000,  1.0000,  0.0000],
 [10.0000,  7.0000, 13.0000,  …,  1.0000,  2.0000,  0.0000],
 [10.0000,  7.0000, 13.0000,  …,  1.0000,  2.0000,  0.0000],
 …,
 [10.0000,  7.0000, 15.0000,  …,  0.0000,  0.0000,  0.0000],
 [10.0000,  7.0000, 13.0000,  …,  0.0000,  0.0000,  0.0000],
 [12.0000, 12.0000, 13.0000,  …, 15.0000, 15.0000, 15.0000]],

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

 [[15.0000,  8.0000,  8.0000,  …,  8.0000, 15.0000,  0.0000],
  [15.0000,  7.0000,  7.0000,  …,  7.0000, 15.0000,  0.0000],
  [15.0000,  7.0000,  6.0000,  …,  7.0000, 15.0000,  0.0000],
  …,
  [ 0.0000,  6.0000,  6.0000,  …,  6.0000,  6.0000,  1.0000],
  [ 0.0000,  6.0000,  6.0000,  …,  6.0000,  6.0000,  1.0000],
  [ 0.0000,  3.0000,  3.0000,  …,  3.0000,  3.0000,  2.0000]],

 [[ 5.0000,  5.0000,  6.0000,  …,  5.0000, 15.0000,  0.0000],
  [ 4.0000,  4.0000,  5.0000,  …,  4.0000, 15.0000,  0.0000],
  [ 4.0000,  4.0000,  5.0000,  …,  4.0000, 15.0000,  0.0000],
```

```
 …,
 [ 4.0000,  4.0000,  4.0000,  …,  4.0000,  4.0000,  0.0000],
 [ 4.0000,  4.0000,  4.0000,  …,  4.0000,  4.0000,  0.0000],
 [ 6.0000,  5.0000,  5.0000,  …,  5.0000,  5.0000,  0.0000]],

…,

[[10.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
 [ 9.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
 [ 9.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
 …,
 [ 0.0000,  1.0000,  1.0000,  …,  1.0000,  1.0000,  7.0000],
 [ 0.0000,  1.0000,  1.0000,  …,  1.0000,  1.0000,  7.0000],
 [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  3.0000]],

[[ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000, 15.0000],
 [ 0.0000,  3.0000,  2.0000,  …,  3.0000,  0.0000, 15.0000],
 [ 0.0000,  3.0000,  2.0000,  …,  3.0000,  0.0000, 15.0000],
 …,
 [10.0000,  7.0000,  7.0000,  …,  7.0000,  7.0000, 11.0000],
 [10.0000,  7.0000,  7.0000,  …,  7.0000,  7.0000, 11.0000],
 [12.0000, 12.0000, 12.0000,  …, 12.0000, 12.0000, 15.0000]],

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


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

 [[ 0.0000,  3.0000,  2.0000,  …,  5.0000,  4.0000,  0.0000],
  [ 0.0000,  4.0000, 11.0000,  …,  4.0000,  7.0000,  5.0000],
  [ 0.0000,  3.0000,  8.0000,  …,  5.0000,  2.0000,  5.0000],
  …,
  [ 0.0000,  5.0000,  7.0000,  …,  7.0000,  8.0000,  7.0000],
  [ 0.0000,  5.0000,  7.0000,  …,  7.0000,  7.0000,  9.0000],
  [ 0.0000,  3.0000,  4.0000,  …,  8.0000,  9.0000, 10.0000]],
```

```
 [[ 3.0000,  2.0000,  4.0000,  …,  3.0000,  3.0000,  0.0000],
  [ 4.0000,  0.0000,  4.0000,  …,  8.0000,  4.0000,  0.0000],
  [ 4.0000,  0.0000,  3.0000,  …, 13.0000,  3.0000,  0.0000],
  …,
  [ 4.0000,  1.0000,  0.0000,  …,  5.0000, 10.0000, 15.0000],
  [ 4.0000,  1.0000,  1.0000,  …,  4.0000,  8.0000, 15.0000],
  [ 6.0000,  3.0000,  1.0000,  …,  4.0000,  5.0000, 15.0000]],

 …,

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

 [[11.0000,  8.0000,  8.0000,  …, 10.0000,  9.0000, 15.0000],
  [10.0000,  4.0000,  2.0000,  …,  1.0000,  4.0000, 12.0000],
  [10.0000,  3.0000,  4.0000,  …,  0.0000,  5.0000, 15.0000],
  …,
  [10.0000,  8.0000,  8.0000,  …,  2.0000,  3.0000,  4.0000],
  [10.0000,  8.0000,  8.0000,  …,  3.0000,  3.0000,  3.0000],
  [12.0000, 12.0000, 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,  0.0000],
  …,
  [ 0.0000,  0.0000,  0.0000,  …,  3.0000,  2.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  3.0000,  2.0000,  1.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  5.0000,  5.0000,  2.0000]]],


…,


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

 [[15.0000,  8.0000,  8.0000,  …, 10.0000, 15.0000,  0.0000],
  [15.0000,  7.0000,  7.0000,  …,  7.0000, 15.0000,  0.0000],
```

```
 [15.0000,  7.0000,  7.0000,  …,  7.0000, 15.0000,  0.0000],
 …,
 [ 4.0000,  6.0000,  6.0000,  …,  6.0000,  9.0000,  0.0000],
 [ 9.0000,  8.0000,  8.0000,  …,  7.0000,  8.0000,  0.0000],
 [ 0.0000,  2.0000,  2.0000,  …,  2.0000,  1.0000,  7.0000]],

[[ 5.0000,  5.0000,  5.0000,  …,  6.0000, 15.0000,  0.0000],
 [ 4.0000,  4.0000,  4.0000,  …,  4.0000, 15.0000,  0.0000],
 [ 4.0000,  4.0000,  4.0000,  …,  4.0000, 15.0000,  0.0000],
 …,
 [ 4.0000,  4.0000,  4.0000,  …,  4.0000, 15.0000,  0.0000],
 [ 2.0000,  4.0000,  4.0000,  …,  3.0000, 15.0000,  0.0000],
 [ 6.0000,  4.0000,  4.0000,  …,  4.0000,  4.0000,  0.0000]],

…,

[[10.0000,  0.0000,  0.0000,  …,  1.0000,  0.0000,  0.0000],
 [ 9.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
 [ 9.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
 …,
 [ 0.0000,  1.0000,  1.0000,  …,  1.0000,  0.0000,  2.0000],
 [ 0.0000,  2.0000,  2.0000,  …,  1.0000,  0.0000,  5.0000],
 [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  1.0000,  3.0000]],

[[ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000, 15.0000],
 [ 0.0000,  3.0000,  3.0000,  …,  3.0000,  2.0000, 15.0000],
 [ 0.0000,  3.0000,  3.0000,  …,  3.0000,  2.0000, 15.0000],
 …,
 [ 6.0000,  6.0000,  6.0000,  …,  5.0000,  4.0000, 13.0000],
 [ 4.0000,  2.0000,  2.0000,  …,  3.0000,  2.0000, 12.0000],
 [13.0000, 14.0000, 14.0000,  …, 14.0000, 14.0000, 15.0000]],

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


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

```
 [[ 0.0000,  4.0000,  4.0000,  …,  4.0000,  4.0000,  0.0000],
  [ 0.0000,  6.0000,  6.0000,  …,  0.0000,  0.0000,  3.0000],
  [ 0.0000,  6.0000,  6.0000,  …, 13.0000,  9.0000, 15.0000],
  …,
  [ 0.0000,  6.0000,  6.0000,  …,  6.0000,  6.0000,  8.0000],
  [ 0.0000,  6.0000,  6.0000,  …,  5.0000,  5.0000,  8.0000],
  [ 0.0000,  3.0000,  3.0000,  …,  7.0000,  7.0000,  7.0000]],

 [[ 3.0000,  3.0000,  3.0000,  …,  3.0000,  3.0000,  0.0000],
  [ 4.0000,  4.0000,  4.0000,  …,  6.0000,  5.0000,  0.0000],
  [ 4.0000,  4.0000,  4.0000,  …,  5.0000,  5.0000, 15.0000],
  …,
  [ 4.0000,  4.0000,  4.0000,  …,  4.0000,  4.0000, 15.0000],
  [ 4.0000,  4.0000,  4.0000,  …,  4.0000,  4.0000, 15.0000],
  [ 6.0000,  5.0000,  5.0000,  …,  4.0000,  4.0000, 14.0000]],

 …,

 [[ 0.0000,  2.0000,  2.0000,  …,  2.0000,  2.0000,  8.0000],
  [ 0.0000,  1.0000,  1.0000,  …,  1.0000,  0.0000,  0.0000],
  [ 0.0000,  1.0000,  1.0000,  …,  2.0000,  0.0000,  0.0000],
  …,
  [ 0.0000,  1.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  1.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  2.0000,  2.0000,  0.0000]],

 [[11.0000, 10.0000, 10.0000,  …, 10.0000, 10.0000, 14.0000],
  [10.0000,  7.0000,  7.0000,  …, 14.0000, 15.0000, 15.0000],
  [10.0000,  7.0000,  7.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [10.0000,  7.0000,  7.0000,  …,  5.0000,  5.0000,  0.0000],
  [10.0000,  7.0000,  7.0000,  …,  5.0000,  5.0000,  0.0000],
  [12.0000, 12.0000, 12.0000,  …,  0.0000,  0.0000,  0.0000]],

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


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

```
         [ 7.0000,  7.0000,  4.0000,  …,  0.0000,  2.0000,  0.0000],
         [ 7.0000,  8.0000,  6.0000,  …,  0.0000,  2.0000,  0.0000],
         [ 4.0000,  4.0000,  4.0000,  …,  0.0000,  1.0000,  0.0000]],

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

        [[ 3.0000,  3.0000,  3.0000,  …,  3.0000,  3.0000,  0.0000],
         [ 6.0000,  5.0000,  5.0000,  …,  4.0000,  5.0000,  0.0000],
         [ 4.0000,  5.0000,  5.0000,  …, 10.0000,  9.0000,  0.0000],
         …,
         [ 5.0000, 10.0000,  9.0000,  …,  4.0000, 13.0000,  0.0000],
         [ 4.0000,  8.0000, 11.0000,  …,  4.0000, 13.0000,  0.0000],
         [ 4.0000,  7.0000, 13.0000,  …,  4.0000, 11.0000,  0.0000]],


       …,


        [[ 0.0000,  2.0000,  2.0000,  …,  2.0000,  2.0000,  8.0000],
         [ 0.0000,  1.0000,  1.0000,  …,  0.0000,  0.0000,  6.0000],
         [ 3.0000,  2.0000,  2.0000,  …,  0.0000,  0.0000,  4.0000],
         …,
         [ 6.0000,  0.0000,  0.0000,  …,  1.0000,  0.0000,  3.0000],
         [ 6.0000,  0.0000,  0.0000,  …,  1.0000,  0.0000,  4.0000],
         [ 8.0000,  6.0000,  2.0000,  …,  0.0000,  0.0000,  2.0000]],

        [[11.0000, 10.0000, 10.0000,  …, 10.0000, 10.0000, 14.0000],
         [ 9.0000, 11.0000, 11.0000,  …, 11.0000,  9.0000, 12.0000],
         [ 0.0000,  0.0000,  0.0000,  …, 10.0000,  8.0000, 12.0000],
         …,
         [15.0000,  0.0000,  0.0000,  …,  6.0000,  7.0000, 14.0000],
         [15.0000,  0.0000,  0.0000,  …,  6.0000,  7.0000, 14.0000],
         [11.0000,  0.0000,  0.0000,  …, 10.0000, 10.0000, 15.0000]],

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

[6]:
```
conv_int = torch.nn.Conv2d(in_channels = 16, out_channels=16, 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([[[[-4.5160e+00, -2.3440e+01, -3.3118e+01,  …, -3.6559e+00,
            5.0178e-01,  3.5125e+00],
          [ 1.2903e+00, -1.9785e+01, -2.6236e+01,  …, -4.8745e+00,
            4.5161e+00,  6.2365e+00],
          [ 1.5054e+00, -1.9570e+01, -2.5806e+01,  …, -2.0071e+00,
            5.9497e+00,  9.1038e+00],
          …,
          [ 4.3010e-01, -2.7885e+01, -4.3512e+01,  …,  1.2831e+01,
            6.5949e+00,  9.6056e+00],
          [-1.1469e+00, -3.0752e+01, -3.0967e+01,  …,  2.7240e+00,
            1.3620e+00,  4.8028e+00],
          [-3.3691e+00, -2.4444e+01, -1.7204e+01,  …, -5.0178e+00,
           -4.7311e+00, -7.4551e+00]],

         [[ 1.1254e+01,  1.9211e+01,  1.0394e+01,  …,  6.2365e+00,
            3.0107e+00, -3.5125e+00],
          [ 1.7634e+01,  2.3297e+01,  1.8208e+01,  …,  6.4515e-01,
            2.2939e+00, -3.5842e-01],
          [ 1.7993e+01,  2.3799e+01,  1.8494e+01,  …, -3.4408e+00,
            1.0322e+01,  1.3476e+01],
          …,
          [ 1.3835e+01,  1.5627e+01,  1.5770e+00,  …,  2.4372e+00,
            1.1828e+01,  1.0466e+01],
          [ 1.1828e+01,  8.3870e+00, -3.2974e+00,  …,  1.0179e+01,
            4.8028e+00,  2.6523e+00],
          [ 1.5340e+01,  1.1899e+01,  6.8099e+00,  …, -2.9103e+01,
           -2.8817e+01, -2.7311e+01]],

         [[ 4.3727e+00,  1.2114e+01, -1.3620e+00,  …, -1.0036e+00,
           -4.0143e+00, -3.9426e+00],
          [-2.6523e+00,  9.3188e-01, -1.2258e+01,  …, -1.7921e+01,
           -1.7419e+01, -1.0824e+01],
          [-3.2258e+00, -9.3188e-01, -1.5269e+01,  …, -3.7060e+01,
           -2.8745e+01, -6.6666e+00],
          …,
          [-4.9462e+00, -1.5770e+00, -1.1254e+01,  …, -7.5268e+00,
           -3.2974e+00, -7.1683e+00],
          [-3.6559e+00, -1.4337e+00, -2.3154e+01,  …, -5.7347e+00,
           -3.1541e+00, -9.3905e+00],
```

```
 [-2.6523e+00, -4.5161e+00, -4.4157e+01,  …, -6.1576e+01,
  -6.2293e+01, -6.4873e+01]],

…,

[[-1.6774e+01, -3.2114e+01, -3.1039e+01,  …, -2.0716e+01,
  -1.7921e+01, -1.3261e+01],
 [-2.1648e+01, -3.8996e+01, -3.1899e+01,  …, -1.0322e+01,
  -1.5699e+01, -1.0753e+01],
 [-2.1290e+01, -3.6559e+01, -3.0035e+01,  …, -4.0860e+00,
  -1.6774e+01, -1.6415e+01],
 …,
 [-1.5340e+01, -2.8458e+01, -2.7598e+01,  …, -2.3799e+01,
  -2.2437e+01, -1.0394e+01],
 [-1.5054e+01, -2.8817e+01, -2.2652e+01,  …, -1.6272e+01,
  -1.5054e+01, -7.9569e+00],
 [-1.7061e+01, -2.8172e+01, -1.3548e+01,  …, -1.5842e+01,
  -1.7347e+01, -9.1755e+00]],

[[-1.9483e-06,  2.1003e+01,  4.0143e+00,  …,  1.6415e+01,
   1.5125e+01,  1.4337e+01],
 [ 1.7921e+00,  2.4086e+01,  5.8064e+00,  …,  1.6057e+01,
   1.4050e+01,  1.6702e+01],
 [ 1.2903e+00,  2.3010e+01,  4.8028e+00,  …,  1.2473e+01,
   4.0860e+00,  1.3907e+01],
 …,
 [-1.2903e+00,  2.0860e+01, -3.0107e+00,  …,  3.2688e+01,
   3.1182e+01,  1.3620e+01],
 [-9.3188e-01,  1.4767e+01, -1.2975e+01,  …,  3.2042e+01,
   3.0107e+01,  1.8781e+01],
 [-1.4337e-01,  1.9355e+00, -3.7275e+00,  …, -1.0394e+01,
  -8.4586e+00, -1.6200e+01]],

[[-1.3190e+01, -1.5197e+01,  3.6559e+00,  …, -1.2043e+01,
  -5.0178e+00, -8.3870e+00],
 [-1.3620e+01, -9.8206e+00,  1.8638e+00,  …, -2.6523e+00,
  -5.6630e+00, -8.6737e+00],
 [-1.4623e+01, -1.0466e+01, -5.7347e-01,  …, -4.5161e+00,
  -9.1038e+00, -1.7706e+01],
 …,
 [-1.4767e+01, -1.4552e+01, -1.7204e+00,  …, -5.8780e+00,
  -1.0681e+01, -5.3763e+00],
 [-1.2186e+01, -1.3476e+01,  8.9604e+00,  …, -1.0896e+01,
  -7.7418e+00, -2.5089e+00],
 [-1.9355e+01, -2.4086e+01,  1.0537e+01,  …, -1.1469e+00,
  -1.0752e+00, -1.1469e+00]]],
```

```
[[[ 1.7204e+00,  7.6701e+00,  9.8923e+00,  …,  7.3834e+00,
    7.5984e+00,  3.4910e+01],
  [-1.3620e+00,  3.4408e+00,  6.2365e+00,  …,  3.2974e+00,
    2.2939e+00,  3.4766e+01],
  [ 1.0752e+00,  5.8064e+00,  9.3905e+00,  …,  3.2974e+00,
    2.2939e+00,  3.4766e+01],
  …,
  [-3.2258e+00, -1.3620e+00, -1.3620e+00,  …, -1.3620e+00,
   -7.1683e-02, -8.8887e+00],
  [ 1.2186e+00,  1.5054e+00,  1.5054e+00,  …,  1.5054e+00,
    1.7921e+00, -8.5303e+00],
  [ 6.8099e+00,  5.9497e+00,  5.9497e+00,  …,  5.9497e+00,
    5.9497e+00, -7.0967e+00]],

 [[ 1.7204e+00,  3.7992e+00,  1.7204e+00,  …,  6.0931e+00,
    9.0321e+00,  1.0824e+01],
  [-1.2903e+00,  1.6236e-06, -2.6523e+00,  …,  3.9426e+00,
    6.5949e+00,  9.8206e+00],
  [-2.2939e+00, -1.0752e+00, -3.0107e+00,  …,  3.9426e+00,
    6.5949e+00,  9.8206e+00],
  …,
  [ 4.0071e+01,  4.1075e+01,  4.1075e+01,  …,  4.1075e+01,
    4.1863e+01,  2.3727e+01],
  [ 3.9426e+00,  3.4408e+00,  3.4408e+00,  …,  3.4408e+00,
    4.1576e+00,  1.0107e+01],
  [ 1.3835e+01,  1.2903e+01,  1.2903e+01,  …,  1.2903e+01,
    1.2903e+01,  1.1613e+01]],

 [[-3.9426e+00, -6.7382e+00, -5.0178e+00,  …, -6.9533e+00,
   -1.1469e+00, -1.3261e+01],
  [ 1.2903e+00, -7.1684e-02, -2.1505e-01,  …, -1.2903e+00,
    3.7275e+00, -4.7311e+00],
  [ 2.5089e+00,  7.8852e-01,  7.1683e-01,  …, -1.2903e+00,
    3.7275e+00, -4.7311e+00],
  …,
  [ 8.6020e+00,  9.2472e+00,  9.2472e+00,  …,  9.2472e+00,
    8.3870e+00,  1.1326e+01],
  [ 1.2688e+01,  1.3692e+01,  1.3692e+01,  …,  1.3692e+01,
    1.2903e+01,  1.2760e+01],
  [-3.2974e+00, -5.4479e+00, -5.4479e+00,  …, -5.4479e+00,
   -5.4479e+00, -6.8816e+00]],


 …,


 [[-1.8064e+01, -2.0573e+01, -1.9426e+01,  …, -2.3154e+01,
   -2.7957e+01, -2.0645e+01],
  [-9.4622e+00, -1.1541e+01, -9.7489e+00,  …, -1.4265e+01,
   -1.8781e+01, -1.1613e+01],
```

```
         [-9.8923e+00, -1.1398e+01, -1.0036e+01,  …, -1.4265e+01,
          -1.8781e+01, -1.1613e+01],
         …,
         [-6.2293e+01, -5.4479e+01, -5.4479e+01,  …, -5.4479e+01,
          -5.4264e+01, -4.5304e+01],
         [-1.4552e+01, -1.0753e+01, -1.0753e+01,  …, -1.0753e+01,
          -1.0753e+01, -9.5339e+00],
         [-1.4265e+01, -1.5412e+01, -1.5412e+01,  …, -1.5412e+01,
          -1.5412e+01, -8.2436e+00]],

        [[ 1.1684e+01,  1.2616e+01,  1.3190e+01,  …,  1.3261e+01,
           1.8064e+01, -1.0537e+01],
         [ 1.0036e+01,  1.1541e+01,  1.0824e+01,  …,  1.0753e+01,
           1.4050e+01, -1.3835e+01],
         [ 1.0036e+01,  1.1541e+01,  1.1899e+01,  …,  1.0753e+01,
           1.4050e+01, -1.3835e+01],
         …,
         [ 3.9354e+01,  3.8566e+01,  3.8566e+01,  …,  3.8566e+01,
           3.4981e+01,  1.7132e+01],
         [ 2.5089e+00,  3.2974e+00,  3.2974e+00,  …,  3.2974e+00,
           3.3691e+00, -6.1648e+00],
         [ 6.4515e-01, -1.1469e+00, -1.1469e+00,  …, -1.1469e+00,
          -1.1469e+00, -8.9604e+00]],

        [[-5.3046e+00, -4.6594e+00, -5.9497e+00,  …, -6.8816e+00,
          -8.0285e+00,  1.5053e+00],
         [-8.2436e+00, -9.8206e+00, -9.5339e+00,  …, -1.0466e+01,
          -9.0321e+00, -5.6630e+00],
         [-9.0321e+00, -9.9640e+00, -9.6056e+00,  …, -1.0466e+01,
          -9.0321e+00, -5.6630e+00],
         …,
         [-3.5412e+01, -3.4121e+01, -3.4121e+01,  …, -3.4121e+01,
          -3.1899e+01, -3.9282e+01],
         [-1.1326e+01, -1.3118e+01, -1.3118e+01,  …, -1.3118e+01,
          -1.2903e+01, -2.7741e+01],
         [-1.3620e+01, -1.6917e+01, -1.6917e+01,  …, -1.6917e+01,
          -1.6917e+01, -2.1075e+01]]],


       [[[-1.9641e+01, -1.2831e+01,  4.3010e+00,  …,  1.1039e+01,
           1.0609e+01, -6.8816e+00],
         [-1.5054e+01, -7.8852e+00,  2.8673e+00,  …,  8.8887e+00,
           2.6164e+01,  1.1254e+01],
         [-1.5770e+01, -6.0931e+00,  2.7240e+00,  …,  9.4622e+00,
           2.7741e+01,  1.5627e+01],
         …,
         [ 7.0967e+00, -1.0753e+00, -2.0860e+01,  …, -5.8064e+00,
          -1.1469e+00,  1.7993e+01],
```

```
 [-1.0036e+00, -1.1541e+01, -1.7993e+01,  …, -4.7311e+00,
   3.7992e+00,  2.0215e+01],
 [-2.9390e+00, -1.1469e+01, -2.3225e+01,  …,  1.5770e+00,
   7.3834e+00,  1.6989e+01]],

[[-1.0036e+01, -8.2436e+00, -2.7240e+00,  …, -1.4265e+01,
  -7.0967e+00,  5.0178e-01],
 [-6.2365e+00, -4.2293e+00,  8.3870e+00,  …, -1.0824e+01,
  -9.3188e+00,  1.7204e+00],
 [-6.8816e+00, -1.2186e+00,  1.3548e+01,  …, -1.0036e+01,
  -1.0681e+01,  2.9390e+00],
 …,
 [ 1.0896e+01,  1.4122e+01,  1.3692e+01,  …,  3.4408e+00,
   2.2222e+00,  3.7992e+00],
 [ 6.3081e+00,  5.9497e+00,  4.2293e+00,  …,  8.6020e-01,
  -1.7204e+00,  3.7992e+00],
 [ 1.1613e+01,  1.2688e+01,  1.1326e+01,  …,  4.4444e+00,
   4.7311e+00,  5.0178e+00]],

[[ 1.6415e+01,  5.0178e-01, -7.1683e-01,  …,  4.9462e+00,
   2.1147e+01,  6.8099e+00],
 [ 7.6701e+00, -4.0860e+00, -7.2400e+00,  …, -1.0609e+01,
  -1.7921e+00,  5.4479e+00],
 [ 7.3834e+00, -4.8745e+00, -3.0107e+00,  …, -7.8852e+00,
  -3.9426e+00,  1.1469e+00],
 …,
 [-1.5054e+00,  1.5054e+00,  2.5806e+00,  …,  1.7921e+00,
   2.2222e+00, -2.5089e+00],
 [ 1.0752e+00, -5.3763e+00, -1.0107e+01,  …, -5.8064e+00,
  -3.5842e+00, -2.5806e+00],
 [-2.5089e+00, -1.0896e+01, -6.8816e+00,  …,  7.8852e-01,
  -2.0071e+00, -8.6020e-01]],


…,


[[-3.8709e+00,  8.4586e+00, -2.8673e-01,  …, -8.1719e+00,
  -1.7204e+00,  1.1899e+01],
 [-1.5842e+01, -1.7562e+01, -2.8817e+01,  …, -2.1147e+01,
  -1.4552e+01, -1.0036e+01],
 [-1.2545e+01, -1.4193e+01, -2.3082e+01,  …, -1.0179e+01,
  -1.0322e+01, -6.6666e+00],
 …,
 [-1.5770e+01, -1.6129e+01, -1.3118e+01,  …, -1.5555e+01,
  -1.5412e+01, -1.3763e+01],
 [-1.8064e+01, -1.5125e+01, -1.1613e+01,  …, -1.0968e+01,
  -1.5340e+01, -1.4982e+01],
 [-1.6344e+01, -1.6272e+01, -2.0215e+01,  …, -1.7204e+01,
  -1.6200e+01, -1.4982e+01]],
```

```
 [[ 8.1002e+00, -1.0609e+01, -1.0753e+00,  …,  8.6020e-01,
    6.8816e+00,  2.7957e+00],
  [ 1.4337e+01, -2.8673e-01,  1.0036e+01,  …, -1.4337e+00,
    3.8709e+00,  1.3620e+01],
  [ 1.4265e+01, -2.6523e+00,  5.5913e+00,  …,  1.0036e+00,
    1.1469e+00,  1.1971e+01],
  …,
  [ 7.4551e+00,  2.0788e+00, -2.2939e+00,  …,  8.8171e+00,
    1.1111e+01, -2.1505e-01],
  [ 7.7418e+00, -2.5806e+00, -1.5770e+00,  …,  6.5949e+00,
    8.1002e+00,  4.7311e+00],
  [ 2.3656e+00, -2.7957e+00, -1.9355e+00,  …,  1.6487e+01,
    1.6989e+01,  1.2043e+01]],

 [[-1.1254e+01, -4.3727e+00, -1.1684e+01,  …, -1.0753e+01,
   -2.5161e+01, -1.9068e+01],
  [-1.0036e+01, -7.9569e+00, -1.2760e+01,  …, -1.6487e+00,
   -1.0896e+01, -2.5734e+01],
  [-1.0394e+01, -1.0251e+01, -1.5985e+01,  …,  1.0254e-06,
   -6.5949e+00, -2.6738e+01],
  …,
  [-8.1002e+00, -1.7491e+01, -1.7419e+01,  …, -1.1398e+01,
   -9.1038e+00, -3.1541e+00],
  [-9.8923e+00, -1.0609e+01, -4.5877e+00,  …, -5.0895e+00,
   -6.0214e+00, -5.0178e+00],
  [-1.6702e+01, -1.4193e+01, -1.6702e+01,  …, -9.6056e+00,
   -6.8099e+00, -7.1683e+00]]],


…,


[[[ 1.2903e+00,  7.5984e+00,  7.7418e+00,  …,  9.3188e+00,
    1.0251e+01,  3.4910e+01],
  [-1.6487e+00,  3.2974e+00,  3.2974e+00,  …,  6.0931e+00,
    5.5913e+00,  3.4551e+01],
  [-1.6487e+00,  3.2974e+00,  3.2974e+00,  …,  5.5913e+00,
    5.3763e+00,  3.5268e+01],
  …,
  [ 4.7311e+00,  4.5161e+00,  4.9462e+00,  …,  3.9426e+00,
    3.6559e+00,  1.5484e+01],
  [ 6.1648e+00,  5.7347e+00,  6.0214e+00,  …,  5.5913e+00,
    6.0931e+00,  1.7204e+01],
  [-5.7347e-01,  1.5770e+00,  1.5770e+00,  …,  1.5053e+00,
    4.8745e+00,  1.2258e+01]],

 [[ 2.4372e+00,  6.2365e+00,  5.9497e+00,  …,  5.2329e+00,
```

```
   1.0107e+01,  1.3620e+01],
 [ 3.5842e-01,  3.9426e+00,  3.9426e+00,  …,  2.2939e+00,
   7.0250e+00,  1.1828e+01],
 [ 3.5842e-01,  3.9426e+00,  3.9426e+00,  …,  1.4337e+00,
   6.4515e+00,  1.0753e+01],
 …,
 [ 4.0860e+00,  5.8780e+00,  6.1648e+00,  …,  5.5196e+00,
   6.3081e+00,  8.7454e+00],
 [ 3.9426e+00,  4.6594e+00,  4.8745e+00,  …,  3.5125e+00,
   4.1576e+00,  7.4551e+00],
 [-2.0215e+01, -1.9713e+01, -1.9641e+01,  …, -1.3835e+01,
  -1.4982e+01, -5.3046e+00]],

[[-5.3763e+00, -7.3117e+00, -7.3117e+00,  …, -4.2293e+00,
   1.4337e+00, -1.0394e+01],
 [ 3.5842e-01, -1.2903e+00, -1.2903e+00,  …, -5.0178e-01,
   4.3727e+00, -4.3010e+00],
 [ 3.5842e-01, -1.2903e+00, -1.2903e+00,  …,  1.2186e+00,
   4.0860e+00, -4.1576e+00],
 …,
 [-4.8745e+00, -5.3046e+00, -5.3763e+00,  …,  4.3010e-01,
  -6.4515e-01, -7.2400e+00],
 [-3.2974e+00, -3.5842e+00, -3.4408e+00,  …, -1.2903e+00,
  -7.1683e-01, -8.3870e+00],
 [-2.9820e+01, -2.9820e+01, -2.9462e+01,  …, -2.6379e+01,
  -2.5806e+01, -3.1182e+01]],


…,

[[-1.7921e+01, -2.2867e+01, -2.3010e+01,  …, -2.7741e+01,
  -3.0752e+01, -2.3297e+01],
 [-1.1326e+01, -1.4265e+01, -1.4265e+01,  …, -1.7061e+01,
  -2.2580e+01, -1.3405e+01],
 [-1.1326e+01, -1.4265e+01, -1.4265e+01,  …, -1.7061e+01,
  -2.2365e+01, -1.2903e+01],
 …,
 [-1.8781e+01, -1.7849e+01, -1.7276e+01,  …, -1.8638e+01,
  -1.8924e+01, -1.0179e+01],
 [-2.2580e+01, -1.8924e+01, -1.8351e+01,  …, -1.8423e+01,
  -1.8208e+01, -8.2436e+00],
 [-3.0465e+01, -3.0752e+01, -3.0967e+01,  …, -3.1612e+01,
  -3.0967e+01, -1.5914e+01]],

[[ 1.2186e+01,  1.4122e+01,  1.4193e+01,  …,  1.5125e+01,
   2.0358e+01, -9.1755e+00],
 [ 9.3188e+00,  1.0753e+01,  1.0753e+01,  …,  1.4337e+01,
   1.7061e+01, -1.2975e+01],
 [ 9.3188e+00,  1.0753e+01,  1.0753e+01,  …,  1.3118e+01,
```

```
          1.7706e+01, -1.3261e+01],
        …,
        [ 5.5913e+00,  4.8745e+00,  3.3691e+00,  …,  5.5196e+00,
          7.9569e+00, -1.9283e+01],
        [ 8.1719e+00,  9.2472e+00,  8.3870e+00,  …,  7.0967e+00,
          1.0322e+01, -1.6559e+01],
        [-5.0178e+00, -5.0895e+00, -4.9462e+00,  …, -3.1541e+00,
         -3.3691e+00, -1.6129e+01]],

       [[-4.7311e+00, -6.8816e+00, -6.8099e+00,  …, -9.3188e+00,
         -1.0681e+01, -1.9355e+00],
        [-8.6020e+00, -1.0466e+01, -1.0466e+01,  …, -1.0968e+01,
         -1.0609e+01, -6.0931e+00],
        [-8.6020e+00, -1.0466e+01, -1.0466e+01,  …, -1.1254e+01,
         -8.4586e+00, -5.7347e+00],
        …,
        [-1.1398e+01, -1.1541e+01, -1.1541e+01,  …, -1.1039e+01,
         -9.6773e+00, -8.0285e+00],
        [-1.1541e+01, -1.1183e+01, -1.0968e+01,  …, -9.8923e+00,
         -9.5339e+00, -6.8816e+00],
        [-2.5806e+00, -6.2365e+00, -6.2365e+00,  …, -7.8135e+00,
         -9.1038e+00, -3.4408e+00]]],


      [[[ 3.6559e+00, -2.5806e+00, -1.6487e+00,  …, -3.8709e+00,
           3.9426e+00, -3.3691e+00],
        [ 1.0107e+01,  1.2903e+00, -3.4910e+01,  …, -2.8817e+01,
         -1.1111e+01,  1.0753e+00],
        [ 1.0107e+01, -5.2329e+00, -3.1756e+01,  …, -3.2186e+01,
         -1.0753e+01,  4.2293e+00],
        …,
        [ 1.0107e+01, -2.2222e+00, -3.7275e+01,  …,  1.5054e+00,
          4.4444e+00,  5.3763e+00],
        [ 1.0107e+01, -2.5089e+00, -3.5842e+01,  …,  3.7275e+00,
          4.1576e+00,  3.0824e+00],
        [ 6.8099e+00, -5.7347e+00, -4.1146e+01,  …,  5.3763e+00,
          6.3798e+00,  3.7992e+00]],

       [[ 3.5125e+00,  3.2258e+00,  2.9390e+00,  …,  3.6630e+01,
          3.9999e+01,  4.6092e+01],
        [ 9.8206e+00,  9.9640e+00, -1.1183e+01,  …, -2.8315e+01,
         -2.9749e+01, -2.7096e+01],
        [ 9.8206e+00,  5.5196e+00, -2.2222e+00,  …,  2.1505e+00,
          4.3010e+00,  7.8135e+00],
        …,
        [ 9.8206e+00,  7.6701e+00,  1.4337e+00,  …,  4.5877e+00,
          3.6559e+00,  1.4337e+00],
        [ 9.8206e+00,  7.6701e+00,  2.7240e+00,  …,  4.7311e+00,
```

```
   5.3763e+00,  2.7240e+00],
 [ 1.3835e+01,  1.1756e+01,  3.5125e+00,  …,  7.0250e+00,
   7.3834e+00,  5.5196e+00]],

[[ 8.7454e+00,  3.2974e+00,  1.0753e+00,  …,  3.0035e+01,
   3.8064e+01,  3.9211e+01],
 [ 1.5770e+00, -2.8673e+00,  3.5125e+00,  …, -8.9604e+00,
  -1.0036e+01, -6.0214e+00],
 [ 1.5770e+00, -5.6630e+00,  2.5806e+00,  …, -7.4551e+00,
  -8.3153e+00, -2.9390e+00],
 …,
 [ 1.5770e+00, -4.5160e+00,  2.0071e+00,  …, -6.5949e+00,
  -8.6737e+00, -3.8709e+00],
 [ 1.5770e+00, -4.5877e+00,  1.6487e+00,  …, -6.9533e+00,
  -7.9569e+00, -5.6630e+00],
 [-3.2974e+00, -7.5984e+00,  5.1612e+00,  …, -2.4372e+00,
  -7.8852e-01,  2.1505e+00]],


…,

[[-1.6057e+01, -5.4479e+00,  1.0036e+00,  …,  3.2258e+00,
   3.9426e+00,  5.7347e+00],
 [-1.9713e+01, -8.3870e+00, -5.3763e+00,  …,  7.1683e+00,
   1.3118e+01,  8.6737e+00],
 [-1.9713e+01, -1.0322e+01, -1.3190e+01,  …, -2.3369e+01,
  -2.0788e+01, -2.6953e+01],
 …,
 [-1.9713e+01, -8.2436e+00, -1.1971e+01,  …, -1.3548e+01,
  -1.3261e+01, -1.5269e+01],
 [-1.9713e+01, -8.4586e+00, -1.2258e+01,  …, -1.7204e+01,
  -1.6774e+01, -1.6917e+01],
 [-1.4265e+01, -8.6737e+00, -1.5197e+01,  …, -1.2760e+01,
  -1.2760e+01, -1.2258e+01]],

[[ 3.5125e+00, -4.6594e+00,  1.5412e+01,  …,  6.9533e+00,
   3.8709e+00,  1.3046e+01],
 [ 5.2329e+00, -3.9426e+00,  1.0107e+01,  …, -1.6774e+01,
  -2.3297e+01, -1.0251e+01],
 [ 5.2329e+00, -5.6630e+00,  1.1254e+01,  …,  4.4444e+00,
   2.5089e+00,  1.3333e+01],
 …,
 [ 5.2329e+00, -4.0860e+00,  1.3261e+01,  …,  7.1683e-01,
  -1.7204e+00,  1.7921e+00],
 [ 5.2329e+00, -3.9426e+00,  1.4122e+01,  …,  5.7347e-01,
   9.3188e-01,  4.0860e+00],
 [ 6.4515e-01, -7.0967e+00,  9.3188e+00,  …,  1.9355e+00,
   2.2939e+00,  3.9426e+00]],
```

```
 [[-8.1719e+00, -1.2258e+01, -1.2473e+01,  …, -3.7347e+01,
   -3.9713e+01, -4.3727e+01],
  [-8.0285e+00, -1.3118e+01, -1.4552e+01,  …,  7.1683e+00,
    1.3118e+01,  8.1719e+00],
  [-8.0285e+00, -1.2473e+01, -1.4838e+01,  …, -4.7311e+00,
   -3.1541e+00, -1.0179e+01],
  …,
  [-8.0285e+00, -1.7347e+01, -1.7849e+01,  …, -9.9640e+00,
   -8.3153e+00, -1.0537e+01],
  [-8.0285e+00, -1.7562e+01, -1.8494e+01,  …, -1.2903e+01,
   -1.1254e+01, -1.0609e+01],
  [-1.3620e+01, -2.2078e+01, -2.1505e+01,  …, -1.1541e+01,
   -1.2616e+01, -1.2473e+01]]],


[[[ 6.8099e+00,  3.5842e+00,  4.1576e+00,  …,  7.5984e+00,
    5.7347e+00, -9.3905e+00],
  [ 2.1505e-01,  1.1469e+00,  1.7921e+00,  …,  4.5161e+01,
    3.3046e+01,  1.0896e+01],
  [ 1.2186e+00,  5.2329e+00,  5.3763e+00,  …,  2.2437e+01,
    4.6523e+01,  3.4551e+01],
  …,
  [ 1.3620e+00, -6.8816e+00, -1.3476e+01,  …,  7.0250e+00,
    7.6701e+00,  1.8638e+01],
  [ 9.1038e+00,  4.0860e+00, -5.8780e+00,  …,  6.0931e+00,
    7.7418e+00,  1.6631e+01],
  [ 3.7275e+00,  5.0895e+00, -4.4444e+00,  …,  4.1576e+00,
    6.7382e+00,  1.1756e+01]],

 [[ 3.8781e+01,  3.7705e+01,  3.8279e+01,  …,  3.2616e+01,
    2.2007e+01,  1.2258e+01],
  [-2.6881e+01, -2.7885e+01, -2.7526e+01,  …, -7.8851e-01,
    1.4050e+01,  1.8494e+01],
  [ 3.9426e+00,  4.8028e+00,  4.3010e+00,  …,  1.7921e+00,
    1.1684e+01,  2.1218e+01],
  …,
  [-1.5340e+01, -2.3584e+01, -1.4122e+01,  …,  3.0824e+00,
    3.4408e+00,  1.0896e+01],
  [-9.4622e+00, -2.1720e+01, -1.9139e+01,  …,  3.5842e+00,
    4.8028e+00,  1.0394e+01],
  [-6.5232e+00, -1.9355e+01, -2.1577e+01,  …,  5.4479e+00,
    7.3117e+00,  1.2186e+01]],

 [[ 4.0143e+01,  3.6415e+01,  3.6415e+01,  …,  3.1612e+01,
    1.4767e+01,  1.1039e+01],
  [-5.0895e+00, -5.6630e+00, -5.7347e+00,  …,  2.5806e+00,
    2.4086e+01,  2.0788e+00],
  [-6.5949e+00, -6.2365e+00, -5.7347e+00,  …, -1.4337e+01,
```

```
  -8.1719e+00,  1.8064e+01],
 …,
 [-1.0609e+01,  2.0071e+00, -7.1683e-01,  …, -7.8852e-01,
  -5.7347e-01, -7.3117e+00],
 [-1.3692e+01, -2.4372e+00,  1.2186e+00,  …,  2.1505e-01,
  -1.1469e+00, -7.3834e+00],
 [-1.9498e+01, -9.3188e+00,  1.4337e-01,  …, -4.0860e+00,
  -5.0178e+00, -1.1254e+01]],

…,

[[ 1.0753e+01,  8.7454e+00,  8.3870e+00,  …,  3.0107e+00,
   1.3620e+00, -8.2436e+00],
 [ 1.0036e+01,  7.0967e+00,  7.1683e+00,  …, -3.0824e+00,
  -8.3153e+00,  5.3763e+00],
 [-2.6093e+01, -3.0322e+01, -3.0035e+01,  …, -1.3978e+01,
  -7.6701e+00, -8.0285e+00],
 …,
 [-2.1792e+01, -2.5806e+01, -2.2437e+01,  …, -2.0932e+01,
  -2.2580e+01, -1.3907e+01],
 [-1.3261e+01, -1.9785e+01, -2.2007e+01,  …, -2.1290e+01,
  -2.2007e+01, -1.5197e+01],
 [-4.2293e+00, -1.2760e+01, -1.8996e+01,  …, -2.1648e+01,
  -2.2580e+01, -1.2473e+01]],

[[ 4.3010e+00,  5.0895e+00,  4.9462e+00,  …,  5.8780e+00,
   1.5770e+01, -6.1648e+00],
 [-1.4408e+01, -1.1756e+01, -1.2043e+01,  …, -1.0251e+01,
  -1.6487e+00, -4.0860e+00],
 [ 8.2436e+00,  1.1828e+01,  1.2616e+01,  …,  5.0178e-01,
  -6.3081e+00, -1.4265e+01],
 …,
 [ 1.9068e+01,  2.7741e+01,  2.2652e+01,  …,  8.6020e+00,
   1.0609e+01, -1.4552e+01],
 [ 1.7706e+01,  1.8064e+01,  2.6666e+01,  …,  8.7454e+00,
   1.0251e+01, -1.3476e+01],
 [ 9.2472e+00,  4.1576e+00,  2.0716e+01,  …,  5.0178e+00,
   5.8780e+00, -1.6200e+01]],

[[-3.6630e+01, -3.9139e+01, -3.8996e+01,  …, -3.3261e+01,
  -1.8064e+01, -2.2939e+01],
 [ 5.0895e+00,  4.2293e+00,  4.6594e+00,  …, -8.8171e+00,
  -3.1397e+01, -1.1326e+01],
 [-1.0179e+01, -1.0322e+01, -1.0394e+01,  …,  2.5089e+00,
  -4.4444e+00, -3.1899e+01],
 …,
 [-7.5268e+00, -2.1792e+01, -1.5054e+01,  …, -1.0322e+01,
  -8.8887e+00, -8.7454e+00],
```

```
          [-6.3798e+00, -1.7132e+01, -1.7276e+01,  …, -1.1398e+01,
           -9.7489e+00, -1.0609e+01],
          [-6.8816e+00, -1.1613e+01, -1.9713e+01,  …, -1.4122e+01,
           -1.4767e+01, -1.3835e+01]]]], device='cuda:0',
       grad_fn=<MulBackward0>)
```

[7]:
```
#### input floating number / weight quantized version

conv_ref = torch.nn.Conv2d(in_channels = 16, out_channels=16, kernel_size = 3,␣
 ↪bias = False)
conv_ref.weight = model.layer1[0].conv1.weight_q

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

```
tensor([[[[-4.6597e+00, -2.5733e+01, -3.2835e+01,  …, -3.5135e+00,
            1.9584e+00,  3.5007e+00],
          [ 7.9587e-01, -2.1969e+01, -2.5796e+01,  …, -5.0442e+00,
            6.1554e+00,  6.2092e+00],
          [ 1.1467e+00, -2.0376e+01, -2.5855e+01,  …, -1.1504e+00,
            5.8581e+00,  9.1889e+00],
          …,
          [-2.9927e+00, -3.3437e+01, -4.7374e+01,  …,  1.8183e+01,
            1.3683e+01,  1.1037e+01],
          [-4.9380e+00, -3.9532e+01, -3.7526e+01,  …,  3.8721e+00,
            2.2386e+00,  1.7452e+00],
          [-4.9245e+00, -2.8319e+01, -2.0530e+01,  …, -8.0153e+00,
           -7.3359e+00, -1.1229e+01]],

         [[ 1.1465e+01,  1.9983e+01,  1.2527e+01,  …,  5.0320e+00,
            4.1526e+00, -4.0398e+00],
          [ 1.7109e+01,  2.3046e+01,  2.0405e+01,  …, -4.4969e-01,
            8.2300e-01, -6.0840e-01],
          [ 1.7087e+01,  2.3309e+01,  1.9142e+01,  …, -3.5624e+00,
            1.0037e+01,  1.3501e+01],
          …,
          [ 1.5420e+01,  1.4059e+01,  3.3124e+00,  …,  4.7894e+00,
            1.6173e+01,  8.3577e+00],
          [ 1.4157e+01,  5.4930e+00, -3.4958e+00,  …,  1.3210e+01,
            8.3999e+00,  6.2509e-01],
          [ 1.7187e+01,  1.1694e+01,  7.1863e+00,  …, -3.3203e+01,
           -3.2593e+01, -3.4612e+01]],

         [[ 3.8603e+00,  1.3019e+01,  3.8444e-01,  …, -8.1820e-01,
           -3.5801e+00, -3.6629e+00],
          [-3.5186e+00,  1.8894e+00, -1.2342e+01,  …, -1.8192e+01,
           -1.6964e+01, -1.1353e+01],
          [-4.2129e+00, -7.8686e-01, -1.4482e+01,  …, -3.7048e+01,
```

```
  -2.8522e+01, -6.0064e+00],
 …,
 [-6.3783e+00, -8.4180e-01, -9.9688e+00,  …, -5.3030e+00,
  -1.8273e+00, -5.8336e+00],
 [-3.9529e+00,  3.3456e+00, -2.6782e+01,  …, -1.5346e+01,
  -1.3757e+01, -1.7843e+01],
 [-2.8527e+00, -1.7662e+00, -4.2052e+01,  …, -8.0405e+01,
  -8.2119e+01, -8.1681e+01]],

…,

[[-1.5511e+01, -3.5544e+01, -3.5127e+01,  …, -2.1384e+01,
  -1.7739e+01, -1.2893e+01],
 [-2.1708e+01, -4.0719e+01, -3.5071e+01,  …, -9.9446e+00,
  -1.5172e+01, -1.0125e+01],
 [-2.1326e+01, -3.8458e+01, -3.3165e+01,  …, -4.5561e+00,
  -1.7077e+01, -1.6924e+01],
 …,
 [-1.3716e+01, -2.5085e+01, -2.8133e+01,  …, -2.9217e+01,
  -2.8888e+01, -1.0673e+01],
 [-1.3669e+01, -2.9900e+01, -2.3494e+01,  …, -1.8446e+01,
  -1.7514e+01, -4.8442e+00],
 [-1.6372e+01, -3.2619e+01, -1.9000e+01,  …, -1.1381e+01,
  -1.2598e+01, -2.3051e+00]],

[[-9.5073e-01,  2.1707e+01,  3.8938e+00,  …,  1.6453e+01,
   1.4763e+01,  1.4196e+01],
 [ 2.3387e+00,  2.5359e+01,  7.3150e+00,  …,  1.6086e+01,
   1.3325e+01,  1.5801e+01],
 [ 1.5336e+00,  2.4032e+01,  5.3468e+00,  …,  1.2673e+01,
   4.3574e+00,  1.3533e+01],
 …,
 [-5.0868e+00,  2.3169e+01, -1.0348e+01,  …,  4.1773e+01,
   4.1370e+01,  2.0081e+01],
 [-4.1793e+00,  1.8196e+01, -1.6875e+01,  …,  3.4009e+01,
   3.1681e+01,  1.9197e+01],
 [-1.2812e+00,  3.9340e+00, -2.4457e+00,  …, -1.5608e+01,
  -1.4492e+01, -2.2184e+01]],

[[-1.3768e+01, -1.5050e+01,  4.7507e+00,  …, -1.2531e+01,
  -4.8946e+00, -8.5683e+00],
 [-1.4009e+01, -9.9581e+00,  2.5446e+00,  …, -2.3308e+00,
  -5.3210e+00, -9.7000e+00],
 [-1.4393e+01, -1.0416e+01, -5.5608e-01,  …, -4.5526e+00,
  -9.3019e+00, -1.8287e+01],
 …,
 [-1.5630e+01, -1.1784e+01, -2.4332e+00,  …, -5.4534e+00,
  -1.0209e+01, -6.3791e+00],
```

```
  [-1.4226e+01, -1.3605e+01,  1.3795e+01,  …, -1.3425e+01,
   -1.0483e+01, -5.8133e+00],
  [-2.0190e+01, -2.3463e+01,  1.2723e+01,  …, -3.6003e+00,
   -3.2418e+00, -4.5670e+00]]],


[[[ 1.9608e+00,  8.9665e+00,  9.9126e+00,  …,  8.1919e+00,
    4.2537e+00,  6.6717e+01],
  [-3.4858e-01,  3.9417e+00,  6.1617e+00,  …,  2.7528e+00,
   -5.2619e+00,  6.7315e+01],
  [ 3.1930e-01,  5.7010e+00,  8.5193e+00,  …,  2.4449e+00,
   -5.2998e+00,  6.7293e+01],
  …,
  [-4.5350e+00, -1.9685e+00, -1.9802e+00,  …, -1.9516e+00,
   -5.4120e+00, -6.5686e+00],
  [ 6.8426e-01, -7.8976e-01, -7.8976e-01,  …, -7.8976e-01,
   -4.6505e-01, -1.2190e+01],
  [ 7.4953e+00,  6.5263e+00,  6.5263e+00,  …,  6.5263e+00,
    6.5263e+00, -8.8971e+00]],

 [[ 5.9856e+00,  5.5899e+00,  4.3091e+00,  …,  8.4432e+00,
    1.2208e+01,  3.4080e+00],
  [-1.1715e+00, -3.6405e-01, -2.7435e+00,  …,  4.2700e+00,
    4.0090e+00, -2.5543e+00],
  [-1.3450e+00, -1.0249e+00, -4.2279e+00,  …,  4.2372e+00,
    3.9641e+00, -2.5763e+00],
  …,
  [ 5.4986e+01,  5.5713e+01,  5.5712e+01,  …,  5.5759e+01,
    5.7337e+01,  3.2802e+01],
  [ 4.5442e+00,  4.6629e+00,  4.6629e+00,  …,  4.6629e+00,
    4.2169e+00,  1.1667e+01],
  [ 1.2839e+01,  1.2327e+01,  1.2327e+01,  …,  1.2327e+01,
    1.2327e+01,  1.3906e+01]],

 [[-6.3444e+00, -6.9196e+00, -5.5757e+00,  …, -8.2592e+00,
   -1.1398e+00, -1.5591e+01],
  [ 1.0395e+00, -3.8055e-01, -9.5284e-02,  …, -1.7352e+00,
    2.2434e+00, -7.6879e+00],
  [ 1.8202e+00,  2.1349e-01, -1.5452e-02,  …, -1.8431e+00,
    2.2034e+00, -7.7210e+00],
  …,
  [ 9.5203e+00,  5.9612e+00,  5.9768e+00,  …,  5.9865e+00,
    5.2210e+00,  3.7801e+00],
  [ 1.5300e+01,  1.6019e+01,  1.6019e+01,  …,  1.6019e+01,
    1.5170e+01,  1.3354e+01],
  [-3.9799e+00, -5.8648e+00, -5.8648e+00,  …, -5.8648e+00,
   -5.8648e+00, -7.0642e+00]],
```

```
 …,

 [[-2.0478e+01, -2.2598e+01, -2.0777e+01,  …, -2.3753e+01,
   -2.4263e+01, -2.1440e+01],
  [-9.8585e+00, -1.2322e+01, -1.0230e+01,  …, -1.3758e+01,
   -1.2040e+01, -7.7946e+00],
  [-9.2807e+00, -1.1539e+01, -1.0363e+01,  …, -1.3590e+01,
   -1.1974e+01, -7.8283e+00],
  …,
  [-7.3275e+01, -6.6248e+01, -6.6265e+01,  …, -6.6251e+01,
   -6.1465e+01, -5.2145e+01],
  [-1.3323e+01, -9.4404e+00, -9.4404e+00,  …, -9.4404e+00,
   -1.0110e+01, -9.4211e+00],
  [-1.5451e+01, -1.6507e+01, -1.6507e+01,  …, -1.6507e+01,
   -1.6507e+01, -7.8081e+00]],

 [[ 1.3795e+01,  1.4588e+01,  1.4677e+01,  …,  1.3607e+01,
    2.2596e+01, -4.7576e+01],
  [ 1.0507e+01,  1.1401e+01,  1.1518e+01,  …,  1.0260e+01,
    1.6043e+01, -5.4500e+01],
  [ 9.9744e+00,  1.1102e+01,  1.1739e+01,  …,  1.0111e+01,
    1.5968e+01, -5.4501e+01],
  …,
  [ 4.6472e+01,  4.7427e+01,  4.7465e+01,  …,  4.7438e+01,
    4.2562e+01,  1.5328e+01],
  [ 2.9481e+00,  2.1755e+00,  2.1755e+00,  …,  2.1755e+00,
    2.2690e+00, -9.8353e+00],
  [ 1.8044e+00,  6.4857e-01,  6.4857e-01,  …,  6.4857e-01,
    6.4857e-01, -1.0068e+01]],

 [[-9.0908e+00, -6.7313e+00, -7.3447e+00,  …, -7.7228e+00,
   -1.7124e+01,  2.6975e+01],
  [-8.7235e+00, -9.4913e+00, -9.2919e+00,  …, -1.0234e+01,
   -1.1948e+01,  2.1066e+01],
  [-8.9716e+00, -9.8008e+00, -9.9500e+00,  …, -1.0049e+01,
   -1.1935e+01,  2.1072e+01],
  …,
  [-4.9971e+01, -4.8766e+01, -4.8695e+01,  …, -4.8819e+01,
   -4.5204e+01, -3.8722e+01],
  [-1.2875e+01, -1.5963e+01, -1.5963e+01,  …, -1.5963e+01,
   -1.5427e+01, -2.9864e+01],
  [-1.3508e+01, -1.6490e+01, -1.6490e+01,  …, -1.6490e+01,
   -1.6490e+01, -2.2278e+01]]],


[[[-1.8759e+01, -1.3571e+01,  2.2548e+00,  …,  1.1345e+01,
    1.0527e+01, -6.3901e+00],
  [-1.4855e+01, -8.0141e+00,  3.0503e+00,  …,  9.0670e+00,
```

```
   2.5303e+01,  1.0627e+01],
 [-1.5508e+01, -5.5893e+00,  3.0969e+00,  …,  9.0317e+00,
   2.6862e+01,  1.4800e+01],
 …,
 [ 7.0900e+00, -1.9803e+00, -2.1520e+01,  …, -5.1502e+00,
  -1.3047e+00,  1.7927e+01],
 [-1.5578e-01, -1.1036e+01, -1.7480e+01,  …, -4.1643e+00,
   3.4397e+00,  1.9330e+01],
 [-3.0249e+00, -1.1836e+01, -2.3154e+01,  …,  2.3917e+00,
   6.9993e+00,  1.7383e+01]],

[[-8.8397e+00, -7.7601e+00, -2.3763e+00,  …, -1.5070e+01,
  -7.4422e+00,  7.1611e-01],
 [-5.9285e+00, -3.6261e+00,  8.7934e+00,  …, -9.5767e+00,
  -9.6715e+00,  1.6190e+00],
 [-6.5246e+00, -1.4535e+00,  1.3247e+01,  …, -1.0018e+01,
  -1.1298e+01,  1.8892e+00],
 …,
 [ 1.0908e+01,  1.3573e+01,  1.3182e+01,  …,  3.3219e+00,
   2.2183e+00,  2.6571e+00],
 [ 6.5088e+00,  6.4235e+00,  5.0449e+00,  …, -4.0747e-01,
  -1.9846e+00,  3.2633e+00],
 [ 1.0968e+01,  1.2070e+01,  1.0268e+01,  …,  4.8822e+00,
   3.6207e+00,  3.5838e+00]],

[[ 1.5291e+01,  3.9310e-01, -7.3584e-01,  …,  4.0508e+00,
   2.0628e+01,  7.1250e+00],
 [ 6.8693e+00, -4.4314e+00, -6.1610e+00,  …, -1.0027e+01,
  -8.1810e-01,  4.5664e+00],
 [ 7.3832e+00, -4.1252e+00, -3.0985e+00,  …, -8.1775e+00,
  -3.6036e+00,  1.5358e+00],
 …,
 [-1.5562e+00,  1.7099e+00,  2.5814e+00,  …,  6.0295e-01,
   1.0128e+00, -3.1992e+00],
 [ 3.9953e-01, -4.6898e+00, -8.7078e+00,  …, -5.6097e+00,
  -4.3097e+00, -3.2127e+00],
 [-2.2729e+00, -1.1109e+01, -7.2910e+00,  …,  1.5962e+00,
  -1.8988e+00, -1.5088e+00]],


…,

[[-4.4813e+00,  7.9173e+00,  3.5198e-01,  …, -8.5432e+00,
  -8.3646e-01,  1.2309e+01],
 [-1.5064e+01, -1.5934e+01, -2.8523e+01,  …, -2.3181e+01,
  -1.5945e+01, -9.3251e+00],
 [-1.2573e+01, -1.3917e+01, -2.2659e+01,  …, -1.1204e+01,
  -1.1605e+01, -6.7238e+00],
 …,
```

```
  [-1.6796e+01, -1.6347e+01, -1.2687e+01,  …, -1.4975e+01,
   -1.5475e+01, -1.2418e+01],
  [-1.7603e+01, -1.4934e+01, -1.1241e+01,  …, -1.2411e+01,
   -1.5266e+01, -1.3535e+01],
  [-1.7599e+01, -1.6508e+01, -1.9629e+01,  …, -1.6115e+01,
   -1.5373e+01, -1.3944e+01]],

 [[ 7.7735e+00, -1.0990e+01, -1.9105e+00,  …,  4.8452e-01,
    6.8461e+00,  1.3034e+00],
  [ 1.3239e+01, -1.3109e+00,  9.4061e+00,  …, -4.9497e-01,
    5.2156e+00,  1.3947e+01],
  [ 1.4563e+01, -2.9316e+00,  5.6282e+00,  …,  1.2749e+00,
    1.9717e+00,  1.1820e+01],
  …,
  [ 7.9129e+00,  2.2544e+00, -2.2184e+00,  …,  9.1478e+00,
    1.1359e+01, -5.2299e-01],
  [ 7.9191e+00, -3.3846e+00, -1.0885e+00,  …,  7.4320e+00,
    8.6430e+00,  4.3297e+00],
  [ 2.9228e+00, -2.2256e+00, -1.5574e+00,  …,  1.7159e+01,
    1.8074e+01,  1.1924e+01]],

 [[-1.1834e+01, -5.5214e+00, -1.1423e+01,  …, -1.2120e+01,
   -2.4766e+01, -1.9900e+01],
  [-1.0217e+01, -7.8127e+00, -1.3928e+01,  …, -2.0444e+00,
   -1.1811e+01, -2.5084e+01],
  [-9.9560e+00, -1.0222e+01, -1.5705e+01,  …,  1.1014e+00,
   -6.8687e+00, -2.6542e+01],
  …,
  [-8.7250e+00, -1.7517e+01, -1.7346e+01,  …, -1.0850e+01,
   -9.1884e+00, -3.1291e+00],
  [-9.4581e+00, -1.0917e+01, -4.9640e+00,  …, -5.6667e+00,
   -6.8806e+00, -5.2214e+00],
  [-1.6825e+01, -1.2906e+01, -1.6300e+01,  …, -9.5445e+00,
   -6.8411e+00, -6.9096e+00]]],


…,


[[[ 6.4074e-01,  8.7874e+00,  8.8274e+00,  …,  9.3949e+00,
    6.8171e+00,  6.4387e+01],
  [-1.9089e+00,  3.0082e+00,  3.4748e+00,  …,  6.3149e+00,
   -5.4054e-01,  6.4372e+01],
  [-1.8700e+00,  3.2801e+00,  3.9363e+00,  …,  5.7115e+00,
   -1.4773e+00,  6.4552e+01],
  …,
  [ 4.1504e+00,  3.3992e+00,  4.3356e+00,  …,  4.0419e+00,
    3.2470e+00,  1.7690e+01],
```

```
 [ 7.1571e+00,  6.3886e+00,  5.9076e+00,  …,  5.5444e+00,
   6.2103e+00,  1.8759e+01],
 [-2.2137e-01,  1.4860e+00,  1.1348e+00,  …,  4.0633e-01,
   3.2047e+00,  1.2869e+01]],

[[ 7.4229e+00,  8.3620e+00,  8.5034e+00,  …,  4.2402e+00,
   9.9218e+00,  6.9166e+00],
 [ 2.1892e-01,  3.5695e+00,  3.5151e+00,  …,  2.7053e+00,
   4.3660e+00,  2.7587e+00],
 [ 1.8107e-01,  3.6308e+00,  3.8150e+00,  …,  2.9404e+00,
   4.0811e+00,  1.0151e+00],
 …,
 [ 4.2122e+00,  4.8762e+00,  4.9767e+00,  …,  5.2311e+00,
   6.0753e+00,  7.9027e+00],
 [ 5.7669e+00,  6.2622e+00,  6.4587e+00,  …,  3.4963e+00,
   4.1494e+00,  7.9110e+00],
 [-2.0188e+01, -2.0229e+01, -2.0015e+01,  …, -1.4498e+01,
  -1.4801e+01, -5.5434e+00]],

[[-6.9579e+00, -8.1363e+00, -7.3679e+00,  …, -4.6615e+00,
   7.5069e-01, -1.3535e+01],
 [ 5.0364e-01, -1.5014e+00, -1.6496e+00,  …, -6.8120e-01,
   2.3540e+00, -7.7832e+00],
 [ 4.4075e-01, -1.3389e+00, -1.5547e+00,  …,  1.0512e+00,
   3.2676e+00, -7.7269e+00],
 …,
 [-3.8899e+00, -4.7426e+00, -5.4429e+00,  …,  1.7184e-01,
  -3.3772e-01, -7.0529e+00],
 [-7.5017e-01, -1.3209e+00, -1.3285e+00,  …, -1.0834e+00,
   3.3814e-01, -7.8669e+00],
 [-3.0521e+01, -3.1011e+01, -3.0507e+01,  …, -2.5652e+01,
  -2.6699e+01, -3.1648e+01]],


…,


[[-2.0352e+01, -2.3520e+01, -2.4015e+01,  …, -2.7380e+01,
  -2.7364e+01, -2.2725e+01],
 [-9.9807e+00, -1.3495e+01, -1.3366e+01,  …, -1.7331e+01,
  -1.7409e+01, -1.0496e+01],
 [-1.0071e+01, -1.3649e+01, -1.3460e+01,  …, -1.7323e+01,
  -1.6830e+01, -9.8644e+00],
 …,
 [-1.9019e+01, -1.8257e+01, -1.7793e+01,  …, -1.8339e+01,
  -1.7762e+01, -9.0382e+00],
 [-2.1478e+01, -1.8496e+01, -1.8311e+01,  …, -1.8982e+01,
  -1.9142e+01, -7.3694e+00],
 [-2.8928e+01, -2.9258e+01, -2.9340e+01,  …, -3.0770e+01,
  -2.9620e+01, -1.4335e+01]],
```

```
 [[ 1.3064e+01,  1.4282e+01,  1.4323e+01,  …,  1.5607e+01,
    2.2382e+01, -4.0227e+01],
  [ 9.8191e+00,  1.1062e+01,  1.0367e+01,  …,  1.3684e+01,
    1.7811e+01, -4.6843e+01],
  [ 9.8707e+00,  1.1233e+01,  1.0666e+01,  …,  1.3437e+01,
    1.7939e+01, -4.7331e+01],
  …,
  [ 4.9637e+00,  4.4912e+00,  4.2012e+00,  …,  5.4252e+00,
    7.5182e+00, -2.3726e+01],
  [ 7.7068e+00,  8.8693e+00,  8.8760e+00,  …,  7.8324e+00,
    9.7799e+00, -1.8897e+01],
  [-6.3873e+00, -7.0745e+00, -6.9692e+00,  …, -3.9625e+00,
   -4.4854e+00, -1.8393e+01]],

 [[-8.9454e+00, -7.5094e+00, -8.3056e+00,  …, -8.4338e+00,
   -1.5584e+01,  1.7735e+01],
  [-8.5585e+00, -1.0140e+01, -9.9552e+00,  …, -1.0945e+01,
   -1.2150e+01,  1.3998e+01],
  [-8.5006e+00, -1.0159e+01, -9.8130e+00,  …, -1.1333e+01,
   -1.1440e+01,  1.5912e+01],
  …,
  [-1.1536e+01, -1.1031e+01, -1.0597e+01,  …, -1.1214e+01,
   -9.4285e+00, -6.0782e+00],
  [-1.2071e+01, -1.1889e+01, -1.2102e+01,  …, -9.6835e+00,
   -9.4967e+00, -5.2127e+00],
  [-2.3209e+00, -5.7957e+00, -6.1483e+00,  …, -8.3349e+00,
   -8.0367e+00, -2.7801e+00]]],


[[[ 4.7015e+00, -2.4771e+00,  4.1577e-01,  …, -2.2243e+00,
    3.7396e+00, -8.1508e-01],
  [ 1.0221e+01, -1.6751e+00, -4.4834e+01,  …, -3.2928e+01,
   -1.3754e+01, -3.2438e-01],
  [ 1.0221e+01, -9.5990e+00, -3.6675e+01,  …, -2.8701e+01,
   -8.3973e+00,  8.0311e+00],
  …,
  [ 1.0221e+01, -1.0701e+01, -4.9303e+01,  …,  1.4567e+00,
    4.9524e+00,  5.7570e+00],
  [ 1.0221e+01, -1.1050e+01, -4.9074e+01,  …,  3.1707e+00,
    3.5666e+00,  2.6834e+00],
  [ 7.4953e+00, -1.3092e+01, -5.0172e+01,  …,  4.5106e+00,
    5.3927e+00,  3.1559e+00]],

 [[ 4.0701e+00,  4.5395e+00,  5.3563e+00,  …,  4.1466e+01,
    4.8754e+01,  5.4827e+01],
  [ 8.7234e+00,  9.8142e+00, -1.7531e+01,  …, -4.0495e+01,
   -4.6792e+01, -5.1494e+01],
```

```
 [ 8.7234e+00,  7.3309e+00, -2.6361e+00,  …,  5.8546e+00,
   8.7054e+00,  1.6407e+01],
 …,
 [ 8.7234e+00,  1.1773e+01, -3.6240e+00,  …,  4.2757e+00,
   2.8692e+00,  2.0650e+00],
 [ 8.7234e+00,  1.1854e+01, -3.0290e+00,  …,  4.8115e+00,
   4.8600e+00,  3.0559e+00],
 [ 1.2839e+01,  1.5699e+01,  1.0213e+00,  …,  6.0862e+00,
   6.0379e+00,  4.0662e+00]],

[[ 7.6247e+00,  2.5126e+00, -5.6479e-01,  …,  3.2755e+01,
   4.2728e+01,  4.2554e+01],
 [ 1.9249e-01, -3.6290e+00,  6.6563e+00,  …, -1.4505e+01,
  -1.9329e+01, -1.7062e+01],
 [ 1.9249e-01, -5.3826e+00,  6.1890e+00,  …, -6.5611e+00,
  -1.0358e+01, -5.9316e+00],
 …,
 [ 1.9249e-01, -4.9544e+00,  4.9813e+00,  …, -6.5192e+00,
  -8.9447e+00, -4.0788e+00],
 [ 1.9249e-01, -5.4278e+00,  4.1611e+00,  …, -6.4724e+00,
  -6.9626e+00, -6.2007e+00],
 [-3.9799e+00, -8.4702e+00,  5.6548e+00,  …, -2.1598e+00,
  -9.7396e-01,  1.1391e+00]],


…,

[[-1.4452e+01, -4.4418e+00,  2.1497e+00,  …, -3.2919e+00,
  -2.1499e+00,  7.9983e-01],
 [-1.9289e+01, -7.4397e+00, -9.1374e+00,  …,  5.4513e+00,
   9.2807e+00,  3.8383e+00],
 [-1.9289e+01, -8.9219e+00, -1.5294e+01,  …, -2.8147e+01,
  -2.8869e+01, -3.2468e+01],
 …,
 [-1.9289e+01, -5.0430e+00, -8.3814e+00,  …, -1.4203e+01,
  -1.3601e+01, -1.4946e+01],
 [-1.9289e+01, -4.7868e+00, -8.4752e+00,  …, -1.7198e+01,
  -1.6578e+01, -1.5964e+01],
 [-1.5451e+01, -4.2726e+00, -1.0142e+01,  …, -1.2647e+01,
  -1.2701e+01, -1.2599e+01]],

[[ 2.7490e+00, -6.0613e+00,  1.7512e+01,  …,  8.8076e+00,
   6.8785e+00,  1.9049e+01],
 [ 5.8133e+00, -6.3287e+00,  1.0974e+01,  …, -2.2793e+01,
  -2.9773e+01, -2.0761e+01],
 [ 5.8133e+00, -9.0321e+00,  1.3233e+01,  …,  5.9030e+00,
   6.0384e+00,  1.7956e+01],
 …,
 [ 5.8133e+00, -1.0561e+01,  1.5313e+01,  …,  2.0060e-01,
```

```
          -9.6218e-01,  1.7378e+00],
         [ 5.8133e+00, -1.0993e+01,  1.5532e+01,  ...,  1.1225e+00,
           1.8814e+00,  4.2525e+00],
         [ 1.8044e+00, -1.3995e+01,  1.1999e+01,  ...,  2.3099e+00,
           3.3446e+00,  4.5827e+00]],

        [[-8.3171e+00, -1.2924e+01, -1.2077e+01,  ..., -4.8797e+01,
          -5.5165e+01, -5.8742e+01],
         [-7.9600e+00, -1.3624e+01, -1.7381e+01,  ...,  9.9681e+00,
           1.6259e+01,  5.5196e+00],
         [-7.9600e+00, -1.4463e+01, -1.7527e+01,  ..., -6.1581e+00,
          -4.1244e+00, -1.6327e+01],
         ...,
         [-7.9600e+00, -2.0359e+01, -1.5277e+01,  ..., -1.0072e+01,
          -8.1767e+00, -1.0702e+01],
         [-7.9600e+00, -2.0476e+01, -1.5122e+01,  ..., -1.2802e+01,
          -1.1729e+01, -1.0512e+01],
         [-1.3508e+01, -2.4820e+01, -1.6449e+01,  ..., -1.1084e+01,
          -1.2061e+01, -1.2574e+01]]],


       [[[ 9.1537e+00,  3.3970e+00,  3.4872e+00,  ...,  6.8960e+00,
           5.3063e+00, -1.2703e+01],
         [ 9.1774e-01,  4.2631e-01,  6.4064e-01,  ...,  4.5872e+01,
           3.4464e+01,  8.0513e+00],
         [-4.0728e-01,  5.1469e+00,  5.3913e+00,  ...,  2.3642e+01,
           4.6875e+01,  3.4377e+01],
         ...,
         [ 1.3264e+00, -6.6643e+00, -1.2113e+01,  ...,  7.6281e+00,
           8.3927e+00,  1.8003e+01],
         [ 7.6131e+00,  4.0600e+00, -6.2050e+00,  ...,  7.4357e+00,
           8.5335e+00,  1.5465e+01],
         [ 1.8885e+00,  5.4769e+00, -4.3746e+00,  ...,  4.7139e+00,
           6.1386e+00,  1.0323e+01]],

        [[ 4.4188e+01,  4.0361e+01,  4.0633e+01,  ...,  3.3716e+01,
           2.1515e+01,  1.2646e+01],
         [-3.3873e+01, -3.1809e+01, -3.2133e+01,  ..., -4.3480e+00,
           1.2734e+01,  1.9137e+01],
         [ 5.4177e+00,  4.5526e+00,  4.6873e+00,  ...,  1.8556e+00,
           1.1335e+01,  1.9837e+01],
         ...,
         [-1.5991e+01, -2.4308e+01, -1.3503e+01,  ...,  3.6778e+00,
           4.8475e+00,  1.1892e+01],
         [-8.8594e+00, -2.1341e+01, -1.8985e+01,  ...,  4.6831e+00,
           5.8138e+00,  1.1333e+01],
         [-4.4197e+00, -1.9613e+01, -2.0956e+01,  ...,  5.6486e+00,
           6.8123e+00,  1.2134e+01]],
```

```
[[ 4.3067e+01,  3.6879e+01,  3.7182e+01,  …,  3.1464e+01,
   1.4829e+01,  1.0406e+01],
 [-8.7329e+00, -7.2054e+00, -7.7407e+00,  …,  1.2911e+00,
   2.4727e+01,  2.7899e+00],
 [-8.4452e+00, -7.5601e+00, -7.5214e+00,  …, -1.6686e+01,
  -8.8463e+00,  2.1093e+01],
 …,
 [-1.1956e+01,  3.4047e+00, -3.0050e-01,  …, -9.2581e-01,
  -1.7254e-01, -7.0192e+00],
 [-1.4636e+01, -1.8818e+00,  1.8258e+00,  …, -5.7471e-01,
  -1.1421e+00, -6.8675e+00],
 [-1.9442e+01, -9.5814e+00,  7.1815e-01,  …, -4.9617e+00,
  -5.7437e+00, -1.1235e+01]],


…,


[[ 1.2877e+01,  8.4026e+00,  8.0334e+00,  …,  5.1331e+00,
   3.3094e+00, -6.6517e+00],
 [ 8.9495e+00,  6.7478e+00,  7.0824e+00,  …, -2.6555e+00,
  -8.8470e+00,  5.8697e+00],
 [-2.8822e+01, -3.0968e+01, -3.0819e+01,  …, -1.3524e+01,
  -7.5763e+00, -7.8364e+00],
 …,
 [-2.3224e+01, -2.6337e+01, -2.2401e+01,  …, -2.2206e+01,
  -2.3383e+01, -1.3466e+01],
 [-1.5551e+01, -2.0942e+01, -2.2883e+01,  …, -2.1685e+01,
  -2.2169e+01, -1.2878e+01],
 [-3.0090e+00, -1.2737e+01, -1.9377e+01,  …, -2.0947e+01,
  -2.1644e+01, -1.0824e+01]],

[[ 4.9949e+00,  5.7275e+00,  5.5158e+00,  …,  4.7120e+00,
   1.5389e+01, -9.2008e+00],
 [-1.6210e+01, -1.3870e+01, -1.4223e+01,  …, -1.2095e+01,
  -2.0113e+00, -6.1936e+00],
 [ 8.8733e+00,  1.1759e+01,  1.2059e+01,  …,  9.4240e-03,
  -5.8969e+00, -2.0225e+01],
 …,
 [ 2.0568e+01,  2.7923e+01,  2.2054e+01,  …,  9.1665e+00,
   1.1057e+01, -1.5805e+01],
 [ 1.9839e+01,  1.8722e+01,  2.7943e+01,  …,  8.8729e+00,
   9.8277e+00, -1.5601e+01],
 [ 9.8013e+00,  4.6062e+00,  2.1883e+01,  …,  4.9823e+00,
   5.3207e+00, -1.7237e+01]],

[[-3.9636e+01, -4.3747e+01, -4.4330e+01,  …, -3.3878e+01,
  -1.8318e+01, -2.3583e+01],
 [ 3.6374e+00,  3.4929e+00,  4.1060e+00,  …, -9.5570e+00,
```

```
          -3.2202e+01, -1.3154e+01],
         [-1.2211e+01, -8.6939e+00, -8.6881e+00,  …,  3.1360e+00,
          -4.7789e+00, -3.3698e+01],
         …,
         [-6.2840e+00, -2.1588e+01, -1.3499e+01,  …, -1.0663e+01,
          -1.0374e+01, -1.0072e+01],
         [-5.0162e+00, -1.7383e+01, -1.7169e+01,  …, -1.1020e+01,
          -9.7810e+00, -1.1129e+01],
         [-5.7763e+00, -1.1241e+01, -2.0904e+01,  …, -1.4545e+01,
          -1.3619e+01, -1.4177e+01]]]], device='cuda:0',
       grad_fn=<CudnnConvolutionBackward>)
```

[8]:
```python
#### input floating number / weight floating number version

conv_ref = torch.nn.Conv2d(in_channels = 16, out_channels=16, kernel_size = 3,␣
 ↪bias = False)
weight = model.layer1[0].conv1.weight
mean = weight.data.mean()
std = weight.data.std()
conv_ref.weight = torch.nn.parameter.Parameter(weight.add(-mean).div(std))

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

```
tensor([[[[-2.2607e+00, -2.4806e+01, -3.1008e+01,  …, -3.5973e+00,
            1.8554e+00,  3.5235e+00],
          [ 3.3864e+00, -2.0572e+01, -2.4636e+01,  …, -4.9534e+00,
            6.4759e+00,  6.3380e+00],
          [ 3.7171e+00, -1.9050e+01, -2.4396e+01,  …, -7.8262e-01,
            6.0045e+00,  8.5060e+00],
          …,
          [-3.5722e-01, -3.1624e+01, -4.4902e+01,  …,  1.8069e+01,
            1.3894e+01,  1.1173e+01],
          [-2.1447e+00, -3.7620e+01, -3.6145e+01,  …,  4.2146e+00,
            2.4184e+00,  2.2785e+00],
          [-2.7171e+00, -2.7564e+01, -1.9787e+01,  …, -5.8661e+00,
           -5.2428e+00, -8.9488e+00]],

         [[ 1.0944e+01,  2.2099e+01,  1.1629e+01,  …,  4.0247e+00,
            2.8597e+00, -5.2097e+00],
          [ 1.7111e+01,  2.5431e+01,  2.0005e+01,  …, -1.0258e+00,
           -1.6340e-01, -1.6222e+00],
          [ 1.7063e+01,  2.5619e+01,  1.8543e+01,  …, -3.5446e+00,
            9.2673e+00,  1.2805e+01],
          …,
          [ 1.5125e+01,  1.6920e+01,  2.4425e+00,  …,  4.1642e+00,
            1.4548e+01,  6.2648e+00],
          [ 1.3905e+01,  8.4446e+00, -3.8387e+00,  …,  1.2654e+01,
```

```
   8.2070e+00, -2.7279e-01],
 [ 1.6499e+01,  1.3795e+01,  8.0555e+00,  …, -3.3909e+01,
  -3.3314e+01, -3.5777e+01]],

[[ 3.4884e+00,  1.2117e+01, -9.9786e-01,  …, -8.9902e-01,
  -3.8987e+00, -3.6257e+00],
 [-3.7175e+00,  1.1520e+00, -1.3428e+01,  …, -1.8521e+01,
  -1.7518e+01, -1.1624e+01],
 [-4.4134e+00, -1.4843e+00, -1.5459e+01,  …, -3.7981e+01,
  -2.8982e+01, -6.1441e+00],
 …,
 [-6.6215e+00, -1.0081e+00, -1.0945e+01,  …, -6.1917e+00,
  -2.1669e+00, -5.8865e+00],
 [-4.0929e+00,  3.3703e+00, -2.7538e+01,  …, -1.5339e+01,
  -1.4128e+01, -1.8259e+01],
 [-3.2674e+00, -2.3998e+00, -4.2927e+01,  …, -8.2682e+01,
  -8.4459e+01, -8.3774e+01]],


…,


[[-1.6114e+01, -3.5703e+01, -3.5443e+01,  …, -2.1780e+01,
  -1.8070e+01, -1.3383e+01],
 [-2.1955e+01, -4.0883e+01, -3.5332e+01,  …, -1.0586e+01,
  -1.5772e+01, -1.0920e+01],
 [-2.1563e+01, -3.8697e+01, -3.3508e+01,  …, -5.7561e+00,
  -1.8123e+01, -1.8097e+01],
 …,
 [-1.4249e+01, -2.5715e+01, -2.7811e+01,  …, -3.0157e+01,
  -2.9566e+01, -1.1260e+01],
 [-1.3948e+01, -2.9879e+01, -2.2453e+01,  …, -1.9480e+01,
  -1.8800e+01, -6.3278e+00],
 [-1.6885e+01, -3.1885e+01, -1.8623e+01,  …, -1.2703e+01,
  -1.3828e+01, -3.7483e+00]],

[[-9.3518e-01,  2.1477e+01,  4.4698e+00,  …,  1.6993e+01,
   1.5430e+01,  1.4968e+01],
 [ 1.4682e+00,  2.4784e+01,  7.7387e+00,  …,  1.6578e+01,
   1.3800e+01,  1.6958e+01],
 [ 6.2748e-01,  2.3337e+01,  5.7627e+00,  …,  1.2711e+01,
   4.5760e+00,  1.4314e+01],
 …,
 [-5.7061e+00,  2.2038e+01, -1.0482e+01,  …,  4.2421e+01,
   4.2607e+01,  2.2153e+01],
 [-4.9141e+00,  1.7116e+01, -1.7067e+01,  …,  3.4860e+01,
   3.2146e+01,  2.0091e+01],
 [-2.5028e+00,  2.8071e+00, -3.8354e+00,  …, -1.6382e+01,
  -1.5358e+01, -2.2338e+01]],
```

```
 [[-1.2947e+01, -1.6006e+01,  4.2763e+00,  …, -1.1959e+01,
   -3.8732e+00, -7.6798e+00],
  [-1.3354e+01, -1.1432e+01,  1.8138e+00,  …, -1.8118e+00,
   -4.2084e+00, -9.0229e+00],
  [-1.3768e+01, -1.1896e+01, -1.2683e+00,  …, -4.5817e+00,
   -9.0787e+00, -1.8195e+01],
  …,
  [-1.5026e+01, -1.3579e+01, -3.4098e+00,  …, -4.7605e+00,
   -9.6189e+00, -4.4635e+00],
  [-1.3590e+01, -1.5296e+01,  1.3302e+01,  …, -1.2151e+01,
   -8.7937e+00, -3.5098e+00],
  [-1.9270e+01, -2.4584e+01,  1.2079e+01,  …, -3.1588e+00,
   -3.1150e+00, -3.8960e+00]]],


[[[ 4.1439e-01,  7.7543e+00,  8.6692e+00,  …,  6.9306e+00,
    1.4993e+00,  7.1186e+01],
  [-1.3085e+00,  3.5837e+00,  5.7443e+00,  …,  2.3576e+00,
   -6.9277e+00,  7.1373e+01],
  [-6.6192e-01,  5.2636e+00,  8.0599e+00,  …,  2.0398e+00,
   -6.9678e+00,  7.1356e+01],
  …,
  [-4.0233e+00, -2.2005e+00, -2.2020e+00,  …, -2.1844e+00,
   -6.1783e+00, -3.8147e+00],
  [ 2.1140e+00,  7.8505e-02,  7.8505e-02,  …,  7.8505e-02,
    2.6618e-01, -1.1540e+01],
  [ 8.4914e+00,  7.2008e+00,  7.2008e+00,  …,  7.2008e+00,
    7.2008e+00, -7.2584e+00]],

 [[ 5.8485e+00,  6.6966e+00,  5.5037e+00,  …,  9.3834e+00,
    1.3919e+01, -9.2770e-01],
  [-1.1754e+00,  4.2678e-01, -1.8247e+00,  …,  4.8260e+00,
    4.6740e+00, -7.2981e+00],
  [-1.3299e+00, -2.0527e-01, -3.2549e+00,  …,  4.7980e+00,
    4.6321e+00, -7.3158e+00],
  …,
  [ 5.7700e+01,  5.8701e+01,  5.8698e+01,  …,  5.8751e+01,
    5.9220e+01,  3.1471e+01],
  [ 4.2732e+00,  4.0724e+00,  4.0724e+00,  …,  4.0724e+00,
    3.5434e+00,  1.1040e+01],
  [ 1.2611e+01,  1.1959e+01,  1.1959e+01,  …,  1.1959e+01,
    1.1959e+01,  1.2994e+01]],

 [[-4.0523e+00, -4.6455e+00, -3.2323e+00,  …, -6.0962e+00,
    2.5807e+00, -1.1709e+01],
  [ 2.8094e+00,  1.6967e+00,  2.0938e+00,  …,  2.0411e-01,
    5.9740e+00, -4.2674e+00],
  [ 3.6145e+00,  2.3208e+00,  2.1943e+00,  …,  8.3034e-02,
```

```
  5.9342e+00, -4.3002e+00],
 …,
 [ 1.2397e+01,  8.7891e+00,  8.8040e+00,  …,  8.8133e+00,
   8.2210e+00,  5.6150e+00],
 [ 1.5652e+01,  1.6080e+01,  1.6080e+01,  …,  1.6080e+01,
   1.4973e+01,  1.2394e+01],
 [-3.5797e+00, -5.4336e+00, -5.4336e+00,  …, -5.4336e+00,
  -5.4336e+00, -7.3298e+00]],

…,

[[-1.9784e+01, -2.2392e+01, -2.0538e+01,  …, -2.3581e+01,
  -2.4784e+01, -2.0617e+01],
 [-9.3026e+00, -1.1943e+01, -9.8735e+00,  …, -1.3362e+01,
  -1.2412e+01, -6.6561e+00],
 [-8.7136e+00, -1.1157e+01, -1.0079e+01,  …, -1.3189e+01,
  -1.2341e+01, -6.6879e+00],
 …,
 [-7.2592e+01, -6.5958e+01, -6.5977e+01,  …, -6.5963e+01,
  -6.1344e+01, -5.2638e+01],
 [-1.4229e+01, -1.0074e+01, -1.0074e+01,  …, -1.0074e+01,
  -1.0898e+01, -9.3832e+00],
 [-1.6086e+01, -1.6892e+01, -1.6892e+01,  …, -1.6892e+01,
  -1.6892e+01, -8.5405e+00]],

[[ 1.2878e+01,  1.2854e+01,  1.3021e+01,  …,  1.1791e+01,
   2.0236e+01, -4.8860e+01],
 [ 1.0370e+01,  1.0619e+01,  1.0819e+01,  …,  9.3653e+00,
   1.5188e+01, -5.5580e+01],
 [ 9.9080e+00,  1.0397e+01,  1.1017e+01,  …,  9.2305e+00,
   1.5115e+01, -5.5579e+01],
 …,
 [ 4.6548e+01,  4.7821e+01,  4.7855e+01,  …,  4.7832e+01,
   4.2880e+01,  1.4453e+01],
 [ 3.5842e+00,  3.5652e+00,  3.5652e+00,  …,  3.5652e+00,
   3.5640e+00, -9.5140e+00],
 [ 5.8782e-01, -2.8850e-01, -2.8850e-01,  …, -2.8850e-01,
  -2.8850e-01, -1.0306e+01]],

[[-9.8261e+00, -6.7614e+00, -7.3915e+00,  …, -7.6334e+00,
  -2.1641e+01,  3.0866e+01],
 [-9.0708e+00, -9.3143e+00, -9.0939e+00,  …, -9.9351e+00,
  -1.6642e+01,  2.6197e+01],
 [-9.3543e+00, -9.6495e+00, -9.7078e+00,  …, -9.7392e+00,
  -1.6621e+01,  2.6200e+01],
 …,
 [-5.1161e+01, -4.9268e+01, -4.9202e+01,  …, -4.9316e+01,
  -4.6878e+01, -3.7110e+01],
```

```
         [-1.2441e+01, -1.5234e+01, -1.5234e+01,  …, -1.5234e+01,
          -1.4557e+01, -2.8611e+01],
         [-1.3123e+01, -1.5976e+01, -1.5976e+01,  …, -1.5976e+01,
          -1.5976e+01, -2.1312e+01]]],


       [[[-1.6850e+01, -1.2462e+01,  2.5122e+00,  …,  1.2443e+01,
           1.1429e+01, -5.3496e+00],
         [-1.3502e+01, -7.1985e+00,  2.7839e+00,  …,  9.5813e+00,
           2.5659e+01,  1.1113e+01],
         [-1.3863e+01, -4.6487e+00,  2.9189e+00,  …,  9.5333e+00,
           2.8180e+01,  1.5602e+01],
         …,
         [ 8.6239e+00, -5.8787e-01, -2.0259e+01,  …, -5.6264e+00,
          -1.9819e+00,  1.7570e+01],
         [ 1.8496e+00, -9.5073e+00, -1.6601e+01,  …, -4.4209e+00,
           2.8628e+00,  1.8657e+01],
         [-1.9825e+00, -1.1322e+01, -2.2441e+01,  …,  2.2029e+00,
           6.7218e+00,  1.6933e+01]],

        [[-7.8857e+00, -7.9912e+00, -2.4734e+00,  …, -1.5139e+01,
          -8.1193e+00, -1.3449e-01],
         [-3.9214e+00, -3.3124e+00,  9.1593e+00,  …, -9.9722e+00,
          -1.0031e+01,  1.7558e+00],
         [-4.5218e+00, -1.4373e+00,  1.3628e+01,  …, -9.7511e+00,
          -1.1716e+01,  1.9593e+00],
         …,
         [ 1.1708e+01,  1.3569e+01,  1.3317e+01,  …,  4.0308e+00,
           2.5100e+00,  1.9819e+00],
         [ 7.1325e+00,  6.4064e+00,  5.8005e+00,  …,  3.9868e-03,
          -1.6044e+00,  3.0760e+00],
         [ 1.1078e+01,  1.2105e+01,  1.0566e+01,  …,  5.8008e+00,
           4.3868e+00,  3.8219e+00]],

        [[ 1.5467e+01,  3.0721e-01, -1.0465e+00,  …,  5.5304e+00,
           2.0703e+01,  6.1167e+00],
         [ 7.3442e+00, -4.6168e+00, -6.0783e+00,  …, -7.9232e+00,
           6.9666e-01,  4.1081e+00],
         [ 7.9641e+00, -4.2745e+00, -3.4133e+00,  …, -6.7579e+00,
          -2.0846e+00,  1.5839e+00],
         …,
         [-1.1147e+00,  1.6628e+00,  2.3536e+00,  …,  2.2190e+00,
           3.0005e+00, -6.7281e-01],
         [ 6.2890e-01, -4.9738e+00, -9.1939e+00,  …, -3.9656e+00,
          -2.2999e+00, -7.9947e-01],
         [-2.3130e+00, -1.1424e+01, -7.5492e+00,  …,  3.8783e+00,
           7.7425e-01,  1.6031e+00]],
```

```
 …,

 [[-5.4207e+00,  7.7221e+00,  9.9075e-02,  …, -8.3200e+00,
   -7.3783e-01,  1.1831e+01],
  [-1.5436e+01, -1.5794e+01, -2.8239e+01,  …, -2.2708e+01,
   -1.5856e+01, -1.0064e+01],
  [-1.3062e+01, -1.3698e+01, -2.2309e+01,  …, -1.0961e+01,
   -1.1639e+01, -7.4180e+00],
  …,
  [-1.7113e+01, -1.6545e+01, -1.2578e+01,  …, -1.4521e+01,
   -1.5153e+01, -1.2008e+01],
  [-1.7904e+01, -1.4619e+01, -1.0892e+01,  …, -1.1930e+01,
   -1.4791e+01, -1.3230e+01],
  [-1.8206e+01, -1.6856e+01, -2.0009e+01,  …, -1.5412e+01,
   -1.4743e+01, -1.3546e+01]],

 [[ 7.4709e+00, -1.0960e+01, -2.1252e+00,  …,  2.8975e-01,
    7.0925e+00,  2.0418e+00],
  [ 1.2001e+01, -2.6690e+00,  8.3250e+00,  …, -1.4034e+00,
    4.5866e+00,  1.3344e+01],
  [ 1.3513e+01, -3.9541e+00,  5.1274e+00,  …,  3.4151e-01,
    1.1077e+00,  1.1408e+01],
  …,
  [ 7.0360e+00,  1.7918e+00, -2.2141e+00,  …,  8.2853e+00,
    1.0252e+01, -1.5901e+00],
  [ 6.9251e+00, -3.7869e+00, -1.5664e+00,  …,  6.7002e+00,
    7.6403e+00,  3.1758e+00],
  [ 1.5851e+00, -3.1128e+00, -2.0755e+00,  …,  1.5740e+01,
    1.6642e+01,  1.0324e+01]],

 [[-1.2092e+01, -4.3744e+00, -1.0691e+01,  …, -1.1490e+01,
   -2.3952e+01, -1.8860e+01],
  [-1.0837e+01, -6.9491e+00, -1.4000e+01,  …, -2.4517e+00,
   -1.0943e+01, -2.4464e+01],
  [-1.0635e+01, -9.1465e+00, -1.5629e+01,  …,  1.1620e+00,
   -5.6716e+00, -2.5967e+01],
  …,
  [-8.7235e+00, -1.7191e+01, -1.7466e+01,  …, -1.0803e+01,
   -9.6168e+00, -3.1556e+00],
  [-9.3901e+00, -1.0525e+01, -5.0503e+00,  …, -5.5392e+00,
   -7.1602e+00, -5.7140e+00],
  [-1.6366e+01, -1.2629e+01, -1.6764e+01,  …, -8.8912e+00,
   -6.5486e+00, -7.0545e+00]]],


…,
```

```
[[[-9.8454e-01,  7.4633e+00,  7.5140e+00,  …,  8.0547e+00,
    4.2421e+00,  6.8263e+01],
  [-2.9162e+00,  2.5960e+00,  3.0550e+00,  …,  5.3811e+00,
   -2.3969e+00,  6.7723e+01],
  [-2.8680e+00,  2.8284e+00,  3.5165e+00,  …,  4.8679e+00,
   -3.3094e+00,  6.8016e+01],
  …,
  [ 4.2556e+00,  3.3935e+00,  4.3343e+00,  …,  4.0765e+00,
    2.9215e+00,  1.9886e+01],
  [ 7.3493e+00,  6.4932e+00,  6.0301e+00,  …,  5.7870e+00,
    6.0374e+00,  2.1047e+01],
  [ 1.1694e+00,  2.6881e+00,  2.3034e+00,  …,  1.3067e+00,
    3.5723e+00,  1.4801e+01]],

 [[ 7.2219e+00,  9.3163e+00,  9.4556e+00,  …,  5.3841e+00,
    1.1504e+01,  2.9697e+00],
  [ 1.1659e-01,  4.1148e+00,  4.0432e+00,  …,  3.5599e+00,
    5.1645e+00, -1.6617e+00],
  [ 7.3544e-02,  4.2104e+00,  4.3565e+00,  …,  3.7325e+00,
    4.8626e+00, -3.4137e+00],
  …,
  [ 4.4910e+00,  5.0306e+00,  5.1075e+00,  …,  5.3378e+00,
    6.2594e+00,  5.4639e+00],
  [ 6.4357e+00,  6.8971e+00,  7.0601e+00,  …,  3.9834e+00,
    4.7066e+00,  6.0130e+00],
  [-1.9788e+01, -2.0010e+01, -1.9785e+01,  …, -1.4326e+01,
   -1.4511e+01, -6.9887e+00]],

 [[-4.7267e+00, -5.9464e+00, -5.1787e+00,  …, -2.5427e+00,
    4.2484e+00, -9.7829e+00],
  [ 2.1587e+00,  4.5000e-01,  3.1772e-01,  …,  1.2663e+00,
    5.8782e+00, -4.4447e+00],
  [ 2.1115e+00,  6.2707e-01,  4.0558e-01,  …,  2.9834e+00,
    6.8160e+00, -4.3636e+00],
  …,
  [-2.9509e+00, -3.8077e+00, -4.5003e+00,  …,  9.2581e-01,
    1.0246e+00, -6.1364e+00],
  [ 8.6183e-01,  1.9255e-01,  1.8740e-01,  …,  1.2616e-01,
    2.0030e+00, -6.9787e+00],
  [-3.0189e+01, -3.1046e+01, -3.0556e+01,  …, -2.5672e+01,
   -2.6247e+01, -3.1530e+01]],


 …,


 [[-1.9672e+01, -2.3350e+01, -2.3839e+01,  …, -2.7358e+01,
   -2.7906e+01, -2.2118e+01],
  [-9.4489e+00, -1.3109e+01, -1.2963e+01,  …, -1.7204e+01,
   -1.7853e+01, -9.5454e+00],
```

```
 [-9.5266e+00, -1.3253e+01, -1.3063e+01,  …, -1.7152e+01,
  -1.7223e+01, -8.8761e+00],
 …,
 [-1.8898e+01, -1.8083e+01, -1.7659e+01,  …, -1.8153e+01,
  -1.7794e+01, -8.7008e+00],
 [-2.0854e+01, -1.8039e+01, -1.7880e+01,  …, -1.8585e+01,
  -1.8943e+01, -6.8481e+00],
 [-2.8612e+01, -2.8709e+01, -2.8793e+01,  …, -3.0557e+01,
  -2.9705e+01, -1.4781e+01]],

[[ 1.2121e+01,  1.2429e+01,  1.2464e+01,  …,  1.3941e+01,
   2.0248e+01, -4.1378e+01],
 [ 9.6041e+00,  1.0128e+01,  9.4617e+00,  …,  1.2801e+01,
   1.6867e+01, -4.7804e+01],
 [ 9.6552e+00,  1.0288e+01,  9.7278e+00,  …,  1.2516e+01,
   1.7040e+01, -4.8334e+01],
 …,
 [ 4.3052e+00,  4.0311e+00,  3.7428e+00,  …,  4.9920e+00,
   7.1641e+00, -2.3678e+01],
 [ 6.7463e+00,  8.0716e+00,  8.0848e+00,  …,  7.1115e+00,
   9.1588e+00, -1.8949e+01],
 [-8.1018e+00, -8.8262e+00, -8.7238e+00,  …, -5.5702e+00,
  -5.9233e+00, -1.9518e+01]],

[[-9.6097e+00, -7.4412e+00, -8.2511e+00,  …, -8.4720e+00,
  -1.9451e+01,  2.1257e+01],
 [-8.8084e+00, -9.8770e+00, -9.6709e+00,  …, -1.0545e+01,
  -1.5971e+01,  1.8629e+01],
 [-8.7499e+00, -9.9143e+00, -9.5087e+00,  …, -1.0908e+01,
  -1.5335e+01,  2.0614e+01],
 …,
 [-1.1308e+01, -1.0867e+01, -1.0413e+01,  …, -1.0990e+01,
  -1.0657e+01, -3.8309e+00],
 [-1.1548e+01, -1.1267e+01, -1.1490e+01,  …, -9.0847e+00,
  -1.0182e+01, -2.8194e+00],
 [-1.6434e+00, -5.2873e+00, -5.6514e+00,  …, -7.8606e+00,
  -8.4224e+00, -1.3367e+00]]],


[[[ 6.5295e+00, -8.2979e-01,  5.6145e-01,  …, -3.8622e+00,
    1.6402e+00, -2.4983e+00],
  [ 1.2065e+01, -2.9821e-01, -4.2284e+01,  …, -3.1361e+01,
   -1.2676e+01,  1.1832e+00],
  [ 1.2065e+01, -7.2636e+00, -3.6376e+01,  …, -3.0322e+01,
   -1.0513e+01,  5.1092e+00],
  …,
  [ 1.2065e+01, -8.1310e+00, -4.7448e+01,  …,  1.6440e+00,
    5.1183e+00,  5.5077e+00],
```

```
 [ 1.2065e+01, -8.4390e+00, -4.7289e+01,  …,  3.2897e+00,
   3.7469e+00,  2.5096e+00],
 [ 8.4914e+00, -1.1117e+01, -4.8936e+01,  …,  5.1191e+00,
   5.9119e+00,  3.1966e+00]],

[[ 4.0093e+00,  3.7532e+00,  6.3546e+00,  …,  4.1553e+01,
   4.8384e+01,  5.4661e+01],
 [ 9.1752e+00,  9.6878e+00, -1.6004e+01,  …, -4.1793e+01,
  -4.8493e+01, -5.3302e+01],
 [ 9.1752e+00,  6.6704e+00,  1.0467e+00,  …,  7.5719e+00,
   9.4933e+00,  1.7965e+01],
 …,
 [ 9.1752e+00,  1.1053e+01, -1.1275e-01,  …,  3.9710e+00,
   2.5145e+00,  2.1332e+00],
 [ 9.1752e+00,  1.1091e+01,  5.7260e-01,  …,  4.7133e+00,
   4.6302e+00,  3.1595e+00],
 [ 1.2611e+01,  1.4479e+01,  3.8360e+00,  …,  5.9669e+00,
   5.9979e+00,  4.3029e+00]],

[[ 8.0488e+00,  2.3671e+00,  7.4421e-01,  …,  3.5506e+01,
   4.5404e+01,  4.5069e+01],
 [ 7.5875e-01, -3.6170e+00,  6.7815e+00,  …, -1.4716e+01,
  -1.9337e+01, -1.6683e+01],
 [ 7.5875e-01, -5.6683e+00,  6.5399e+00,  …, -6.0036e+00,
  -9.0312e+00, -3.2786e+00],
 …,
 [ 7.5875e-01, -5.4634e+00,  5.4038e+00,  …, -5.0388e+00,
  -7.3321e+00, -2.1595e+00],
 [ 7.5875e-01, -5.9552e+00,  4.6198e+00,  …, -4.9525e+00,
  -5.4327e+00, -4.1904e+00],
 [-3.5797e+00, -8.8910e+00,  6.3459e+00,  …, -3.0556e-01,
   8.4523e-01,  3.2518e+00]],


…,


[[-1.4810e+01, -5.0797e+00,  2.9744e-01,  …, -4.0780e+00,
  -2.5034e+00,  5.6489e-01],
 [-1.9368e+01, -7.7907e+00, -1.0079e+01,  …,  7.5891e+00,
   1.1734e+01,  6.2142e+00],
 [-1.9368e+01, -9.1371e+00, -1.5608e+01,  …, -2.8184e+01,
  -2.8644e+01, -3.2463e+01],
 …,
 [-1.9368e+01, -5.5006e+00, -8.7075e+00,  …, -1.3599e+01,
  -1.2953e+01, -1.4393e+01],
 [-1.9368e+01, -5.2579e+00, -8.8187e+00,  …, -1.6589e+01,
  -1.5992e+01, -1.5558e+01],
 [-1.6086e+01, -5.1694e+00, -1.0830e+01,  …, -1.1459e+01,
  -1.1583e+01, -1.1803e+01]],
```

```
 [[ 2.7063e+00, -4.9008e+00,  1.7939e+01,  …,  9.1650e+00,
    7.3501e+00,  1.8702e+01],
  [ 4.9387e+00, -6.1576e+00,  1.0090e+01,  …, -2.3999e+01,
   -3.0809e+01, -2.2050e+01],
  [ 4.9387e+00, -8.5868e+00,  1.2228e+01,  …,  3.9709e+00,
    4.2101e+00,  1.5413e+01],
  …,
  [ 4.9387e+00, -1.0098e+01,  1.5002e+01,  …, -6.1059e-01,
   -1.7844e+00,  1.0207e+00],
  [ 4.9387e+00, -1.0522e+01,  1.5185e+01,  …,  2.4848e-01,
    1.0591e+00,  3.5897e+00],
  [ 5.8782e-01, -1.4046e+01,  1.1680e+01,  …,  1.6924e+00,
    2.7230e+00,  3.8663e+00]],

 [[-8.0870e+00, -1.2511e+01, -1.3010e+01,  …, -4.8066e+01,
   -5.3985e+01, -5.7193e+01],
  [-7.8535e+00, -1.3063e+01, -1.8370e+01,  …,  1.0642e+01,
    1.7672e+01,  6.0903e+00],
  [-7.8535e+00, -1.3761e+01, -1.8791e+01,  …, -6.5629e+00,
   -4.0963e+00, -1.8031e+01],
  …,
  [-7.8535e+00, -1.9474e+01, -1.7608e+01,  …, -9.9772e+00,
   -8.0511e+00, -1.1736e+01],
  [-7.8535e+00, -1.9585e+01, -1.7509e+01,  …, -1.2691e+01,
   -1.1679e+01, -1.1657e+01],
  [-1.3123e+01, -2.3740e+01, -1.9004e+01,  …, -1.1134e+01,
   -1.2109e+01, -1.3575e+01]]],


[[[ 7.3984e+00,  1.9167e+00,  1.9984e+00,  …,  6.9435e+00,
    5.5297e+00, -1.0940e+01],
  [ 1.0299e+00,  7.2632e-01,  9.4511e-01,  …,  4.5067e+01,
    3.4722e+01,  9.5597e+00],
  [-2.3260e+00,  3.3881e+00,  3.6279e+00,  …,  2.2756e+01,
    4.5223e+01,  3.5888e+01],
  …,
  [ 3.6966e+00, -5.8094e+00, -1.1996e+01,  …,  7.7071e+00,
    8.2692e+00,  1.9888e+01],
  [ 9.8089e+00,  5.1904e+00, -5.7970e+00,  …,  7.5980e+00,
    8.4712e+00,  1.7400e+01],
  [ 3.9874e+00,  7.2816e+00, -3.4557e+00,  …,  4.5881e+00,
    5.7754e+00,  1.1867e+01]],

 [[ 4.3693e+01,  4.0087e+01,  4.0355e+01,  …,  3.2532e+01,
    2.0599e+01,  1.1320e+01],
  [-3.4119e+01, -3.2311e+01, -3.2656e+01,  …, -5.0186e+00,
    1.1147e+01,  1.6870e+01],
```

```
 [ 6.1092e+00,  5.9656e+00,  6.1086e+00,  …,  1.6047e+00,
   1.1372e+01,  1.7640e+01],
 …,
 [-1.5933e+01, -2.3528e+01, -1.2394e+01,  …,  4.0930e+00,
   5.1811e+00,  9.8775e+00],
 [-9.1058e+00, -2.1383e+01, -1.7946e+01,  …,  5.0012e+00,
   6.1170e+00,  9.3937e+00],
 [-5.3034e+00, -2.0274e+01, -2.0702e+01,  …,  5.5012e+00,
   6.6627e+00,  1.0036e+01]],

[[ 4.4605e+01,  3.9021e+01,  3.9355e+01,  …,  3.2793e+01,
   1.5673e+01,  1.0452e+01],
 [-8.3758e+00, -6.8887e+00, -7.4102e+00,  …,  2.9410e+00,
   2.5348e+01,  2.6016e+00],
 [-6.6771e+00, -5.8545e+00, -5.8016e+00,  …, -1.3833e+01,
  -5.3859e+00,  2.1783e+01],
 …,
 [-1.0933e+01,  3.9532e+00,  4.2995e-01,  …,  4.1596e-02,
   1.2448e+00, -6.2744e+00],
 [-1.3656e+01, -1.2193e+00,  2.4193e+00,  …,  2.6658e-01,
   1.7925e-01, -6.2393e+00],
 [-1.8623e+01, -9.1626e+00,  9.9968e-01,  …, -4.2938e+00,
  -4.6910e+00, -1.0754e+01]],


…,


[[ 1.2688e+01,  7.9959e+00,  7.6441e+00,  …,  5.0151e+00,
   2.2987e+00, -7.4081e+00],
 [ 1.0104e+01,  8.2903e+00,  8.6585e+00,  …, -1.4595e+00,
  -7.6047e+00,  6.0276e+00],
 [-2.8410e+01, -3.1094e+01, -3.0977e+01,  …, -1.3669e+01,
  -7.3390e+00, -7.5032e+00],
 …,
 [-2.3254e+01, -2.6763e+01, -2.2805e+01,  …, -2.2150e+01,
  -2.3504e+01, -1.3333e+01],
 [-1.5767e+01, -2.0965e+01, -2.3487e+01,  …, -2.1635e+01,
  -2.2263e+01, -1.2786e+01],
 [-3.0407e+00, -1.2310e+01, -1.9896e+01,  …, -2.1414e+01,
  -2.2228e+01, -1.1082e+01]],

[[ 5.9799e+00,  6.6177e+00,  6.3868e+00,  …,  5.7973e+00,
   1.6513e+01, -7.8578e+00],
 [-1.7335e+01, -1.4906e+01, -1.5241e+01,  …, -1.3048e+01,
  -2.3237e+00, -5.1464e+00],
 [ 7.7008e+00,  1.0180e+01,  1.0464e+01,  …, -1.0794e+00,
  -6.8453e+00, -2.0235e+01],
 …,
 [ 2.1316e+01,  2.7259e+01,  2.1562e+01,  …,  8.7503e+00,
```

```
          1.0560e+01, -1.5720e+01],
        [ 2.1133e+01,  1.8481e+01,  2.7258e+01,  …,  8.4273e+00,
          9.3403e+00, -1.5475e+01],
        [ 1.1975e+01,  5.4535e+00,  2.1681e+01,  …,  4.3099e+00,
          4.6465e+00, -1.7332e+01]],

       [[-3.9053e+01, -4.3121e+01, -4.3678e+01,  …, -3.2674e+01,
         -1.7873e+01, -2.2388e+01],
        [ 5.0219e+00,  4.4095e+00,  5.0390e+00,  …, -9.3153e+00,
         -3.2281e+01, -1.2226e+01],
        [-1.2199e+01, -8.7430e+00, -8.7397e+00,  …,  2.7149e+00,
         -5.2448e+00, -3.1864e+01],
        …,
        [-6.0349e+00, -2.1224e+01, -1.3011e+01,  …, -1.0174e+01,
         -1.0998e+01, -8.0515e+00],
        [-4.7189e+00, -1.6513e+01, -1.6908e+01,  …, -1.0634e+01,
         -1.0362e+01, -9.2599e+00],
        [-6.1957e+00, -1.0560e+01, -2.0869e+01,  …, -1.4142e+01,
         -1.3965e+01, -1.2320e+01]]]], device='cuda:0',
    grad_fn=<CudnnConvolutionBackward>)
```

[9]:
```
difference = abs( output_ref - output_recovered )
print(difference.mean())  ## It should be small, e.g.,2.3 in my trainned model
```

```
tensor(1.4554, device='cuda:0', grad_fn=<MeanBackward0>)
```

[ ]:

[ ]:

[ ]:

[ ]: