# KW_VIP 과제3

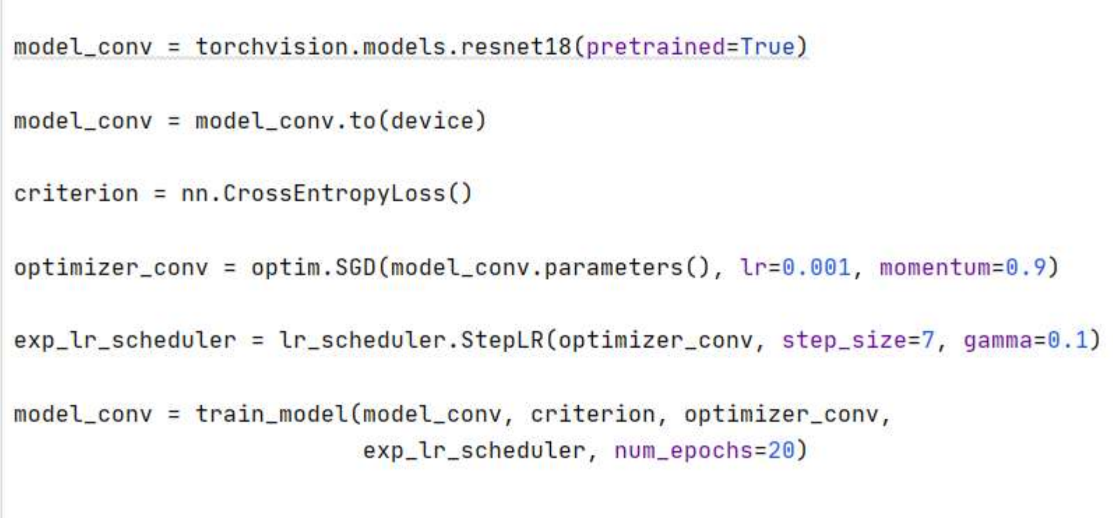
2017741009 로봇학부 장우현

## *과제 목표

Assignment3와 같은 데이터셋인 개미와 벌의 학습을 Pretrained된 resnet18 Strategy1 & 3 방법으로 학습시키고, 결과를 Assignment3에서 만든 나만의 네트워크와 비교해보자

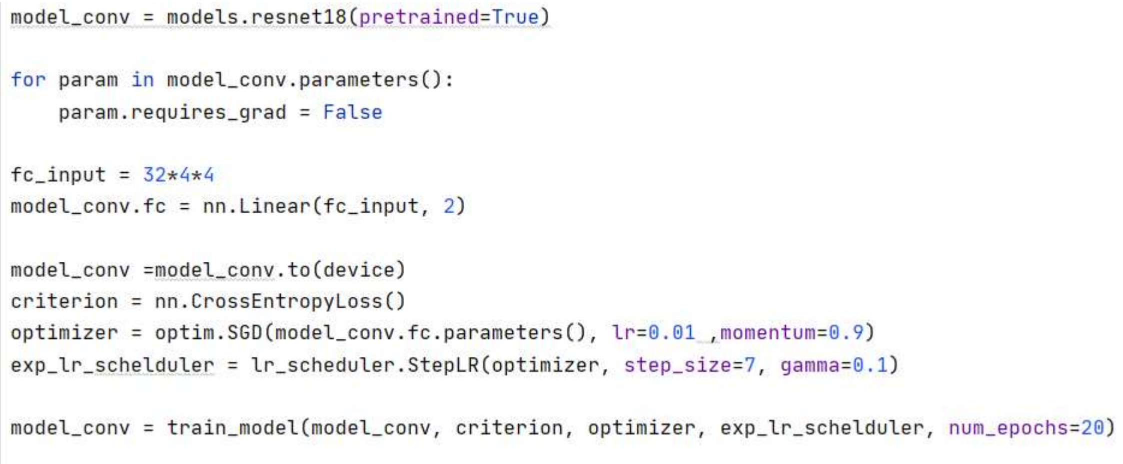
## *구현 방법

데이터셋을 로드하는 파트와 트레이닝 함수 파트는 나만의 네트워크와 동일하다.



```
model_conv = torchvision.models.resnet18(pretrained=True)

model_conv = model_conv.to(device)

criterion = nn.CrossEntropyLoss()

optimizer_conv = optim.SGD(model_conv.parameters(), lr=0.001, momentum=0.9)

exp_lr_scheduler = lr_scheduler.StepLR(optimizer_conv, step_size=7, gamma=0.1)

model_conv = train_model(model_conv, criterion, optimizer_conv,
                         exp_lr_scheduler, num_epochs=20)
```

Strategy 1을 먼저 살펴보면, model_conv라는 이름으로 pretrained된 resnet18을 불러와서 conv파트와 classifiar파트 모두 training 시킬 것 이므로, 어느 부분도 freeze 하지 않고 그대로 최적화를 진행, 바로 training 함수에 대입하여 학습시킨다.

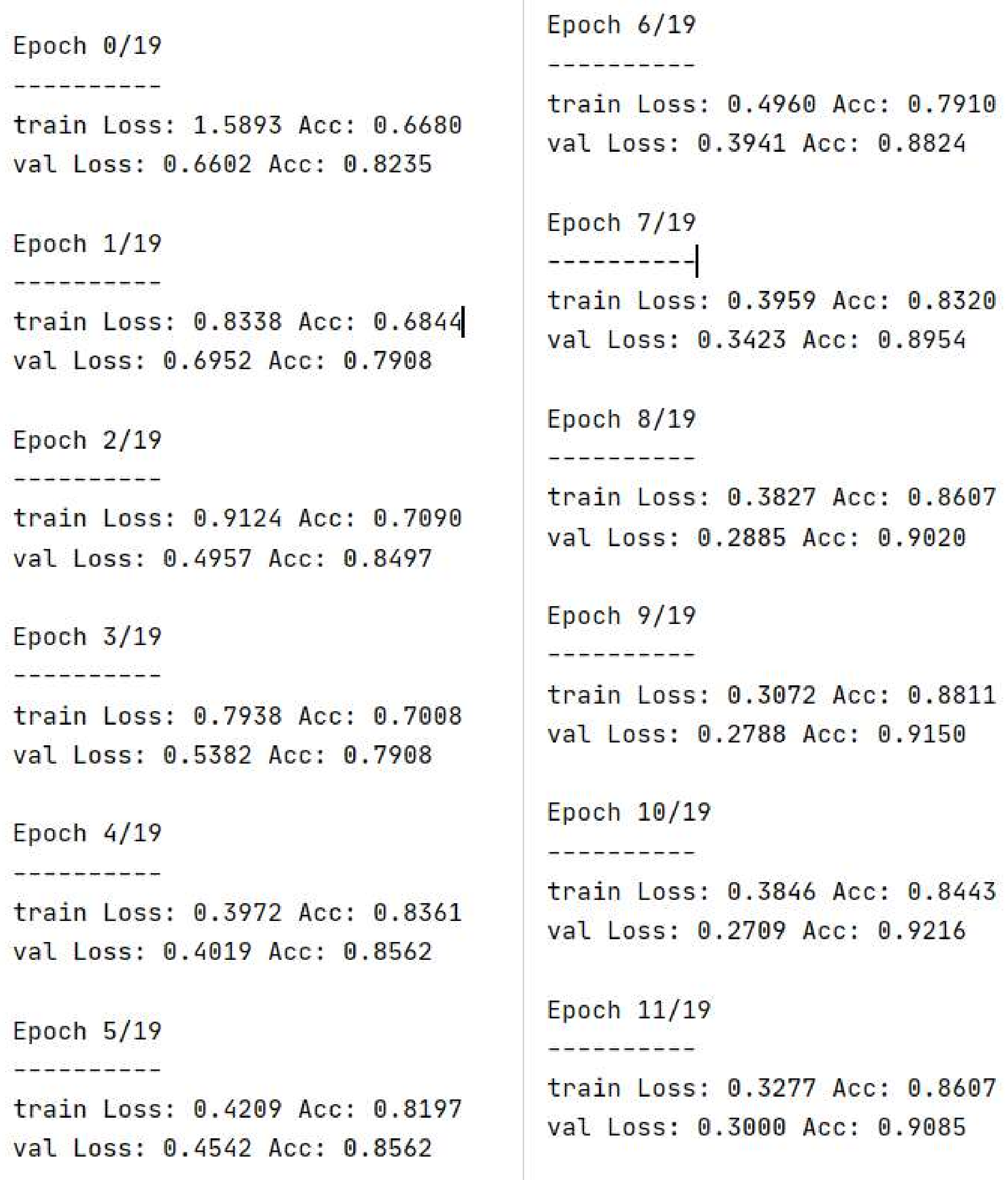


```
model_conv = models.resnet18(pretrained=True)

for param in model_conv.parameters():
    param.requires_grad = False

fc_input = 32*4*4
model_conv.fc = nn.Linear(fc_input, 2)

model_conv =model_conv.to(device)
criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(model_conv.fc.parameters(), lr=0.01 ,momentum=0.9)
exp_lr_schelduler = lr_scheduler.StepLR(optimizer, step_size=7, gamma=0.1)

model_conv = train_model(model_conv, criterion, optimizer, exp_lr_schelduler, num_epochs=20)
```

 위의 코드는 strategy 3인데, 여기서 strategy 1과 다른 점은, strategy 3은 conv파트는 다시 학습시키지 않게 하기위해서 해당 부분을 for문을 통하여 freeze한다는 것 이다. 그리고 model_conv.fc에 resnet18의 classifiar를 따로 불러와서, freeze를 안한 파트인 fc만 최적화를 하고 이를 training 함수에 대입하여 학습시킨다.

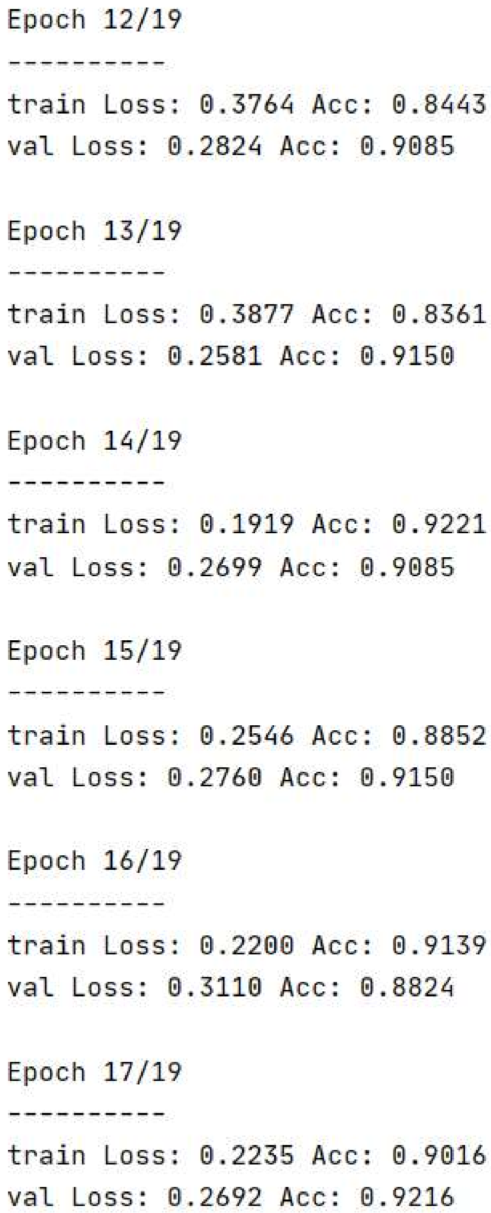
## *과제 결과

-Strategy 1

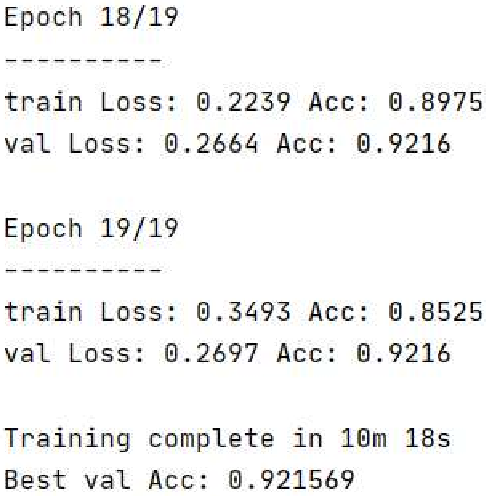


```
Epoch 0/19
----------
train Loss: 1.5893 Acc: 0.6680
val Loss: 0.6602 Acc: 0.8235

Epoch 1/19
----------
train Loss: 0.8338 Acc: 0.6844
val Loss: 0.6952 Acc: 0.7908

Epoch 2/19
----------
train Loss: 0.9124 Acc: 0.7090
val Loss: 0.4957 Acc: 0.8497

Epoch 3/19
----------
train Loss: 0.7938 Acc: 0.7008
val Loss: 0.5382 Acc: 0.7908

Epoch 4/19
----------
train Loss: 0.3972 Acc: 0.8361
val Loss: 0.4019 Acc: 0.8562

Epoch 5/19
----------
train Loss: 0.4209 Acc: 0.8197
val Loss: 0.4542 Acc: 0.8562
```

```
Epoch 6/19
----------
train Loss: 0.4960 Acc: 0.7910
val Loss: 0.3941 Acc: 0.8824

Epoch 7/19
----------
train Loss: 0.3959 Acc: 0.8320
val Loss: 0.3423 Acc: 0.8954

Epoch 8/19
----------
train Loss: 0.3827 Acc: 0.8607
val Loss: 0.2885 Acc: 0.9020

Epoch 9/19
----------
train Loss: 0.3072 Acc: 0.8811
val Loss: 0.2788 Acc: 0.9150

Epoch 10/19
----------
train Loss: 0.3846 Acc: 0.8443
val Loss: 0.2709 Acc: 0.9216

Epoch 11/19
----------
train Loss: 0.3277 Acc: 0.8607
val Loss: 0.3000 Acc: 0.9085
```

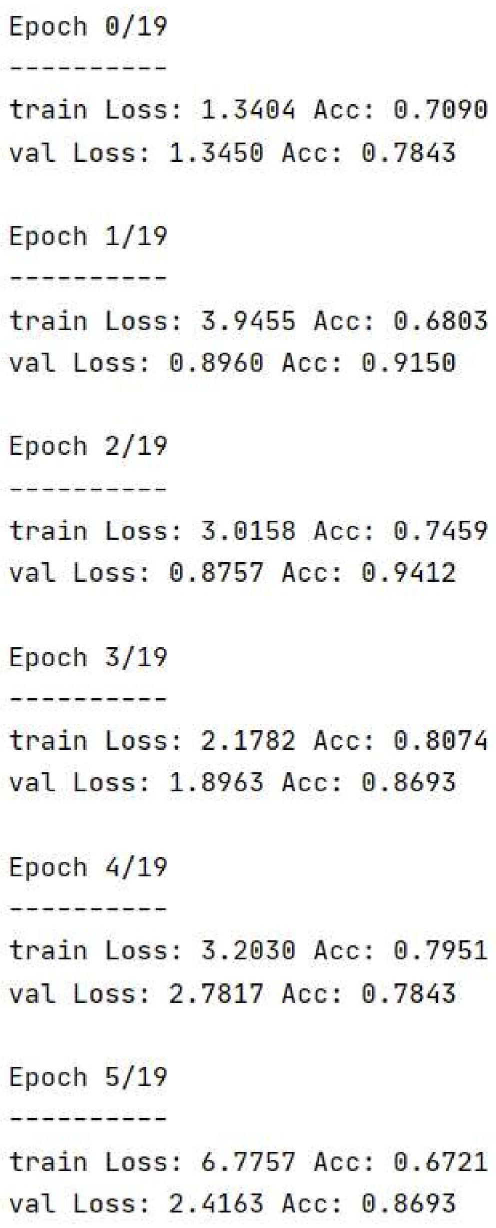


```
Epoch 12/19
----------
train Loss: 0.3764 Acc: 0.8443
val Loss: 0.2824 Acc: 0.9085

Epoch 13/19
----------
train Loss: 0.3877 Acc: 0.8361
val Loss: 0.2581 Acc: 0.9150

Epoch 14/19
----------
train Loss: 0.1919 Acc: 0.9221
val Loss: 0.2699 Acc: 0.9085

Epoch 15/19
----------
train Loss: 0.2546 Acc: 0.8852
val Loss: 0.2760 Acc: 0.9150

Epoch 16/19
----------
train Loss: 0.2200 Acc: 0.9139
val Loss: 0.3110 Acc: 0.8824

Epoch 17/19
----------
train Loss: 0.2235 Acc: 0.9016
val Loss: 0.2692 Acc: 0.9216
```

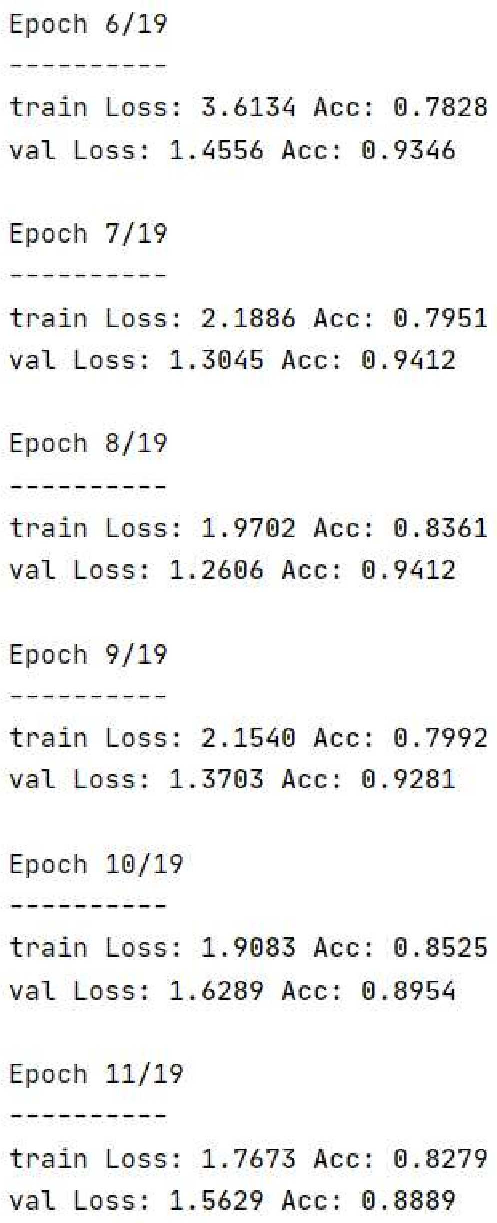


```
Epoch 18/19
----------
train Loss: 0.2239 Acc: 0.8975
val Loss: 0.2664 Acc: 0.9216

Epoch 19/19
----------
train Loss: 0.3493 Acc: 0.8525
val Loss: 0.2697 Acc: 0.9216

Training complete in 10m 18s
Best val Acc: 0.921569
```

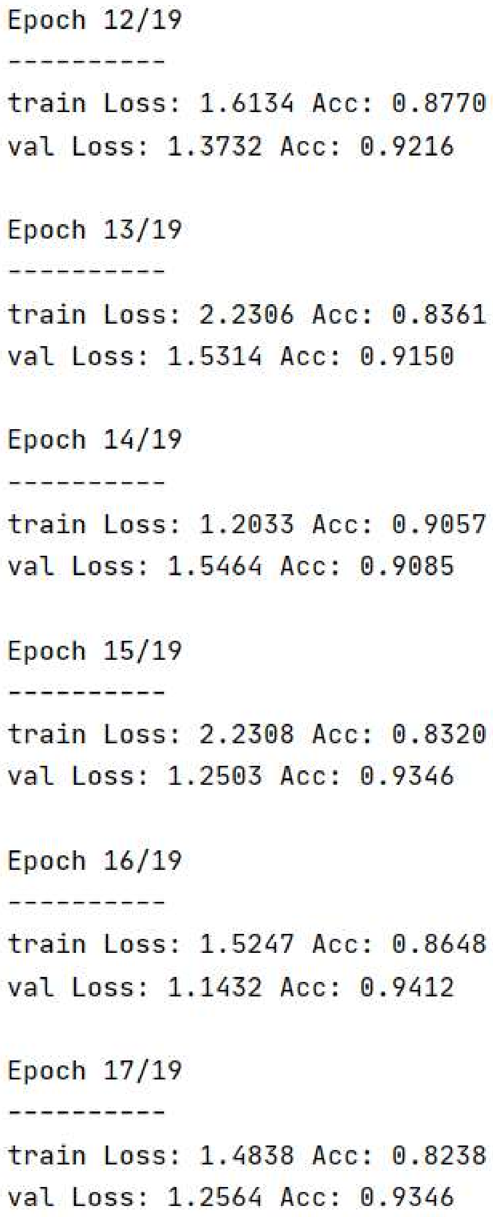
-Strategy 3



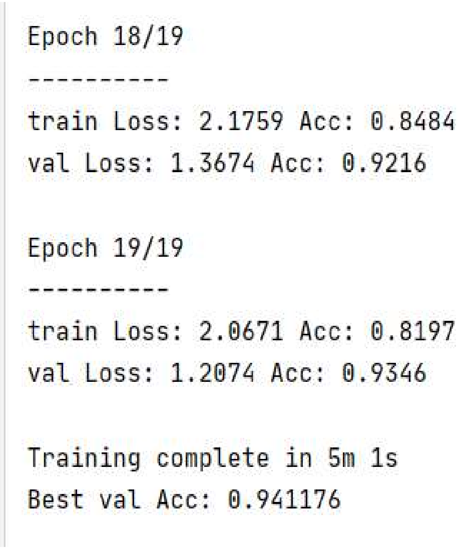
```
Epoch 0/19
----------
train Loss: 1.3404 Acc: 0.7090
val Loss: 1.3450 Acc: 0.7843

Epoch 1/19
----------
train Loss: 3.9455 Acc: 0.6803
val Loss: 0.8960 Acc: 0.9150

Epoch 2/19
----------
train Loss: 3.0158 Acc: 0.7459
val Loss: 0.8757 Acc: 0.9412

Epoch 3/19
----------
train Loss: 2.1782 Acc: 0.8074
val Loss: 1.8963 Acc: 0.8693

Epoch 4/19
----------
train Loss: 3.2030 Acc: 0.7951
val Loss: 2.7817 Acc: 0.7843

Epoch 5/19
----------
train Loss: 6.7757 Acc: 0.6721
val Loss: 2.4163 Acc: 0.8693
```



```
Epoch 6/19
----------
train Loss: 3.6134 Acc: 0.7828
val Loss: 1.4556 Acc: 0.9346

Epoch 7/19
----------
train Loss: 2.1886 Acc: 0.7951
val Loss: 1.3045 Acc: 0.9412

Epoch 8/19
----------
train Loss: 1.9702 Acc: 0.8361
val Loss: 1.2606 Acc: 0.9412

Epoch 9/19
----------
train Loss: 2.1540 Acc: 0.7992
val Loss: 1.3703 Acc: 0.9281

Epoch 10/19
----------
train Loss: 1.9083 Acc: 0.8525
val Loss: 1.6289 Acc: 0.8954

Epoch 11/19
----------
train Loss: 1.7673 Acc: 0.8279
val Loss: 1.5629 Acc: 0.8889
```



```
Epoch 12/19
----------
train Loss: 1.6134 Acc: 0.8770
val Loss: 1.3732 Acc: 0.9216

Epoch 13/19
----------
train Loss: 2.2306 Acc: 0.8361
val Loss: 1.5314 Acc: 0.9150

Epoch 14/19
----------
train Loss: 1.2033 Acc: 0.9057
val Loss: 1.5464 Acc: 0.9085

Epoch 15/19
----------
train Loss: 2.2308 Acc: 0.8320
val Loss: 1.2503 Acc: 0.9346

Epoch 16/19
----------
train Loss: 1.5247 Acc: 0.8648
val Loss: 1.1432 Acc: 0.9412

Epoch 17/19
----------
train Loss: 1.4838 Acc: 0.8238
val Loss: 1.2564 Acc: 0.9346
```



```
Epoch 18/19
----------
train Loss: 2.1759 Acc: 0.8484
val Loss: 1.3674 Acc: 0.9216

Epoch 19/19
----------
train Loss: 2.0671 Acc: 0.8197
val Loss: 1.2074 Acc: 0.9346

Training complete in 5m 1s
Best val Acc: 0.941176
```

- 결과 비교표

| Epoch | 본인 네트워크 | | Strategy 1 | | Strategy 3 | |
|---|---|---|---|---|---|---|
| | Val loss | Acc | Val loss | Acc | Val loss | Acc |
| 1 | 0.6618 | 0.5882 | 0.6602 | 0.8235 | 1.3450 | 0.7843 |
| 2 | 0.7695 | 0.5425 | 0.6952 | 0.7908 | 0.8960 | 0.9150 |
| 3 | 0.6396 | 0.6340 | 0.4957 | 0.8497 | 0.8757 | 0.9412 |
| 4 | 0.6475 | 0.6078 | 0.5382 | 0.7908 | 1.8963 | 0.8693 |
| 5 | 0.6485 | 0.6144 | 0.4019 | 0.8562 | 2.7817 | 0.7843 |
| 6 | 0.7338 | 0.5359 | 0.4542 | 0.8562 | 2.4163 | 0.8693 |
| 7 | 0.6579 | 0.6013 | 0.3941 | 0.8824 | 1.4556 | 0.9346 |
| 8 | 0.6553 | 0.6144 | 0.3423 | 0.8954 | 1.3045 | 0.9412 |
| 9 | 0.6497 | 0.6144 | 0.2885 | 0.9020 | 1.2606 | 0.9412 |
| 10 | 0.6472 | 0.6209 | 0.2788 | 0.9150 | 1.3703 | 0.9281 |
| 11 | 0.6472 | 0.6275 | 0.2709 | 0.9216 | 1.6289 | 0.8954 |
| 12 | 0.6440 | 0.6275 | 0.3000 | 0.9085 | 1.5629 | 0.8889 |
| 13 | 0.6417 | 0.6275 | 0.2824 | 0.9085 | 1.3732 | 0.9216 |
| 14 | 0.6442 | 0.6275 | 0.2581 | 0.9150 | 1.5314 | 0.9150 |
| 15 | 0.6442 | 0.6275 | 0.2699 | 0.9085 | 1.5464 | 0.9085 |
| 16 | 0.6443 | 0.6275 | 0.2760 | 0.9150 | 1.2503 | 0.9346 |
| 17 | 0.6445 | 0.6275 | 0.3110 | 0.8824 | 1.1432 | 0.9412 |
| 18 | 0.6443 | 0.6275 | 0.2692 | 0.9216 | 1.2564 | 0.9346 |
| 19 | 0.6444 | 0.6275 | 0.2664 | 0.9216 | 1.3674 | 0.9216 |
| 20 | 0.6443 | 0.6275 | 0.2697 | 0.9216 | 1.2074 | 0.9346 |