결과 보고서

제 목: 전이학습을 이용한 분류기 구현

과 목 명:	딥러닝프로그래밍및실습
학 부:	수학과
학 번:	20190501
이 름:	김 태 균
제 출 일:	2024년 5월 17일(금)
담당교수:	한 영 준

*** 과제 수행 내용**

: Pytorch에서 제공되는 사전 학습된 VGGNet-16 모델의 전이학습을 이용한 분류기를 구현한 후, Fashion-MNIST dataset을 이용한 학습

(1) Set pretrained model

```
    """ Set pretrained model """
    model = vgg16(pretrained=True)

    # classifier
    fc = nn.Sequential(
        nn.Linear(512*7*7, 4096),
        nn.ReLU(),
        nn.Dropout(), # dropout layer 정의
        nn.Linear(4096, 4096),
        nn.ReLU(),
        nn.Dropout(),
        nn.Linear(4096, 10, bias=False),
        nn.Softmax(dim=1)
    )
    model.classifier = fc

    device = "cuda" if torch.cuda.is_available() else "cpu"
    model.to(device)
    print(model)

    summary(model,input_size=(3, 224, 224))
```

(2) Data preprocessing

```
""" Dataset preproceesing & load """
transforms = Compose([
    Grayscale(num_output_channels=3),
    Resize(224),
    RandomCrop((224, 224), padding=4),
    RandomHorizontalFlip(p=0.5),
    ToTensor(),
    Normalize(mean=(0.4914, 0.4822, 0.4465), std=(0.247, 0.243, 0.261))
])

training_data = FashionMNIST(root="./data", train=True, download=True, transform=transforms)
test_data = FashionMNIST(root="./data", train=False, download=True, transform=transforms)

train_loader = DataLoader(training_data, batch_size=128, shuffle=True)
test_loader = DataLoader(test_data, batch_size=128, shuffle=False)

# freezing pretrained parameter
for name, param in model.features.named_parameters():
    param.requires_grad = False

params_to_update = []
for name, param in model.classifier.named_parameters():
    param.requires_grad = True
    params_to_update.append(param)
```

(3) Using Fashion-MNIST dataset, Training and Validation

```
67  """ Training """
68  lr = 1e-4
69  loss_fn = nn.CrossEntropyLoss()
70  optim = Adam(params=params_to_update, lr=lr)
71
72  losses = []
73  size = len(train_loader.dataset)
74
75  for epoch in range(2):
76      start_time = time.time()
77      epoch_loss = 0.0
78
79      iterator = tqdm.tqdm(train_loader)
80      for data, label in iterator:
81          optim.zero_grad()
82
83          preds = model(data.to(device))
84          loss = loss_fn(preds, label.to(device))
85          loss.backward()
86          optim.step()
87
88          epoch_loss += loss.item() * data.size(0)
89
90          iterator.set_description(f"epoch:{epoch+1} loss:{loss.item()}")
91
92      epoch_loss = epoch_loss / size
93      losses.append(epoch_loss)
94
95      calculation_time = time.time() - start_time
96      print(f"Calculation time on training set : {calculation_time:0.3f} sec")
97
```

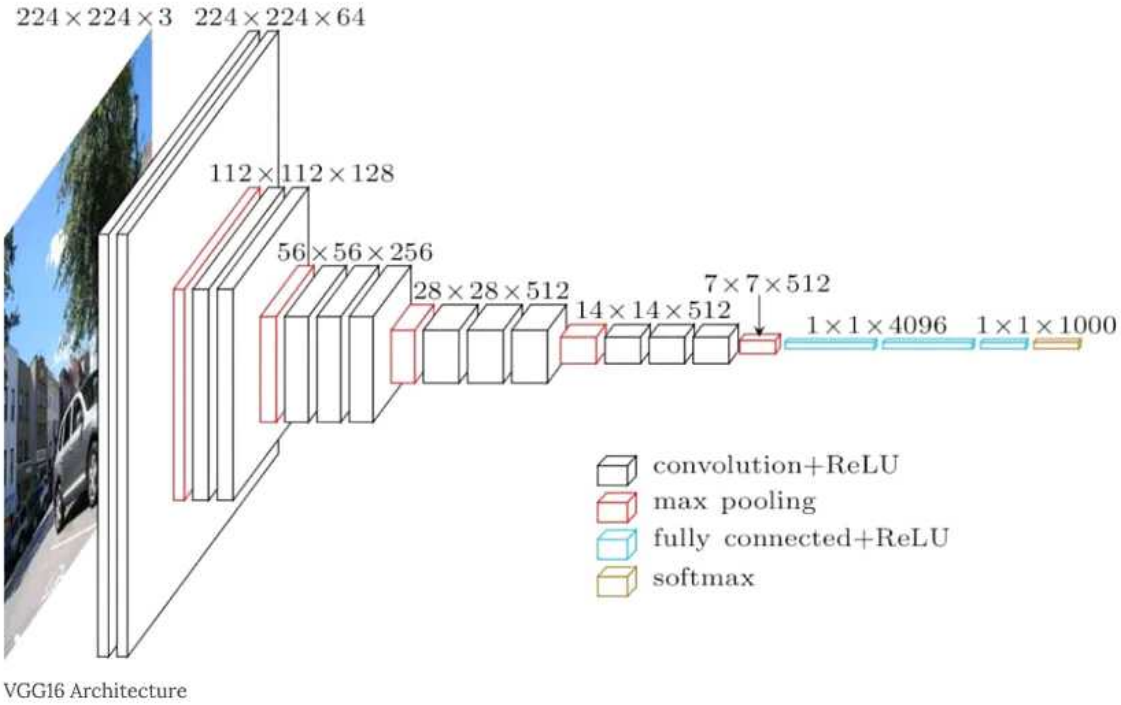
```
113   """ Validation """
114   num_corr = 0
115   correct_samples = []
116   incorrect_samples = []
117
118   model.eval()
119   with torch.no_grad():
120       start_time = time.time()
121       iterator = tqdm.tqdm(test_loader)
122       for data, label in iterator:
123           output = model(data.to(device))
124           preds = output.data.max(1)[1]
125
126           corr = preds.eq(label.to(device).data).sum().item()
127           num_corr += corr
128
129           for i in range(len(preds)):
130               if preds[i] == label[i]:
131                   correct_samples.append((data[i], preds[i]))
132               else:
133                   incorrect_samples.append((data[i], preds[i], label[i]))
134
135       validation_time = time.time() - start_time
136       print(f"Calculation time on validation set: {validation_time:.3f} sec")
137
```

```
160   print(f"Accuracy:{num_corr/len(test_data)}")
```

(4) Important part

```
41   """ Dataset preproceesing & load """
42   transforms = Compose([
43       Grayscale(num_output_channels=3),
44       Resize(224),
45       RandomCrop((224, 224), padding=4),
46       RandomHorizontalFlip(p=0.5),
47       ToTensor(),
48       Normalize(mean=(0.4914, 0.4822, 0.4465), std=(0.247, 0.243, 0.261))
49   ])
```

: Fashion-MNIST dataset은 RGB가 아닌 흑백 이미지로, channel의 수가 3이 아닌 1이다. VGGNet-16의 input 이미지는 아래의 사진과 같이 3개의 channel 수를 갖기 때문에 data load시 이 점을 고려해야 한다. 즉, transforms시 Grayscale 함수를 통해 channel의 수를 3으로 조정함으로써 training 및 validation 과정에서 차원 조정 문제를 해결하는 것이 핵심이다.



VGG16 Architecture

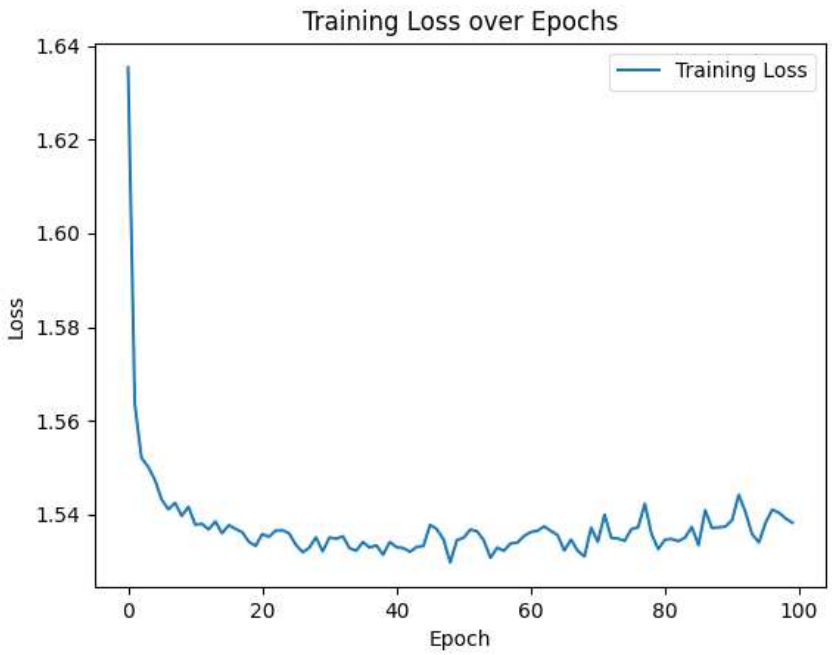
(5) Execution time and cost function trend during training

hyper parameter : epoch=100, batch size=256, lr=1e-4

device : GPU

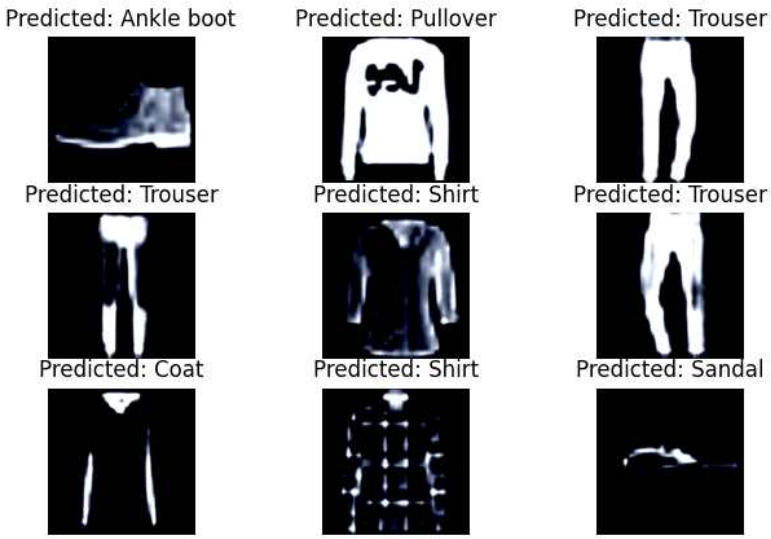
execution time for 1 epoch= 약 110초

trend :



(6) Accuracy of test dataset, print correctly classified samples

- Accuracy : 0.912
- Correctly classified samples (3x3)



(7) Print incorrectly classified samples

- Incorrectly classified samples (3x3)