IE4424 Machine Learning Design and Application

Image Classification Using Convolutional Neural Network (CNN)

Week 4: Design Lab Briefing

Dr Yap Kim Hui

Email: ekhyap@ntu.edu.sg

Objective

- Study the concepts and models for image classification using convolutional neural networks and transfer learning.
- Exposure to Pytorch Deep Learning framework for practical applications.

Introduction

- Image Classification: Predict the image category.
- Transfer Learning: Finetune a model pretrained on a large dataset such that it can work well on target dataset.
- Focus on high-level understanding first, rather than detailed code syntax.

Data Preprocessing

- Define Dataset:

```
transform = transforms.Compose(
    [transforms.ToTensor(),
     transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))])

trainset = torchvision.datasets.CIFAR10(root='./data', train=True,
                                        download=True, transform=transform)
```

- Construct Dataloader:

```
trainloader = torch.utils.data.DataLoader(trainset, batch_size=4,
                                          shuffle=True, num_workers=0)
```

- Get Samples from Dataloader:

```
dataiter = iter(trainloader)
images, labels = dataiter.next()
```

Pytorch CNN Implementation

```
import torch.nn as nn
import torch.nn.functional as F


class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(3, 6, 3, padding=1)
        self.pool = nn.MaxPool2d(2, 2)
        self.conv2 = nn.Conv2d(6, 16, 3, padding=1)
        self.fc1 = nn.Linear(16 * 8 * 8, 120)
        self.fc2 = nn.Linear(120, 84)
        self.fc3 = nn.Linear(84, 10)

    def forward(self, x):
        x = self.pool(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))
        x = x.view(-1, 16 * 8 * 8)
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = self.fc3(x)
        return x


net = Net()
```

Layer Definition

Feed Forward Logic

Loss Calculation and Back Propagation

```
# create your optimizer
optimizer = optim.SGD(net.parameters(), lr=0.01)

# in your training loop:
optimizer.zero_grad()   # zero the gradient buffers
output = net(input)
loss = criterion(output, target)
loss.backward()
optimizer.step()    # Does the update
```

Saving and Loading Model

- Saving trained model

```
PATH = './cifar_net.pth'
torch.save(net.state_dict(), PATH)
```

- Loading trained model

```
net = Net()
net.load_state_dict(torch.load(PATH))
```

Why Transfer Learning?

- Deep neural network needs a long time to train.
- Large quantity of data is required.
- CV tasks often share certain similarities.

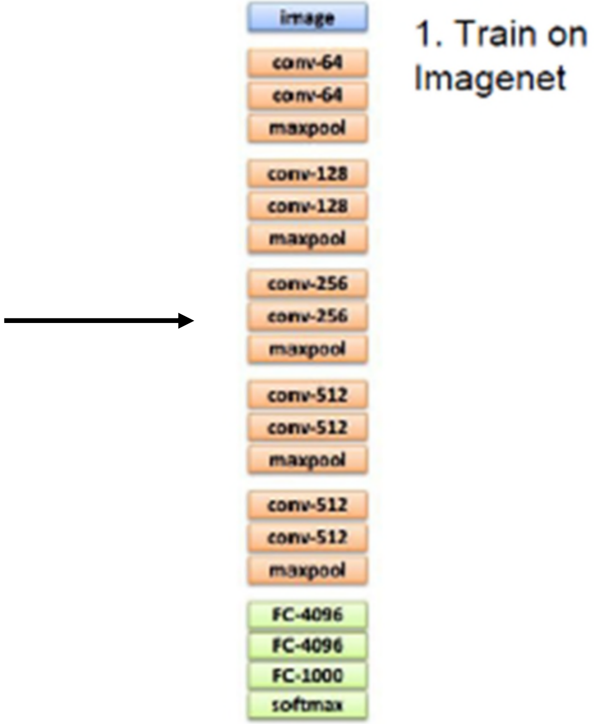
Transfer Leaning

Source: www.mathworks.com/discovery/transfer-learning.html

Transfer Learning

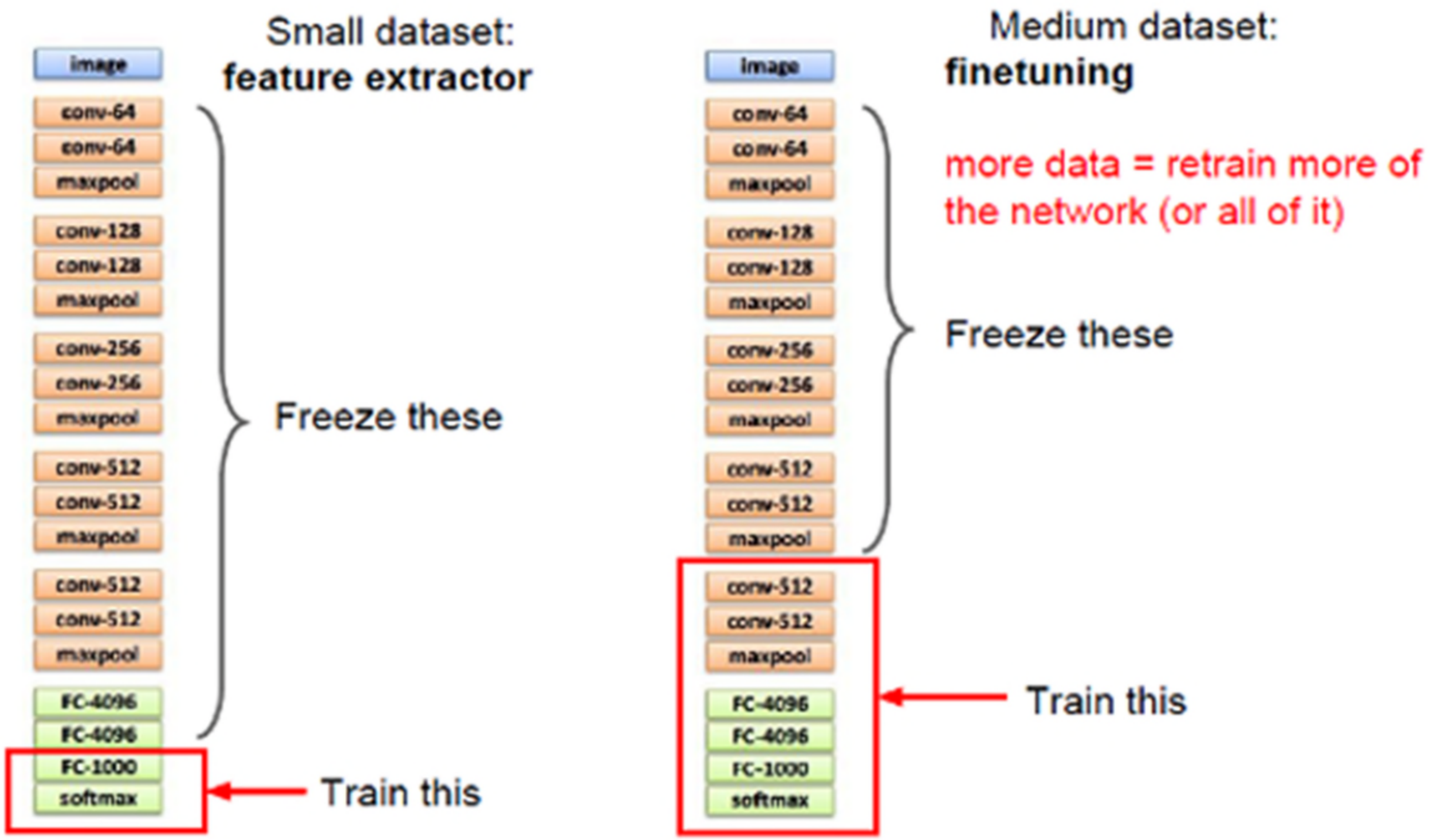
- Step 1: Pretrain on large-size dataset

14,197,122 images



Transfer Learning

- Step 2: Finetune partial layers as needed



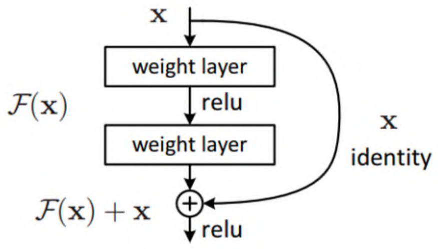
Transfer Learning with Pytorch

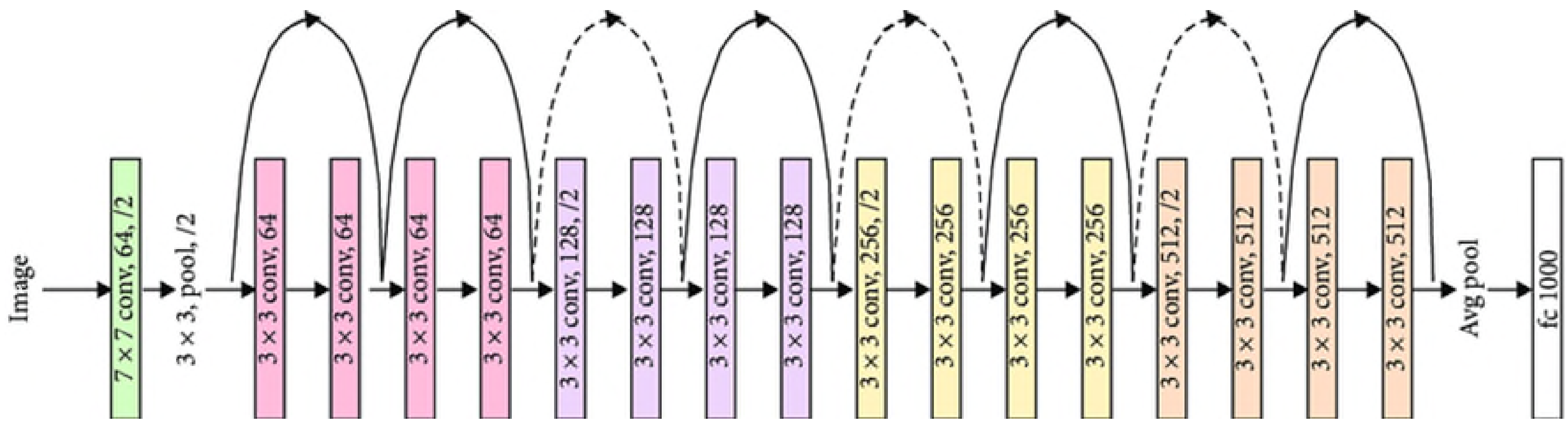
- Freeze parameters in early layers of pretrained model
- Modify structure of late layer(s) of pretrained model
- Train the specific late layer(s)

```
model_resnet18 = torchvision.models.resnet18(pretrained=True)
for param in model_resnet18.parameters():
    param.requires_grad = False

# Parameters of newly constructed modules have requires_grad=True by default
num_ftrs = model_resnet18.fc.in_features
model_resnet18.fc = nn.Linear(num_ftrs, 2)
```

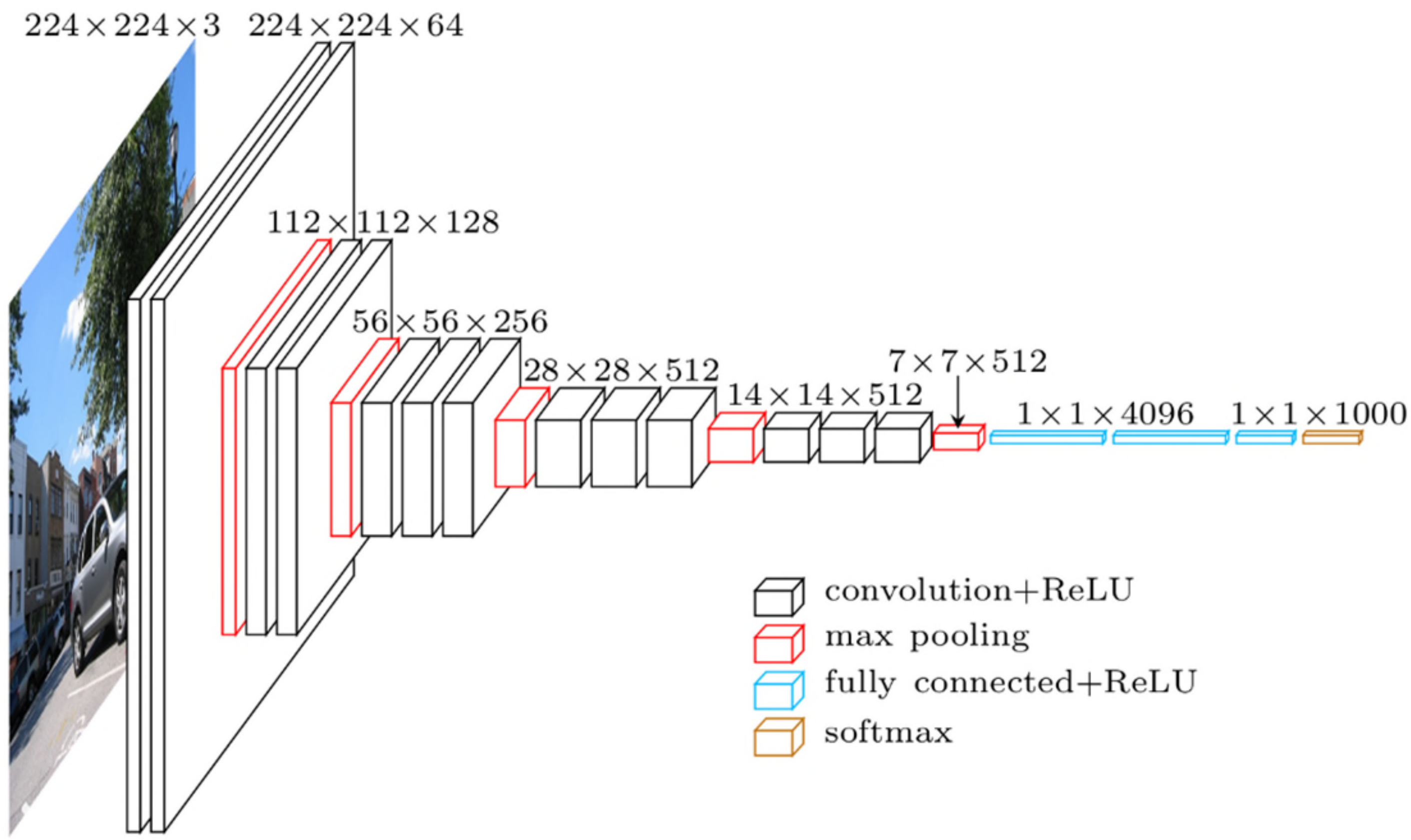
ResNet-18 Architecture





Source: Transfer Learning with ResNet in Pytorch

VGG Architecture



Lab Instructions

1. Read the instructions and complete the exercises in Part1_Image_Classification.ipynb and Part2_Transfer_Learning.ipynb.
2. Get the answer sheet from lab staff. Follow the instructions and answer the questions in the answer sheet.
3. Write your full name and matriculation no clearly on the answer sheet.
4. Submit the completed answer sheet to the TA at the end of lab.
5. Around last 30 min of the lab, you will be given a few short questions to answer individually. This last part will be closed book and centred on the lecture note and conducted lab.