

# KW-VIP 과제 2

2017706106 전자공학과 현민기

## \*과제 목표

-CIFAR 10 이미지를 연결하여,  
정확도를 도출해내는 프로그램 작성.

## \*구현 방법

```
class ConvNet(nn.Module):
    def __init__(self, num_classes=10):
        super(ConvNet, self).__init__()
        self.layer1 = nn.Sequential(
            nn.Conv2d(3, 16, kernel_size=5, padding=2),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2)
        )
        self.layer2 = nn.Sequential(
            nn.Conv2d(16, 32, kernel_size=5, padding=2),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2)
        )
        self.fc = nn.Linear(32*8*8, num_classes)

    def forward(self, x):
        out = self.layer1(x)
        out = self.layer2(out)
        out = out.reshape(out.size(0), -1)
        out = self.fc(out)
        return out
```

CNN 선언; CIFAR 이미지는 3채널이기 때문에, Conv2d의 채널을 3채널로 적용.

필터의 크기는 kernel size 5\*5 적용

```
num_epochs = 5
num_classes = 10
batch_size = 100
learning_rate = 0.001
```

training 횟수를 5번으로 지정.

```

for epoch in range(num_epochs): # 데이터셋을 수차례 반복합니다.
    for i, (images, labels) in enumerate(trainloader, 0):
        # [inputs, labels]의 목록인data로부터 입력을 받은 후;
        images = images.to(device)
        labels = labels.to(device)
        # 변화도(Gradient) 매개변수를0으로 만들고
        optimizer.zero_grad()

        # 순전파+ 역전파+ 최적화를 한 후
        outputs = model(images)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()

print('Finished Training')

```

Train과정. num\_epochs를 5로 설정해 두었기 때문에, 총 5회 학습을 진행함.

loss에 criterion을 대입하며 오차를 계산, 최적화 시킴.

```

classes = ('plane', 'car', 'bird', 'cat',
'deer', 'dog', 'frog', 'horse', 'ship', 'truck')

model.eval()
class_correct = list(0. for i in range(10))
class_total = list(0. for i in range(10))

with torch.no_grad():
    for (images, labels) in testloader:
        images = images.to(device)
        labels = labels.to(device)

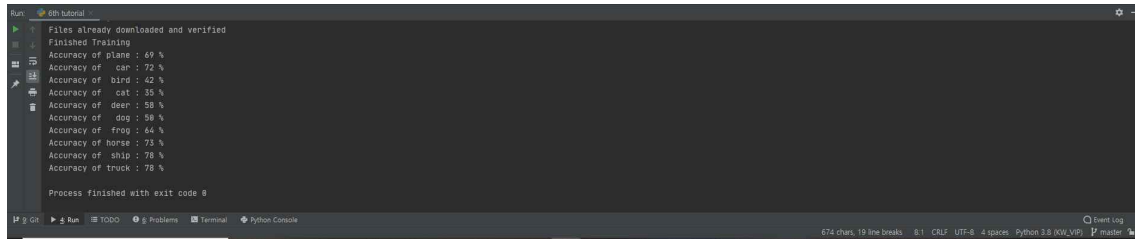
        outputs = model(images)
        _, predicted = torch.max(outputs, 1)
        c = (predicted == labels).squeeze()
        for i in range(4):
            label = labels[i]
            class_correct[label] += c[i].item()
            class_total[label] += 1

for i in range(10):
    print('Accuracy of %5s : %2d %%' % (
        classes[i], 100 * class_correct[i] / class_total[i]))

```

모델을 테스트 한 후, 정확도를 계산하여 출력함.

## \*구현 결과



```
Run: 6th tutorial
Files already downloaded and verified
Finished Training
Accuracy of plane : 69 %
Accuracy of car : 72 %
Accuracy of bird : 42 %
Accuracy of cat : 35 %
Accuracy of deer : 58 %
Accuracy of dog : 58 %
Accuracy of frog : 44 %
Accuracy of horse : 73 %
Accuracy of ship : 78 %
Accuracy of truck : 78 %

Process finished with exit code 0
```

The screenshot shows a Jupyter Notebook interface with a terminal window. The terminal output displays the results of a machine learning model's performance on a dataset. The model has achieved accuracies of 69% for plane, 72% for car, 42% for bird, 35% for cat, 58% for deer, 58% for dog, 44% for frog, 73% for horse, 78% for ship, and 78% for truck. The process finished with exit code 0.