HW6-1

November 17, 2023

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

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

import torchvision
import torchvision.transforms as transforms

from models import *

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


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

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


train_dataset = torchvision.datasets.CIFAR10(
    root='./data',
    train=True,
    download=True,
    transform=transforms.Compose([
        transforms.RandomCrop(32, padding=4),
        transforms.RandomHorizontalFlip(),
```

```
        transforms.ToTensor(),
        normalize,
    ]))
trainloader = torch.utils.data.DataLoader(train_dataset, batch_size=batch_size,␣
 ↪shuffle=True, num_workers=2)


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

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


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

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

    model.train()

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

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

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

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

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

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


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



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

    # switch to evaluate mode
    model.eval()

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

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

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

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

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

```python
            end = time.time()

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

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


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

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

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


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

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

    def update(self, val, n=1):
        self.val = val
        self.sum += val * n
        self.count += n
        self.avg = self.sum / self.count
```

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


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

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

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

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

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

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

lr = 4e-2
weight_decay = 1e-4
epochs = 60
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)+"_hw6"
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)
```

[2]:
```
PATH = "result/VGG16_quant_hw6/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: 8957/10000 (90%)
```

[ ]:

[ ]:

[ ]:

[ ]:

[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 inputs from selected layer ##########
save_output = SaveOutput()
i = 0

for layer in model.modules():
    i = i+1
    if isinstance(layer, QuantConv2d):
        print(i,"-th layer prehooked")
        layer.register_forward_pre_hook(save_output)
####################################################

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

```
3 -th layer prehooked
7 -th layer prehooked
12 -th layer prehooked
16 -th layer prehooked
21 -th layer prehooked
25 -th layer prehooked
29 -th layer prehooked
34 -th layer prehooked
38 -th layer prehooked
42 -th layer prehooked
47 -th layer prehooked
51 -th layer prehooked
55 -th layer prehooked
```

[4]:

```python
weight_q = model.features[3].weight_q
w_alpha = model.features[3].weight_quant.wgt_alpha
w_bit = 4

weight_int = weight_q / (w_alpha / (2**(w_bit-1)-1))
print(weight_int)
```

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

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

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

         …,

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

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

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

```
[[[-0.0000, -3.0000,  0.0000],
  [ 5.0000,  6.0000,  7.0000],
  [ 1.0000, -7.0000, -7.0000]],

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

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

 …,

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

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

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


[[[ 4.0000,  7.0000,  7.0000],
  [-3.0000, -6.0000,  0.0000],
  [-4.0000, -2.0000, -0.0000]],

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

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

 …,

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

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

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


…,


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

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

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

 …,

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

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

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


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

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

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

```
        …,

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

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

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


       [[[ 6.0000,  0.0000, -5.0000],
         [ 7.0000, -0.0000, -7.0000],
         [ 6.0000,  1.0000, -7.0000]],

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

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

        …,

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

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

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

[5]:
```
act = save_output.outputs[1][0]
act_alpha  = model.features[3].act_alpha
act_bit = 4
act_quant_fn = act_quantization(act_bit)
```

```python
act_q = act_quant_fn(act, act_alpha)

act_int = act_q / (act_alpha / (2**act_bit-1))
print(act_int)
```

```
tensor([[[[ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 4.0000,  5.0000,  6.0000,  …,  7.0000,  4.0000,  3.0000],
          [ 3.0000,  4.0000,  6.0000,  …,  6.0000,  5.0000,  3.0000],
          …,
          [10.0000,  4.0000,  4.0000,  …,  8.0000,  0.0000,  2.0000],
          [10.0000,  3.0000,  2.0000,  …,  3.0000,  3.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]],

         [[ 1.0000,  1.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 1.0000,  2.0000,  2.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 1.0000,  2.0000,  2.0000,  …,  0.0000,  0.0000,  0.0000],
          …,
          [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  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,  …,  3.0000,  3.0000,  9.0000],
          [ 0.0000,  0.0000,  0.0000,  …,  1.0000,  2.0000, 12.0000],
          [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  2.0000, 13.0000],
          …,
          [15.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
          [15.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
          [15.0000,  0.0000,  2.0000,  …,  3.0000,  1.0000,  0.0000]],


         …,

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

         [[ 0.0000,  0.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
          [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
          …,
          [ 2.0000,  3.0000,  2.0000,  …,  3.0000,  3.0000,  0.0000],
          [ 1.0000,  2.0000,  3.0000,  …,  2.0000,  2.0000,  1.0000],
          [ 6.0000,  8.0000, 10.0000,  …, 10.0000, 11.0000,  8.0000]],
```

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


[[[15.0000, 14.0000, 15.0000,  …, 15.0000, 15.0000, 15.0000],
  [15.0000, 12.0000, 12.0000,  …, 12.0000, 12.0000, 15.0000],
  [15.0000, 12.0000, 12.0000,  …, 12.0000, 12.0000, 15.0000],
  …,
  [ 8.0000,  3.0000,  1.0000,  …,  8.0000, 10.0000, 15.0000],
  [ 8.0000,  2.0000,  3.0000,  …,  9.0000, 12.0000, 15.0000],
  [ 7.0000,  2.0000,  3.0000,  …,  6.0000,  8.0000, 15.0000]],

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

 …,

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

 [[ 1.0000,  9.0000,  9.0000,  …,  9.0000,  9.0000,  8.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [ 0.0000,  2.0000,  3.0000,  …,  0.0000,  0.0000,  0.0000],
```

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

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


[[[15.0000, 15.0000, 15.0000,  …, 14.0000, 15.0000, 15.0000],
  [14.0000, 11.0000,  8.0000,  …, 11.0000, 12.0000, 15.0000],
  [13.0000, 11.0000, 10.0000,  …, 11.0000, 14.0000, 15.0000],
  …,
  [ 2.0000,  3.0000,  1.0000,  …,  5.0000,  1.0000,  0.0000],
  [ 0.0000,  3.0000,  2.0000,  …,  5.0000,  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,  1.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  3.0000,  3.0000,  3.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  3.0000,  3.0000,  3.0000],
  …,
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]],

 [[ 6.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [11.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [10.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [ 5.0000,  1.0000,  0.0000,  …,  0.0000,  1.0000,  0.0000],
  [ 5.0000,  0.0000,  0.0000,  …,  0.0000,  1.0000,  0.0000],
  [ 5.0000,  1.0000,  0.0000,  …,  1.0000,  1.0000,  0.0000]],


 …,

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

 [[ 0.0000,  4.0000,  3.0000,  …, 10.0000, 10.0000,  9.0000],
  [ 0.0000,  0.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
```

```
  [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [ 3.0000,  0.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 4.0000,  1.0000,  2.0000,  …,  0.0000,  0.0000,  1.0000],
  [ 9.0000, 10.0000, 10.0000,  …, 11.0000, 12.0000, 11.0000]],

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


…,


[[[ 3.0000,  1.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [11.0000,  7.0000, 11.0000,  …, 12.0000,  8.0000,  0.0000],
  [10.0000,  8.0000, 10.0000,  …, 15.0000, 10.0000,  2.0000],
  …,
  [11.0000, 12.0000, 15.0000,  …, 14.0000,  3.0000,  2.0000],
  [ 9.0000, 10.0000, 15.0000,  …,  4.0000,  6.0000,  4.0000],
  [13.0000, 14.0000, 15.0000,  …,  0.0000,  4.0000,  2.0000]],

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

 [[ 0.0000,  3.0000,  1.0000,  …, 11.0000, 15.0000,  7.0000],
  [ 0.0000,  3.0000,  1.0000,  …, 13.0000, 15.0000, 10.0000],
  [ 0.0000,  3.0000,  2.0000,  …, 13.0000, 15.0000,  8.0000],
  …,
  [ 0.0000,  0.0000,  2.0000,  …,  7.0000,  4.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  3.0000,  1.0000,  0.0000],
  [ 0.0000,  0.0000,  0.0000,  …,  3.0000,  1.0000,  0.0000]],

 …,

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

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

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

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


[[[15.0000, 15.0000, 15.0000,  …, 15.0000, 15.0000, 15.0000],
  [13.0000,  9.0000, 10.0000,  …,  8.0000,  9.0000, 13.0000],
  [12.0000, 10.0000, 10.0000,  …,  9.0000,  9.0000, 13.0000],
  …,
  [ 9.0000,  7.0000,  7.0000,  …,  9.0000,  4.0000,  2.0000],
  [ 9.0000,  6.0000,  8.0000,  …,  7.0000,  4.0000,  4.0000],
  [ 6.0000,  3.0000,  3.0000,  …,  2.0000,  3.0000,  0.0000]],

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

 [[ 6.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [10.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [10.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  …,
  [ 7.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 7.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 5.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000]],

 …,

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

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

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

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


[[[ 1.0000,  3.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
  [ 3.0000,  5.0000,  2.0000,  …,  2.0000,  0.0000,  0.0000],
  [ 5.0000,  1.0000,  3.0000,  …,  2.0000,  1.0000,  0.0000],
  …,
  [11.0000,  8.0000,  8.0000,  …,  2.0000,  1.0000,  0.0000],
  [10.0000,  4.0000,  9.0000,  …,  0.0000,  0.0000,  0.0000],
  [13.0000, 11.0000,  9.0000,  …,  0.0000,  0.0000,  0.0000]],

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

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

```
        …,

        [[ 6.0000,  7.0000,  7.0000,  …, 14.0000, 15.0000,  8.0000],
         [ 1.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
         [ 2.0000,  1.0000,  1.0000,  …,  0.0000,  0.0000,  0.0000],
         …,
         [ 2.0000,  4.0000,  3.0000,  …,  1.0000,  0.0000,  0.0000],
         [ 1.0000,  2.0000,  2.0000,  …,  1.0000,  0.0000,  0.0000],
         [ 1.0000,  3.0000,  4.0000,  …,  0.0000,  0.0000,  0.0000]],

        [[ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  0.0000],
         [ 1.0000,  0.0000,  0.0000,  …,  2.0000,  2.0000,  1.0000],
         [ 1.0000,  0.0000,  0.0000,  …,  2.0000,  1.0000,  1.0000],
         …,
         [ 0.0000,  0.0000,  0.0000,  …,  0.0000,  0.0000,  1.0000],
         [ 0.0000,  0.0000,  0.0000,  …,  1.0000,  1.0000,  1.0000],
         [ 0.0000,  0.0000,  0.0000,  …, 13.0000, 14.0000, 11.0000]],

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

[6]:
```
conv_int = torch.nn.Conv2d(in_channels = 64, out_channels=64, kernel_size = 3,␣
 ↪padding=1)
conv_int.weight = torch.nn.parameter.Parameter(weight_int)
conv_int.bias = model.features[3].bias
output_int = conv_int(act_int)
output_recovered = output_int * (act_alpha / (2**act_bit-1)) * (w_alpha /␣
 ↪(2**(w_bit-1)-1))
print(output_recovered)
```

```
tensor([[[[-2.1890e+00,  2.0640e+01,  9.5693e+00,  …,  1.1258e+01,
             2.0389e+01,  5.2537e+00],
           [-9.9445e+00,  3.1710e+01,  2.3704e+01,  …,  3.0834e+01,
             4.5845e+01,  2.3517e+01],
           [-1.4385e+01,  2.4830e+01,  2.2328e+01,  …,  2.9146e+01,
             4.6158e+01,  2.2141e+01],
           …,
           [ 2.9271e+01, -7.4428e+00,  1.3697e+01,  …,  2.3141e+00,
            -2.2453e+01, -9.4442e+00],
           [ 2.2391e+01, -1.4010e+01,  8.6936e+00,  …, -3.8777e+00,
            -3.0772e+01, -1.5011e+01],
           [ 2.6394e+01,  8.0682e+00,  2.3204e+01,  …,  1.7450e+01,
```

```
  -6.1919e+00,  1.2509e-01]],

[[-2.2766e+01, -3.6338e+01, -4.0466e+01,  …, -5.1036e+01,
  -5.3163e+01, -3.9528e+01],
 [-3.1898e+01, -4.1404e+01, -3.9716e+01,  …, -2.6519e+01,
  -1.7888e+01, -7.9431e+00],
 [-2.4455e+01, -3.1835e+01, -3.9841e+01,  …, -2.8395e+01,
  -3.2961e+01, -2.2516e+01],
 …,
 [-4.0654e+00, -1.8576e+01, -3.0209e+01,  …,  6.8798e-01,
   3.2523e+00, -7.1926e+00],
 [-1.0758e+01, -2.8895e+01, -2.3454e+01,  …, -3.4024e+01,
  -3.3711e+01, -2.8520e+01],
 [-2.4392e+01, -4.8534e+01, -5.3725e+01,  …, -6.1981e+01,
  -4.2593e+01, -2.2453e+01]],

[[-2.5706e+01, -3.0647e+01, -3.1084e+01,  …, -3.1272e+01,
  -3.1772e+01, -2.6394e+01],
 [-4.3406e+01, -5.7541e+01, -6.8861e+01,  …, -4.9222e+01,
  -4.4406e+01, -4.6408e+01],
 [-2.2891e+01, -2.6894e+01, -3.5650e+01,  …, -3.1835e+01,
  -3.0709e+01, -3.6463e+01],
 …,
 [-3.0834e+01, -2.8207e+01, -2.1015e+01,  …, -8.8813e+00,
  -1.5011e+01, -3.7464e+01],
 [-2.3204e+01, -1.0633e+01, -6.7548e+00,  …, -1.5386e+01,
   1.3760e+00, -1.9514e+01],
 [-2.9959e+01, -4.2030e+01, -4.5595e+01,  …, -4.3218e+01,
  -3.1397e+01, -3.2711e+01]],


…,


[[-2.3454e+01, -4.5282e+01, -2.2266e+01,  …, -2.3454e+01,
  -2.5768e+01, -3.0021e+00],
 [-2.5768e+01, -6.2169e+01, -2.4830e+01,  …, -2.2391e+01,
  -2.5893e+01,  8.5060e+00],
 [-2.6206e+01, -6.4858e+01, -3.3524e+01,  …, -3.2398e+01,
  -4.1342e+01,  1.8138e+00],
 …,
 [-1.5136e+01,  6.1231e+01, -1.3072e+01,  …,  5.1286e+00,
   1.5699e+01, -7.5053e-01],
 [-1.9701e+01,  5.6915e+01, -2.3016e+01,  …, -8.3809e+00,
   2.3829e+01, -1.6824e+01],
 [-6.3170e+00,  4.7471e+01, -7.2551e+00,  …,  5.2537e+00,
   2.3517e+01, -5.0661e+00]],

[[ 1.0633e+00,  3.8152e+00, -2.5017e-01,  …, -8.9438e+00,
  -1.2321e+01, -1.9514e+01],
```

```
 [ 1.8576e+01,  3.4775e+01,  3.8089e+01,  …,  5.8917e+01,
   5.9292e+01,  2.8708e+01],
 [-8.1307e-01,  8.0682e+00,  7.5053e+00,  …,  1.3384e+01,
   2.2829e+01,  1.1070e+01],
 …,
 [ 1.2259e+01, -1.0633e+01, -2.4392e+01,  …,  8.9438e+00,
  -1.6887e+01, -2.2453e+01],
 [-6.5046e+00, -2.5268e+01, -3.2898e+01,  …, -3.1585e+01,
  -3.4462e+01, -3.1460e+01],
 [ 2.3016e+01,  1.4448e+01,  1.7700e+01,  …,  4.5032e+00,
   9.6318e+00, -7.6304e+00]],

[[-2.5643e+00, -4.8159e+00, -1.3885e+01,  …, -4.6283e+00,
   1.9013e+01, -1.2509e+00],
 [ 9.5693e+00,  9.2565e+00, -8.1307e+00,  …,  8.7562e+00,
   4.7784e+01,  8.0057e+00],
 [ 5.1912e+00,  9.8820e+00, -1.2446e+01,  …,  5.1286e+00,
   4.4031e+01,  1.2884e+01],
 …,
 [-1.6637e+01,  2.5330e+01,  1.3635e+01,  …,  4.0654e+01,
   5.2099e+01, -2.2578e+01],
 [-2.0765e+01,  1.5198e+01,  2.4517e+01,  …,  2.5080e+01,
   3.9465e+01,  7.8180e+00],
 [-1.4698e+01, -2.2516e+00,  1.2259e+01,  …,  4.1279e+00,
   1.1258e+01,  1.4886e+01]]],


[[[ 9.1314e+00,  2.9396e+00,  4.7534e+00,  …,  5.8791e+00,
    3.4399e+00, -5.0035e+00],
  [ 3.4399e+00,  2.6269e+00,  1.8138e+00,  …,  3.0647e+00,
    4.1905e+00, -1.3760e+00],
  [ 3.3148e+00,  4.1279e+00,  8.1307e-01,  …,  9.3817e-01,
    3.0647e+00, -2.9396e+00],
  …,
  [ 1.0132e+01,  1.7763e+01,  3.0522e+01,  …,  2.3767e+00,
   -6.2545e-02, -5.0035e+00],
  [ 5.6915e+00,  1.5011e+01,  2.6519e+01,  …, -1.1258e+00,
   -3.0647e+00, -5.9417e+00],
  [ 7.0049e+00,  1.6449e+01,  2.3579e+01,  …,  9.0689e+00,
    7.3802e+00,  4.6908e+00]],

 [[ 3.0146e+01,  7.2426e+01,  6.7298e+01,  …,  6.7298e+01,
    6.1418e+01,  4.2092e+01],
  [-5.8479e+01, -8.3622e+01, -9.7506e+01,  …, -9.8820e+01,
   -1.0426e+02, -7.6429e+01],
  [-3.5025e+01, -3.4337e+01, -3.7464e+01,  …, -3.5525e+01,
   -4.3468e+01, -4.0341e+01],
  …,
```

```
 [-2.9646e+01, -4.8409e+01, -4.1217e+01,  …, -1.2696e+01,
  -2.5893e+01, -3.0834e+01],
 [-1.3259e+01, -2.5330e+01, -2.7269e+01,  …, -2.4017e+01,
  -3.3899e+01, -3.1647e+01],
 [-1.1508e+01, -2.4955e+01, -1.9952e+01,  …, -2.5018e+00,
  -2.7519e+00, -1.2509e+01]],

[[-2.5080e+01, -3.1272e+01, -2.8145e+01,  …, -3.0021e+01,
  -2.9521e+01, -3.9841e+01],
 [-6.1418e+01, -7.0800e+01, -5.7791e+01,  …, -5.7478e+01,
  -5.0973e+01, -4.9723e+01],
 [-4.2718e+01, -4.6658e+01, -3.2273e+01,  …, -3.1335e+01,
  -3.0959e+01, -4.4907e+01],
 …,
 [-1.9514e+01, -1.7387e+01, -7.5678e+00,  …, -1.9764e+01,
  -2.6394e+01, -4.2030e+01],
 [-1.5761e+01, -1.5198e+01, -1.2759e+01,  …, -3.1272e+01,
  -3.2023e+01, -4.3155e+01],
 [-1.8576e+01, -2.3016e+01, -2.1140e+01,  …, -4.4969e+01,
  -4.2092e+01, -4.5094e+01]],


…,


[[-5.3163e+00, -3.4399e+01, -1.4510e+01,  …, -1.4635e+01,
  -2.2766e+01,  5.6290e-01],
 [-5.8166e+00, -4.7096e+01, -2.0640e+01,  …, -2.0952e+01,
  -3.3086e+01, -5.5039e+00],
 [ 5.0035e+00, -3.9090e+01, -8.8187e+00,  …, -9.5693e+00,
  -1.9639e+01, -2.1265e+00],
 …,
 [-1.9576e+01, -1.4198e+01, -1.2321e+01,  …,  1.7512e+00,
  -4.3781e+00, -4.7534e+00],
 [-1.7387e+01, -1.5198e+01, -2.1765e+01,  …, -5.0035e+00,
  -1.0758e+01, -5.8166e+00],
 [-1.0195e+01, -1.0633e+01, -1.5261e+01,  …,  1.1571e+01,
   7.8806e+00, -1.6261e+00]],

[[ 2.3329e+01,  3.0084e+01,  3.2210e+01,  …,  3.3961e+01,
   3.2773e+01,  3.2210e+01],
 [-2.2766e+01, -3.7276e+01, -3.8715e+01,  …, -3.8777e+01,
  -3.3711e+01, -8.8813e+00],
 [ 7.1926e+00,  7.5053e+00,  1.2696e+01,  …,  1.1821e+01,
   6.1293e+00,  1.0007e+01],
 …,
 [ 2.3141e+00,  4.4406e+00,  1.4323e+01,  …,  7.2551e+00,
  -2.6894e+00,  3.8152e+00],
 [ 1.6887e+00,  7.7555e+00,  1.1070e+01,  …,  1.5636e+00,
   1.5011e+00,  7.5053e-01],
```

```
 [ 6.5671e+00,  1.0883e+01,  1.5011e+01,  …, -1.0633e+00,
  -8.1307e+00,  6.0042e+00]],

[[-3.4337e+01,  5.8166e+00, -3.4399e+00,  …, -1.1258e+00,
  -3.1272e+01,  7.0675e+00],
 [-5.4101e+01,  9.8194e+00, -4.3781e+00,  …, -2.4392e+00,
  -5.5914e+01,  3.9403e+00],
 [-5.1599e+01,  1.8701e+01,  7.8806e+00,  …,  7.7555e+00,
  -4.8284e+01,  6.2544e+00],
 …,
 [-2.5143e+01,  4.9410e+01,  3.8402e+01,  …, -1.0945e+01,
  -7.3677e+01,  5.3163e+00],
 [-1.8263e+01,  4.3468e+01,  1.9264e+01,  …, -8.8813e+00,
  -6.7110e+01,  6.6922e+00],
 [-1.0195e+01,  1.9451e+01,  2.5018e-01,  …, -6.7548e+00,
  -4.3281e+01,  5.6290e+00]]],


[[[ 8.5060e+00, -9.7569e+00, -2.3141e+00,  …,  6.5671e+00,
    2.0014e+00, -6.2544e+00],
  [ 9.6318e+00, -8.6311e+00, -3.8152e+00,  …,  3.3148e+00,
    3.5650e+00, -1.2509e+00],
  [ 1.2196e+01, -3.6276e+00, -1.7512e+00,  …, -2.6894e+00,
    1.7512e+00, -2.4392e+00],
  …,
  [ 2.6957e+01,  1.7575e+01,  3.1460e+01,  …,  3.0271e+01,
    1.8138e+01,  1.0758e+01],
  [ 2.0264e+01,  8.8187e+00,  2.0577e+01,  …,  3.0584e+01,
    1.5136e+01,  7.8180e+00],
  [ 1.9576e+01,  1.3760e+01,  2.3329e+01,  …,  3.2023e+01,
    1.5573e+01,  1.0945e+01]],

 [[ 3.1835e+01,  7.0550e+01,  5.8979e+01,  …,  6.3295e+01,
    5.9042e+01,  4.4782e+01],
  [-3.5463e+01, -4.9785e+01, -4.6846e+01,  …, -9.3691e+01,
   -9.7756e+01, -7.1363e+01],
  [-3.5087e+01, -4.2467e+01, -4.3468e+01,  …, -3.8402e+01,
   -5.2224e+01, -4.1654e+01],
  …,
  [-1.7387e+01, -3.8027e+01, -2.8583e+01,  …, -3.4900e+01,
   -4.4594e+01, -3.2023e+01],
  [-8.4435e+00, -3.3961e+01, -2.8895e+01,  …, -4.7033e+01,
   -4.6595e+01, -3.6276e+01],
  [-1.3697e+01, -2.5768e+01, -2.5393e+01,  …, -5.2600e+01,
   -3.1772e+01,  1.8763e-01]],

 [[-1.3447e+01, -2.1515e+01, -2.1515e+01,  …, -2.5956e+01,
   -2.5205e+01, -3.7714e+01],
```

```
 [-3.9465e+01, -3.8214e+01, -1.5198e+01,  …, -5.7353e+01,
  -5.2099e+01, -5.2537e+01],
 [-4.1342e+01, -4.7784e+01, -3.6839e+01,  …, -3.9716e+01,
  -3.6964e+01, -5.1036e+01],
 …,
 [-2.9646e+01, -1.5198e+01, -1.9264e+01,  …, -3.3148e+01,
  -1.5261e+01, -4.3656e+01],
 [-1.9389e+01, -3.6276e+00, -4.8159e+00,  …, -2.3642e+01,
  -2.3141e+00, -3.5275e+01],
 [-1.6512e+01, -1.5886e+01, -1.9389e+01,  …, -1.1195e+01,
  -1.3134e+00, -2.6706e+01]],

…,

[[-3.5025e+00, -1.4260e+01, -1.4635e+01,  …, -1.4823e+01,
  -1.9139e+01, -5.3788e+00],
 [-2.0014e+00, -9.2565e+00, -7.6304e+00,  …, -2.1453e+01,
  -2.6581e+01, -1.3447e+01],
 [ 3.9403e+00, -5.8166e+00, -4.5657e+00,  …, -7.6929e+00,
  -1.3510e+01, -1.3510e+01],
 …,
 [-4.3218e+01,  4.6345e+01, -9.8820e+00,  …, -2.2516e+01,
  -1.2071e+01,  4.4406e+00],
 [-4.5157e+01,  3.6025e+01, -1.6387e+01,  …, -3.3211e+01,
  -1.2446e+01, -6.1293e+00],
 [-2.7457e+01,  2.6206e+01, -6.3795e+00,  …, -2.1140e+01,
  -6.5046e+00,  1.6261e+00]],

[[ 1.3447e+01,  1.5949e+01,  5.5664e+00,  …,  3.6213e+01,
   2.8395e+01,  2.8958e+01],
 [-2.3454e+01, -3.2460e+01, -3.3461e+01,  …, -3.6776e+01,
  -3.0334e+01, -7.3177e+00],
 [-3.1272e-01, -2.0640e+00,  1.6449e+01,  …,  4.1279e+00,
   1.6261e+00,  1.0570e+01],
 …,
 [ 7.2551e+00,  7.6929e+00,  6.5671e+00,  …,  7.5678e+00,
   2.7519e+00, -1.0320e+01],
 [-8.5060e+00, -1.1383e+01, -9.6943e+00,  …, -1.2384e+01,
  -1.1571e+01, -1.2696e+01],
 [ 2.1077e+01,  1.5511e+01,  1.4948e+01,  …,  9.2565e+00,
   5.5664e+00, -5.7541e+00]],

[[-4.4782e+01, -2.5643e+00,  3.8777e+01,  …, -1.9087e-06,
  -4.5845e+01,  7.0049e+00],
 [-7.1050e+01, -9.1314e+00,  6.7735e+01,  …,  3.7526e-01,
  -8.0870e+01,  7.6304e+00],
 [-6.5359e+01, -7.1926e+00,  7.7367e+01,  …,  1.1195e+01,
  -8.1495e+01,  1.4760e+01],
```

```
  …,
  [ 1.6074e+01, -7.3802e+00,  1.3322e+01,  …,  8.8187e+00,
    7.7555e+01,  1.1070e+01],
  [ 1.5261e+01, -1.5261e+01,  6.5671e+00,  …,  2.2078e+01,
    6.5734e+01,  2.9396e+00],
  [ 8.3184e+00, -2.1015e+01, -1.0695e+01,  …,  9.1940e+00,
    2.4580e+01, -8.9438e+00]]],


…,


[[[ 8.6936e+00,  1.8388e+01,  6.6922e+00,  …,  3.6088e+01,
    1.4323e+01, -9.8820e+00],
  [ 4.2530e+00,  3.1960e+01,  1.2571e+01,  …,  6.9487e+01,
    4.2467e+01,  3.2523e+00],
  [ 1.5636e+00,  3.5150e+01,  1.0007e+01,  …,  6.3044e+01,
    3.4837e+01,  1.9389e+00],
  …,
  [-1.7700e+01,  3.4337e+01,  3.2898e+01,  …,  2.7645e+01,
   -5.6290e+00, -4.0028e+00],
  [-1.6762e+01,  3.5588e+01,  4.1154e+01,  …,  3.6151e+01,
    5.9417e+00, -4.6283e+00],
  [-1.6887e+01,  1.9201e+01,  2.8082e+01,  …,  2.9396e+01,
    1.5886e+01,  7.4428e+00]],

 [[ 4.8159e+00,  1.1946e+01,  3.8152e+00,  …, -4.6908e+01,
   -5.0348e+01, -3.5650e+01],
  [-6.0230e+01, -8.4310e+01, -8.2183e+01,  …, -6.6172e+01,
   -1.9889e+01, -1.6262e+00],
  [-3.1898e+01, -3.4524e+01, -4.1905e+01,  …, -3.9903e+01,
   -3.3273e+01, -2.2954e+01],
  …,
  [-2.7332e+01, -2.7394e+01, -3.2335e+01,  …, -2.4455e+01,
   -3.2210e+01, -2.6706e+01],
  [-4.3343e+01, -4.8972e+01, -5.0035e+01,  …, -1.0482e+02,
   -6.0918e+01, -2.8082e+01],
  [ 6.8799e-01,  1.4135e+01,  1.5198e+01,  …,  1.2008e+01,
   -1.7950e+01, -1.6949e+01]],

 [[-2.6331e+01, -3.6401e+01, -3.6401e+01,  …, -6.4295e+01,
   -4.2405e+01, -2.9271e+01],
  [-7.4177e+01, -9.2128e+01, -8.8813e+01,  …, -1.0238e+02,
   -7.2864e+01, -4.9160e+01],
  [-3.0522e+01, -3.4962e+01, -2.6018e+01,  …, -7.0175e+01,
   -6.3107e+01, -4.9347e+01],
  …,
  [-3.8777e+01, -4.5032e+01, -3.9403e+01,  …, -7.3052e+01,
```

```
  -4.6658e+01, -3.2711e+01],
 [-3.4024e+01, -4.6158e+01, -4.5157e+01,  …, -3.6713e+01,
  -3.2648e+01, -3.0396e+01],
 [-1.4886e+01, -1.9764e+01, -1.8513e+01,  …, -1.6574e+01,
  -2.9083e+01, -2.6269e+01]],

…,

[[-1.5323e+01, -5.0035e+01, -5.7541e+00,  …, -3.8715e+01,
   1.3697e+01,  3.8027e+01],
 [-2.3704e+01, -7.3677e+01,  5.0036e-01,  …, -4.5344e+01,
   3.8152e+01,  7.5303e+01],
 [-2.5831e+01, -7.7742e+01,  7.6929e+00,  …, -4.0341e+01,
   4.0966e+01,  7.1676e+01],
 …,
 [-1.9889e+01, -8.7437e+01, -4.6220e+01,  …, -1.0820e+01,
   2.2704e+01,  1.9451e+01],
 [-1.8951e+01, -7.6617e+01, -4.3593e+01,  …, -2.5581e+01,
   1.1008e+01, -2.4392e+00],
 [-2.6894e+00, -3.2773e+01, -1.5323e+01,  …, -2.0640e+00,
   1.9764e+01,  5.0661e+00]],

[[ 2.8458e+01,  2.9146e+01,  1.7950e+01,  …,  2.5893e+01,
  -9.1940e+00, -2.3829e+01],
 [ 6.3170e+00,  1.0883e+01,  1.5011e+01,  …,  8.8813e+01,
   8.1995e+01,  3.1522e+01],
 [ 8.0056e+00,  1.3947e+01,  1.0945e+01,  …,  2.3329e+01,
   2.8895e+01,  1.0883e+01],
 …,
 [-2.3642e+01, -7.8806e+00,  2.8770e+00,  …,  3.1272e+01,
   3.3023e+01,  4.4406e+00],
 [-2.2453e+01, -8.0682e+00,  7.8806e+00,  …, -5.8229e+01,
  -1.1571e+01,  9.3816e-01],
 [-2.4079e+01, -2.1328e+01, -8.5685e+00,  …,  6.1293e+00,
  -1.1008e+01, -9.1940e+00]],

[[-1.7575e+01,  1.8138e+01, -6.5671e+00,  …,  1.9701e+01,
   4.6720e+01,  8.8187e+00],
 [-2.3329e+01,  3.0396e+01, -8.5060e+00,  …,  4.6783e+01,
   8.1558e+01,  2.2578e+01],
 [-2.1828e+01,  2.8395e+01, -7.5053e-01,  …,  5.5977e+01,
   7.9744e+01,  1.9201e+01],
 …,
 [-1.9889e+01, -4.8096e+01, -4.0278e+01,  …,  1.7512e+00,
   4.6971e+01,  4.9410e+00],
 [-1.1258e+01, -4.4156e+01, -4.5094e+01,  …, -5.2475e+01,
   5.6915e+00,  7.1926e+00],
 [-3.7526e+00, -2.9208e+01, -3.7089e+01,  …, -4.1029e+01,
```

```
         -3.2023e+01, -8.3809e+00]]],


       [[[ 1.1508e+01, -1.6887e+00,  7.0049e+00,  …,  1.3134e+00,
          -1.8138e+00, -5.2537e+00],
         [ 1.2634e+01, -2.3767e+00,  4.0654e+00,  …, -1.5011e+00,
          -5.3788e+00, -7.9431e+00],
         [ 1.6449e+01,  2.5018e-01,  7.5678e+00,  …, -6.2543e-02,
          -4.2530e+00, -6.3170e+00],
         …,
         [ 1.7262e+01, -5.6289e-01,  7.3177e+00,  …,  1.2634e+01,
           2.1890e+00,  1.8138e+00],
         [ 1.6074e+01, -2.7519e+00,  1.5011e+00,  …,  5.3163e+00,
          -2.3767e+00, -1.8138e+00],
         [ 1.5386e+01,  1.0320e+01,  1.1946e+01,  …,  1.0007e+01,
           4.5032e+00,  4.7534e+00]],

        [[ 3.5713e+01,  6.8298e+01,  6.3858e+01,  …,  6.4733e+01,
           5.5289e+01,  3.9153e+01],
         [-3.8840e+01, -5.5414e+01, -5.9417e+01,  …, -6.6422e+01,
          -6.5922e+01, -5.1224e+01],
         [-1.8826e+01, -1.9889e+01, -2.6331e+01,  …, -2.4142e+01,
          -3.0271e+01, -3.1335e+01],
         …,
         [-2.4767e+01, -3.8402e+01, -2.8833e+01,  …, -2.7332e+01,
          -4.3406e+01, -3.3336e+01],
         [-1.8826e+01, -2.0327e+01, -2.2141e+01,  …, -3.4087e+01,
          -2.0952e+01, -1.2947e+01],
         [-5.4413e+00, -1.9514e+01, -2.4705e+01,  …, -2.8270e+01,
          -3.2961e+01, -2.3454e+01]],

        [[-1.9326e+01, -2.5706e+01, -2.2016e+01,  …, -2.3829e+01,
          -1.9952e+01, -2.8958e+01],
         [-3.3524e+01, -3.4900e+01, -2.6018e+01,  …, -2.8082e+01,
          -2.4330e+01, -3.0647e+01],
         [-2.9646e+01, -3.2210e+01, -2.7144e+01,  …, -2.8395e+01,
          -2.4893e+01, -3.3836e+01],
         …,
         [-3.2648e+01, -3.6526e+01, -3.2711e+01,  …, -3.8027e+01,
          -3.6713e+01, -3.2210e+01],
         [-1.6762e+01, -2.0514e+01, -2.1265e+01,  …, -1.2759e+01,
          -2.0952e+01, -3.0709e+01],
         [-2.9834e+01, -3.8902e+01, -3.6025e+01,  …, -1.5636e+01,
          -2.3266e+01, -2.5643e+01]],

        …,

        [[ 1.8763e-01, -8.8187e+00, -7.8180e+00,  …, -6.8173e+00,
```

```
         -1.0445e+01,  2.4392e+00],
        [-4.3781e-01, -1.5636e+00, -7.1926e+00,  …, -4.8784e+00,
         -1.3447e+01, -1.3760e+00],
        [ 1.6887e+00,  1.0257e+01, -8.7562e-01,  …,  3.9403e+00,
         -6.8173e+00, -1.8138e+00],
        …,
        [-2.5018e+00,  1.9451e+01,  2.1265e+00,  …,  1.0007e+00,
          1.7512e+00,  7.1300e+00],
        [-4.7534e+00,  1.9389e+01,  1.1258e+00,  …, -1.7512e+00,
         -1.2509e+00,  1.2509e+00],
        [ 5.0035e+00,  2.4955e+01,  1.0445e+01,  …,  7.1926e+00,
          5.9417e+00,  2.6269e+00]],

       [[ 8.5060e+00,  1.3697e+01,  1.9201e+01,  …,  2.0264e+01,
          2.4580e+01,  2.6143e+01],
        [-2.5768e+01, -4.8159e+01, -4.7221e+01,  …, -4.6345e+01,
         -4.0278e+01, -1.4323e+01],
        [ 8.9438e+00,  2.1265e+00,  5.1286e+00,  …,  5.0035e+00,
         -3.5650e+00,  8.7562e-01],
        …,
        [ 9.5067e+00, -8.1308e-01,  4.5032e+00,  …,  1.1258e+00,
          2.3767e+00, -6.0668e+00],
        [ 1.0945e+01, -6.6922e+00, -7.5678e+00,  …, -1.6699e+01,
         -1.4760e+01, -8.7562e+00],
        [ 1.3009e+01,  8.3809e+00,  6.8799e+00,  …,  2.8145e+00,
          1.1258e+00, -9.0689e+00]],

       [[-2.3266e+01,  9.9445e+00, -4.0028e+00,  …,  3.4399e+00,
         -1.9076e+01,  8.1933e+00],
        [-3.8214e+01,  9.9445e+00, -5.1286e+00,  …,  3.6276e+00,
         -3.3899e+01,  1.0820e+01],
        [-3.1522e+01,  1.2446e+01,  3.8777e+00,  …,  8.8813e+00,
         -3.1522e+01,  9.3191e+00],
        …,
        [-1.7200e+01,  9.3816e+00,  3.5025e+00,  …, -7.7555e+00,
          3.3148e+01, -8.1307e+00],
        [-1.3635e+01,  8.3184e+00, -1.2509e-01,  …, -9.8820e+00,
          3.2335e+01, -1.2571e+01],
        [-9.8194e+00,  3.4399e+00,  4.3781e-01,  …, -1.4698e+01,
          1.3635e+01, -1.2134e+01]]],


      [[[-3.2523e+00,  7.5053e+00,  8.8813e+00,  …,  8.8187e+00,
          4.5657e+00,  6.3795e+00],
        [ 2.8145e+00,  1.8263e+01,  2.3517e+01,  …,  3.2273e+01,
          1.5323e+01,  1.2634e+01],
        [ 4.5032e+00,  1.5886e+01,  1.9639e+01,  …,  4.1154e+01,
          1.7137e+01,  1.0320e+01],
```

```
 …,
 [ 1.2509e-01,  1.3822e+01,  1.4385e+00,  …,  3.6213e+01,
   1.3947e+01,  1.1320e+01],
 [ 2.8145e+00,  1.4948e+01,  3.1272e+00,  …,  2.8708e+01,
   7.9431e+00,  1.0382e+01],
 [ 4.3781e-01,  1.2509e+01,  7.6929e+00,  …,  2.9708e+01,
   1.2571e+01,  1.3635e+01]],

[[-3.5900e+01, -5.6915e+01, -6.5922e+01,  …, -6.8111e+01,
  -6.4358e+01, -5.3225e+01],
 [ 1.4385e+00,  1.0883e+01,  2.3079e+01,  …,  1.5573e+01,
   2.2141e+01,  1.2446e+01],
 [-2.0327e+01, -4.1217e+01, -4.2092e+01,  …, -3.5150e+01,
  -2.9959e+01, -1.9076e+01],
 …,
 [-5.2600e+01, -9.0189e+01, -9.9445e+01,  …, -3.2836e+01,
  -2.7895e+01, -1.9576e+01],
 [-4.3406e+01, -1.9514e+01,  9.1940e+00,  …, -3.9028e+01,
  -4.5032e+01, -3.9090e+01],
 [ 8.5060e+00, -7.3802e+00, -4.6032e+01,  …, -4.7346e+01,
  -2.8020e+01, -1.3760e+00]],

[[-2.4079e+01, -1.8388e+01, -1.5448e+01,  …, -1.8513e+01,
  -2.0327e+01, -4.2968e+01],
 [-1.1508e+01, -4.3781e+00, -1.0070e+01,  …,  1.5261e+01,
   1.4823e+01, -3.6964e+01],
 [-2.7770e+01, -2.5268e+01, -3.1210e+01,  …, -2.8145e+01,
  -1.0507e+01, -4.7471e+01],
 …,
 [-4.7471e+01, -5.2725e+01, -3.1835e+01,  …, -1.3572e+01,
  -6.0042e+00, -4.7659e+01],
 [-1.9952e+01, -3.2398e+01, -4.5219e+01,  …, -1.9514e+01,
  -6.8173e+00, -4.2155e+01],
 [-2.9896e+01, -4.2155e+01, -4.5907e+01,  …, -1.9514e+01,
  -8.1307e+00, -3.0459e+01]],


…,

[[-2.8333e+01, -7.7555e+00, -1.8826e+01,  …, -1.8826e+01,
  -2.4017e+01, -1.8013e+01],
 [-2.4705e+01,  5.3788e+00, -1.6887e+00,  …,  1.2196e+01,
   1.3384e+01,  1.2384e+01],
 [-2.9083e+01, -4.2530e+00, -5.1912e+00,  …,  1.8762e-01,
   8.1933e+00,  7.9431e+00],
 …,
 [-3.7527e+00, -3.4962e+01, -4.5032e+00,  …, -4.2530e+00,
   8.6311e+00, -1.8763e-01],
 [-5.3788e+00, -2.2641e+01, -1.5636e+00,  …, -1.1883e+01,
```

```
          -4.3042e-08, -1.3134e+01],
         [ 5.2537e+00, -6.2543e-02,  1.8763e+00,  …, -4.1279e+00,
           1.2509e+00, -1.6887e+00]],

        [[-1.0945e+01, -8.3809e+00, -1.3885e+01,  …, -3.7839e+01,
          -4.6971e+01, -3.4587e+01],
         [ 3.0209e+01,  4.7721e+01,  5.2662e+01,  …,  5.8917e+01,
           3.3148e+01,  3.6276e+00],
         [-2.5018e-01, -4.2530e+00, -9.4442e+00,  …,  1.6512e+01,
           5.0661e+00, -7.8180e+00],
         …,
         [-1.4510e+01, -1.4135e+01, -6.7548e+00,  …,  5.0036e-01,
          -2.1890e+00, -9.9445e+00],
         [ 1.1946e+01, -2.8145e+00,  6.2544e+00,  …, -1.0445e+01,
          -2.2829e+01, -1.7137e+01],
         [ 8.7562e+00,  1.3635e+01, -3.0021e+00,  …,  1.3009e+01,
           1.1633e+01, -4.8784e+00]],

        [[-1.3760e+00, -9.5693e+00, -8.4435e+00,  …, -5.1286e+00,
           8.3809e+00, -2.7082e+01],
         [ 1.8138e+01,  1.5886e+01,  4.2530e+00,  …,  2.6519e+01,
           5.7541e+01, -8.9438e+00],
         [ 1.4385e+01,  2.5956e+01, -5.5039e+00,  …,  3.2023e+01,
           6.6047e+01,  2.4392e+00],
         …,
         [-3.2523e+01, -9.5693e+00, -1.4010e+01,  …,  4.1029e+01,
           5.9479e+01,  6.8799e+00],
         [-1.9326e+01,  3.1272e+00, -1.6449e+01,  …,  2.8895e+01,
           4.5970e+01,  2.6269e+00],
         [-1.3572e+01,  4.7534e+00, -1.8638e+01,  …, -1.1258e+00,
           1.5261e+01, -8.0057e+00]]]], device='cuda:0',
    grad_fn=<MulBackward0>)
```

[7]:

```
conv_ref = torch.nn.Conv2d(in_channels = 64, out_channels=64, kernel_size = 3,
 ↪padding=1)
conv_ref.weight = model.features[3].weight_q
conv_ref.bias = model.features[3].bias
output_ref = conv_ref(act)
print(output_ref)
```

```
tensor([[[[-3.3602e+00,  2.0532e+01,  9.7798e+00,  …,  1.0963e+01,
            2.2540e+01,  5.5590e+00],
          [-1.0819e+01,  3.1853e+01,  2.4546e+01,  …,  3.1208e+01,
            4.9200e+01,  2.5180e+01],
          [-1.4340e+01,  2.5186e+01,  2.2635e+01,  …,  2.9151e+01,
            4.9394e+01,  2.3629e+01],
          …,
          [ 3.8949e+01, -1.7879e+01,  1.3632e+01,  …,  2.4148e+00,
```

```
  -2.3749e+01, -9.8002e+00],
 [ 3.1316e+01, -2.1118e+01,  8.2696e+00,  …, -4.5471e+00,
  -3.5107e+01, -1.6342e+01],
 [ 3.2810e+01,  5.0826e+00,  2.3477e+01,  …,  1.7478e+01,
  -1.0417e+01, -1.1214e+00]],

[[-2.3923e+01, -3.7368e+01, -4.1696e+01,  …, -5.3020e+01,
  -5.5052e+01, -4.1891e+01],
 [-3.3807e+01, -4.3517e+01, -4.1983e+01,  …, -2.7630e+01,
  -1.6717e+01, -6.2841e+00],
 [-2.4024e+01, -3.1968e+01, -4.0776e+01,  …, -2.8662e+01,
  -3.2673e+01, -2.2955e+01],
 …,
 [-5.2887e+00, -2.0726e+01, -3.0393e+01,  …, -1.0850e+00,
   6.5697e-01, -7.7698e+00],
 [-1.3645e+01, -3.2533e+01, -2.3336e+01,  …, -3.5640e+01,
  -3.4426e+01, -2.9809e+01],
 [-2.6411e+01, -5.2874e+01, -5.6952e+01,  …, -6.5751e+01,
  -4.4761e+01, -2.3863e+01]],

[[-2.6988e+01, -3.2802e+01, -3.3092e+01,  …, -3.1469e+01,
  -3.0571e+01, -2.7239e+01],
 [-4.5271e+01, -5.9726e+01, -6.9796e+01,  …, -4.9065e+01,
  -4.3409e+01, -4.6985e+01],
 [-2.1994e+01, -2.5883e+01, -3.3747e+01,  …, -3.1320e+01,
  -3.0498e+01, -3.7101e+01],
 …,
 [-3.5888e+01, -2.7828e+01, -1.9982e+01,  …, -9.4207e+00,
  -1.2443e+01, -4.0185e+01],
 [-2.8309e+01, -1.0907e+01, -6.0301e+00,  …, -1.3052e+01,
   2.9593e+00, -2.2651e+01],
 [-3.5215e+01, -4.4771e+01, -4.6855e+01,  …, -4.3335e+01,
  -3.2498e+01, -3.4648e+01]],


…,

[[-2.2553e+01, -4.7165e+01, -2.2930e+01,  …, -2.4353e+01,
  -2.8652e+01, -4.1579e+00],
 [-2.4592e+01, -6.2499e+01, -2.5296e+01,  …, -2.2715e+01,
  -2.9512e+01,  7.8098e+00],
 [-2.6059e+01, -6.4663e+01, -3.4260e+01,  …, -3.1749e+01,
  -4.3138e+01,  5.9124e-01],
 …,
 [-1.2920e+01,  8.6443e+01, -1.3655e+01,  …,  4.4109e+00,
   1.6460e+01, -5.3103e+00],
 [-1.7768e+01,  7.8135e+01, -2.4463e+01,  …, -9.9244e+00,
   2.7352e+01, -2.2258e+01],
 [-5.6614e+00,  6.0189e+01, -9.0390e+00,  …,  3.4452e+00,
```

```
   2.6293e+01, -7.9918e+00]],

 [[ 1.3179e+00,  3.6501e+00,  4.8317e-02,  …, -9.9882e+00,
   -1.2519e+01, -2.1081e+01],
  [ 2.0983e+01,  3.6723e+01,  3.8912e+01,  …,  6.0071e+01,
    6.1826e+01,  3.1400e+01],
  [-1.1199e+00,  9.0141e+00,  7.3010e+00,  …,  1.3291e+01,
    2.4056e+01,  1.0953e+01],
  …,
  [ 1.9961e+01, -1.3177e+01, -2.4490e+01,  …,  1.0173e+01,
   -2.0147e+01, -2.6611e+01],
  [ 1.1920e+00, -2.5519e+01, -3.3356e+01,  …, -3.3124e+01,
   -3.6023e+01, -3.3838e+01],
  [ 3.0828e+01,  1.7732e+01,  1.7058e+01,  …,  3.5045e+00,
    1.2047e+01, -8.1261e+00]],

 [[-4.2574e+00, -3.9318e+00, -1.2868e+01,  …, -5.1914e+00,
    2.1534e+01,  9.9363e-01],
  [ 8.3369e+00,  1.1759e+01, -5.8651e+00,  …,  7.8045e+00,
    5.0657e+01,  1.1184e+01],
  [ 4.6659e+00,  1.2253e+01, -1.0776e+01,  …,  3.7100e+00,
    4.5354e+01,  1.4105e+01],
  …,
  [-1.7381e+01,  3.1880e+01,  1.5133e+01,  …,  4.6786e+01,
    6.1220e+01, -2.4280e+01],
  [-2.2319e+01,  1.9012e+01,  2.7018e+01,  …,  2.9902e+01,
    4.9004e+01,  8.2395e+00],
  [-1.5481e+01, -2.2783e+00,  1.4711e+01,  …,  6.9925e+00,
    1.8499e+01,  1.6478e+01]]],


[[[ 1.3837e+01,  2.1585e+00,  8.3291e+00,  …,  9.4705e+00,
    2.3134e-01, -1.1244e+01],
  [ 4.9605e+00,  8.1396e-03,  3.4845e-02,  …,  1.3601e+00,
   -3.3197e+00, -1.3929e+01],
  [ 3.3270e+00,  2.5936e+00,  4.1224e-01,  …,  8.2704e-01,
   -3.3251e+00, -1.4248e+01],
  …,
  [ 7.8371e+00,  1.9134e+01,  3.1944e+01,  …,  2.3542e+00,
   -3.9489e+00, -1.1212e+01],
  [ 4.7181e+00,  1.7243e+01,  2.6319e+01,  …,  4.9089e-01,
   -5.6954e+00, -1.0959e+01],
  [ 6.4820e+00,  1.7359e+01,  2.3404e+01,  …,  1.0691e+01,
    6.0804e+00,  1.9082e+00]],

 [[ 4.8498e+01,  1.0288e+02,  9.4974e+01,  …,  9.5835e+01,
    8.8995e+01,  6.6519e+01],
  [-7.8764e+01, -1.1662e+02, -1.2861e+02,  …, -1.3069e+02,
```

```
  -1.4441e+02, -1.0969e+02],
 [-3.6860e+01, -3.5385e+01, -3.8928e+01,  …, -3.6570e+01,
  -5.0545e+01, -5.2141e+01],
 …,
 [-3.1790e+01, -5.4460e+01, -4.4599e+01,  …, -1.3139e+01,
  -3.0341e+01, -3.8587e+01],
 [-1.7494e+01, -2.9859e+01, -2.8613e+01,  …, -2.5463e+01,
  -3.9966e+01, -4.0180e+01],
 [-1.6176e+01, -2.8456e+01, -1.9244e+01,  …,  3.7725e+00,
   1.7992e+00, -1.6594e+01]],

[[-2.7704e+01, -3.2212e+01, -2.9889e+01,  …, -3.2626e+01,
  -2.1903e+01, -5.8310e+01],
 [-8.2865e+01, -9.3464e+01, -8.1879e+01,  …, -8.1989e+01,
  -6.3534e+01, -8.3607e+01],
 [-4.7236e+01, -4.7852e+01, -3.3410e+01,  …, -3.1592e+01,
  -1.7378e+01, -6.5645e+01],
 …,
 [-2.2436e+01, -1.8312e+01, -7.5501e+00,  …, -1.9991e+01,
  -1.9210e+01, -5.2657e+01],
 [-1.9603e+01, -1.7581e+01, -1.4326e+01,  …, -3.2147e+01,
  -2.4638e+01, -5.3452e+01],
 [-2.0615e+01, -2.3337e+01, -2.0938e+01,  …, -4.6939e+01,
  -4.1865e+01, -5.4597e+01]],


…,


[[ 4.5415e+00, -3.2413e+01, -1.2382e+01,  …, -1.2416e+01,
  -2.6981e+01,  2.1894e+00],
 [-2.9711e-02, -4.4443e+01, -1.9767e+01,  …, -1.9903e+01,
  -3.8620e+01, -5.2451e+00],
 [ 9.8255e+00, -4.0188e+01, -9.0577e+00,  …, -9.1275e+00,
  -2.3425e+01, -5.6888e+00],
 …,
 [-1.9808e+01, -2.0154e+01, -1.5468e+01,  …,  1.3466e+00,
  -6.4676e+00, -6.1376e+00],
 [-1.7548e+01, -2.1183e+01, -2.2500e+01,  …, -5.0870e+00,
  -1.0759e+01, -3.8363e+00],
 [-1.0832e+01, -1.2980e+01, -1.5850e+01,  …,  1.1871e+01,
   1.0601e+01,  5.4281e-01]],

[[ 3.7932e+01,  4.8426e+01,  5.2147e+01,  …,  5.4677e+01,
   5.5227e+01,  5.2782e+01],
 [-2.8144e+01, -4.9257e+01, -4.7968e+01,  …, -4.7877e+01,
  -4.7781e+01, -7.2223e+00],
 [ 8.1257e+00,  5.4209e+00,  1.2217e+01,  …,  1.1522e+01,
   3.7753e-01,  1.4958e+01],
 …,
```

```
 [ 3.3636e+00,  6.9906e+00,  1.4065e+01,  …,  7.0420e+00,
  -5.5089e+00,  5.3101e+00],
 [ 3.2230e+00,  7.8386e+00,  1.0831e+01,  …,  5.1825e+00,
   1.9586e+00,  4.4800e+00],
 [ 6.5787e+00,  9.9820e+00,  1.5702e+01,  …, -8.3511e-01,
  -7.5057e+00,  1.0192e+01]],

[[-4.2402e+01,  9.8400e+00, -4.0778e+00,  …, -1.6237e+00,
  -7.1301e+01,  8.2484e+00],
 [-6.4577e+01,  1.4055e+01, -3.1034e+00,  …, -1.3341e+00,
  -1.1282e+02,  9.1984e+00],
 [-6.0276e+01,  1.9283e+01,  6.6087e+00,  …,  6.6310e+00,
  -1.0813e+02,  1.3551e+01],
 …,
 [-2.7000e+01,  6.0519e+01,  4.1895e+01,  …, -1.1309e+01,
  -1.0492e+02,  7.6147e+00],
 [-1.6291e+01,  5.1364e+01,  1.9140e+01,  …, -7.9915e+00,
  -9.3152e+01,  1.0063e+01],
 [-9.1710e+00,  2.2921e+01,  1.8902e-01,  …, -3.3819e+00,
  -6.0958e+01,  8.6303e+00]]],


[[[ 1.0682e+01, -1.4441e+01, -1.8923e+00,  …,  9.8341e+00,
   -1.3444e+00, -1.3494e+01],
  [ 1.4441e+01, -1.3021e+01, -3.9512e+00,  …,  1.6794e+00,
   -3.9345e+00, -1.4500e+01],
  [ 1.6730e+01, -6.2703e+00, -1.8917e+00,  …, -2.6925e+00,
   -4.6632e+00, -1.4538e+01],
  …,
  [ 2.8028e+01,  1.5557e+01,  3.1439e+01,  …,  2.9450e+01,
    1.5070e+01,  9.0465e+00],
  [ 2.1313e+01,  7.4001e+00,  2.0205e+01,  …,  3.0549e+01,
    1.0227e+01,  4.8106e+00],
  [ 2.0039e+01,  1.3158e+01,  2.3638e+01,  …,  3.2523e+01,
    1.1129e+01,  8.4305e+00]],

 [[ 3.5718e+01,  7.5889e+01,  6.2034e+01,  …,  9.1616e+01,
    8.5663e+01,  7.0331e+01],
  [-3.8451e+01, -5.2729e+01, -4.6633e+01,  …, -1.2593e+02,
   -1.4119e+02, -1.0983e+02],
  [-3.6793e+01, -4.5157e+01, -4.5287e+01,  …, -3.7609e+01,
   -5.8926e+01, -5.5828e+01],
  …,
  [-1.7641e+01, -3.7698e+01, -2.7904e+01,  …, -3.5798e+01,
   -4.7635e+01, -3.4972e+01],
  [-8.7768e+00, -3.2664e+01, -2.8921e+01,  …, -4.8951e+01,
   -4.8130e+01, -4.1318e+01],
  [-1.2481e+01, -2.5457e+01, -2.4223e+01,  …, -5.3968e+01,
```

```
  -3.3908e+01,  2.7117e-01]],

[[-1.3069e+01, -2.1274e+01, -2.2079e+01,  …, -2.7928e+01,
  -1.3828e+01, -5.7155e+01],
 [-4.0148e+01, -3.6890e+01, -1.6339e+01,  …, -8.1764e+01,
  -6.5248e+01, -9.1622e+01],
 [-4.3046e+01, -4.7647e+01, -3.7509e+01,  …, -3.8401e+01,
  -2.1789e+01, -7.6047e+01],
 …,
 [-3.1515e+01, -1.3588e+01, -1.8999e+01,  …, -3.3998e+01,
  -1.3110e+01, -4.6566e+01],
 [-2.0106e+01, -2.7688e+00, -3.8855e+00,  …, -2.6257e+01,
  -4.3930e-01, -4.0385e+01],
 [-1.6193e+01, -1.5539e+01, -1.9065e+01,  …, -1.1973e+01,
  -2.7045e+00, -3.5035e+01]],

…,

[[-1.0723e+00, -9.5719e+00, -1.3737e+01,  …, -1.2558e+01,
  -2.5505e+01, -3.5902e+00],
 [ 2.2669e+00, -3.6719e+00, -6.3902e+00,  …, -2.0458e+01,
  -3.5370e+01, -1.3433e+01],
 [ 7.2937e+00,  3.6495e-01, -4.9715e+00,  …, -7.0387e+00,
  -2.0395e+01, -1.8374e+01],
 …,
 [-4.5213e+01,  5.2583e+01, -9.0764e+00,  …, -2.1922e+01,
  -9.5013e+00, -1.8857e-01],
 [-4.6873e+01,  4.0173e+01, -1.6912e+01,  …, -3.5363e+01,
  -7.4127e+00, -1.3233e+01],
 [-2.8497e+01,  2.7787e+01, -6.3206e+00,  …, -2.3511e+01,
  -1.7113e+00, -8.4030e-01]],

[[ 1.5686e+01,  1.4931e+01,  7.8311e+00,  …,  5.6989e+01,
   5.1463e+01,  5.0841e+01],
 [-2.2549e+01, -3.6965e+01, -3.4297e+01,  …, -4.6715e+01,
  -4.3239e+01, -2.8514e+00],
 [ 3.5709e+00, -2.6442e+00,  1.7116e+01,  …,  5.7474e+00,
  -3.5554e+00,  1.7345e+01],
 …,
 [ 7.4397e+00,  7.5916e+00,  5.6584e+00,  …,  8.5836e+00,
   2.5787e+00, -1.2863e+01],
 [-7.6612e+00, -1.1558e+01, -1.1713e+01,  …, -1.1276e+01,
  -1.4074e+01, -1.5818e+01],
 [ 2.2611e+01,  1.6234e+01,  1.5259e+01,  …,  9.3320e+00,
   8.4813e+00, -1.1796e+01]],

[[-4.8805e+01,  8.7325e-01,  4.1364e+01,  …,  1.2062e+00,
  -9.0862e+01,  7.7336e+00],
```

```
  [-7.6160e+01, -4.3552e+00,  7.0548e+01,  …,  4.4618e+00,
   -1.4592e+02,  1.2420e+01],
  [-6.9574e+01, -3.1224e+00,  7.7183e+01,  …,  1.1031e+01,
   -1.4945e+02,  2.1959e+01],
  …,
  [ 1.9987e+01, -1.2190e+01,  1.2787e+01,  …,  1.1160e+01,
    9.0232e+01,  1.0558e+01],
  [ 1.9674e+01, -1.9645e+01,  6.2222e+00,  …,  2.3847e+01,
    8.1682e+01,  1.2654e+00],
  [ 1.0971e+01, -2.2517e+01, -1.0564e+01,  …,  1.0129e+01,
    3.8411e+01, -1.2292e+01]]],


…,


[[[ 7.9181e+00,  1.8677e+01,  7.0680e+00,  …,  5.1486e+01,
    2.0191e+01, -2.1308e+01],
  [ 3.3370e+00,  3.2760e+01,  1.3153e+01,  …,  9.6486e+01,
    5.4103e+01, -7.0558e+00],
  [ 1.2851e+00,  3.6242e+01,  1.1087e+01,  …,  8.5010e+01,
    4.5864e+01, -4.5907e+00],
  …,
  [-1.8274e+01,  3.4886e+01,  3.2580e+01,  …,  2.9489e+01,
   -6.2296e+00, -3.5357e+00],
  [-1.7159e+01,  3.7251e+01,  4.1439e+01,  …,  4.0362e+01,
    7.0424e+00, -4.6864e+00],
  [-1.7081e+01,  1.9688e+01,  2.6903e+01,  …,  3.0807e+01,
    1.6406e+01,  6.8026e+00]],

 [[ 4.5971e+00,  1.1130e+01,  3.9044e+00,  …, -5.1492e+01,
   -5.5264e+01, -3.5904e+01],
  [-6.0606e+01, -8.5093e+01, -8.2206e+01,  …, -7.6749e+01,
   -1.7725e+01,  8.7558e-01],
  [-3.2826e+01, -3.5278e+01, -4.1446e+01,  …, -4.1448e+01,
   -3.7811e+01, -2.7703e+01],
  …,
  [-2.7960e+01, -2.7839e+01, -3.2577e+01,  …, -1.6976e+01,
   -2.6823e+01, -2.6402e+01],
  [-4.3875e+01, -5.2347e+01, -5.4646e+01,  …, -1.2532e+02,
   -6.9013e+01, -2.7821e+01],
  [ 2.7838e-01,  1.8774e+01,  2.0588e+01,  …,  1.4464e+01,
   -1.6903e+01, -1.7600e+01]],

 [[-2.6583e+01, -3.7437e+01, -3.6420e+01,  …, -6.8614e+01,
   -4.2517e+01, -2.7118e+01],
  [-7.3880e+01, -9.2638e+01, -8.7177e+01,  …, -1.1231e+02,
   -7.7278e+01, -4.9366e+01],
```

```
 [-3.0996e+01, -3.5635e+01, -2.5510e+01,  …, -7.7920e+01,
  -7.0675e+01, -5.0449e+01],
 …,
 [-4.0139e+01, -4.4021e+01, -3.9027e+01,  …, -7.6650e+01,
  -4.8830e+01, -3.2633e+01],
 [-3.5128e+01, -4.5106e+01, -4.5208e+01,  …, -4.1418e+01,
  -3.3520e+01, -3.1095e+01],
 [-1.3940e+01, -1.8511e+01, -2.1045e+01,  …, -1.7165e+01,
  -2.9560e+01, -2.6267e+01]],

…,

[[-1.5151e+01, -5.2504e+01, -4.9176e+00,  …, -5.6165e+01,
   1.6643e+01,  5.4511e+01],
 [-2.4683e+01, -7.6417e+01,  5.0500e-01,  …, -6.2095e+01,
   4.5419e+01,  1.0019e+02],
 [-2.6342e+01, -7.9411e+01,  6.8174e+00,  …, -4.8736e+01,
   4.9430e+01,  9.5588e+01],
 …,
 [-1.9792e+01, -8.8888e+01, -4.6190e+01,  …, -4.4108e+00,
   2.5499e+01,  1.9357e+01],
 [-1.8799e+01, -7.7923e+01, -4.3412e+01,  …, -2.4517e+01,
   1.2028e+01, -1.9439e+00],
 [-2.3954e+00, -3.2983e+01, -1.4987e+01,  …, -1.7596e+00,
   1.9609e+01,  4.8548e+00]],

[[ 2.8543e+01,  2.9480e+01,  1.8585e+01,  …,  3.0010e+01,
  -1.1100e+01, -3.2437e+01],
 [ 5.2327e+00,  1.0642e+01,  1.5606e+01,  …,  1.0545e+02,
   9.2562e+01,  3.0747e+01],
 [ 7.7668e+00,  1.4418e+01,  1.2131e+01,  …,  4.3001e+01,
   4.0900e+01,  1.1418e+01],
 …,
 [-2.5969e+01, -1.0543e+01,  2.6046e+00,  …,  3.6496e+01,
   3.5358e+01,  4.1639e+00],
 [-2.2583e+01, -7.3608e+00,  8.9936e+00,  …, -6.8982e+01,
  -1.5086e+01,  1.4791e-02],
 [-2.3089e+01, -2.1360e+01, -7.6628e+00,  …,  1.0215e+01,
  -1.2082e+01, -1.0544e+01]],

[[-1.8385e+01,  1.9594e+01, -6.7155e+00,  …,  3.2149e+01,
   5.3443e+01,  1.4855e+01],
 [-2.4619e+01,  3.1190e+01, -7.6170e+00,  …,  6.0968e+01,
   8.9614e+01,  3.0625e+01],
 [-2.3313e+01,  2.8397e+01,  6.6465e-01,  …,  6.5365e+01,
   8.6787e+01,  2.4046e+01],
 …,
 [-2.0580e+01, -5.1761e+01, -4.0636e+01,  …,  6.8802e+00,
```

```
    4.9438e+01,  4.5296e+00],
  [-1.1297e+01, -5.1084e+01, -4.6124e+01,  …, -5.7604e+01,
    5.3683e+00,  6.0946e+00],
  [-2.6919e+00, -3.3926e+01, -3.8528e+01,  …, -4.4823e+01,
   -3.4275e+01, -9.3765e+00]]],


[[[ 1.2969e+01, -4.3294e+00,  6.2346e+00,  …,  9.4737e-01,
   -3.7193e+00, -8.0256e+00],
  [ 1.5948e+01, -4.9786e+00,  3.7219e+00,  …, -1.5602e+00,
   -7.3961e+00, -1.1329e+01],
  [ 1.9373e+01, -3.2648e+00,  6.5128e+00,  …, -1.0896e+00,
   -6.4196e+00, -9.6600e+00],
  …,
  [ 1.8367e+01, -4.0416e-01,  7.0468e+00,  …,  1.2114e+01,
    2.5492e+00,  2.5098e+00],
  [ 1.6664e+01, -2.4093e+00,  1.5200e+00,  …,  4.5427e+00,
   -2.2893e+00, -1.7442e+00],
  [ 1.5453e+01,  1.0399e+01,  1.1790e+01,  …,  9.1446e+00,
    4.5816e+00,  4.7779e+00]],

 [[ 3.7602e+01,  7.1608e+01,  6.7031e+01,  …,  6.5241e+01,
    5.5795e+01,  4.0867e+01],
  [-4.0566e+01, -5.7039e+01, -6.0354e+01,  …, -6.6122e+01,
   -6.7554e+01, -5.3195e+01],
  [-2.0404e+01, -2.1549e+01, -2.6636e+01,  …, -2.4435e+01,
   -3.1342e+01, -3.3527e+01],
  …,
  [-2.4516e+01, -3.7435e+01, -2.7665e+01,  …, -2.7364e+01,
   -4.3529e+01, -3.3675e+01],
  [-1.8450e+01, -2.1503e+01, -2.3618e+01,  …, -3.3504e+01,
   -1.9605e+01, -1.1163e+01],
  [-6.0736e+00, -1.9750e+01, -2.3700e+01,  …, -2.9707e+01,
   -3.3889e+01, -2.4417e+01]],

 [[-1.9014e+01, -2.5431e+01, -2.2939e+01,  …, -2.3810e+01,
   -1.7053e+01, -3.1923e+01],
  [-3.2197e+01, -3.1876e+01, -2.3426e+01,  …, -2.7149e+01,
   -2.0291e+01, -3.4646e+01],
  [-3.0375e+01, -3.2578e+01, -2.7400e+01,  …, -2.7924e+01,
   -2.1117e+01, -3.8666e+01],
  …,
  [-3.3167e+01, -3.6592e+01, -3.2541e+01,  …, -3.6660e+01,
   -3.6946e+01, -3.1187e+01],
  [-1.7767e+01, -2.1163e+01, -1.9521e+01,  …, -1.4186e+01,
   -2.1532e+01, -3.0468e+01],
  [-3.0193e+01, -3.8881e+01, -3.5809e+01,  …, -1.6271e+01,
   -2.3943e+01, -2.5264e+01]],
```

```
 …,

 [[ 1.2947e+00, -6.3369e+00, -7.7937e+00,  …, -6.9400e+00,
   -1.2297e+01,  1.2272e+00],
  [ 1.2873e+00,  2.9619e+00, -6.5153e+00,  …, -5.9328e+00,
   -1.4603e+01, -2.6121e+00],
  [ 3.5555e+00,  1.4116e+01, -1.2584e+00,  …,  2.6513e+00,
   -8.0155e+00, -2.1817e+00],
  …,
  [-2.7296e+00,  2.0867e+01,  3.0444e+00,  …,  9.0831e-01,
    5.8475e-01,  6.7515e+00],
  [-4.4488e+00,  1.9397e+01,  1.8275e+00,  …, -1.7283e+00,
   -2.1300e+00, -9.7471e-02],
  [ 4.8988e+00,  2.4770e+01,  1.0573e+01,  …,  6.7292e+00,
    5.8364e+00,  1.8172e+00]],

 [[ 8.7984e+00,  1.2665e+01,  1.9328e+01,  …,  1.9456e+01,
    2.4159e+01,  2.6752e+01],
  [-2.4573e+01, -5.1288e+01, -4.9272e+01,  …, -4.5635e+01,
   -4.1156e+01, -1.3606e+01],
  [ 1.0491e+01, -5.3142e-01,  4.7258e+00,  …,  3.6874e+00,
   -3.3697e+00,  3.0334e+00],
  …,
  [ 1.0640e+01, -4.0648e-01,  4.3765e+00,  …,  2.1168e+00,
    1.9931e+00, -6.0621e+00],
  [ 1.1091e+01, -6.4875e+00, -6.7388e+00,  …, -1.7356e+01,
   -1.5051e+01, -8.5471e+00],
  [ 1.3529e+01,  8.9424e+00,  7.4137e+00,  …,  2.1501e+00,
   -3.8354e-02, -9.8307e+00]],

 [[-2.4619e+01,  1.1506e+01, -4.3226e+00,  …,  3.5924e+00,
   -2.8256e+01,  8.4391e+00],
  [-4.0242e+01,  1.3081e+01, -5.5457e+00,  …,  3.6791e+00,
   -4.7459e+01,  1.1695e+01],
  [-3.3556e+01,  1.6659e+01,  3.1992e+00,  …,  8.2448e+00,
   -4.5542e+01,  1.0465e+01],
  …,
  [-1.7012e+01,  1.0206e+01,  2.6427e+00,  …, -6.8355e+00,
    3.2383e+01, -7.7538e+00],
  [-1.3175e+01,  8.7209e+00, -5.5861e-01,  …, -9.0308e+00,
    3.1196e+01, -1.1606e+01],
  [-9.3254e+00,  3.4223e+00,  2.6070e-01,  …, -1.4218e+01,
    1.1750e+01, -1.1379e+01]]],


[[[-3.1247e+00,  7.2372e+00,  8.5881e+00,  …,  9.2347e+00,
    5.4720e+00,  6.0711e+00],
```

```
 [ 2.8783e+00,  1.8319e+01,  2.3706e+01,  …,  3.2528e+01,
   1.4629e+01,  1.2123e+01],
 [ 5.6463e+00,  1.6723e+01,  1.9412e+01,  …,  4.0679e+01,
   1.1331e+01,  7.5378e+00],
 …,
 [-1.0168e+00,  1.4143e+01,  1.9684e+00,  …,  3.7215e+01,
   8.8453e+00,  8.2956e+00],
 [ 1.7879e+00,  1.4773e+01,  3.5527e+00,  …,  3.0252e+01,
   2.6164e+00,  6.6859e+00],
 [-4.6712e-01,  1.2207e+01,  7.9163e+00,  …,  2.9844e+01,
   7.9800e+00,  1.1241e+01]],

[[-3.4612e+01, -5.6271e+01, -6.6479e+01,  …, -6.8138e+01,
  -7.2180e+01, -6.5375e+01],
 [-1.7271e-01,  1.0488e+01,  2.3724e+01,  …,  1.6456e+01,
   2.9681e+01,  1.9312e+01],
 [-1.9424e+01, -4.0563e+01, -4.1525e+01,  …, -3.5124e+01,
  -3.0261e+01, -2.1116e+01],
 …,
 [-5.3287e+01, -9.0813e+01, -9.8308e+01,  …, -3.4871e+01,
  -3.0209e+01, -2.3319e+01],
 [-4.3251e+01, -1.8591e+01,  1.0022e+01,  …, -3.9696e+01,
  -4.5247e+01, -4.5088e+01],
 [ 8.5040e+00, -7.1211e+00, -4.5816e+01,  …, -4.7342e+01,
  -3.1974e+01, -2.2024e+00]],

[[-2.5045e+01, -1.8607e+01, -1.5573e+01,  …, -1.7514e+01,
  -1.5082e+01, -4.6250e+01],
 [-1.1336e+01, -4.4910e+00, -1.0710e+01,  …,  1.5955e+01,
   2.4577e+01, -4.4297e+01],
 [-2.8287e+01, -2.5458e+01, -3.2588e+01,  …, -2.7111e+01,
  -9.2459e+00, -5.2294e+01],
 …,
 [-4.6050e+01, -5.1198e+01, -3.1108e+01,  …, -1.3818e+01,
  -5.8782e+00, -5.2125e+01],
 [-2.0420e+01, -3.3330e+01, -4.4297e+01,  …, -2.0447e+01,
  -4.9166e+00, -4.7276e+01],
 [-2.9446e+01, -4.1673e+01, -4.4295e+01,  …, -1.9658e+01,
  -8.8599e+00, -3.8871e+01]],


…,


[[-2.8390e+01, -7.2303e+00, -1.8860e+01,  …, -1.8477e+01,
  -2.2794e+01, -2.6199e+01],
 [-2.5008e+01,  5.4478e+00, -1.3013e+00,  …,  1.2074e+01,
   1.6956e+01,  6.3513e+00],
 [-2.9048e+01, -4.7285e+00, -5.2490e+00,  …, -1.3814e-01,
   1.4298e+01,  3.1702e+00],
```

```
       …,
       [-3.6954e+00, -3.6661e+01, -4.6082e+00,  …, -6.1819e+00,
         1.5410e+01, -3.6481e+00],
       [-5.3776e+00, -2.3309e+01, -1.5093e+00,  …, -1.2986e+01,
         7.8112e+00, -2.0553e+01],
       [ 5.4829e+00, -1.1023e+00,  1.4891e+00,  …, -5.7294e+00,
         5.7604e+00, -5.4535e+00]],

      [[-1.1548e+01, -8.8188e+00, -1.3786e+01,  …, -3.8128e+01,
        -5.2170e+01, -3.8860e+01],
       [ 3.1793e+01,  4.7593e+01,  5.2851e+01,  …,  6.0211e+01,
         3.6815e+01,  4.7683e+00],
       [-6.3659e-02, -4.2025e+00, -9.3852e+00,  …,  1.6154e+01,
         5.4802e+00, -9.5988e+00],
       …,
       [-1.5312e+01, -1.4793e+01, -6.2053e+00,  …,  1.5291e+00,
        -3.2100e-01, -1.2138e+01],
       [ 1.2831e+01, -2.8055e+00,  6.2049e+00,  …, -9.6234e+00,
        -2.3170e+01, -2.0248e+01],
       [ 8.5889e+00,  1.4089e+01, -2.7157e+00,  …,  1.2957e+01,
         1.5592e+01, -9.3640e+00]],

      [[-2.5304e+00, -1.0193e+01, -8.0289e+00,  …, -5.0835e+00,
         1.7803e+01, -2.5016e+01],
       [ 1.8456e+01,  1.5262e+01,  5.0764e+00,  …,  2.6703e+01,
         7.3676e+01, -5.7064e+00],
       [ 1.3624e+01,  2.6040e+01, -3.8575e+00,  …,  3.5317e+01,
         8.2580e+01,  2.7106e+00],
       …,
       [-3.2110e+01, -8.5863e+00, -1.4313e+01,  …,  4.1767e+01,
         7.5013e+01,  6.0838e+00],
       [-1.9167e+01,  2.8595e+00, -1.5935e+01,  …,  2.9167e+01,
         6.4185e+01,  9.8547e-01],
       [-1.3637e+01,  5.0763e+00, -1.7887e+01,  …, -8.2019e-01,
         3.1124e+01, -1.0624e+01]]]], device='cuda:0',
   grad_fn=<ConvolutionBackward0>)
```

[ ]:

0.0.1 Tiling

[8]:

```
import torch
import math
import torch.nn as nn


a_int = act_int[0, :, :, :]
```

```
nis = a_int.size(1)
njs = a_int.size(2)

w_int = torch.reshape(weight_int, (weight_int.size(0), weight_int.size(1), -1)) # merge ki, kj index to kij

padding = 1
stride = 1
array_size = 16  # change to 16 for hw6

oc = w_int.size(0)
ic = w_int.size(1)

w_int_tile = torch.reshape(w_int, (4, array_size, 4, array_size, -1))
a_int_tile = torch.reshape(a_int, (4, array_size, nis, njs))

nig = range(a_int.size(1))  # ni group [0, 1, ..., 31]
njg = range(a_int.size(2))  # nj group
icg = range(int(w_int.size(1)))  # input channel [0, ..., 63]
ocg = range(int(w_int.size(0)))  # output channel
kijg = range(w_int.size(2))  # [0, .. 8]
ki_dim = int(math.sqrt(w_int.size(2)))  # Kernel's 1 dim size

######## Padding before Convolution #######
# a_pad = torch.zeros(len(icg), len(nig)+padding*2, len(njg)+padding*2).cuda()
# # a_pad.size() = [64, 32+2pad, 32+2pad]
# a_pad[ :, padding:padding+len(nig), padding:padding+len(njg)] = a_int.cuda()
# a_pad = torch.reshape(a_pad, (a_pad.size(0), -1))  ## mergin ni and nj index into nij
# # a_pad.size() = [64, (32+2pad)*(32+2pad)]

# Padding
a_pad_tile = torch.zeros(a_int_tile.size(0), a_int_tile.size(1), nis + padding * 2, njs + padding * 2).cuda()
a_pad_tile[:, :, padding:padding + len(nig), padding:padding + len(njg)] = a_int_tile.cuda()
a_pad_tile = torch.reshape(a_pad_tile, (a_pad_tile.size(0), a_pad_tile.size(1), -1))

# Matrix Mult
p_nijg = range(a_pad_tile.size(-1))
psum_tile = torch.zeros(4, array_size, 4, len(p_nijg), len(kijg)).cuda()

for kij in kijg:
    for out_tile in range(4):
        for in_tile in range(4):
            for nij in p_nijg:
```

```
                m_tile = nn.Linear(array_size, array_size, bias=False)
                m_tile.weight = torch.nn.Parameter(w_int_tile[out_tile, :,␣
↪in_tile, :, kij])
                psum_tile[out_tile, :, in_tile, nij, kij] =␣
↪m_tile(a_pad_tile[in_tile, :, nij]).cuda()

psum = torch.reshape(psum_tile, (oc, 4, len(p_nijg), len(kijg)))

# Output dimension
a_pad_ni_dim = int(math.sqrt(len(p_nijg)))
o_ni_dim = int((a_pad_ni_dim - (ki_dim - 1) - 1) / stride + 1)
o_nijg = range(o_ni_dim**2)
out = torch.zeros(oc, len(o_nijg)).cuda()

# SFP Acc
for o_nij in o_nijg:
    for in_tile in range(4):
        for kij in kijg:
            nij_ = int(o_nij / o_ni_dim) * a_pad_ni_dim + o_nij % o_ni_dim +␣
↪int(kij / ki_dim) * a_pad_ni_dim + kij % ki_dim
            out[:, o_nij] = out[:, o_nij] + psum[:, in_tile, nij_, kij]
```

[9]:
```
out_2D = torch.reshape(out, (out.size(0), o_ni_dim, -1)) # nij -> ni & nj
difference = (out_2D - output_int[0,:,:,:])
print(difference.abs().sum())
print(difference.sum())
```

```
tensor(3.6061, device='cuda:0', grad_fn=<SumBackward0>)
tensor(-0.1326, device='cuda:0', grad_fn=<SumBackward0>)
```

[10]:
```
output_int[0,:,:,:]
```

[10]:
```
tensor([[[  -35.0000,   330.0000,   153.0001,  …,   180.0000,
            326.0000,    84.0000],
         [ -159.0000,   507.0000,   379.0000,  …,   493.0001,
            732.9999,   376.0000],
         [ -230.0000,   397.0000,   357.0001,  …,   466.0000,
            738.0001,   354.0000],
         …,
         [  467.9999,  -119.0001,   219.0000,  …,    37.0001,
           -359.0001,  -151.0000],
         [  358.0000,  -224.0000,   139.0000,  …,   -62.0000,
           -492.0000,  -240.0000],
         [  422.0000,   129.0000,   371.0000,  …,   279.0000,
            -99.0000,     2.0000]],

        [[ -363.9998,  -580.9999,  -647.0000,  …,  -816.0000,
```

```
  -850.0000,  -632.0000],
 [ -509.9999,  -661.9999,  -635.0000,  …,  -424.0000,
  -286.0000,  -127.0001],
 [ -391.0001,  -509.0001,  -637.0001,  …,  -454.0001,
  -527.0000,  -360.0001],
 …,
 [  -65.0001,  -296.9999,  -482.9999,  …,    10.9999,
    52.0000,  -115.0000],
 [ -172.0000,  -462.0001,  -375.0001,  …,  -544.0000,
  -539.0000,  -456.0000],
 [ -390.0000,  -776.0000,  -858.9999,  …,  -991.0000,
  -681.0000,  -359.0000]],

[[ -410.9999,  -489.9999,  -496.9999,  …,  -499.9999,
  -508.0000,  -422.0000],
 [ -694.0001,  -920.0001, -1101.0000,  …,  -787.0002,
  -709.9999,  -742.0000],
 [ -366.0002,  -430.0002,  -570.0001,  …,  -509.0002,
  -491.0000,  -583.0000],
 …,
 [ -492.9999,  -450.9998,  -335.9999,  …,  -142.0000,
  -240.0000,  -599.0000],
 [ -371.0000,  -170.0000,  -108.0000,  …,  -246.0000,
    22.0000,  -311.9999],
 [ -479.0000,  -672.0000,  -729.0000,  …,  -691.0000,
  -501.9999,  -522.9999]],


…,


[[ -375.0000,  -723.9999,  -356.0001,  …,  -375.0000,
  -412.0000,   -48.0000],
 [ -412.0000,  -993.9999,  -397.0002,  …,  -357.9999,
  -413.9999,   135.9999],
 [ -419.0000, -1037.0000,  -536.0001,  …,  -517.9999,
  -661.0000,    29.0000],
 …,
 [ -242.0001,   978.9999,  -209.0001,  …,    82.0000,
   251.0000,   -12.0000],
 [ -315.0000,   909.9999,  -368.0000,  …,  -134.0000,
   381.0000,  -269.0000],
 [ -101.0000,   759.0000,  -116.0000,  …,    84.0000,
   376.0000,   -81.0000]],

[[   17.0001,    61.0000,    -3.9999,  …,  -143.0000,
  -197.0000,  -312.0000],
 [  297.0002,   556.0000,   609.0001,  …,   942.0001,
   948.0000,   459.0000],
```

```
 [  -12.9999,   129.0001,   120.0000,  …,   214.0000,
    364.9999,   177.0000],
 …,
 [  196.0000,  -169.9999,  -389.9999,  …,   143.0000,
   -270.0000,  -359.0000],
 [ -104.0000,  -404.0000,  -526.0000,  …,  -505.0001,
   -551.0000,  -503.0000],
 [  367.9999,   230.9999,   283.0000,  …,    72.0000,
    154.0000,  -122.0000]],

[[  -41.0001,   -76.9996,  -222.0000,  …,   -74.0000,
    304.0001,   -20.0000],
 [  153.0000,   148.0002,  -129.9998,  …,   140.0000,
    763.9999,   128.0000],
 [   82.9999,   158.0002,  -198.9999,  …,    81.9999,
    704.0001,   206.0000],
 …,
 [ -265.9998,   405.0000,   217.9999,  …,   650.0001,
    832.9999,  -360.9999],
 [ -332.0000,   243.0000,   392.0000,  …,   401.0000,
    631.0000,   125.0000],
 [ -235.0000,   -36.0000,   196.0000,  …,    66.0000,
    180.0000,   237.9999]]], device='cuda:0', grad_fn=<SliceBackward0>)
```