Imports

In [1]:

```
import torch
import torchvision
import torchvision.transforms as transforms
from torch.utils.data import Dataset, DataLoader
import torch.nn as nn
import torch.nn.functional as F
import sys
import numpy as np
import os
```

Utilising GPU using Pytorch

In [2]:

```
# cpu-gpu
a = torch.randn((3, 4))
print(a.device)

device = torch.device("cuda")
a = a.to(device)
print(a.device)

# a more generic code
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
```

```
cpu
cuda:0
```

In [3]:

```
!nvidia-smi
```

```
Sun Sep 18 08:59:58 2022
+-----------------------------------------------------------------------------
----------+
| NVIDIA-SMI 460.32.03    Driver Version: 460.32.03    CUDA Version:
11.2     |
|-------------------------------+----------------------+------------
----------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Un
corr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  C
ompute M. |
|                               |                      |
MIG M. |
|===============================+======================+============
==========|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |
0 |
| N/A   54C    P0    28W /  70W |    612MiB / 15109MiB |      2%
Default |
|                               |                      |
N/A |
+-------------------------------+----------------------+------------
----------+

+-----------------------------------------------------------------------------
----------+
| Processes:
|
|  GPU   GI   CI        PID   Type   Process name                  G
PU Memory |
|        ID   ID                                                   U
sage      |
|=============================================================================
==========|
+-----------------------------------------------------------------------------
----------+
```

Dataset and Transforms

In [4]:

```
train_transform = transforms.Compose([
  transforms.RandomCrop(32, padding=4),
  transforms.RandomHorizontalFlip(),
  transforms.ToTensor(),
  transforms.Normalize((0.4914, 0.4822, 0.4465), (0.2023, 0.1994, 0.2010)),
])
test_transform = transforms.Compose([
  transforms.ToTensor(),
  transforms.Normalize((0.4914, 0.4822, 0.4465), (0.2023, 0.1994, 0.2010)),
])

train_dset = torchvision.datasets.CIFAR10(root="data/", train=True, transform=tr
ain_transform, download=True)
test_dset = torchvision.datasets.CIFAR10(root="data/", train=False, transform=te
st_transform, download=True)
```

```
Downloading https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
to data/cifar-10-python.tar.gz


Extracting data/cifar-10-python.tar.gz to data/
Files already downloaded and verified
```

In [5]:

```
print(f"# of train samples: {len(train_dset)}")
print(f"# of test samples: {len(test_dset)}")
```

```
# of train samples: 50000
# of test samples: 10000
```

In [6]:

```
train_loader = DataLoader(train_dset, batch_size=100, shuffle=True, num_workers=
2)
test_loader = DataLoader(test_dset, batch_size=100, shuffle=False, num_workers=2
)
```

In [7]:

```
print(f"# of train batches: {len(train_loader)}")
print(f"# of test batches: {len(test_loader)}")
```

```
# of train batches: 500
# of test batches: 100
```

In [8]:

```
print("sample i/o sizes")
data = next(iter(train_loader))
img, target = data
print(f"input size: {img.shape}")
print(f"output size: {target.shape}")
```

```
sample i/o sizes
input size: torch.Size([100, 3, 32, 32])
output size: torch.Size([100])
```

LeNet

In [9]:

```
class LeNet(nn.Module):
  def __init__(self):
    super(LeNet, self).__init__()
    self.conv1 = nn.Conv2d(3, 6, kernel_size=5)
    self.conv2 = nn.Conv2d(6, 16, kernel_size=5)
    # TODO: missing input feature size
    self.fc1   = nn.Linear(16*5*5, 120)
    self.fc2   = nn.Linear(120, 84)
    # TODO: missing output feature size
    self.fc3   = nn.Linear(84, 10)  ##10 CLASSES
    self.activ = nn.ReLU()

  # TODO: add maxpool operation of given kernel size
  # https://pytorch.org/docs/stable/nn.functional.html
  def pool(self, x, kernel_size=2):
    out = F.max_pool2d(x, kernel_size=2)
    return out

  def forward(self, x):
    out = self.activ(self.conv1(x))
    out = self.pool(out)
    out = self.activ(self.conv2(out))
    out = self.pool(out)

    # TODO: flatten
    out = out.view(out.size(0),-1) ##OR We can do out.view(out.size(0),-1)
    out = self.activ(self.fc1(out))
    out = self.activ(self.fc2(out))
    out = self.fc3(out)
    return out
```

VGG

In [17]:

```
class VGG(nn.Module):
  CONFIGS = {
      "vgg11": [64, "pool", 128, "pool", 256, 256, "pool", 512, 512, "pool", 512
, 512, "pool"],
      "vgg13": [64, 64, "pool", 128, 128, "pool", 256, 256, "pool", 512, 512, "p
ool", 512, 512, "pool"],
      "vgg16": [64, 64, "pool", 128, 128, "pool", 256, 256, 256, "pool", 512, 51
2, 512, "pool", 512, 512, 512, "pool"],
      "vgg19": [64, 64, "pool", 128, 128, "pool", 256, 256, 256, 256, "pool", 51
2, 512, 512, 512, "pool", 512, 512, 512, 512, "pool"],
  }
  def __init__(self, cfg):
    super(VGG, self).__init__()
    # TODO: missing input dimension
    in_dim = 3
    layers = []
    for layer in self.CONFIGS[cfg]:
        if layer == "pool":
            # TODO: add maxpool module of given kernel size, stride (here 2 eac
h)
            # https://pytorch.org/docs/stable/nn.html
            maxpool = nn.MaxPool2d(kernel_size=2, stride=2)
            layers.append(maxpool)
        else:
            # TODO: add sequential module consisting of convolution (kernel size
= 3, padding = 1), batchnorm, relu
            # https://pytorch.org/docs/stable/generated/torch.nn.Sequential.htm
l?highlight=sequential#torch.nn.Sequential
            block = nn.Sequential(
          nn.Conv2d(in_dim,layer,kernel_size = 3, padding = 1),
          nn.BatchNorm2d(layer),
          nn.ReLU()
        )
            layers.append(block)
            in_dim = layer
    # TODO: add average pool to collapse spatial dimensions
    #avgpool = F.lp_pool2d(input, norm_type, kernel_size=2) """difference"""
    avgpool = nn.AvgPool2d(kernel_size=1, stride=1)
    layers.append(avgpool)
    self.layers = nn.Sequential(*layers)
    # TODO: missing output features
    self.fc = nn.Linear(512, 10) ##10 CLASSES

  def forward(self, x):
    out = self.layers(x)
    # TODO: flatten
    out = out.view(out.size(0),-1)
    out = self.fc(out)
    return out
```

ResNet

In [11]:

```
class BasicBlock(nn.Module):
  expansion = 1

  def __init__(self, in_dim, dim, stride=1):
    super(BasicBlock, self).__init__()
    self.conv1 = nn.Conv2d(in_dim, dim, kernel_size=3, stride=stride, padding=1,
bias=False)
    self.bn1 = nn.BatchNorm2d(dim)
    self.conv2 = nn.Conv2d(dim, dim, kernel_size=3, stride=1, padding=1, bias=Fa
lse)
    self.bn2 = nn.BatchNorm2d(dim)
    self.activ = nn.ReLU()

    self.shortcut = nn.Identity()
    # TODO: missing condition for parameterized shortcut connection (hint: when
 input and output dimensions don't match - both spatial, feature)
    if (stride != 1 or in_dim != self.expansion*dim):
        # TODO: add sequential module consisting of 1x1 convolution (given strid
e, bias=False), batchnorm
        self.shortcut = nn.Sequential(
                nn.Conv2d(in_dim, self.expansion*dim, kernel_size=1, stride=stri
de),
                nn.BatchNorm2d(self.expansion*dim)
            )

  def forward(self, x):
    out = self.activ(self.bn1(self.conv1(x)))
    out = self.bn2(self.conv2(out))
    # TODO: missing residual connection
    out = out + self.shortcut(x)
    out = self.activ(out)
    return out


class Bottleneck(nn.Module):
  expansion = 4

  def __init__(self, in_dim, dim, stride=1):
    super(Bottleneck, self).__init__()
    self.conv1 = nn.Conv2d(in_dim, dim, kernel_size=1, bias=False)
    self.bn1 = nn.BatchNorm2d(dim)
    self.conv2 = nn.Conv2d(dim, dim, kernel_size=3, stride=stride, padding=1, bi
as=False)
    self.bn2 = nn.BatchNorm2d(dim)
    self.conv3 = nn.Conv2d(dim, self.expansion * dim, kernel_size=1, bias=False)
    self.bn3 = nn.BatchNorm2d(self.expansion*dim)
    self.activ = nn.ReLU()

    self.shortcut = nn.Identity()
    # TODO: missing condition for parameterized shortcut connection (hint: when
 input and output dimensions don't match - both spatial, feature)
    if (stride != 1 or in_dim != self.expansion*dim):
        # TODO: add sequential module consisting of 1x1 convolution (given strid
e, bias=False), batchnorm
        self.shortcut = nn.Sequential(
                nn.Conv2d(in_dim, self.expansion*dim, kernel_size=1, stride=stri
de),
                nn.BatchNorm2d(self.expansion*dim)
            )
```

```
  def forward(self, x):
    out = self.activ(self.bn1(self.conv1(x)))
    out = self.activ(self.bn2(self.conv2(out)))
    out = self.bn3(self.conv3(out))
    # TODO: missing residual connection
    out = out + self.shortcut(x)
    out = self.activ(out)
    return out


class ResNet(nn.Module):
  CONFIGS = {
      "resnet18": (BasicBlock, [2, 2, 2, 2]),
      "resnet34": (BasicBlock, [3, 4, 6, 3]),
      "resnet50": (Bottleneck, [3, 4, 6, 3]),
      "resnet101": (Bottleneck, [3, 4, 23, 3]),
      "resnet152": (Bottleneck, [3, 8, 36, 3]),
  }
  def __init__(self, cfg):
    super(ResNet, self).__init__()
    block, num_blocks = self.CONFIGS[cfg]
    self.in_dim = 64
    self.conv1 = nn.Conv2d(3, 64, kernel_size=3, stride=1, padding=1, bias=False
)
    self.bn1 = nn.BatchNorm2d(64)
    self.layer1 = self._make_layer(block, 64, num_blocks[0], stride=1)
    self.layer2 = self._make_layer(block, 128, num_blocks[1], stride=2)
    self.layer3 = self._make_layer(block, 256, num_blocks[2], stride=2)
    self.layer4 = self._make_layer(block, 512, num_blocks[3], stride=2)
    self.activ = nn.ReLU()
    # TODO: missing output features
    self.linear = nn.Linear(512*block.expansion,10)

  def _make_layer(self, block, dim, num_blocks, stride):
    strides = [stride] + [1]*(num_blocks-1)
    layers = []
    for stride in strides:
        # TODO: create layers within block
        layer = block(self.in_dim, dim, stride)
        layers.append(layer)
        # TODO: update in_dim based on block output size
        self.in_dim = dim * block.expansion
    return nn.Sequential(*layers)

  def forward(self, x):
    out = self.activ(self.bn1(self.conv1(x)))
    out = self.layer1(out)
    out = self.layer2(out)
    out = self.layer3(out)
    out = self.layer4(out)
    # TODO: average pool and flatten
    # pooling = nn.AvgPool2d(4)
    out = F.avg_pool2d(out,4)
    out = out.view(out.size(0),-1)
    out = self.linear(out)
    return out
```

Utility functions (can ignore)

In [12]:

```
def pbar(p=0, msg="", bar_len=20):
    sys.stdout.write("\033[K")
    sys.stdout.write("\x1b[2K" + "\r")
    block = int(round(bar_len * p))
    text = "Progress: [{}] {}% {}".format(
        "\x1b[32m" + "=" * (block - 1) + ">" + "\033[0m" + "-" * (bar_len - bloc
k),
        round(p * 100, 2),
        msg,
    )
    print(text, end="\r")
    if p == 1:
        print()


class AvgMeter:
    def __init__(self):
        self.reset()

    def reset(self):
        self.metrics = {}

    def add(self, batch_metrics):
        if self.metrics == {}:
            for key, value in batch_metrics.items():
                self.metrics[key] = [value]
        else:
            for key, value in batch_metrics.items():
                self.metrics[key].append(value)

    def get(self):
        return {key: np.mean(value) for key, value in self.metrics.items()}

    def msg(self):
        avg_metrics = {key: np.mean(value) for key, value in self.metrics.items
()}
        return "".join(["[{}] {:.5f} ".format(key, value) for key, value in avg_
metrics.items()])
```

Training

In [13]:

```
def train(model, optim, lr_sched=None, epochs=200, device=torch.device("cuda" if
torch.cuda.is_available() else "cpu"), criterion=None, metric_meter=None, out_di
r="out/"):
  model.to(device)
  best_acc = 0
  for epoch in range(epochs):
    model.train()
    metric_meter.reset()
    for indx, (img, target) in enumerate(train_loader):
      # TODO: send to device (cpu or gpu)
      img = img.to(device)
      target = target.to(device)

      # TODO: missing forward pass
      out = model(img)
      loss = criterion(out, target)
      # TODO: missing backward, parameter update
      optim.zero_grad()
      loss.backward()
      optim.step()

      metric_meter.add({"train loss": loss.item()})
      pbar(indx / len(train_loader), msg=metric_meter.msg())
    pbar(1, msg=metric_meter.msg())

    model.eval()
    metric_meter.reset()
    for indx, (img, target) in enumerate(test_loader):
      # TODO: send to device (cpu or gpu)
      img = img.to(device)
      target = target.to(device)

      # TODO: missing forward pass
      out = model(img)
      loss = criterion(out, target)
      # TODO: compute accuracy
      classes = torch.argmax(out, dim=1)
      acc_t = torch.mean((classes == target).float())
      acc=acc_t.cpu().detach().numpy()

      metric_meter.add({"test loss": loss.item(), "test acc": acc})
      pbar(indx / len(test_loader), msg=metric_meter.msg())
    pbar(1, msg=metric_meter.msg())

    test_metrics = metric_meter.get()
    if test_metrics["test acc"] > best_acc:
      print(
          "\x1b[33m"
          + f"test acc improved from {round(best_acc, 5)} to {round(test_metrics
['test acc'], 5)}"
          + "\033[0m"
      )
      best_acc = test_metrics['test acc']
      torch.save(model.state_dict(), os.path.join(out_dir, "best.ckpt"))
    lr_sched.step()
```

Run Experiments

In [14]:

```
def run_experiment(model_name="lenet", model_cfg=None, epochs=200):
  if model_name == "lenet":
    model = LeNet()
  elif model_name == "vgg":
    model = VGG(model_cfg)
  elif model_name == "resnet":
    model = ResNet(model_cfg)
  else:
    raise NotImplementedError()
  optim = torch.optim.SGD(model.parameters(), lr=1e-1, momentum=0.9, weight_deca
y=5e-4)
  lr_sched = torch.optim.lr_scheduler.CosineAnnealingLR(optim, T_max=epochs)
  criterion = nn.CrossEntropyLoss()
  metric_meter = AvgMeter()
  out_dir = f"{model_name}_{model_cfg}"
  os.makedirs(out_dir, exist_ok=True)
  train(model, optim, lr_sched, epochs=epochs, criterion=criterion, metric_meter
=metric_meter, out_dir=out_dir)
```

In [21]:

```
# run_experiment(model_name="lenet")
# run_experiment(model_name="vgg",model_cfg="vgg16")
run_experiment(model_name="resnet",model_cfg="resnet18")
```

```
Progress: [==================>] 100% [train loss] 1.96899
Progress: [==================>] 100% [test loss] 1.62717 [test acc]
0.40000
test acc improved from 0 to 0.4000000059604645
Progress: [==================>] 100% [train loss] 1.42769
Progress: [==================>] 100% [test loss] 1.30875 [test acc]
0.52040
test acc improved from 0.4000000059604645 to 0.5203999876976013
Progress: [==================>] 100% [train loss] 1.14282
Progress: [==================>] 100% [test loss] 1.17196 [test acc]
0.60720
test acc improved from 0.5203999876976013 to 0.607200026512146
Progress: [==================>] 100% [train loss] 0.96004
Progress: [==================>] 100% [test loss] 1.05006 [test acc]
0.63590
test acc improved from 0.607200026512146 to 0.6359000205993652
Progress: [==================>] 100% [train loss] 0.83885
Progress: [==================>] 100% [test loss] 0.79605 [test acc]
0.71920
test acc improved from 0.6359000205993652 to 0.7192000150680542
Progress: [==================>] 100% [train loss] 0.72772
Progress: [==================>] 100% [test loss] 0.78887 [test acc]
0.72950
test acc improved from 0.7192000150680542 to 0.7294999957084656
Progress: [==================>] 100% [train loss] 0.65510
Progress: [==================>] 100% [test loss] 0.89532 [test acc]
0.70260
Progress: [==================>] 100% [train loss] 0.60810
Progress: [==================>] 100% [test loss] 0.88356 [test acc]
0.70890
Progress: [==================>] 100% [train loss] 0.57897
Progress: [==================>] 100% [test loss] 0.67864 [test acc]
0.77330
test acc improved from 0.7294999957084656 to 0.7732999920845032
Progress: [==================>] 100% [train loss] 0.54857
Progress: [==================>] 100% [test loss] 0.88229 [test acc]
0.71420
Progress: [==================>] 100% [train loss] 0.52581
Progress: [==================>] 100% [test loss] 0.69746 [test acc]
0.77400
test acc improved from 0.7732999920845032 to 0.7739999890327454
Progress: [==================>] 100% [train loss] 0.50858
Progress: [==================>] 100% [test loss] 0.68934 [test acc]
0.77400
Progress: [==================>] 100% [train loss] 0.49648
Progress: [==================>] 100% [test loss] 0.59074 [test acc]
0.80520
test acc improved from 0.7739999890327454 to 0.8051999807357788
Progress: [==================>] 100% [train loss] 0.47788
Progress: [==================>] 100% [test loss] 0.60258 [test acc]
0.79900
Progress: [==================>] 100% [train loss] 0.47511
Progress: [==================>] 100% [test loss] 0.73770 [test acc]
0.75990
Progress: [==================>] 100% [train loss] 0.45714
Progress: [==================>] 100% [test loss] 0.67051 [test acc]
0.77870
Progress: [==================>] 100% [train loss] 0.44941
Progress: [==================>] 100% [test loss] 0.52482 [test acc]
0.82020
test acc improved from 0.8051999807357788 to 0.8202000260353088
```

```
Progress: [====================>] 100% [train loss] 0.44353
Progress: [====================>] 100% [test loss] 0.55082 [test acc]
0.81780
Progress: [====================>] 100% [train loss] 0.43842
Progress: [====================>] 100% [test loss] 0.60326 [test acc]
0.80860
Progress: [====================>] 100% [train loss] 0.43978
Progress: [====================>] 100% [test loss] 0.61095 [test acc]
0.80800
Progress: [====================>] 100% [train loss] 0.42592
Progress: [====================>] 100% [test loss] 0.55704 [test acc]
0.81100
Progress: [====================>] 100% [train loss] 0.42471
Progress: [====================>] 100% [test loss] 0.51768 [test acc]
0.82590
test acc improved from 0.8202000260353088 to 0.8259000182151794
Progress: [====================>] 100% [train loss] 0.41826
Progress: [====================>] 100% [test loss] 0.53869 [test acc]
0.81830
Progress: [====================>] 100% [train loss] 0.41239
Progress: [====================>] 100% [test loss] 0.51870 [test acc]
0.82110
Progress: [====================>] 100% [train loss] 0.40652
Progress: [====================>] 100% [test loss] 0.54714 [test acc]
0.82250
Progress: [====================>] 100% [train loss] 0.40554
Progress: [====================>] 100% [test loss] 0.48999 [test acc]
0.83020
test acc improved from 0.8259000182151794 to 0.8302000164985657
Progress: [====================>] 100% [train loss] 0.40248
Progress: [====================>] 100% [test loss] 0.87510 [test acc]
0.74310
Progress: [====================>] 100% [train loss] 0.40098
Progress: [====================>] 100% [test loss] 0.56110 [test acc]
0.81730
Progress: [====================>] 100% [train loss] 0.39452
Progress: [====================>] 100% [test loss] 0.47634 [test acc]
0.84360
test acc improved from 0.8302000164985657 to 0.8435999751091003
Progress: [====================>] 100% [train loss] 0.39467
Progress: [====================>] 100% [test loss] 0.57342 [test acc]
0.80610
Progress: [====================>] 100% [train loss] 0.39095
Progress: [====================>] 100% [test loss] 0.66437 [test acc]
0.78130
Progress: [====================>] 100% [train loss] 0.39730
Progress: [====================>] 100% [test loss] 0.69390 [test acc]
0.78630
Progress: [====================>] 100% [train loss] 0.38521
Progress: [====================>] 100% [test loss] 0.67412 [test acc]
0.78860
Progress: [====================>] 100% [train loss] 0.38071
Progress: [====================>] 100% [test loss] 0.51810 [test acc]
0.82330
Progress: [====================>] 100% [train loss] 0.37327
Progress: [====================>] 100% [test loss] 0.49662 [test acc]
0.83580
Progress: [====================>] 100% [train loss] 0.38448
Progress: [====================>] 100% [test loss] 0.64336 [test acc]
0.78460
Progress: [====================>] 100% [train loss] 0.37193
```

```
Progress: [====================>] 100% [test loss] 0.55583 [test acc]
0.81700
Progress: [====================>] 100% [train loss] 0.37717
Progress: [====================>] 100% [test loss] 0.54079 [test acc]
0.81970
Progress: [====================>] 100% [train loss] 0.37635
Progress: [====================>] 100% [test loss] 0.52814 [test acc]
0.82690
Progress: [====================>] 100% [train loss] 0.36458
Progress: [====================>] 100% [test loss] 0.63601 [test acc]
0.79330
Progress: [====================>] 100% [train loss] 0.36578
Progress: [====================>] 100% [test loss] 0.47166 [test acc]
0.83990
Progress: [====================>] 100% [train loss] 0.36761
Progress: [====================>] 100% [test loss] 0.48229 [test acc]
0.83630
Progress: [====================>] 100% [train loss] 0.36273
Progress: [====================>] 100% [test loss] 0.57234 [test acc]
0.81510
Progress: [====================>] 100% [train loss] 0.36207
Progress: [====================>] 100% [test loss] 0.46691 [test acc]
0.84440
test acc improved from 0.8435999751091003 to 0.8443999886512756
Progress: [====================>] 100% [train loss] 0.36142
Progress: [====================>] 100% [test loss] 0.53558 [test acc]
0.82550
Progress: [====================>] 100% [train loss] 0.35933
Progress: [====================>] 100% [test loss] 0.51908 [test acc]
0.83490
Progress: [====================>] 100% [train loss] 0.35842
Progress: [====================>] 100% [test loss] 0.55695 [test acc]
0.83010
Progress: [====================>] 100% [train loss] 0.35375
Progress: [====================>] 100% [test loss] 0.50472 [test acc]
0.83250
Progress: [====================>] 100% [train loss] 0.35035
Progress: [====================>] 100% [test loss] 0.45941 [test acc]
0.84750
test acc improved from 0.8443999886512756 to 0.8475000262260437
Progress: [====================>] 100% [train loss] 0.35449
Progress: [====================>] 100% [test loss] 0.62662 [test acc]
0.80670
Progress: [====================>] 100% [train loss] 0.34949
Progress: [====================>] 100% [test loss] 0.59870 [test acc]
0.80710
Progress: [====================>] 100% [train loss] 0.34351
Progress: [====================>] 100% [test loss] 0.47300 [test acc]
0.84040
Progress: [====================>] 100% [train loss] 0.34619
Progress: [====================>] 100% [test loss] 0.45297 [test acc]
0.84860
test acc improved from 0.8475000262260437 to 0.8485999703407288
Progress: [====================>] 100% [train loss] 0.34341
Progress: [====================>] 100% [test loss] 0.51144 [test acc]
0.83350
Progress: [====================>] 100% [train loss] 0.33708
Progress: [====================>] 100% [test loss] 0.49847 [test acc]
0.83740
Progress: [====================>] 100% [train loss] 0.33574
Progress: [====================>] 100% [test loss] 0.54500 [test acc]
```

```
0.82380
Progress: [==================>] 100% [train loss] 0.33447
Progress: [==================>] 100% [test loss] 0.42858 [test acc]
0.85700
test acc improved from 0.8485999703407288 to 0.8569999933242798
Progress: [==================>] 100% [train loss] 0.33318
Progress: [==================>] 100% [test loss] 0.56287 [test acc]
0.81600
Progress: [==================>] 100% [train loss] 0.33150
Progress: [==================>] 100% [test loss] 0.57027 [test acc]
0.81870
Progress: [==================>] 100% [train loss] 0.33096
Progress: [==================>] 100% [test loss] 0.46827 [test acc]
0.84280
Progress: [==================>] 100% [train loss] 0.32493
Progress: [==================>] 100% [test loss] 0.38142 [test acc]
0.87070
test acc improved from 0.8569999933242798 to 0.8707000017166138
Progress: [==================>] 100% [train loss] 0.32439
Progress: [==================>] 100% [test loss] 0.42481 [test acc]
0.85890
Progress: [==================>] 100% [train loss] 0.32692
Progress: [==================>] 100% [test loss] 0.37594 [test acc]
0.87740
test acc improved from 0.8707000017166138 to 0.8773999810218811
Progress: [==================>] 100% [train loss] 0.31397
Progress: [==================>] 100% [test loss] 0.54652 [test acc]
0.82890
Progress: [==================>] 100% [train loss] 0.32422
Progress: [==================>] 100% [test loss] 0.41169 [test acc]
0.86640
Progress: [==================>] 100% [train loss] 0.31524
Progress: [==================>] 100% [test loss] 0.52031 [test acc]
0.83920
Progress: [==================>] 100% [train loss] 0.31706
Progress: [==================>] 100% [test loss] 0.57739 [test acc]
0.82560
Progress: [==================>] 100% [train loss] 0.30806
Progress: [==================>] 100% [test loss] 0.44301 [test acc]
0.86090
Progress: [==================>] 100% [train loss] 0.31420
Progress: [==================>] 100% [test loss] 0.44451 [test acc]
0.85500
Progress: [==================>] 100% [train loss] 0.30690
Progress: [==================>] 100% [test loss] 0.78737 [test acc]
0.76790
Progress: [==================>] 100% [train loss] 0.30281
Progress: [==================>] 100% [test loss] 0.46534 [test acc]
0.84420
Progress: [==================>] 100% [train loss] 0.30797
Progress: [==================>] 100% [test loss] 0.48297 [test acc]
0.84310
Progress: [==================>] 100% [train loss] 0.30440
Progress: [==================>] 100% [test loss] 0.61996 [test acc]
0.81130
Progress: [==================>] 100% [train loss] 0.29853
Progress: [==================>] 100% [test loss] 0.45145 [test acc]
0.85010
Progress: [==================>] 100% [train loss] 0.29514
Progress: [==================>] 100% [test loss] 0.47248 [test acc]
0.84270
```

```
Progress: [====================>] 100% [train loss] 0.29495
Progress: [====================>] 100% [test loss] 0.48510 [test acc]
0.84660
Progress: [====================>] 100% [train loss] 0.29170
Progress: [====================>] 100% [test loss] 0.51233 [test acc]
0.83670
Progress: [====================>] 100% [train loss] 0.29335
Progress: [====================>] 100% [test loss] 0.38806 [test acc]
0.86970
Progress: [====================>] 100% [train loss] 0.28458
Progress: [====================>] 100% [test loss] 0.42297 [test acc]
0.86290
Progress: [====================>] 100% [train loss] 0.28123
Progress: [====================>] 100% [test loss] 0.44120 [test acc]
0.85870
Progress: [====================>] 100% [train loss] 0.28636
Progress: [====================>] 100% [test loss] 0.48045 [test acc]
0.84550
Progress: [====================>] 100% [train loss] 0.28032
Progress: [====================>] 100% [test loss] 0.44403 [test acc]
0.85740
Progress: [====================>] 100% [train loss] 0.27905
Progress: [====================>] 100% [test loss] 0.46895 [test acc]
0.84610
Progress: [====================>] 100% [train loss] 0.27313
Progress: [====================>] 100% [test loss] 0.42061 [test acc]
0.86000
Progress: [====================>] 100% [train loss] 0.27775
Progress: [====================>] 100% [test loss] 0.45554 [test acc]
0.85430
Progress: [====================>] 100% [train loss] 0.26963
Progress: [====================>] 100% [test loss] 0.34449 [test acc]
0.88580
test acc improved from 0.8773999810218811 to 0.8858000040054321
Progress: [====================>] 100% [train loss] 0.27057
Progress: [====================>] 100% [test loss] 0.43751 [test acc]
0.85370
Progress: [====================>] 100% [train loss] 0.27216
Progress: [====================>] 100% [test loss] 0.45633 [test acc]
0.85050
Progress: [====================>] 100% [train loss] 0.26066
Progress: [====================>] 100% [test loss] 0.55835 [test acc]
0.82590
Progress: [====================>] 100% [train loss] 0.25943
Progress: [====================>] 100% [test loss] 0.33865 [test acc]
0.89110
test acc improved from 0.8858000040054321 to 0.8910999894142151
Progress: [====================>] 100% [train loss] 0.26441
Progress: [====================>] 100% [test loss] 0.49664 [test acc]
0.84090
Progress: [====================>] 100% [train loss] 0.26071
Progress: [====================>] 100% [test loss] 0.37983 [test acc]
0.87180
Progress: [====================>] 100% [train loss] 0.25506
Progress: [====================>] 100% [test loss] 0.38165 [test acc]
0.87780
Progress: [====================>] 100% [train loss] 0.24963
Progress: [====================>] 100% [test loss] 0.39964 [test acc]
0.86570
Progress: [====================>] 100% [train loss] 0.25063
Progress: [====================>] 100% [test loss] 0.40858 [test acc]
```

```
0.86250
Progress: [==================>] 100% [train loss] 0.24119
Progress: [==================>] 100% [test loss] 0.33186 [test acc]
0.88910
Progress: [==================>] 100% [train loss] 0.24089
Progress: [==================>] 100% [test loss] 0.37751 [test acc]
0.87550
Progress: [==================>] 100% [train loss] 0.23261
Progress: [==================>] 100% [test loss] 0.39837 [test acc]
0.86970
Progress: [==================>] 100% [train loss] 0.24128
Progress: [==================>] 100% [test loss] 0.45556 [test acc]
0.85590
Progress: [==================>] 100% [train loss] 0.23596
Progress: [==================>] 100% [test loss] 0.41921 [test acc]
0.86440
Progress: [==================>] 100% [train loss] 0.23392
Progress: [==================>] 100% [test loss] 0.33629 [test acc]
0.88700
Progress: [==================>] 100% [train loss] 0.22336
Progress: [==================>] 100% [test loss] 0.37555 [test acc]
0.87940
Progress: [==================>] 100% [train loss] 0.22882
Progress: [==================>] 100% [test loss] 0.40209 [test acc]
0.87130
Progress: [==================>] 100% [train loss] 0.22279
Progress: [==================>] 100% [test loss] 0.35255 [test acc]
0.88820
Progress: [==================>] 100% [train loss] 0.21585
Progress: [==================>] 100% [test loss] 0.38386 [test acc]
0.87600
Progress: [==================>] 100% [train loss] 0.22113
Progress: [==================>] 100% [test loss] 0.35797 [test acc]
0.88460
Progress: [==================>] 100% [train loss] 0.21232
Progress: [==================>] 100% [test loss] 0.46750 [test acc]
0.85130
Progress: [==================>] 100% [train loss] 0.21075
Progress: [==================>] 100% [test loss] 0.31709 [test acc]
0.89580
test acc improved from 0.8910999894142151 to 0.895799994468689
Progress: [==================>] 100% [train loss] 0.21164
Progress: [==================>] 100% [test loss] 0.34251 [test acc]
0.88470
Progress: [==================>] 100% [train loss] 0.20127
Progress: [==================>] 100% [test loss] 0.39833 [test acc]
0.87420
Progress: [==================>] 100% [train loss] 0.19692
Progress: [==================>] 100% [test loss] 0.34861 [test acc]
0.88860
Progress: [==================>] 100% [train loss] 0.20119
Progress: [==================>] 100% [test loss] 0.30947 [test acc]
0.89630
test acc improved from 0.895799994468689 to 0.8963000178337097
Progress: [==================>] 100% [train loss] 0.19402
Progress: [==================>] 100% [test loss] 0.35381 [test acc]
0.88220
Progress: [==================>] 100% [train loss] 0.19161
Progress: [==================>] 100% [test loss] 0.36576 [test acc]
0.88000
Progress: [==================>] 100% [train loss] 0.18913
```

```
Progress: [====================>] 100% [test loss] 0.36921 [test acc]
0.88190
Progress: [====================>] 100% [train loss] 0.18352
Progress: [====================>] 100% [test loss] 0.32657 [test acc]
0.89930
test acc improved from 0.8963000178337097 to 0.8992999792098999
Progress: [====================>] 100% [train loss] 0.18195
Progress: [====================>] 100% [test loss] 0.34270 [test acc]
0.89080
Progress: [====================>] 100% [train loss] 0.17829
Progress: [====================>] 100% [test loss] 0.29765 [test acc]
0.90310
test acc improved from 0.8992999792098999 to 0.9031000137329102
Progress: [====================>] 100% [train loss] 0.17458
Progress: [====================>] 100% [test loss] 0.32729 [test acc]
0.89500
Progress: [====================>] 100% [train loss] 0.16755
Progress: [====================>] 100% [test loss] 0.32060 [test acc]
0.89600
Progress: [====================>] 100% [train loss] 0.17244
Progress: [====================>] 100% [test loss] 0.36438 [test acc]
0.88400
Progress: [====================>] 100% [train loss] 0.16585
Progress: [====================>] 100% [test loss] 0.33950 [test acc]
0.89070
Progress: [====================>] 100% [train loss] 0.15843
Progress: [====================>] 100% [test loss] 0.31900 [test acc]
0.89680
Progress: [====================>] 100% [train loss] 0.15950
Progress: [====================>] 100% [test loss] 0.35182 [test acc]
0.88120
Progress: [====================>] 100% [train loss] 0.15362
Progress: [====================>] 100% [test loss] 0.29933 [test acc]
0.90570
test acc improved from 0.9031000137329102 to 0.9057000279426575
Progress: [====================>] 100% [train loss] 0.15350
Progress: [====================>] 100% [test loss] 0.32296 [test acc]
0.89780
Progress: [====================>] 100% [train loss] 0.14738
Progress: [====================>] 100% [test loss] 0.39842 [test acc]
0.87750
Progress: [====================>] 100% [train loss] 0.14188
Progress: [====================>] 100% [test loss] 0.29429 [test acc]
0.90370
Progress: [====================>] 100% [train loss] 0.13943
Progress: [====================>] 100% [test loss] 0.30238 [test acc]
0.90260
Progress: [====================>] 100% [train loss] 0.14142
Progress: [====================>] 100% [test loss] 0.30168 [test acc]
0.90540
Progress: [====================>] 100% [train loss] 0.13140
Progress: [====================>] 100% [test loss] 0.34181 [test acc]
0.89560
Progress: [====================>] 100% [train loss] 0.13338
Progress: [====================>] 100% [test loss] 0.33349 [test acc]
0.89790
Progress: [====================>] 100% [train loss] 0.12288
Progress: [====================>] 100% [test loss] 0.29281 [test acc]
0.91260
test acc improved from 0.9057000279426575 to 0.9125999808311462
Progress: [====================>] 100% [train loss] 0.12673
```

```
Progress: [====================>] 100% [test loss] 0.34711 [test acc]
0.89740
Progress: [====================>] 100% [train loss] 0.12020
Progress: [====================>] 100% [test loss] 0.29039 [test acc]
0.90720
Progress: [====================>] 100% [train loss] 0.11134
Progress: [====================>] 100% [test loss] 0.28254 [test acc]
0.91050
Progress: [====================>] 100% [train loss] 0.11238
Progress: [====================>] 100% [test loss] 0.27572 [test acc]
0.91590
test acc improved from 0.9125999808311462 to 0.9158999919891357
Progress: [====================>] 100% [train loss] 0.11346
Progress: [====================>] 100% [test loss] 0.28896 [test acc]
0.91260
Progress: [====================>] 100% [train loss] 0.10739
Progress: [====================>] 100% [test loss] 0.31716 [test acc]
0.90780
Progress: [====================>] 100% [train loss] 0.10350
Progress: [====================>] 100% [test loss] 0.28575 [test acc]
0.91320
Progress: [====================>] 100% [train loss] 0.09778
Progress: [====================>] 100% [test loss] 0.30986 [test acc]
0.90730
Progress: [====================>] 100% [train loss] 0.09249
Progress: [====================>] 100% [test loss] 0.30657 [test acc]
0.90730
Progress: [====================>] 100% [train loss] 0.09544
Progress: [====================>] 100% [test loss] 0.27599 [test acc]
0.91430
Progress: [====================>] 100% [train loss] 0.08734
Progress: [====================>] 100% [test loss] 0.26746 [test acc]
0.91900
test acc improved from 0.9158999919891357 to 0.9190000295639038
Progress: [====================>] 100% [train loss] 0.08836
Progress: [====================>] 100% [test loss] 0.26982 [test acc]
0.91800
Progress: [====================>] 100% [train loss] 0.08357
Progress: [====================>] 100% [test loss] 0.29305 [test acc]
0.91610
Progress: [====================>] 100% [train loss] 0.07740
Progress: [====================>] 100% [test loss] 0.29995 [test acc]
0.91160
Progress: [====================>] 100% [train loss] 0.07624
Progress: [====================>] 100% [test loss] 0.28792 [test acc]
0.91600
Progress: [====================>] 100% [train loss] 0.07269
Progress: [====================>] 100% [test loss] 0.25572 [test acc]
0.92460
test acc improved from 0.9190000295639038 to 0.9246000051498413
Progress: [====================>] 100% [train loss] 0.06933
Progress: [====================>] 100% [test loss] 0.28496 [test acc]
0.91800
Progress: [====================>] 100% [train loss] 0.06581
Progress: [====================>] 100% [test loss] 0.26699 [test acc]
0.92300
Progress: [====================>] 100% [train loss] 0.05947
Progress: [====================>] 100% [test loss] 0.28290 [test acc]
0.92170
Progress: [====================>] 100% [train loss] 0.05359
Progress: [====================>] 100% [test loss] 0.26249 [test acc]
```

```
0.92610
test acc improved from 0.9246000051498413 to 0.9261000156402588
Progress: [====================>] 100% [train loss] 0.05700
Progress: [====================>] 100% [test loss] 0.26074 [test acc]
0.92540
Progress: [====================>] 100% [train loss] 0.04575
Progress: [====================>] 100% [test loss] 0.24699 [test acc]
0.92940
test acc improved from 0.9261000156402588 to 0.9294000267982483
Progress: [====================>] 100% [train loss] 0.04870
Progress: [====================>] 100% [test loss] 0.24634 [test acc]
0.93030
test acc improved from 0.9294000267982483 to 0.9302999973297119
Progress: [====================>] 100% [train loss] 0.04454
Progress: [====================>] 100% [test loss] 0.25128 [test acc]
0.93150
test acc improved from 0.9302999973297119 to 0.9315000176429749
Progress: [====================>] 100% [train loss] 0.04289
Progress: [====================>] 100% [test loss] 0.23907 [test acc]
0.93410
test acc improved from 0.9315000176429749 to 0.9340999722480774
Progress: [====================>] 100% [train loss] 0.04237
Progress: [====================>] 100% [test loss] 0.24316 [test acc]
0.93230
Progress: [====================>] 100% [train loss] 0.04010
Progress: [====================>] 100% [test loss] 0.26237 [test acc]
0.92720
Progress: [====================>] 100% [train loss] 0.03271
Progress: [====================>] 100% [test loss] 0.25308 [test acc]
0.93220
Progress: [====================>] 100% [train loss] 0.02577
Progress: [====================>] 100% [test loss] 0.22864 [test acc]
0.93470
test acc improved from 0.9340999722480774 to 0.9347000122070312
Progress: [====================>] 100% [train loss] 0.02530
Progress: [====================>] 100% [test loss] 0.27456 [test acc]
0.92820
Progress: [====================>] 100% [train loss] 0.02404
Progress: [====================>] 100% [test loss] 0.23866 [test acc]
0.93620
test acc improved from 0.9347000122070312 to 0.9362000226974487
Progress: [====================>] 100% [train loss] 0.02231
Progress: [====================>] 100% [test loss] 0.22740 [test acc]
0.93700
test acc improved from 0.9362000226974487 to 0.9369999766349792
Progress: [====================>] 100% [train loss] 0.01722
Progress: [====================>] 100% [test loss] 0.24521 [test acc]
0.93630
Progress: [====================>] 100% [train loss] 0.01589
Progress: [====================>] 100% [test loss] 0.22152 [test acc]
0.94160
test acc improved from 0.9369999766349792 to 0.9416000247001648
Progress: [====================>] 100% [train loss] 0.01481
Progress: [====================>] 100% [test loss] 0.22621 [test acc]
0.94060
Progress: [====================>] 100% [train loss] 0.01195
Progress: [====================>] 100% [test loss] 0.21094 [test acc]
0.94300
test acc improved from 0.9416000247001648 to 0.9430000185966492
Progress: [====================>] 100% [train loss] 0.00985
Progress: [====================>] 100% [test loss] 0.21435 [test acc]
```

```
0.94580
test acc improved from 0.9430000185966492 to 0.9458000063896179
Progress: [====================>] 100% [train loss] 0.01005
Progress: [====================>] 100% [test loss] 0.20744 [test acc]
0.94740
test acc improved from 0.9458000063896179 to 0.9473999738693237
Progress: [====================>] 100% [train loss] 0.00737
Progress: [====================>] 100% [test loss] 0.20380 [test acc]
0.94780
test acc improved from 0.9473999738693237 to 0.9477999806404114
```

Interrupted before 200 epochs (approx. =200)

```
---------------------------------------------------------------------
-------
KeyboardInterrupt                         Traceback (most recent cal
l last)
<ipython-input-21-728359c21e28> in <module>
      1 # run_experiment(model_name="lenet")
      2 # run_experiment(model_name="vgg",model_cfg="vgg16")
----> 3 run_experiment(model_name="resnet",model_cfg="resnet18")

<ipython-input-14-0d8cd2d0ba80> in run_experiment(model_name, model_
cfg, epochs)
     14     out_dir = f"{model_name}_{model_cfg}"
     15     os.makedirs(out_dir, exist_ok=True)
---> 16     train(model, optim, lr_sched, epochs=epochs, criterion=cri
terion, metric_meter=metric_meter, out_dir=out_dir)

<ipython-input-13-99c4e1e223f0> in train(model, optim, lr_sched, epo
chs, device, criterion, metric_meter, out_dir)
     18         optim.step()
     19
---> 20         metric_meter.add({"train loss": loss.item()})
     21         pbar(indx / len(train_loader), msg=metric_meter.msg())
     22       pbar(1, msg=metric_meter.msg())

KeyboardInterrupt:
```

In [ ]:

Questions

- Train and report test set metrics on three model types - LeNet, VGG, ResNet.
- Which model performs the best and why?
- Which model performs the worst and why?
- BONUS (extra marks): Modify the LeNet model's convolution layers and compare performance against number of layers (depth), number of nodes per layer (width). (Require atleast 3 data points each for width and depth). Feel free to reduce the number of epochs to obtain results quickly.

Answers:

For 200 epochs:

Model name	Train Loss	Test Loss	Test Accuracy
LeNet	0.826	0.799	0.716
VGGNet16	0.0018	0.285	0.938
ResNet18	0.010	0.207	0.947

From the above table, LeNet performs the worst as it is a much shallower network than the other two i.e. VggNet16 and Resnet18. The VggNet16 performance is close enough to that of ResNet as per the experiment done and is better than LeNet as they being deeper and hence are more capable to extract the overall information contained in the image.

In [ ]: