_vgg16_quant_os_prune

December 3, 2024

[1]:

```
import os
import time
import shutil

import torch
import torch.nn as nn

import torchvision
import torchvision.transforms as transforms

import torch.nn.utils.prune as prune

from models import *
from models.prune_util import *

import os
os.environ["CUDA_DEVICE_ORDER"]="PCI_BUS_ID"
os.environ["CUDA_VISIBLE_DEVICES"]="0"

import gc
gc.collect()
torch.cuda.empty_cache()

global best_prec

batch_size = 64
model_name = f"VGG16_new_os_iter_prune_0.78_q"
fdir = 'result/' + model_name
model = VGG16()
os_prune_vgg16(model, 0.78)
checkpoint = torch.load(f"result/VGG16_new_os_iter_prune_0.78/model_best.pth.
 ↪tar")
model.load_state_dict(checkpoint['state_dict'])
model.cuda()

device = torch.device("cuda")
```

```
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)


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)


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

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

    model.train()

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

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

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

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

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


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

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

    # switch to evaluate mode
    model.eval()

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

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

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

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

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

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


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

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

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


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

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

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


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


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

def train_model(model, fdir, criterion, optimizer, epochs):
    os.makedirs(fdir, exist_ok=True)

    best_prec = 0

    #model = nn.DataParallel(model).cuda()
    model.cuda()
    criterion = criterion.cuda()
    #cudnn.benchmark = True


    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)

def val_model(model):

    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)))
```

```
Pruning 50 ic-slices out of 64 ic-slices (78.1% pruned)
Pruning 50 ic-slices out of 64 ic-slices (78.1% pruned)
Pruning 100 ic-slices out of 128 ic-slices (78.1% pruned)
Pruning 100 ic-slices out of 128 ic-slices (78.1% pruned)
Pruning 200 ic-slices out of 256 ic-slices (78.1% pruned)
Pruning 200 ic-slices out of 256 ic-slices (78.1% pruned)
Pruning 200 ic-slices out of 256 ic-slices (78.1% pruned)
Pruning 399 ic-slices out of 512 ic-slices (77.9% pruned)
Pruning 399 ic-slices out of 512 ic-slices (77.9% pruned)
Pruning 399 ic-slices out of 512 ic-slices (77.9% pruned)
Pruning 399 ic-slices out of 512 ic-slices (77.9% pruned)
Pruning 399 ic-slices out of 512 ic-slices (77.9% pruned)
```

```
/tmp/ipykernel_63507/652440560.py:31: FutureWarning: You are using `torch.load`
with `weights_only=False` (the current default value), which uses the default
pickle module implicitly. It is possible to construct malicious pickle data
which will execute arbitrary code during unpickling (See
https://github.com/pytorch/pytorch/blob/main/SECURITY.md#untrusted-models for
more details). In a future release, the default value for `weights_only` will be
flipped to `True`. This limits the functions that could be executed during
unpickling. Arbitrary objects will no longer be allowed to be loaded via this
mode unless they are explicitly allowlisted by the user via
`torch.serialization.add_safe_globals`. We recommend you start setting
`weights_only=True` for any use case where you don't have full control of the
loaded file. Please open an issue on GitHub for any issues related to this
experimental feature.
  checkpoint =
torch.load(f"result/VGG16_new_os_iter_prune_0.78/model_best.pth.tar")
```

```
Files already downloaded and verified
Files already downloaded and verified
```

[2]:

```
val_model(model)
```

```
Test set: Accuracy: 8668/10000 (87%)
```

[3]:

```
quantize_pruned(model)

val_model(model)
```

```
Test set: Accuracy: 1003/10000 (10%)
```

[4]:

```
lr = 3e-3

criterion = nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(model.parameters(), lr=lr)
train_model(model, fdir, criterion, optimizer, 100)
```

```
Epoch: [0][0/782]       Time 0.173 (0.173)      Data 0.020 (0.020)      Loss
1.4387 (1.4387)    Prec 56.250% (56.250%)
Epoch: [0][100/782]     Time 0.067 (0.068)      Data 0.012 (0.016)      Loss
0.7347 (0.9887)    Prec 73.438% (67.141%)
Epoch: [0][200/782]     Time 0.067 (0.068)      Data 0.014 (0.017)      Loss
0.6829 (0.8627)    Prec 81.250% (71.152%)
Epoch: [0][300/782]     Time 0.058 (0.067)      Data 0.023 (0.017)      Loss
0.7700 (0.7957)    Prec 71.875% (73.406%)
Epoch: [0][400/782]     Time 0.066 (0.067)      Data 0.013 (0.017)      Loss
0.7631 (0.7527)    Prec 65.625% (74.716%)
Epoch: [0][500/782]     Time 0.067 (0.067)      Data 0.013 (0.016)      Loss
0.6015 (0.7195)    Prec 78.125% (75.646%)
Epoch: [0][600/782]     Time 0.067 (0.067)      Data 0.013 (0.016)      Loss
0.5384 (0.6885)    Prec 79.688% (76.630%)
Epoch: [0][700/782]     Time 0.066 (0.067)      Data 0.012 (0.016)      Loss
```

```
0.6657 (0.6642)    Prec 81.250% (77.441%)
Validation starts
Test: [0/157]   Time 0.033 (0.033)      Loss 0.6285 (0.6285)    Prec 81.250%
(81.250%)
Test: [100/157] Time 0.033 (0.033)      Loss 0.6762 (0.5937)    Prec 79.688%
(80.554%)
 * Prec 80.700%
best acc: 80.700000
Epoch: [1][0/782]       Time 0.055 (0.055)      Data 0.020 (0.020)      Loss
0.5163 (0.5163)    Prec 81.250% (81.250%)
Epoch: [1][100/782]     Time 0.066 (0.067)      Data 0.012 (0.013)      Loss
0.3526 (0.5094)    Prec 87.500% (82.457%)
Epoch: [1][200/782]     Time 0.067 (0.067)      Data 0.017 (0.014)      Loss
0.2990 (0.5078)    Prec 90.625% (82.478%)
Epoch: [1][300/782]     Time 0.067 (0.067)      Data 0.012 (0.015)      Loss
0.5374 (0.5051)    Prec 79.688% (82.729%)
Epoch: [1][400/782]     Time 0.068 (0.067)      Data 0.013 (0.015)      Loss
0.3316 (0.5044)    Prec 87.500% (82.746%)
Epoch: [1][500/782]     Time 0.066 (0.067)      Data 0.012 (0.015)      Loss
0.5308 (0.5028)    Prec 82.812% (82.850%)
Epoch: [1][600/782]     Time 0.066 (0.067)      Data 0.012 (0.015)      Loss
0.4872 (0.4976)    Prec 81.250% (83.049%)
Epoch: [1][700/782]     Time 0.067 (0.067)      Data 0.012 (0.015)      Loss
0.2464 (0.4921)    Prec 93.750% (83.214%)
Validation starts
Test: [0/157]   Time 0.032 (0.032)      Loss 0.6379 (0.6379)    Prec 76.562%
(76.562%)
Test: [100/157] Time 0.031 (0.032)      Loss 0.6368 (0.5566)    Prec 78.125%
(81.683%)
 * Prec 81.380%
best acc: 81.380000
Epoch: [2][0/782]       Time 0.054 (0.054)      Data 0.017 (0.017)      Loss
0.4469 (0.4469)    Prec 85.938% (85.938%)
Epoch: [2][100/782]     Time 0.067 (0.067)      Data 0.012 (0.015)      Loss
0.5799 (0.4458)    Prec 81.250% (84.437%)
Epoch: [2][200/782]     Time 0.066 (0.067)      Data 0.012 (0.015)      Loss
0.2558 (0.4451)    Prec 87.500% (84.569%)
Epoch: [2][300/782]     Time 0.081 (0.067)      Data 0.025 (0.016)      Loss
0.5199 (0.4441)    Prec 81.250% (84.697%)
Epoch: [2][400/782]     Time 0.067 (0.067)      Data 0.012 (0.016)      Loss
0.3393 (0.4410)    Prec 82.812% (84.749%)
Epoch: [2][500/782]     Time 0.067 (0.067)      Data 0.025 (0.016)      Loss
0.4661 (0.4401)    Prec 82.812% (84.740%)
Epoch: [2][600/782]     Time 0.066 (0.067)      Data 0.012 (0.016)      Loss
0.4827 (0.4415)    Prec 84.375% (84.726%)
Epoch: [2][700/782]     Time 0.066 (0.067)      Data 0.014 (0.015)      Loss
0.4847 (0.4399)    Prec 87.500% (84.692%)
Validation starts
```

```
Test: [0/157]   Time 0.042 (0.042)      Loss 0.6080 (0.6080)    Prec 78.125%
(78.125%)
Test: [100/157] Time 0.031 (0.031)      Loss 0.5666 (0.5259)    Prec 82.812%
(82.317%)
 * Prec 82.520%
best acc: 82.520000
Epoch: [3][0/782]       Time 0.067 (0.067)      Data 0.021 (0.021)      Loss
0.4982 (0.4982)    Prec 78.125% (78.125%)
Epoch: [3][100/782]     Time 0.067 (0.067)      Data 0.012 (0.014)      Loss
0.4800 (0.4166)    Prec 78.125% (85.319%)
Epoch: [3][200/782]     Time 0.067 (0.067)      Data 0.012 (0.015)      Loss
0.3631 (0.4093)    Prec 84.375% (85.634%)
Epoch: [3][300/782]     Time 0.066 (0.067)      Data 0.025 (0.015)      Loss
0.3002 (0.4097)    Prec 90.625% (85.803%)
Epoch: [3][400/782]     Time 0.066 (0.067)      Data 0.012 (0.015)      Loss
0.4937 (0.4099)    Prec 84.375% (85.887%)
Epoch: [3][500/782]     Time 0.069 (0.067)      Data 0.012 (0.015)      Loss
0.3281 (0.4134)    Prec 87.500% (85.841%)
Epoch: [3][600/782]     Time 0.067 (0.067)      Data 0.012 (0.015)      Loss
0.2723 (0.4109)    Prec 92.188% (85.940%)
Epoch: [3][700/782]     Time 0.067 (0.067)      Data 0.017 (0.015)      Loss
0.4651 (0.4096)    Prec 87.500% (86.009%)
Validation starts
Test: [0/157]   Time 0.041 (0.041)      Loss 0.6682 (0.6682)    Prec 76.562%
(76.562%)
Test: [100/157] Time 0.031 (0.031)      Loss 0.5772 (0.4791)    Prec 84.375%
(84.421%)
 * Prec 84.230%
best acc: 84.230000
Epoch: [4][0/782]       Time 0.068 (0.068)      Data 0.020 (0.020)      Loss
0.2113 (0.2113)    Prec 93.750% (93.750%)
Epoch: [4][100/782]     Time 0.066 (0.067)      Data 0.025 (0.016)      Loss
0.4706 (0.3761)    Prec 85.938% (86.989%)
Epoch: [4][200/782]     Time 0.066 (0.067)      Data 0.025 (0.016)      Loss
0.3334 (0.3877)    Prec 92.188% (86.707%)
Epoch: [4][300/782]     Time 0.066 (0.067)      Data 0.012 (0.016)      Loss
0.6545 (0.3896)    Prec 75.000% (86.534%)
Epoch: [4][400/782]     Time 0.079 (0.067)      Data 0.013 (0.016)      Loss
0.4184 (0.3925)    Prec 85.938% (86.366%)
Epoch: [4][500/782]     Time 0.080 (0.067)      Data 0.013 (0.016)      Loss
0.4509 (0.3892)    Prec 89.062% (86.524%)
Epoch: [4][600/782]     Time 0.067 (0.067)      Data 0.023 (0.016)      Loss
0.4059 (0.3869)    Prec 84.375% (86.556%)
Epoch: [4][700/782]     Time 0.059 (0.067)      Data 0.012 (0.016)      Loss
0.3918 (0.3874)    Prec 84.375% (86.577%)
Validation starts
Test: [0/157]   Time 0.042 (0.042)      Loss 0.7254 (0.7254)    Prec 81.250%
(81.250%)
```

```
Test: [100/157] Time 0.032 (0.032)      Loss 0.4934 (0.5331)    Prec 84.375%
(82.936%)
 * Prec 83.410%
best acc: 84.230000
Epoch: [5][0/782]       Time 0.063 (0.063)      Data 0.018 (0.018)      Loss
0.5463 (0.5463)    Prec 84.375% (84.375%)
Epoch: [5][100/782]     Time 0.080 (0.069)      Data 0.020 (0.016)      Loss
0.3303 (0.3789)    Prec 87.500% (87.113%)
Epoch: [5][200/782]     Time 0.076 (0.070)      Data 0.012 (0.016)      Loss
0.3625 (0.3774)    Prec 85.938% (87.306%)
Epoch: [5][300/782]     Time 0.069 (0.069)      Data 0.014 (0.016)      Loss
0.2467 (0.3749)    Prec 92.188% (87.230%)
Epoch: [5][400/782]     Time 0.067 (0.070)      Data 0.014 (0.016)      Loss
0.2644 (0.3734)    Prec 92.188% (87.204%)
Epoch: [5][500/782]     Time 0.066 (0.070)      Data 0.014 (0.017)      Loss
0.2668 (0.3724)    Prec 90.625% (87.229%)
Epoch: [5][600/782]     Time 0.067 (0.070)      Data 0.018 (0.017)      Loss
0.3488 (0.3738)    Prec 89.062% (87.162%)
Epoch: [5][700/782]     Time 0.068 (0.070)      Data 0.013 (0.016)      Loss
0.4182 (0.3713)    Prec 82.812% (87.293%)
Validation starts
Test: [0/157]   Time 0.036 (0.036)      Loss 0.5558 (0.5558)    Prec 82.812%
(82.812%)
Test: [100/157] Time 0.032 (0.033)      Loss 0.5938 (0.5169)    Prec 78.125%
(83.014%)
 * Prec 83.530%
best acc: 84.230000
Epoch: [6][0/782]       Time 0.075 (0.075)      Data 0.019 (0.019)      Loss
0.3723 (0.3723)    Prec 82.812% (82.812%)
Epoch: [6][100/782]     Time 0.066 (0.069)      Data 0.012 (0.017)      Loss
0.3776 (0.3500)    Prec 85.938% (87.701%)
Epoch: [6][200/782]     Time 0.066 (0.069)      Data 0.012 (0.017)      Loss
0.2301 (0.3501)    Prec 92.188% (87.803%)
Epoch: [6][300/782]     Time 0.065 (0.069)      Data 0.012 (0.016)      Loss
0.3576 (0.3555)    Prec 87.500% (87.443%)
Epoch: [6][400/782]     Time 0.068 (0.069)      Data 0.012 (0.016)      Loss
0.3940 (0.3597)    Prec 89.062% (87.329%)
Epoch: [6][500/782]     Time 0.066 (0.069)      Data 0.012 (0.016)      Loss
0.4635 (0.3596)    Prec 82.812% (87.363%)
Epoch: [6][600/782]     Time 0.080 (0.069)      Data 0.012 (0.016)      Loss
0.4442 (0.3566)    Prec 85.938% (87.458%)
Epoch: [6][700/782]     Time 0.072 (0.069)      Data 0.013 (0.016)      Loss
0.3136 (0.3573)    Prec 85.938% (87.418%)
Validation starts
Test: [0/157]   Time 0.040 (0.040)      Loss 0.5201 (0.5201)    Prec 82.812%
(82.812%)
Test: [100/157] Time 0.036 (0.033)      Loss 0.5104 (0.4709)    Prec 82.812%
(83.942%)
```

```
 * Prec 84.160%
best acc: 84.230000
Epoch: [7][0/782]       Time 0.052 (0.052)      Data 0.017 (0.017)      Loss
0.2722 (0.2722)    Prec 89.062% (89.062%)
Epoch: [7][100/782]     Time 0.065 (0.069)      Data 0.012 (0.012)      Loss
0.3669 (0.3379)    Prec 87.500% (88.181%)
Epoch: [7][200/782]     Time 0.067 (0.069)      Data 0.012 (0.013)      Loss
0.2528 (0.3446)    Prec 92.188% (87.966%)
Epoch: [7][300/782]     Time 0.073 (0.069)      Data 0.012 (0.013)      Loss
0.3744 (0.3413)    Prec 85.938% (87.998%)
Epoch: [7][400/782]     Time 0.070 (0.069)      Data 0.012 (0.013)      Loss
0.2796 (0.3394)    Prec 89.062% (88.010%)
Epoch: [7][500/782]     Time 0.082 (0.069)      Data 0.012 (0.013)      Loss
0.3060 (0.3391)    Prec 84.375% (88.043%)
Epoch: [7][600/782]     Time 0.067 (0.069)      Data 0.013 (0.013)      Loss
0.4865 (0.3456)    Prec 85.938% (87.887%)
Epoch: [7][700/782]     Time 0.066 (0.069)      Data 0.012 (0.012)      Loss
0.3364 (0.3471)    Prec 90.625% (87.814%)
Validation starts
Test: [0/157]   Time 0.044 (0.044)      Loss 0.5684 (0.5684)    Prec 82.812%
(82.812%)
Test: [100/157] Time 0.032 (0.032)      Loss 0.6045 (0.4575)    Prec 82.812%
(85.675%)
 * Prec 85.520%
best acc: 85.520000
Epoch: [8][0/782]       Time 0.053 (0.053)      Data 0.018 (0.018)      Loss
0.3203 (0.3203)    Prec 89.062% (89.062%)
Epoch: [8][100/782]     Time 0.078 (0.067)      Data 0.012 (0.013)      Loss
0.1750 (0.3321)    Prec 95.312% (88.846%)
Epoch: [8][200/782]     Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.2982 (0.3321)    Prec 87.500% (88.433%)
Epoch: [8][300/782]     Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.3502 (0.3343)    Prec 85.938% (88.367%)
Epoch: [8][400/782]     Time 0.082 (0.067)      Data 0.012 (0.013)      Loss
0.2993 (0.3358)    Prec 93.750% (88.221%)
Epoch: [8][500/782]     Time 0.067 (0.067)      Data 0.012 (0.012)      Loss
0.4060 (0.3402)    Prec 85.938% (88.061%)
Epoch: [8][600/782]     Time 0.068 (0.067)      Data 0.012 (0.012)      Loss
0.3750 (0.3392)    Prec 87.500% (88.127%)
Epoch: [8][700/782]     Time 0.067 (0.067)      Data 0.012 (0.012)      Loss
0.3071 (0.3409)    Prec 89.062% (88.109%)
Validation starts
Test: [0/157]   Time 0.043 (0.043)      Loss 0.5354 (0.5354)    Prec 84.375%
(84.375%)
Test: [100/157] Time 0.031 (0.032)      Loss 0.5039 (0.4467)    Prec 85.938%
(85.458%)
 * Prec 85.320%
best acc: 85.520000
```

```
Epoch: [9][0/782]       Time 0.048 (0.048)      Data 0.014 (0.014)      Loss
0.2906 (0.2906)    Prec 84.375% (84.375%)
Epoch: [9][100/782]     Time 0.067 (0.067)      Data 0.013 (0.013)      Loss
0.4879 (0.3340)    Prec 87.500% (88.444%)
Epoch: [9][200/782]     Time 0.066 (0.067)      Data 0.012 (0.013)      Loss
0.2935 (0.3378)    Prec 90.625% (88.262%)
Epoch: [9][300/782]     Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.2814 (0.3366)    Prec 89.062% (88.273%)
Epoch: [9][400/782]     Time 0.067 (0.067)      Data 0.013 (0.013)      Loss
0.2189 (0.3321)    Prec 93.750% (88.303%)
Epoch: [9][500/782]     Time 0.056 (0.067)      Data 0.012 (0.013)      Loss
0.2481 (0.3316)    Prec 92.188% (88.426%)
Epoch: [9][600/782]     Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.4197 (0.3352)    Prec 87.500% (88.353%)
Epoch: [9][700/782]     Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.2051 (0.3338)    Prec 92.188% (88.403%)
Validation starts
Test: [0/157]   Time 0.036 (0.036)      Loss 0.4923 (0.4923)    Prec 82.812%
(82.812%)
Test: [100/157] Time 0.031 (0.031)      Loss 0.6052 (0.4344)    Prec 79.688%
(85.907%)
 * Prec 85.900%
best acc: 85.900000
Epoch: [10][0/782]      Time 0.051 (0.051)      Data 0.018 (0.018)      Loss
0.4092 (0.4092)    Prec 87.500% (87.500%)
Epoch: [10][100/782]    Time 0.059 (0.067)      Data 0.013 (0.015)      Loss
0.2738 (0.3129)    Prec 90.625% (89.016%)
Epoch: [10][200/782]    Time 0.067 (0.067)      Data 0.024 (0.014)      Loss
0.2670 (0.3146)    Prec 90.625% (88.822%)
Epoch: [10][300/782]    Time 0.081 (0.067)      Data 0.012 (0.014)      Loss
0.4313 (0.3151)    Prec 87.500% (89.052%)
Epoch: [10][400/782]    Time 0.066 (0.067)      Data 0.013 (0.014)      Loss
0.4553 (0.3182)    Prec 85.938% (88.988%)
Epoch: [10][500/782]    Time 0.076 (0.067)      Data 0.012 (0.014)      Loss
0.4165 (0.3217)    Prec 84.375% (88.863%)
Epoch: [10][600/782]    Time 0.081 (0.067)      Data 0.012 (0.014)      Loss
0.2723 (0.3241)    Prec 90.625% (88.738%)
Epoch: [10][700/782]    Time 0.066 (0.067)      Data 0.012 (0.014)      Loss
0.2164 (0.3229)    Prec 90.625% (88.802%)
Validation starts
Test: [0/157]   Time 0.041 (0.041)      Loss 0.6895 (0.6895)    Prec 78.125%
(78.125%)
Test: [100/157] Time 0.031 (0.032)      Loss 0.6020 (0.4593)    Prec 81.250%
(85.118%)
 * Prec 84.960%
best acc: 85.900000
Epoch: [11][0/782]      Time 0.055 (0.055)      Data 0.021 (0.021)      Loss
0.4376 (0.4376)    Prec 84.375% (84.375%)
```

```
Epoch: [11][100/782]    Time 0.067 (0.067)      Data 0.012 (0.014)      Loss
0.2769 (0.3081)    Prec 90.625% (89.403%)
Epoch: [11][200/782]    Time 0.056 (0.067)      Data 0.021 (0.014)      Loss
0.4523 (0.3143)    Prec 87.500% (89.055%)
Epoch: [11][300/782]    Time 0.067 (0.067)      Data 0.013 (0.015)      Loss
0.3058 (0.3115)    Prec 90.625% (89.120%)
Epoch: [11][400/782]    Time 0.081 (0.067)      Data 0.025 (0.015)      Loss
0.1639 (0.3145)    Prec 93.750% (89.062%)
Epoch: [11][500/782]    Time 0.067 (0.067)      Data 0.013 (0.015)      Loss
0.3476 (0.3127)    Prec 87.500% (89.134%)
Epoch: [11][600/782]    Time 0.067 (0.067)      Data 0.025 (0.015)      Loss
0.2169 (0.3139)    Prec 93.750% (89.096%)
Epoch: [11][700/782]    Time 0.066 (0.067)      Data 0.012 (0.015)      Loss
0.3339 (0.3164)    Prec 84.375% (88.991%)
Validation starts
Test: [0/157]   Time 0.042 (0.042)      Loss 0.5320 (0.5320)    Prec 82.812%
(82.812%)
Test: [100/157] Time 0.031 (0.032)      Loss 0.5328 (0.4409)    Prec 79.688%
(85.582%)
 * Prec 85.550%
best acc: 85.900000
Epoch: [12][0/782]      Time 0.064 (0.064)      Data 0.015 (0.015)      Loss
0.2792 (0.2792)    Prec 92.188% (92.188%)
Epoch: [12][100/782]    Time 0.061 (0.068)      Data 0.014 (0.013)      Loss
0.3029 (0.3093)    Prec 89.062% (89.449%)
Epoch: [12][200/782]    Time 0.071 (0.069)      Data 0.013 (0.013)      Loss
0.3185 (0.3095)    Prec 84.375% (89.405%)
Epoch: [12][300/782]    Time 0.066 (0.069)      Data 0.012 (0.013)      Loss
0.2357 (0.3083)    Prec 93.750% (89.348%)
Epoch: [12][400/782]    Time 0.066 (0.068)      Data 0.013 (0.013)      Loss
0.2761 (0.3074)    Prec 87.500% (89.347%)
Epoch: [12][500/782]    Time 0.066 (0.068)      Data 0.013 (0.013)      Loss
0.3031 (0.3112)    Prec 89.062% (89.250%)
Epoch: [12][600/782]    Time 0.066 (0.067)      Data 0.012 (0.013)      Loss
0.2406 (0.3087)    Prec 85.938% (89.346%)
Epoch: [12][700/782]    Time 0.049 (0.067)      Data 0.012 (0.013)      Loss
0.2793 (0.3112)    Prec 90.625% (89.259%)
Validation starts
Test: [0/157]   Time 0.031 (0.031)      Loss 0.4463 (0.4463)    Prec 85.938%
(85.938%)
Test: [100/157] Time 0.031 (0.031)      Loss 0.4596 (0.4409)    Prec 84.375%
(85.288%)
 * Prec 85.440%
best acc: 85.900000
Epoch: [13][0/782]      Time 0.056 (0.056)      Data 0.018 (0.018)      Loss
0.2230 (0.2230)    Prec 92.188% (92.188%)
Epoch: [13][100/782]    Time 0.066 (0.066)      Data 0.012 (0.013)      Loss
0.3019 (0.2986)    Prec 90.625% (89.480%)
```

```
Epoch: [13][200/782]    Time 0.066 (0.066)      Data 0.012 (0.012)      Loss
0.2039 (0.2987)    Prec 90.625% (89.560%)
Epoch: [13][300/782]    Time 0.066 (0.066)      Data 0.012 (0.012)      Loss
0.4097 (0.2987)    Prec 85.938% (89.540%)
Epoch: [13][400/782]    Time 0.055 (0.066)      Data 0.012 (0.012)      Loss
0.3254 (0.3022)    Prec 90.625% (89.491%)
Epoch: [13][500/782]    Time 0.067 (0.066)      Data 0.012 (0.012)      Loss
0.2570 (0.3011)    Prec 87.500% (89.518%)
Epoch: [13][600/782]    Time 0.051 (0.066)      Data 0.012 (0.012)      Loss
0.4356 (0.2985)    Prec 82.812% (89.556%)
Epoch: [13][700/782]    Time 0.073 (0.067)      Data 0.012 (0.013)      Loss
0.3405 (0.3017)    Prec 85.938% (89.395%)
Validation starts
Test: [0/157]   Time 0.044 (0.044)      Loss 0.4922 (0.4922)    Prec 85.938%
(85.938%)
Test: [100/157] Time 0.032 (0.033)      Loss 0.6277 (0.4317)    Prec 84.375%
(86.077%)
 * Prec 86.190%
best acc: 86.190000
Epoch: [14][0/782]      Time 0.063 (0.063)      Data 0.018 (0.018)      Loss
0.2245 (0.2245)    Prec 96.875% (96.875%)
Epoch: [14][100/782]    Time 0.094 (0.070)      Data 0.026 (0.014)      Loss
0.5881 (0.2888)    Prec 89.062% (90.362%)
Epoch: [14][200/782]    Time 0.067 (0.069)      Data 0.016 (0.014)      Loss
0.2402 (0.2944)    Prec 92.188% (89.941%)
Epoch: [14][300/782]    Time 0.071 (0.070)      Data 0.013 (0.013)      Loss
0.1407 (0.2989)    Prec 95.312% (89.691%)
Epoch: [14][400/782]    Time 0.079 (0.070)      Data 0.012 (0.013)      Loss
0.2708 (0.3028)    Prec 85.938% (89.499%)
Epoch: [14][500/782]    Time 0.067 (0.070)      Data 0.013 (0.014)      Loss
0.3864 (0.3032)    Prec 84.375% (89.496%)
Epoch: [14][600/782]    Time 0.070 (0.070)      Data 0.013 (0.014)      Loss
0.3263 (0.3005)    Prec 84.375% (89.559%)
Epoch: [14][700/782]    Time 0.067 (0.069)      Data 0.024 (0.014)      Loss
0.3595 (0.3017)    Prec 85.938% (89.522%)
Validation starts
Test: [0/157]   Time 0.040 (0.040)      Loss 0.4792 (0.4792)    Prec 85.938%
(85.938%)
Test: [100/157] Time 0.032 (0.032)      Loss 0.6689 (0.4263)    Prec 79.688%
(86.231%)
 * Prec 86.260%
best acc: 86.260000
Epoch: [15][0/782]      Time 0.053 (0.053)      Data 0.019 (0.019)      Loss
0.3615 (0.3615)    Prec 85.938% (85.938%)
Epoch: [15][100/782]    Time 0.070 (0.069)      Data 0.013 (0.013)      Loss
0.2011 (0.3015)    Prec 93.750% (89.295%)
Epoch: [15][200/782]    Time 0.071 (0.069)      Data 0.012 (0.013)      Loss
0.1681 (0.2926)    Prec 95.312% (89.731%)
```

```
Epoch: [15][300/782]    Time 0.068 (0.069)      Data 0.012 (0.013)      Loss
0.4032 (0.2951)    Prec 84.375% (89.675%)
Epoch: [15][400/782]    Time 0.069 (0.069)      Data 0.012 (0.013)      Loss
0.2882 (0.2929)    Prec 89.062% (89.717%)
Epoch: [15][500/782]    Time 0.066 (0.069)      Data 0.012 (0.013)      Loss
0.1945 (0.2945)    Prec 95.312% (89.643%)
Epoch: [15][600/782]    Time 0.066 (0.069)      Data 0.013 (0.013)      Loss
0.2649 (0.2967)    Prec 89.062% (89.707%)
Epoch: [15][700/782]    Time 0.082 (0.068)      Data 0.012 (0.013)      Loss
0.3953 (0.2984)    Prec 85.938% (89.651%)
Validation starts
Test: [0/157]   Time 0.037 (0.037)      Loss 0.4754 (0.4754)    Prec 87.500%
(87.500%)
Test: [100/157] Time 0.031 (0.031)      Loss 0.5311 (0.4543)    Prec 82.812%
(85.473%)
 * Prec 85.490%
best acc: 86.260000
Epoch: [16][0/782]      Time 0.072 (0.072)      Data 0.019 (0.019)      Loss
0.3101 (0.3101)    Prec 87.500% (87.500%)
Epoch: [16][100/782]    Time 0.065 (0.067)      Data 0.024 (0.013)      Loss
0.2178 (0.2795)    Prec 95.312% (90.455%)
Epoch: [16][200/782]    Time 0.067 (0.067)      Data 0.012 (0.014)      Loss
0.3059 (0.2810)    Prec 87.500% (90.275%)
Epoch: [16][300/782]    Time 0.065 (0.067)      Data 0.013 (0.013)      Loss
0.4226 (0.2852)    Prec 85.938% (90.049%)
Epoch: [16][400/782]    Time 0.056 (0.067)      Data 0.012 (0.013)      Loss
0.1969 (0.2862)    Prec 93.750% (89.994%)
Epoch: [16][500/782]    Time 0.066 (0.067)      Data 0.012 (0.013)      Loss
0.2077 (0.2890)    Prec 93.750% (89.770%)
Epoch: [16][600/782]    Time 0.089 (0.067)      Data 0.013 (0.013)      Loss
0.2531 (0.2911)    Prec 90.625% (89.751%)
Epoch: [16][700/782]    Time 0.078 (0.067)      Data 0.012 (0.013)      Loss
0.3320 (0.2897)    Prec 89.062% (89.727%)
Validation starts
Test: [0/157]   Time 0.045 (0.045)      Loss 0.4519 (0.4519)    Prec 82.812%
(82.812%)
Test: [100/157] Time 0.031 (0.031)      Loss 0.7962 (0.4526)    Prec 81.250%
(85.752%)
 * Prec 85.980%
best acc: 86.260000
Epoch: [17][0/782]      Time 0.051 (0.051)      Data 0.018 (0.018)      Loss
0.4575 (0.4575)    Prec 81.250% (81.250%)
Epoch: [17][100/782]    Time 0.057 (0.067)      Data 0.012 (0.014)      Loss
0.4422 (0.2696)    Prec 84.375% (90.285%)
Epoch: [17][200/782]    Time 0.067 (0.067)      Data 0.014 (0.014)      Loss
0.2946 (0.2926)    Prec 87.500% (89.544%)
Epoch: [17][300/782]    Time 0.065 (0.067)      Data 0.012 (0.014)      Loss
0.3099 (0.2916)    Prec 89.062% (89.665%)
```

```
Epoch: [17][400/782]    Time 0.046 (0.067)      Data 0.012 (0.013)      Loss
0.2977 (0.2896)    Prec 90.625% (89.678%)
Epoch: [17][500/782]    Time 0.055 (0.067)      Data 0.012 (0.013)      Loss
0.2485 (0.2915)    Prec 87.500% (89.655%)
Epoch: [17][600/782]    Time 0.068 (0.067)      Data 0.012 (0.013)      Loss
0.1458 (0.2909)    Prec 95.312% (89.666%)
Epoch: [17][700/782]    Time 0.079 (0.068)      Data 0.013 (0.013)      Loss
0.1896 (0.2894)    Prec 93.750% (89.787%)
Validation starts
Test: [0/157]   Time 0.048 (0.048)      Loss 0.5368 (0.5368)    Prec 82.812%
(82.812%)
Test: [100/157] Time 0.031 (0.032)      Loss 0.6371 (0.4314)    Prec 78.125%
(86.170%)
 * Prec 86.280%
best acc: 86.280000
Epoch: [18][0/782]      Time 0.060 (0.060)      Data 0.021 (0.021)      Loss
0.2070 (0.2070)    Prec 90.625% (90.625%)
Epoch: [18][100/782]    Time 0.066 (0.067)      Data 0.026 (0.014)      Loss
0.2068 (0.2741)    Prec 90.625% (90.377%)
Epoch: [18][200/782]    Time 0.065 (0.067)      Data 0.013 (0.014)      Loss
0.1611 (0.2694)    Prec 92.188% (90.516%)
Epoch: [18][300/782]    Time 0.055 (0.067)      Data 0.016 (0.014)      Loss
0.2864 (0.2799)    Prec 87.500% (90.194%)
Epoch: [18][400/782]    Time 0.068 (0.067)      Data 0.014 (0.014)      Loss
0.2174 (0.2837)    Prec 90.625% (90.041%)
Epoch: [18][500/782]    Time 0.066 (0.067)      Data 0.012 (0.015)      Loss
0.6793 (0.2803)    Prec 81.250% (90.204%)
Epoch: [18][600/782]    Time 0.079 (0.067)      Data 0.027 (0.015)      Loss
0.1647 (0.2788)    Prec 96.875% (90.245%)
Epoch: [18][700/782]    Time 0.066 (0.067)      Data 0.017 (0.015)      Loss
0.2261 (0.2780)    Prec 92.188% (90.268%)
Validation starts
Test: [0/157]   Time 0.042 (0.042)      Loss 0.4978 (0.4978)    Prec 89.062%
(89.062%)
Test: [100/157] Time 0.032 (0.032)      Loss 0.5231 (0.4478)    Prec 78.125%
(85.257%)
 * Prec 85.100%
best acc: 86.280000
Epoch: [19][0/782]      Time 0.057 (0.057)      Data 0.021 (0.021)      Loss
0.2057 (0.2057)    Prec 92.188% (92.188%)
Epoch: [19][100/782]    Time 0.047 (0.067)      Data 0.012 (0.013)      Loss
0.3975 (0.2805)    Prec 85.938% (90.300%)
Epoch: [19][200/782]    Time 0.054 (0.067)      Data 0.012 (0.013)      Loss
0.3301 (0.2773)    Prec 89.062% (90.299%)
Epoch: [19][300/782]    Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.1158 (0.2796)    Prec 96.875% (90.163%)
Epoch: [19][400/782]    Time 0.068 (0.067)      Data 0.012 (0.013)      Loss
0.2609 (0.2779)    Prec 92.188% (90.224%)
```

```
Epoch: [19][500/782]    Time 0.066 (0.067)      Data 0.012 (0.013)      Loss
0.3735 (0.2771)    Prec 84.375% (90.226%)
Epoch: [19][600/782]    Time 0.088 (0.067)      Data 0.012 (0.013)      Loss
0.3018 (0.2787)    Prec 90.625% (90.206%)
Epoch: [19][700/782]    Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.3582 (0.2784)    Prec 85.938% (90.204%)
Validation starts
Test: [0/157]   Time 0.050 (0.050)      Loss 0.5622 (0.5622)    Prec 85.938%
(85.938%)
Test: [100/157] Time 0.031 (0.031)      Loss 0.4428 (0.4464)    Prec 84.375%
(85.814%)
 * Prec 85.920%
best acc: 86.280000
Epoch: [20][0/782]      Time 0.065 (0.065)      Data 0.017 (0.017)      Loss
0.2226 (0.2226)    Prec 93.750% (93.750%)
Epoch: [20][100/782]    Time 0.056 (0.068)      Data 0.012 (0.013)      Loss
0.4169 (0.2796)    Prec 90.625% (90.548%)
Epoch: [20][200/782]    Time 0.058 (0.068)      Data 0.012 (0.013)      Loss
0.3933 (0.2739)    Prec 85.938% (90.493%)
Epoch: [20][300/782]    Time 0.067 (0.068)      Data 0.015 (0.013)      Loss
0.3193 (0.2734)    Prec 89.062% (90.480%)
Epoch: [20][400/782]    Time 0.045 (0.068)      Data 0.012 (0.013)      Loss
0.3302 (0.2761)    Prec 87.500% (90.387%)
Epoch: [20][500/782]    Time 0.056 (0.068)      Data 0.012 (0.013)      Loss
0.3016 (0.2791)    Prec 87.500% (90.248%)
Epoch: [20][600/782]    Time 0.067 (0.068)      Data 0.012 (0.013)      Loss
0.3873 (0.2785)    Prec 85.938% (90.271%)
Epoch: [20][700/782]    Time 0.064 (0.068)      Data 0.018 (0.013)      Loss
0.3185 (0.2788)    Prec 85.938% (90.262%)
Validation starts
Test: [0/157]   Time 0.043 (0.043)      Loss 0.4173 (0.4173)    Prec 84.375%
(84.375%)
Test: [100/157] Time 0.031 (0.032)      Loss 0.4313 (0.4032)    Prec 92.188%
(86.726%)
 * Prec 86.850%
best acc: 86.850000
Epoch: [21][0/782]      Time 0.055 (0.055)      Data 0.017 (0.017)      Loss
0.1782 (0.1782)    Prec 93.750% (93.750%)
Epoch: [21][100/782]    Time 0.068 (0.067)      Data 0.012 (0.013)      Loss
0.1685 (0.2481)    Prec 92.188% (90.950%)
Epoch: [21][200/782]    Time 0.066 (0.067)      Data 0.012 (0.013)      Loss
0.4227 (0.2535)    Prec 89.062% (90.928%)
Epoch: [21][300/782]    Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.2739 (0.2546)    Prec 90.625% (90.988%)
Epoch: [21][400/782]    Time 0.068 (0.067)      Data 0.012 (0.013)      Loss
0.3166 (0.2631)    Prec 85.938% (90.707%)
Epoch: [21][500/782]    Time 0.055 (0.067)      Data 0.012 (0.013)      Loss
0.2322 (0.2643)    Prec 90.625% (90.672%)
```

```
Epoch: [21][600/782]    Time 0.078 (0.067)      Data 0.025 (0.013)      Loss
0.2672 (0.2662)    Prec 89.062% (90.674%)
Epoch: [21][700/782]    Time 0.066 (0.067)      Data 0.012 (0.013)      Loss
0.3101 (0.2693)    Prec 90.625% (90.589%)
Validation starts
Test: [0/157]   Time 0.044 (0.044)      Loss 0.4102 (0.4102)    Prec 84.375%
(84.375%)
Test: [100/157] Time 0.033 (0.032)      Loss 0.4511 (0.4094)    Prec 82.812%
(86.510%)
 * Prec 86.710%
best acc: 86.850000
Epoch: [22][0/782]      Time 0.049 (0.049)      Data 0.015 (0.015)      Loss
0.2869 (0.2869)    Prec 89.062% (89.062%)
Epoch: [22][100/782]    Time 0.066 (0.067)      Data 0.013 (0.013)      Loss
0.2792 (0.2661)    Prec 90.625% (90.470%)
Epoch: [22][200/782]    Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.1609 (0.2692)    Prec 95.312% (90.470%)
Epoch: [22][300/782]    Time 0.067 (0.067)      Data 0.024 (0.013)      Loss
0.2983 (0.2695)    Prec 90.625% (90.485%)
Epoch: [22][400/782]    Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.1951 (0.2722)    Prec 93.750% (90.481%)
Epoch: [22][500/782]    Time 0.069 (0.067)      Data 0.012 (0.013)      Loss
0.1848 (0.2711)    Prec 93.750% (90.500%)
Epoch: [22][600/782]    Time 0.070 (0.068)      Data 0.013 (0.013)      Loss
0.4442 (0.2689)    Prec 82.812% (90.550%)
Epoch: [22][700/782]    Time 0.066 (0.068)      Data 0.012 (0.013)      Loss
0.2630 (0.2692)    Prec 89.062% (90.558%)
Validation starts
Test: [0/157]   Time 0.041 (0.041)      Loss 0.4564 (0.4564)    Prec 82.812%
(82.812%)
Test: [100/157] Time 0.031 (0.031)      Loss 0.4094 (0.4502)    Prec 84.375%
(85.783%)
 * Prec 85.850%
best acc: 86.850000
Epoch: [23][0/782]      Time 0.053 (0.053)      Data 0.019 (0.019)      Loss
0.2583 (0.2583)    Prec 92.188% (92.188%)
Epoch: [23][100/782]    Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.1079 (0.2622)    Prec 96.875% (90.718%)
Epoch: [23][200/782]    Time 0.066 (0.067)      Data 0.012 (0.013)      Loss
0.2981 (0.2625)    Prec 90.625% (90.858%)
Epoch: [23][300/782]    Time 0.058 (0.067)      Data 0.013 (0.013)      Loss
0.1800 (0.2632)    Prec 93.750% (90.703%)
Epoch: [23][400/782]    Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.2169 (0.2668)    Prec 93.750% (90.664%)
Epoch: [23][500/782]    Time 0.068 (0.067)      Data 0.012 (0.013)      Loss
0.2833 (0.2673)    Prec 92.188% (90.659%)
Epoch: [23][600/782]    Time 0.066 (0.067)      Data 0.012 (0.013)      Loss
0.2908 (0.2687)    Prec 93.750% (90.648%)
```

```
Epoch: [23][700/782]    Time 0.055 (0.067)      Data 0.012 (0.013)      Loss
0.1923 (0.2692)    Prec 93.750% (90.618%)
Validation starts
Test: [0/157]   Time 0.043 (0.043)      Loss 0.4530 (0.4530)    Prec 85.938%
(85.938%)
Test: [100/157] Time 0.031 (0.032)      Loss 0.5335 (0.4673)    Prec 81.250%
(85.644%)
 * Prec 85.810%
best acc: 86.850000
Epoch: [24][0/782]      Time 0.051 (0.051)      Data 0.018 (0.018)      Loss
0.2868 (0.2868)    Prec 89.062% (89.062%)
Epoch: [24][100/782]    Time 0.079 (0.067)      Data 0.012 (0.014)      Loss
0.1946 (0.2621)    Prec 90.625% (91.012%)
Epoch: [24][200/782]    Time 0.067 (0.067)      Data 0.012 (0.014)      Loss
0.1420 (0.2620)    Prec 95.312% (90.889%)
Epoch: [24][300/782]    Time 0.059 (0.067)      Data 0.013 (0.013)      Loss
0.1239 (0.2575)    Prec 95.312% (90.983%)
Epoch: [24][400/782]    Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.1815 (0.2556)    Prec 95.312% (91.116%)
Epoch: [24][500/782]    Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.1181 (0.2578)    Prec 95.312% (91.037%)
Epoch: [24][600/782]    Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.1685 (0.2604)    Prec 93.750% (90.929%)
Epoch: [24][700/782]    Time 0.066 (0.067)      Data 0.012 (0.013)      Loss
0.1914 (0.2625)    Prec 95.312% (90.843%)
Validation starts
Test: [0/157]   Time 0.032 (0.032)      Loss 0.2949 (0.2949)    Prec 85.938%
(85.938%)
Test: [100/157] Time 0.031 (0.031)      Loss 0.4150 (0.4440)    Prec 85.938%
(85.938%)
 * Prec 86.280%
best acc: 86.850000
Epoch: [25][0/782]      Time 0.068 (0.068)      Data 0.021 (0.021)      Loss
0.3560 (0.3560)    Prec 89.062% (89.062%)
Epoch: [25][100/782]    Time 0.066 (0.067)      Data 0.012 (0.013)      Loss
0.2187 (0.2508)    Prec 90.625% (90.965%)
Epoch: [25][200/782]    Time 0.066 (0.067)      Data 0.012 (0.013)      Loss
0.2616 (0.2477)    Prec 90.625% (90.990%)
Epoch: [25][300/782]    Time 0.081 (0.067)      Data 0.025 (0.013)      Loss
0.3649 (0.2544)    Prec 89.062% (90.827%)
Epoch: [25][400/782]    Time 0.067 (0.067)      Data 0.013 (0.013)      Loss
0.2409 (0.2577)    Prec 92.188% (90.816%)
Epoch: [25][500/782]    Time 0.081 (0.067)      Data 0.012 (0.013)      Loss
0.2947 (0.2607)    Prec 89.062% (90.744%)
Epoch: [25][600/782]    Time 0.067 (0.067)      Data 0.013 (0.013)      Loss
0.2170 (0.2610)    Prec 90.625% (90.797%)
Epoch: [25][700/782]    Time 0.067 (0.067)      Data 0.012 (0.013)      Loss
0.3593 (0.2608)    Prec 87.500% (90.803%)
```

```
Validation starts
Test: [0/157]   Time 0.044 (0.044)      Loss 0.4701 (0.4701)    Prec 84.375%
(84.375%)
Test: [100/157] Time 0.031 (0.031)      Loss 0.4854 (0.4484)    Prec 89.062%
(85.876%)
 * Prec 85.440%
best acc: 86.850000
Epoch: [26][0/782]      Time 0.053 (0.053)      Data 0.017 (0.017)      Loss
0.1517 (0.1517)    Prec 95.312% (95.312%)
Epoch: [26][100/782]    Time 0.066 (0.067)      Data 0.012 (0.014)      Loss
0.2558 (0.2493)    Prec 92.188% (91.429%)
Epoch: [26][200/782]    Time 0.057 (0.067)      Data 0.013 (0.013)      Loss
0.1324 (0.2465)    Prec 96.875% (91.535%)
Epoch: [26][300/782]    Time 0.066 (0.067)      Data 0.012 (0.013)      Loss
0.4317 (0.2521)    Prec 84.375% (91.315%)
Epoch: [26][400/782]    Time 0.056 (0.067)      Data 0.012 (0.013)      Loss
0.1870 (0.2558)    Prec 95.312% (91.096%)
Epoch: [26][500/782]    Time 0.046 (0.067)      Data 0.012 (0.013)      Loss
0.2583 (0.2600)    Prec 93.750% (90.971%)
Epoch: [26][600/782]    Time 0.053 (0.067)      Data 0.012 (0.013)      Loss
0.5544 (0.2612)    Prec 81.250% (90.906%)
Epoch: [26][700/782]    Time 0.066 (0.067)      Data 0.026 (0.013)      Loss
0.2896 (0.2628)    Prec 89.062% (90.810%)
Validation starts
Test: [0/157]   Time 0.034 (0.034)      Loss 0.4580 (0.4580)    Prec 85.938%
(85.938%)
Test: [100/157] Time 0.031 (0.031)      Loss 0.4338 (0.4093)    Prec 85.938%
(87.268%)
 * Prec 87.140%
best acc: 87.140000
Epoch: [27][0/782]      Time 0.084 (0.084)      Data 0.021 (0.021)      Loss
0.3004 (0.3004)    Prec 87.500% (87.500%)
Epoch: [27][100/782]    Time 0.066 (0.067)      Data 0.027 (0.016)      Loss
0.3959 (0.2508)    Prec 87.500% (91.337%)
Epoch: [27][200/782]    Time 0.082 (0.067)      Data 0.012 (0.016)      Loss
0.3243 (0.2548)    Prec 89.062% (91.200%)
Epoch: [27][300/782]    Time 0.067 (0.067)      Data 0.017 (0.016)      Loss
0.3368 (0.2555)    Prec 87.500% (91.191%)
Epoch: [27][400/782]    Time 0.067 (0.067)      Data 0.012 (0.016)      Loss
0.1498 (0.2570)    Prec 93.750% (91.046%)
Epoch: [27][500/782]    Time 0.067 (0.067)      Data 0.012 (0.016)      Loss
0.2586 (0.2599)    Prec 93.750% (90.893%)
Epoch: [27][600/782]    Time 0.066 (0.067)      Data 0.024 (0.016)      Loss
0.2907 (0.2580)    Prec 89.062% (91.010%)
Epoch: [27][700/782]    Time 0.066 (0.067)      Data 0.012 (0.016)      Loss
0.2533 (0.2592)    Prec 93.750% (90.975%)
Validation starts
Test: [0/157]   Time 0.042 (0.042)      Loss 0.4252 (0.4252)    Prec 85.938%
```

```
(85.938%)
Test: [100/157] Time 0.032 (0.031)      Loss 0.6346 (0.4605)    Prec 79.688%
(85.613%)
 * Prec 85.360%
best acc: 87.140000
Epoch: [28][0/782]      Time 0.061 (0.061)      Data 0.017 (0.017)      Loss
0.3024 (0.3024)    Prec 89.062% (89.062%)
Epoch: [28][100/782]    Time 0.066 (0.067)      Data 0.013 (0.013)      Loss
0.4033 (0.2549)    Prec 84.375% (90.873%)
Epoch: [28][200/782]    Time 0.058 (0.067)      Data 0.012 (0.014)      Loss
0.1031 (0.2570)    Prec 98.438% (90.882%)
Epoch: [28][300/782]    Time 0.054 (0.067)      Data 0.012 (0.014)      Loss
0.2233 (0.2584)    Prec 95.312% (90.812%)
Epoch: [28][400/782]    Time 0.055 (0.067)      Data 0.012 (0.014)      Loss
0.3946 (0.2576)    Prec 82.812% (90.941%)
Epoch: [28][500/782]    Time 0.066 (0.067)      Data 0.012 (0.014)      Loss
0.2711 (0.2580)    Prec 92.188% (90.952%)
Epoch: [28][600/782]    Time 0.054 (0.067)      Data 0.013 (0.013)      Loss
0.2103 (0.2579)    Prec 89.062% (90.906%)
Epoch: [28][700/782]    Time 0.066 (0.067)      Data 0.013 (0.013)      Loss
0.2559 (0.2569)    Prec 92.188% (90.937%)
Validation starts
Test: [0/157]   Time 0.044 (0.044)      Loss 0.5125 (0.5125)    Prec 85.938%
(85.938%)
Test: [100/157] Time 0.032 (0.032)      Loss 0.5209 (0.4319)    Prec 85.938%
(86.773%)
 * Prec 86.810%
best acc: 87.140000
Epoch: [29][0/782]      Time 0.053 (0.053)      Data 0.018 (0.018)      Loss
0.1871 (0.1871)    Prec 92.188% (92.188%)
Epoch: [29][100/782]    Time 0.068 (0.069)      Data 0.012 (0.013)      Loss
0.0861 (0.2455)    Prec 98.438% (91.058%)
Epoch: [29][200/782]    Time 0.067 (0.068)      Data 0.012 (0.013)      Loss
0.2522 (0.2369)    Prec 90.625% (91.527%)
Epoch: [29][300/782]    Time 0.068 (0.068)      Data 0.012 (0.013)      Loss
0.3396 (0.2400)    Prec 87.500% (91.533%)
Epoch: [29][400/782]    Time 0.073 (0.068)      Data 0.012 (0.013)      Loss
0.2374 (0.2432)    Prec 93.750% (91.525%)
Epoch: [29][500/782]    Time 0.068 (0.068)      Data 0.013 (0.013)      Loss
0.3035 (0.2447)    Prec 87.500% (91.367%)
Epoch: [29][600/782]    Time 0.057 (0.068)      Data 0.013 (0.013)      Loss
0.3614 (0.2455)    Prec 93.750% (91.306%)
Epoch: [29][700/782]    Time 0.068 (0.068)      Data 0.012 (0.013)      Loss
0.2193 (0.2461)    Prec 92.188% (91.300%)
Validation starts
Test: [0/157]   Time 0.035 (0.035)      Loss 0.4936 (0.4936)    Prec 84.375%
(84.375%)
Test: [100/157] Time 0.031 (0.032)      Loss 0.7299 (0.4557)    Prec 78.125%
```

```
(85.953%)
 * Prec 85.830%
best acc: 87.140000
Epoch: [30][0/782]      Time 0.067 (0.067)      Data 0.016 (0.016)      Loss
0.2071 (0.2071)    Prec 93.750% (93.750%)
Epoch: [30][100/782]    Time 0.068 (0.068)      Data 0.012 (0.013)      Loss
0.2632 (0.2313)    Prec 90.625% (92.079%)
Epoch: [30][200/782]    Time 0.068 (0.069)      Data 0.012 (0.013)      Loss
0.4393 (0.2383)    Prec 85.938% (91.659%)
Epoch: [30][300/782]    Time 0.068 (0.068)      Data 0.013 (0.013)      Loss
0.2124 (0.2411)    Prec 95.312% (91.518%)
Epoch: [30][400/782]    Time 0.078 (0.069)      Data 0.012 (0.013)      Loss
0.2700 (0.2417)    Prec 90.625% (91.506%)
Epoch: [30][500/782]    Time 0.067 (0.069)      Data 0.013 (0.013)      Loss
0.2567 (0.2418)    Prec 90.625% (91.389%)
Epoch: [30][600/782]    Time 0.070 (0.069)      Data 0.012 (0.013)      Loss
0.1785 (0.2429)    Prec 95.312% (91.363%)
Epoch: [30][700/782]    Time 0.067 (0.069)      Data 0.012 (0.013)      Loss
0.2251 (0.2450)    Prec 93.750% (91.329%)
Validation starts
Test: [0/157]   Time 0.044 (0.044)      Loss 0.3133 (0.3133)    Prec 87.500%
(87.500%)
Test: [100/157] Time 0.032 (0.032)      Loss 0.3746 (0.4367)    Prec 84.375%
(86.463%)
 * Prec 86.480%
best acc: 87.140000
Epoch: [31][0/782]      Time 0.051 (0.051)      Data 0.017 (0.017)      Loss
0.2813 (0.2813)    Prec 87.500% (87.500%)
Epoch: [31][100/782]    Time 0.069 (0.068)      Data 0.012 (0.014)      Loss
0.2273 (0.2357)    Prec 89.062% (91.708%)
Epoch: [31][200/782]    Time 0.078 (0.068)      Data 0.012 (0.014)      Loss
0.3110 (0.2379)    Prec 89.062% (91.737%)
Epoch: [31][300/782]    Time 0.082 (0.068)      Data 0.012 (0.014)      Loss
0.2829 (0.2413)    Prec 89.062% (91.482%)
Epoch: [31][400/782]    Time 0.069 (0.068)      Data 0.013 (0.013)      Loss
0.0766 (0.2408)    Prec 98.438% (91.615%)
Epoch: [31][500/782]    Time 0.055 (0.069)      Data 0.012 (0.014)      Loss
0.2914 (0.2442)    Prec 92.188% (91.498%)
Epoch: [31][600/782]    Time 0.066 (0.069)      Data 0.012 (0.014)      Loss
0.1987 (0.2448)    Prec 92.188% (91.462%)
Epoch: [31][700/782]    Time 0.066 (0.069)      Data 0.013 (0.014)      Loss
0.1873 (0.2466)    Prec 93.750% (91.401%)
Validation starts
Test: [0/157]   Time 0.032 (0.032)      Loss 0.4569 (0.4569)    Prec 84.375%
(84.375%)
Test: [100/157] Time 0.031 (0.033)      Loss 0.3920 (0.4177)    Prec 81.250%
(86.881%)
 * Prec 86.960%
```

```
best acc: 87.140000
Epoch: [32][0/782]      Time 0.058 (0.058)      Data 0.025 (0.025)      Loss
0.1147 (0.1147)    Prec 96.875% (96.875%)
Epoch: [32][100/782]    Time 0.066 (0.072)      Data 0.014 (0.013)      Loss
0.2268 (0.2163)    Prec 89.062% (92.234%)
Epoch: [32][200/782]    Time 0.080 (0.070)      Data 0.012 (0.013)      Loss
0.4403 (0.2214)    Prec 85.938% (92.203%)
Epoch: [32][300/782]    Time 0.070 (0.070)      Data 0.013 (0.013)      Loss
0.2759 (0.2297)    Prec 89.062% (92.021%)
Epoch: [32][400/782]    Time 0.066 (0.070)      Data 0.012 (0.013)      Loss
0.2390 (0.2338)    Prec 93.750% (91.856%)
Epoch: [32][500/782]    Time 0.066 (0.071)      Data 0.013 (0.013)      Loss
0.3242 (0.2347)    Prec 90.625% (91.798%)
Epoch: [32][600/782]    Time 0.069 (0.070)      Data 0.014 (0.014)      Loss
0.1459 (0.2394)    Prec 93.750% (91.644%)
Epoch: [32][700/782]    Time 0.068 (0.070)      Data 0.013 (0.014)      Loss
0.1723 (0.2411)    Prec 92.188% (91.619%)
Validation starts
Test: [0/157]   Time 0.043 (0.043)      Loss 0.4651 (0.4651)    Prec 85.938%
(85.938%)
Test: [100/157] Time 0.031 (0.032)      Loss 0.6440 (0.4269)    Prec 81.250%
(87.515%)
 * Prec 87.360%
best acc: 87.360000
Epoch: [33][0/782]      Time 0.053 (0.053)      Data 0.019 (0.019)      Loss
0.2934 (0.2934)    Prec 90.625% (90.625%)
Epoch: [33][100/782]    Time 0.069 (0.070)      Data 0.012 (0.014)      Loss
0.2498 (0.2310)    Prec 93.750% (91.723%)
Epoch: [33][200/782]    Time 0.056 (0.069)      Data 0.013 (0.013)      Loss
0.2289 (0.2407)    Prec 90.625% (91.550%)
Epoch: [33][300/782]    Time 0.083 (0.069)      Data 0.012 (0.013)      Loss
0.2750 (0.2420)    Prec 87.500% (91.544%)
Epoch: [33][400/782]    Time 0.066 (0.068)      Data 0.012 (0.013)      Loss
0.1191 (0.2421)    Prec 95.312% (91.490%)
Epoch: [33][500/782]    Time 0.067 (0.068)      Data 0.012 (0.013)      Loss
0.2080 (0.2421)    Prec 92.188% (91.564%)
Epoch: [33][600/782]    Time 0.067 (0.068)      Data 0.012 (0.013)      Loss
0.2811 (0.2447)    Prec 89.062% (91.415%)
Epoch: [33][700/782]    Time 0.066 (0.068)      Data 0.012 (0.013)      Loss
0.2068 (0.2432)    Prec 93.750% (91.463%)
Validation starts
Test: [0/157]   Time 0.043 (0.043)      Loss 0.3853 (0.3853)    Prec 85.938%
(85.938%)
Test: [100/157] Time 0.032 (0.032)      Loss 0.5060 (0.4132)    Prec 81.250%
(86.850%)
 * Prec 87.040%
best acc: 87.360000
Epoch: [34][0/782]      Time 0.086 (0.086)      Data 0.033 (0.033)      Loss
```

```
0.1795 (0.1795)    Prec 95.312% (95.312%)
Epoch: [34][100/782]    Time 0.066 (0.067)      Data 0.012 (0.014)      Loss
0.2584 (0.2361)    Prec 92.188% (91.383%)
Epoch: [34][200/782]    Time 0.058 (0.067)      Data 0.013 (0.013)      Loss
0.2915 (0.2325)    Prec 85.938% (91.604%)
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[4], line 5
      3 criterion = nn.CrossEntropyLoss()
      4 optimizer = torch.optim.Adam(model.parameters(), lr=lr)
----> 5 train_model(model, fdir, criterion, optimizer, 100)

Cell In[1], line 211, in train_model(model, fdir, criterion, optimizer, epochs)
    208 for epoch in range(0, epochs):
    209     adjust_learning_rate(optimizer, epoch)
--> 211     train(trainloader, model, criterion, optimizer, epoch)
    213     # evaluate on test set
    214     print("Validation starts")

Cell In[1], line 80, in train(trainloader, model, criterion, optimizer, epoch)
     76 for i, (input, target) in enumerate(trainloader):
     77     # measure data loading time
     78     data_time.update(time.time() - end)
---> 80     input, target = input.cuda(), target.cuda()
     82     # compute output
     83     output = model(input)

KeyboardInterrupt:
```

[5]:

```
model = VGG16()
os_prune_vgg16(model, 0.78)
quantize_pruned(model)

PATH = f"{fdir}/model_best.pth.tar"
checkpoint = torch.load(PATH)
model.load_state_dict(checkpoint['state_dict'])
model.cuda()

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()
```

```
model.features[40].register_forward_pre_hook(save_output)

val_model(model)

print_sparsity(model)
```

```
Pruning 50 ic-slices out of 64 ic-slices (78.1% pruned)
Pruning 50 ic-slices out of 64 ic-slices (78.1% pruned)
Pruning 100 ic-slices out of 128 ic-slices (78.1% pruned)
Pruning 100 ic-slices out of 128 ic-slices (78.1% pruned)
Pruning 200 ic-slices out of 256 ic-slices (78.1% pruned)
Pruning 200 ic-slices out of 256 ic-slices (78.1% pruned)
Pruning 200 ic-slices out of 256 ic-slices (78.1% pruned)
Pruning 399 ic-slices out of 512 ic-slices (77.9% pruned)
Pruning 399 ic-slices out of 512 ic-slices (77.9% pruned)
Pruning 399 ic-slices out of 512 ic-slices (77.9% pruned)
Pruning 399 ic-slices out of 512 ic-slices (77.9% pruned)
Pruning 399 ic-slices out of 512 ic-slices (77.9% pruned)
```

```
/tmp/ipykernel_63507/3884249786.py:6: FutureWarning: You are using `torch.load`
with `weights_only=False` (the current default value), which uses the default
pickle module implicitly. It is possible to construct malicious pickle data
which will execute arbitrary code during unpickling (See
https://github.com/pytorch/pytorch/blob/main/SECURITY.md#untrusted-models for
more details). In a future release, the default value for `weights_only` will be
flipped to `True`. This limits the functions that could be executed during
unpickling. Arbitrary objects will no longer be allowed to be loaded via this
mode unless they are explicitly allowlisted by the user via
`torch.serialization.add_safe_globals`. We recommend you start setting
`weights_only=True` for any use case where you don't have full control of the
loaded file. Please open an issue on GitHub for any issues related to this
experimental feature.
  checkpoint = torch.load(PATH)
```

```

Test set: Accuracy: 8736/10000 (87%)

layer 3 sparsity: 0.781
layer 7 sparsity: 0.781
layer 10 sparsity: 0.773
layer 14 sparsity: 0.773
layer 17 sparsity: 0.773
layer 20 sparsity: 0.773
layer 24 sparsity: 0.773
layer 27 sparsity: 0.777
layer 30 sparsity: 0.777
layer 34 sparsity: 0.777
layer 37 sparsity: 0.777
layer 40 sparsity: 0.777
```

[ ]:
```
l = model.features[40]
a = [i for i in range(l.weight_mask.size(1)) if l.weight_mask[0,i].sum() > 0]
print(l.weight_q[0,a[:2],:,:])
```

```
tensor([[[ 0.0000, -1.4775, -1.4775],
         [ 0.7387,  0.7387,  0.0000],
         [ 0.0000,  0.7387,  0.0000]],

        [[-0.0000,  0.0000,  0.7387],
         [ 0.0000,  0.7387,  1.4775],
         [ 0.7387,  0.7387,  0.7387]]], device='cuda:0',
       grad_fn=<IndexBackward0>)
```

[ ]:
```
print(save_output.outputs[0][0][:2,:2])
```

```
tensor([[[[0.0000, 0.0807],
          [0.1303, 0.0000]],

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


        [[[0.5222, 0.3067],
          [0.0775, 0.9398]],

         [[0.0000, 0.0000],
          [0.0000, 0.0000]]]], device='cuda:0')
```

[ ]:
```
print(f'Weight int: \n{(model.features[40].weight_q.data / (model.features[40].
  ↪weight_quant.wgt_alpha.data.item()/(2**(4-1)-1)))[0,a[0]]}')
x = save_output.outputs[0][0]
x_alpha = model.features[40].act_alpha.data.item()
x_delta = x_alpha / (2**(4)-1)
act_q = act_quantization(4)
x_q = act_q(x, x_alpha)
print(f'Act int: \n{(x_q/x_delta)[:2,:2]}')
```

```
Weight int:
tensor([[ 0.0000, -2.0000, -2.0000],
        [ 1.0000,  1.0000,  0.0000],
        [ 0.0000,  1.0000,  0.0000]], device='cuda:0')
Act int:
tensor([[[[0., 0.],
          [0., 0.]],

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


        [[[1., 1.],
          [0., 2.]],

         [[0., 0.],
          [0., 0.]]]], device='cuda:0')
```