플레이데이터 DA 28기 3차 미니 프로젝트 2조

CNN 날씨 분류

2조 이우진 김수환 이정준 김수민

기간 : 23. 4. 13 ~ 20

1. 주제 설명
2. 전처리
   2-1. 데이터 분리
   2-2. 데이터 증강
3. 모델 훈련
   ALEXNET, VGGNET, GOOGLENET, RESNET
4. 성능 향상
   4-1. 데이터 증강
   4-2. LR&BATCH 변경
   4-3. SCHEDULER 변경
   4-4. CLASSIFIER-BLOCK 변경
5. 결과

1. 주제 설명

1-1. 주제 설명

Rainy

Cloudy

Foggy

Snowy

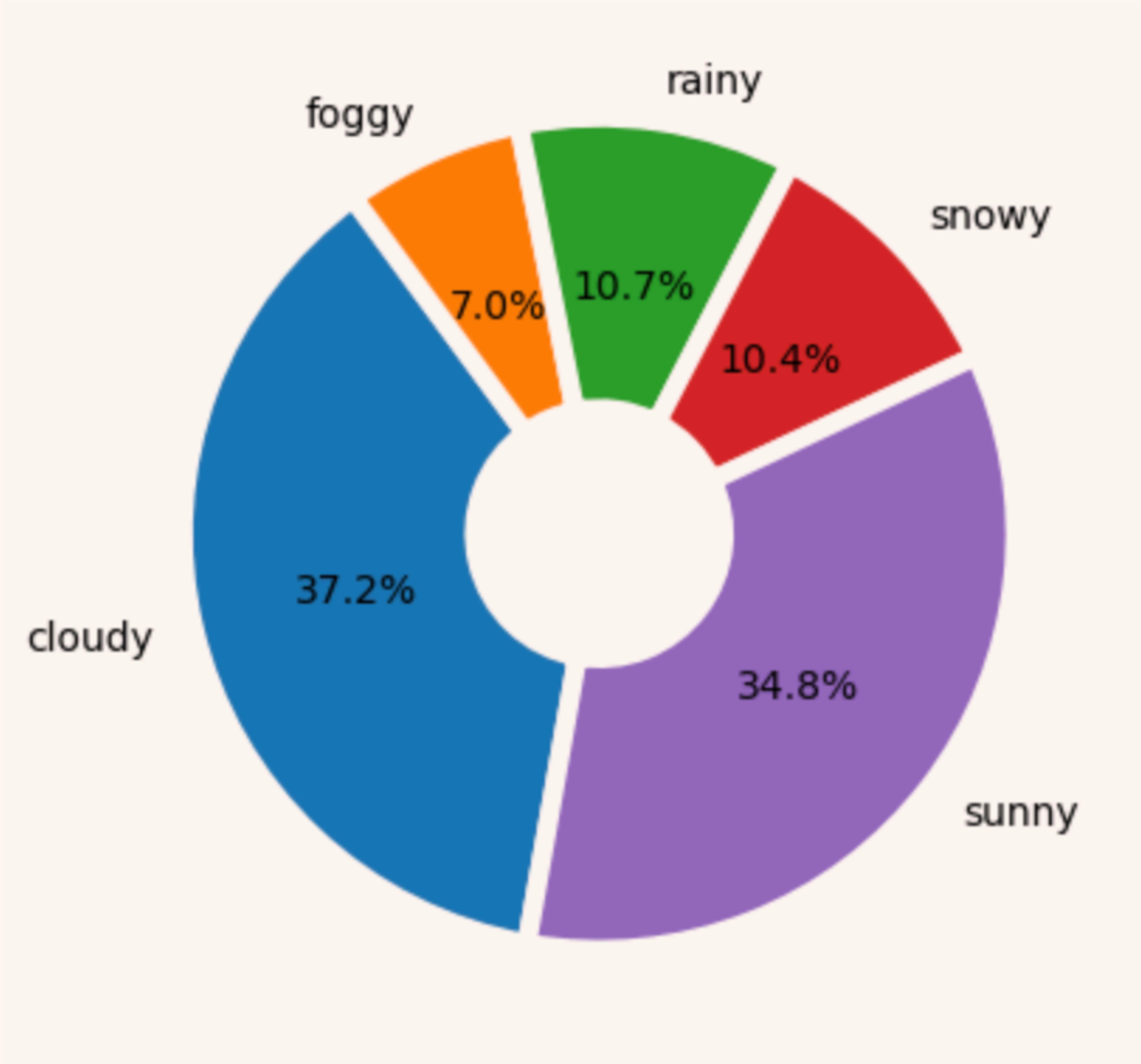
Sunny

2. 전처리

2-1. 데이터 분리

Cat and dog 데이터셋의 폴더 구조

Weather 데이터셋의 폴더 구조



2-1. 데이터 분리 (리스트 버전)

```python
def train_test_valid(dataset, valid_ratio, test_ratio):
    cloudy_list = []
    foggy_list = []
    rainy_list = []
    snowy_list = []
    sunny_list = []

    train_index = []
    valid_index = []
    test_index = []

    for i in range(len(dataset)):
        label = dataset[i][1]
        if label == 0:
            cloudy_list.append(i)
        elif label == 1:
            foggy_list.append(i)
        elif label == 2:
            rainy_list.append(i)
        elif label == 3:
            snowy_list.append(i)
        elif label == 4:
            sunny_list.append(i)

    for cnt, weather_index in enumerate((cloudy_list, foggy_list, rainy_list, snowy_list, sunny_list)):
        train_num = round(round(len(weather_index)*(1.0 - test_ratio)) * (1.0 - valid_ratio))
        valid_num = round(round(len(weather_index)*(1.0 - test_ratio)) * valid_ratio)
        test_num = round(len(weather_index)*test_ratio)

        # print(cnt, len(weather_index), train_num, valid_num, test_num)

        train_index += weather_index[:train_num]
        valid_index += weather_index[train_num: train_num+valid_num]
        test_index += weather_index[train_num+valid_num: train_num+valid_num+test_num]

    return train_index, valid_index, test_index
```

```python
train_index, valid_index, test_index = train_test_valid(dataset_custom, valid_ratio=0.2, test_ratio=0.2)

print(len(train_index), len(valid_index), len(test_index))

trainset = Subset(dataset_custom, train_index)
validset = Subset(dataset_custom, valid_index)
testset = Subset(dataset_custom, test_index)
```

```
11546 2886 3607
```

정답 레이블별로 리스트를 만들어 인덱스를 저장

미리 지정해 둔 비율에 맞춰 데이터셋의 길이를 지정

리턴할 train_index, valid_index, test_index

2-1. 데이터 분리 (데이터 프레임 버전)

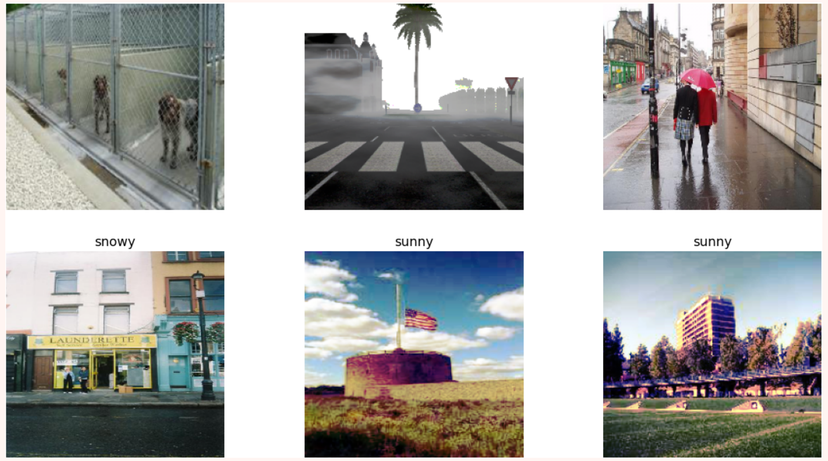
```python
class mySplitFP():
    def __init__(self, root, labels_map):
        self.root, self.labels_map = root, labels_map
        self.dfs = None

    def __call__(self):
        return self.split_join_dfs()

    def create_dfs(self):
        cloudy_df = pd.DataFrame(data={'filepath':glob.glob(self.root+self.labels_map[0]+'/*.*'), 'label':0})
        foggy_df = pd.DataFrame(data={'filepath':glob.glob(self.root+self.labels_map[1]+'/*.*'), 'label':1})
        rainy_df = pd.DataFrame(data={'filepath':glob.glob(self.root+self.labels_map[2]+'/*.*'), 'label':2})
        snowy_df = pd.DataFrame(data={'filepath':glob.glob(self.root+self.labels_map[3]+'/*.*'), 'label':3})
        sunny_df = pd.DataFrame(data={'filepath':glob.glob(self.root+self.labels_map[4]+'/*.*'), 'label':4})
        self.dfs = [cloudy_df, foggy_df, rainy_df, snowy_df, sunny_df]

    def split_join_dfs(self, train_p=0.65, valid_p=0.15, test_p=0.2):
        self.create_dfs()
        idxs_dict = {'train':[], 'valid':[], 'test':[]}
        for df in self.dfs:
            train_n, valid_n = len(df)//(1/train_p), len(df)//(1/valid_p)
            df['cut'] = pd.cut(df.index, bins=[-np.inf, train_n, train_n+valid_n, np.inf], labels=['train','valid','test'])
            for k in idxs_dict.keys():
                idxs_dict[k].append(df.loc[df['cut']==k, ['filepath','label']])
        for k,v in idxs_dict.items():
            if k=='train': train_df = pd.concat(v, axis=0).reset_index()
            elif k=='valid': valid_df = pd.concat(v, axis=0).reset_index()
            elif k=='test': test_df = pd.concat(v, axis=0).reset_index()
        return train_df, valid_df, test_df
```

```python
trainset = mycreateDataset(train_df, transform)
validset = mycreateDataset(valid_df, transform)
testset = mycreateDataset(test_df, transform)
```

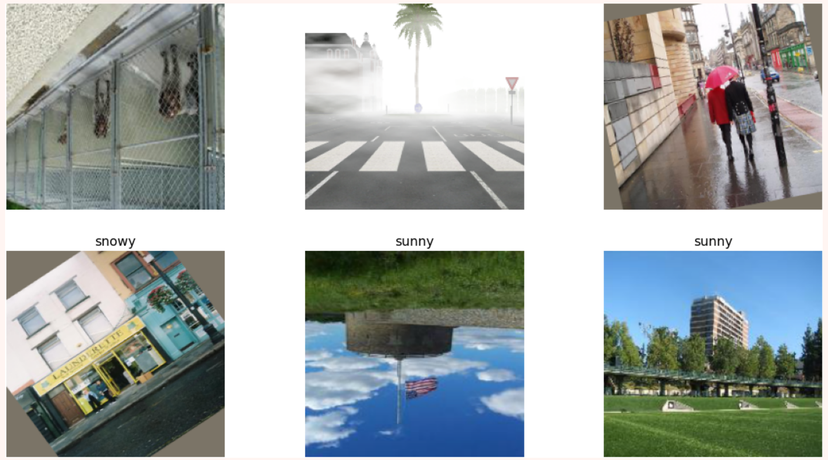
원본 데이터에서 정답 레이블에 맞춰 각각의 데이터 프레임 생성

비율을 정해 각 데이터의 개수를 저장하는 train_n, valid_n, test_n 생성

2-2. 데이터 증강: Transform



대비 + 평활화



회전 + 상하좌우 반전
(Best)

3. 모델 훈련

3. 모델: AlexNet, VggNet, GoogleNet, ResNet

배치 사이즈	학습률	AlexNet	ResNet	GoogLeNet	VGGNet
16	0.0001				16_bn L : 0.622, Acc : 0.836
	0.001	alexNet Test Loss : 0.901 Test Accuracy : 0.681	101 L : 0.377, Acc : 0.864 152 L : 0.375, Acc : 0.865	L : 0.96 Acc : 0.626	
	0.01		101 L : 0.384, Acc : 0.860 152 L : 0.439, Acc : 0.845		
32	0.0001				16_bn L : 0.586, Acc : 0.846
	0.001		152 L : 0.361, Acc : 0.871		16_bn L : 0.631, Acc : 0.841
	0.01				16_bn L : 1.394, Acc : 0.371
64	0.0001				16_bn L : 0.591, Acc : 0.836
	0.001		152 L : 0.387, Acc : 0.842		16_bn L : 0.633, Acc : 0.838
	0.01				16_bn L : 1.373, Acc : 0.388

여러 가지의 **ResNet** 모델 학습 진행

ResNet - 하이퍼 파라미터 세부 조정

모델	스케줄러	배치사이즈	Test Loss	Test Accuracy	전처리기
ResNet34	Cosine(10) (lr = 0.001)	16	0.495	0.809	
	StepLR (lr = 0.001)	16	0.490	808	
	MultiStep (Lr = 0.001)	16	0.475	0.814	
	StepLR (Lr = 0.001)	32	0.485	0.808	
	CosineAnnealing WarmRestarts (Lr = 0.0001)	32	0.481	0.810	
	CosineAnnealing WarmRestarts (Lr = 0.001)	32	0.499	0.805	
	MultiStep (Lr = 0.001)	32	0.492	0.811	
ResNet50	ReduceLR (lr = 0.001)	32	0.380	0.858	
	ReduceLR (lr = 0.001)	32	0.548	0.783	colorjitter
	Cosine(10) (lr = 0.001)	32	0.405	0.843	
	CosineAnnealing WarmRestarts (Lr = 0.001)	32	0.428	0.850	
	StepLR (lr = 0.001)	32	0.413	0.840	
	MultiStep (Lr = 0.001)	32	0.399	0.847	
	MultiStep (Lr = 0.0001)	32	0.381	0.857	
ResNet101	ReduceLR (lr = 0.001)	16	0.377	0.864	horiflip(p=0.3)
	ReduceLR (lr = 0.01)	16	0.384	0.860	horiflip(p=0.3)
	ReduceLR (lr = 0.001)	16	0.491	0.816	
	ReduceLR (lr = 0.001)	16	0.504	0.807	Resize, RandomEqualize(p=0.3), ColorJitter(brightness=0.4)
	ReduceLR (lr = 0.001)	32	0.404	0.849	vertiflip, horiflip, rotate
	ReduceLR (lr = 0.001)	32	0.401	0.844	autocontrast, hori, verti, rotate
	ReduceLR (lr = 0.0001)	32	0.412	0.854	autocontrast, hori, verti, rotate
	ReduceLR (lr = 0.0001)	32	0.399	0.851	vertiflip, horiflip, rotate
	Cosine(10) (Lr = 0.001)	16	0.414	0.849	Resize, RandomAutocontrast(p=0.4), ToTensor
	Cosine(10) (Lr = 0.0001)	16	0.373	0.857	Resize, RandomAutocontrast(p=0.4), ToTensor
	Cosine(10) (Lr = 0.001)	16	0.453	0.830	Resize, RandomEqualize(p=0.4), ToTensor
	Cosine(10) (Lr = 0.0001)	16	0.433	0.838	Resize, RandomEqualize(p=0.4), ToTensor

ResNet34 →

ResNet50 →

ResNet101 →

•
•
•

•
•
•

	Cosine(10) (Lr = 0.0001)	16	0.433	0.838	Resize, RandomEqualize(p=0.4), ToTensor
	Cosine(10) (Lr = 0.0001)	32	0.383	0.863	***Best*** Resize, RandomAutocontrast(p=0.4), ToTensor
	Cosine(10) (Lr = 0.001)	32	0.457	0.820	Resize, RandomEqualize(p=0.4), ToTensor
	Cosine(10) (Lr = 0.0001)	32	0.439	0.835	Resize, RandomEqualize(p=0.4), ToTensor
	Cosine(10) (Lr = 0.001)	32	0.415	0.839	Resize, RandomAutocontrast(p=0.4), ToTensor
ResNet152	ReduceLR (lr = 0.001)	16	0.375	0.865	horiflip(p=0.3)
	ReduceLR (lr = 0.001)	16	0.361	0.865	horiflip(p=0.3), + fc: 3 layers
	ReduceLR (lr = 0.001)	16	0.363	0.864	vertiflip, horiflip, rotate, + fc: 3 layers
	ReduceLR (lr = 0.01)	16	0.450	0.853	horiflip(p=0.3)
	ReduceLR (lr = 0.0001)	16	0.371	0.860	vertiflip, horiflip, rotate, + fc: 3 layers
	ReduceLR (lr = 0.01)	32	0.434	0.834	vertiflip, horiflip, rotate
	ReduceLR (lr = 0.001)	32	0.361	0.845	horiflip(p=0.3)
	ReduceLR (lr = 0.001)	32	0.361	0.871	vertiflip, horiflip, rotate
	Cosine(50) (lr=0.001)	32	0.411	0.844	vertiflip, horiflip, rotate
	Cosine(10) (Lr = 0.0001)	16	0.387	0.844	vertiflip, horiflip, rotate, + fc: 3 layers
	Cosine(10) (Lr = 0.0001)	32	0.398	0.847	vertiflip, horiflip, rotate, + fc: 3 layers
	Cosine(10) (lr = 0.001)	32	0.383	0.853	vertiflip, horiflip, rotate
	CosineRestart (lr = 0.001)	32	0.409	0.849	vertiflip, horiflip, rotate
	StepLR (lr = 0.001)	32	0.402	0.849	vertiflip, horiflip, rotate
	MultiStep (Lr = 0.001)	32	0.405	0.843	vertiflip, horiflip, rotate
	ReduceLR (lr = 0.01)	64	0.406	0.847	vertiflip, horiflip, rotate
	ReduceLR (lr = 0.001)	64	0.387	0.842	vertiflip, horiflip, rotate

← ResNet152

4. 성능 향상

4-1. 데이터 증강

: Transform

4-2. learning rate, batch size 변경

: 0.001, 0.0001 / 16, 32, 64

4-3. classifier block 변경

4-4. scheduler 변경

ReduceLROnPlateau **CosineAnnealingL**

CosineAnnealingWarmRestart **StepLR**

MultiStepLR

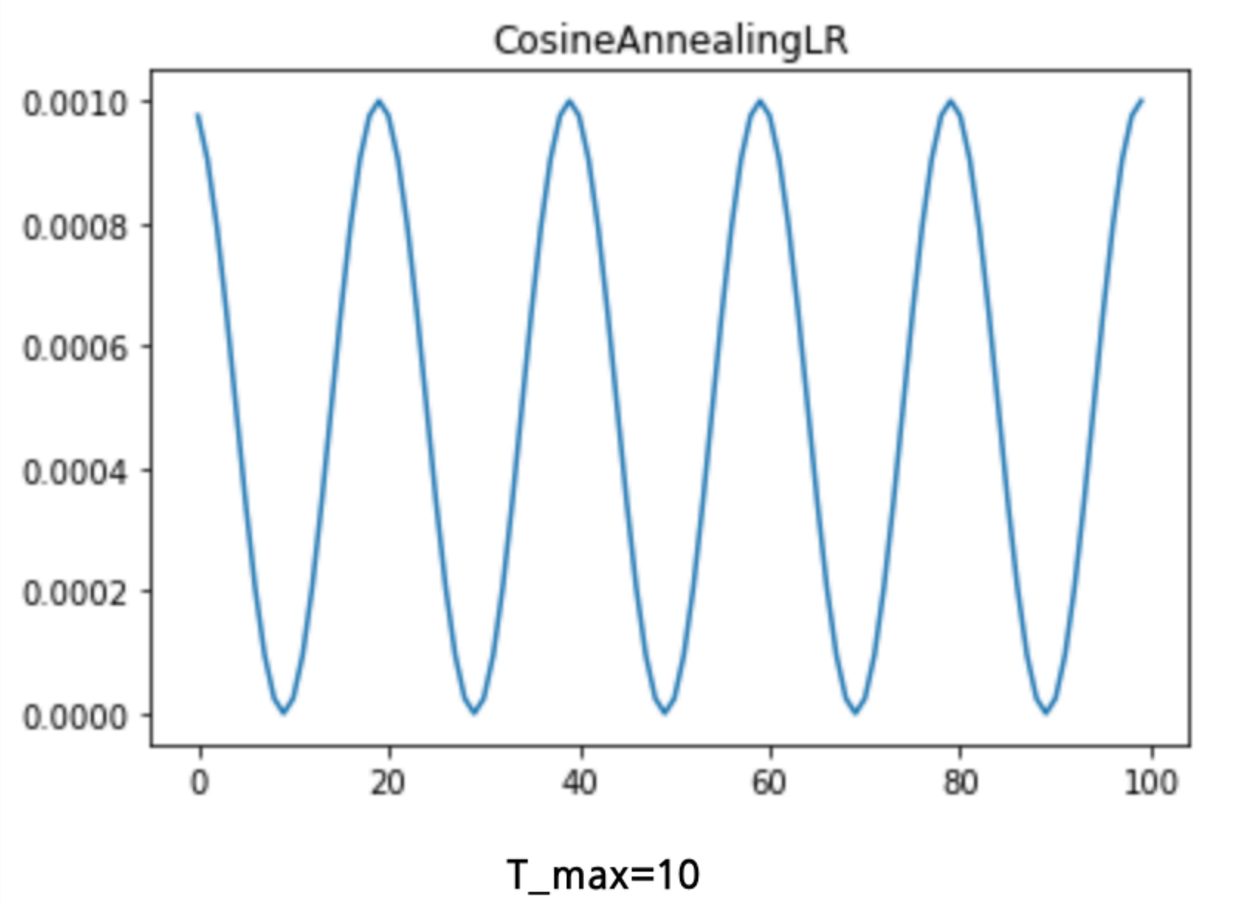
4-3. classifier block 변경

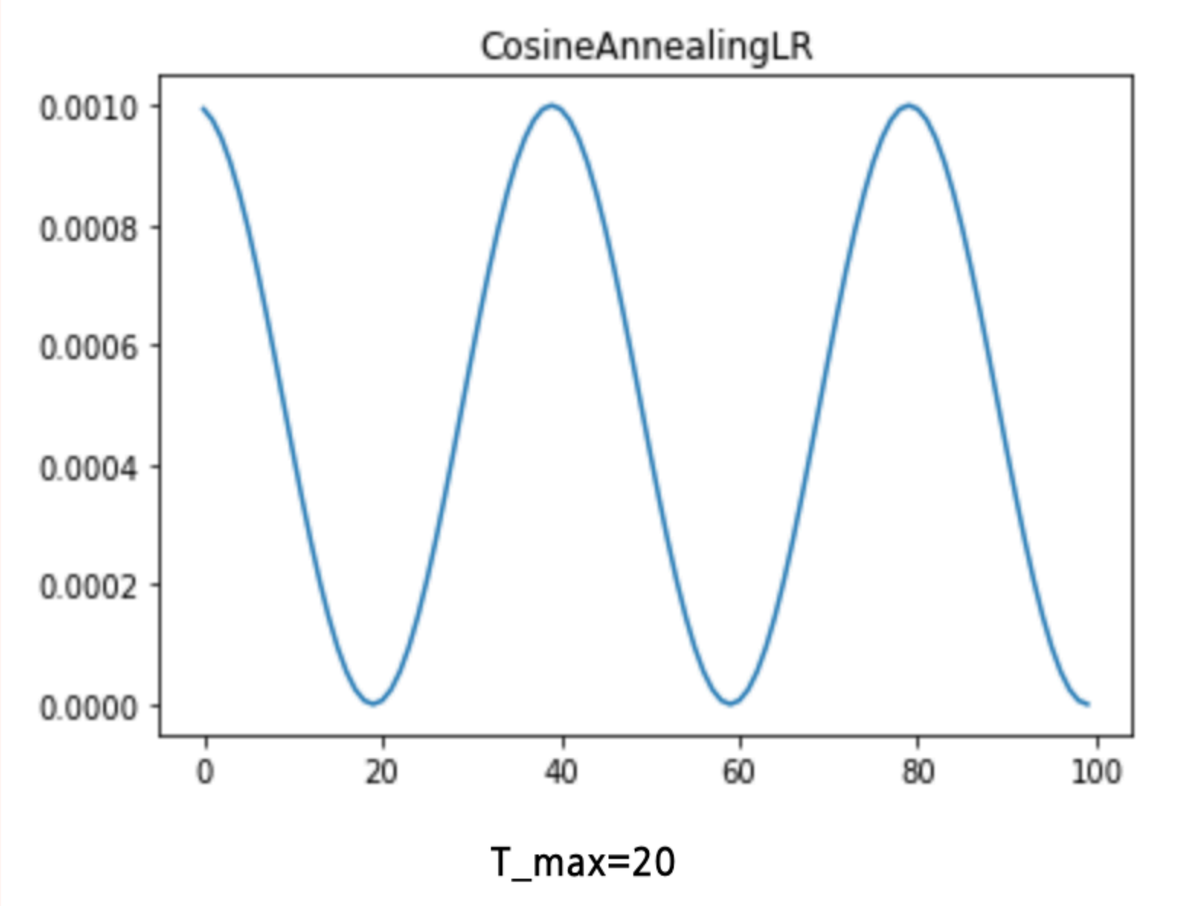
```
model.fc = nn.Sequential(
                    nn.Linear(2048,512),
                    nn.BatchNorm1d(512),
                    nn.ReLU(),
                    nn.Linear(512,5))
```

```
model.fc = nn.Sequential(nn.Linear(2048,512),
                         nn.BatchNorm1d(512),
                         nn.ReLU(),
                         nn.Dropout(0.3),
                         nn.Linear(512,248),
                         nn.BatchNorm1d(248),
                         nn.ReLU(),
                         nn.Linear(248,5))
```

스케줄러 - CosineAnnealingLR

Cosine함수의 개형을 따라가며 최대로 떨어질 수 있는 학습률(eta_min)까지 학습률을 떨어뜨리고 다시 초기 학습률까지 상승시키기를 반복한다.
scheduler = optim.lr_scheduler.CosineAnnealingLR(optimizer, T_max=50, eta_min=0)
T_max = 최대 iteration
eta_min = 최소로 떨어질 수 있는 learning rate





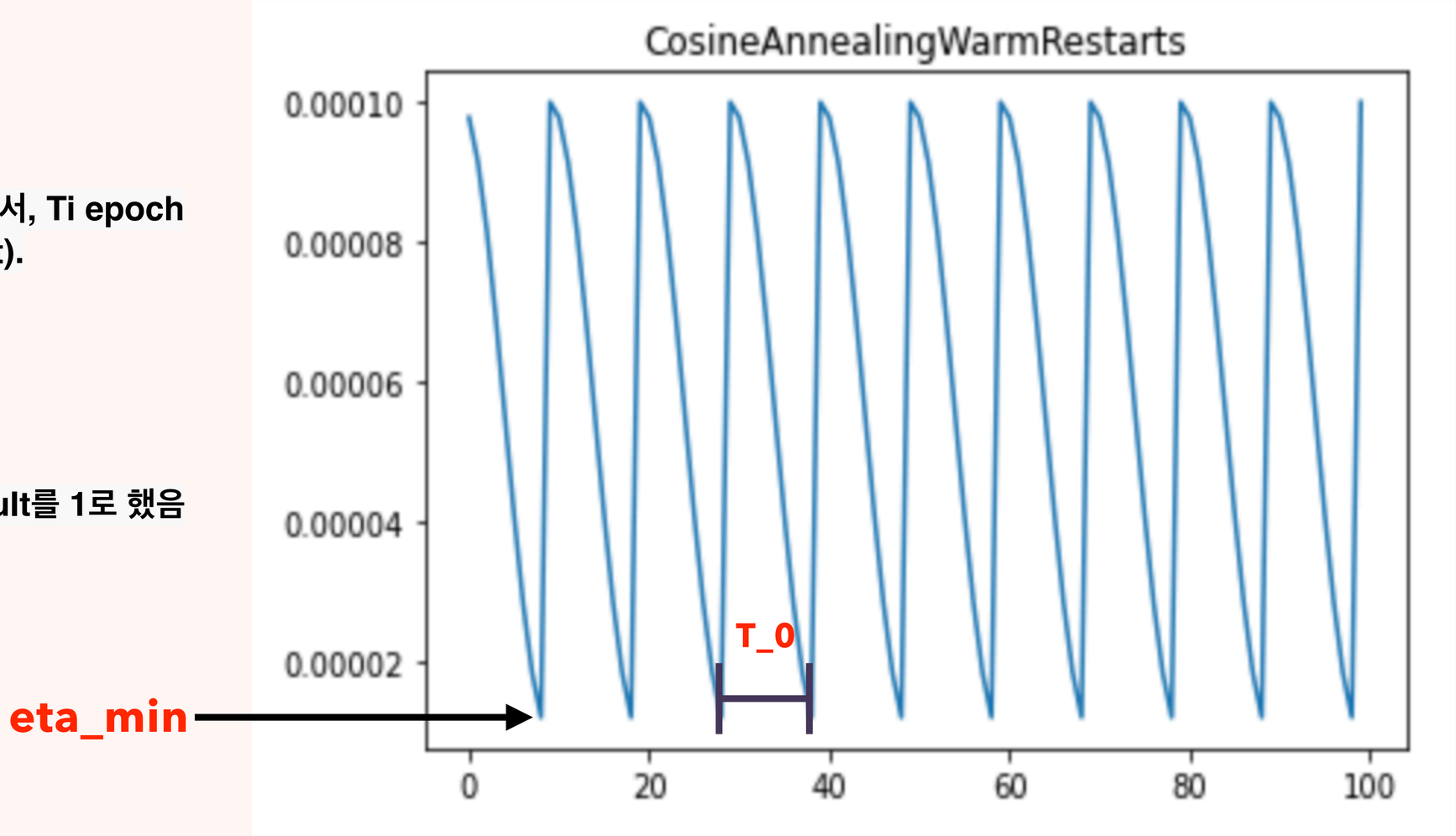
스케줄러 - CosineAnnealingWarmRestart

optimizer, T_0=10, T_mult=1, eta_min=0.00001

CosineAnnealing과 동일하게 진행되면서, Ti epoch 에서 다시 최대 학습률로 상승한다(restart).

T_0 : 한번 restart하기까지의 주기
T_mult : 주기를 늘리는 비율
eta_min : 최소 학습률

CosineAnnealing과 비교를 위해 T_mult를 1로 했음

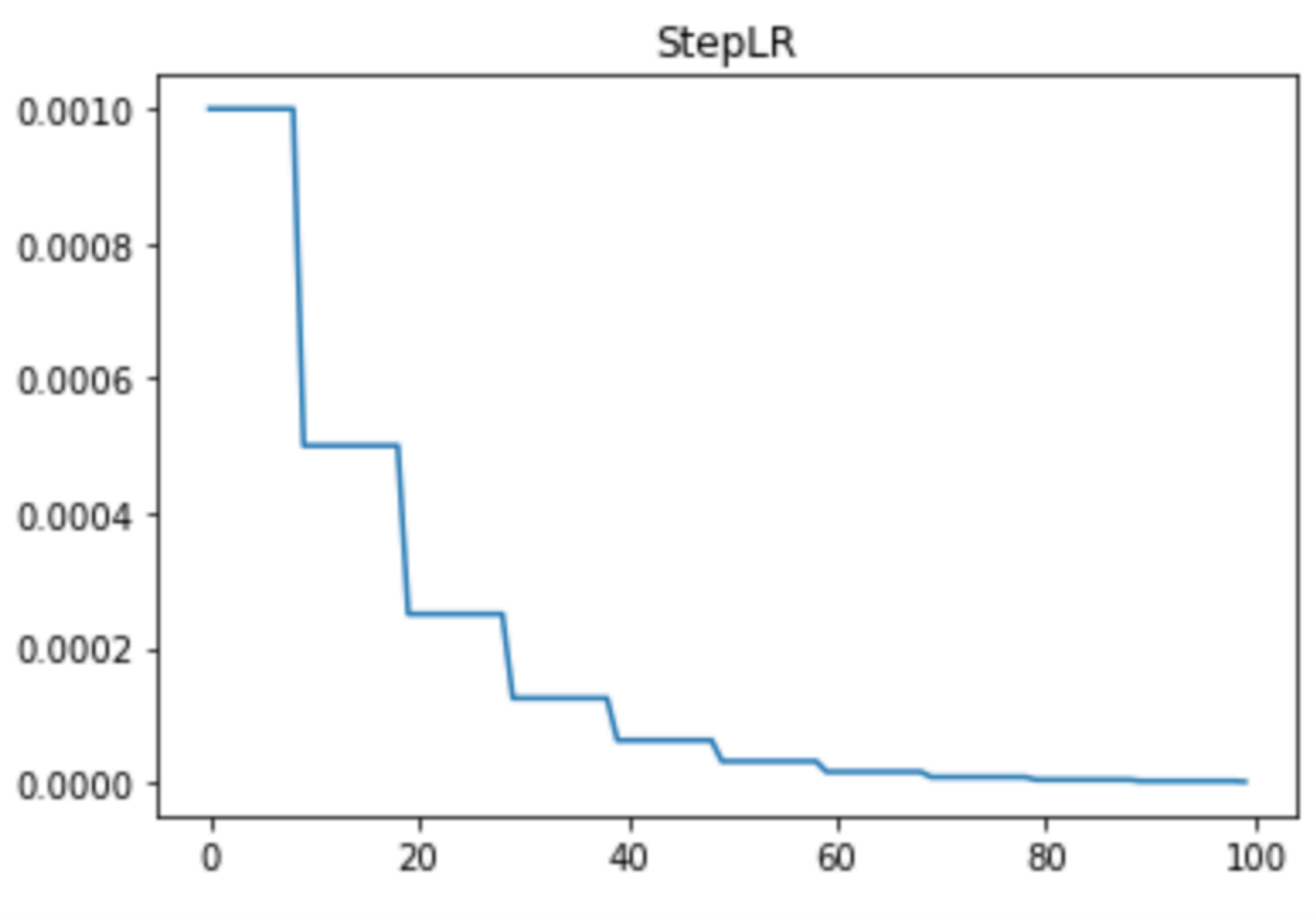


스케줄러 - StepLR

지정된 step_size에서 학습률을 감소시킨다.

step_size : 매 step_size마다 학습률을 감소시킨다.

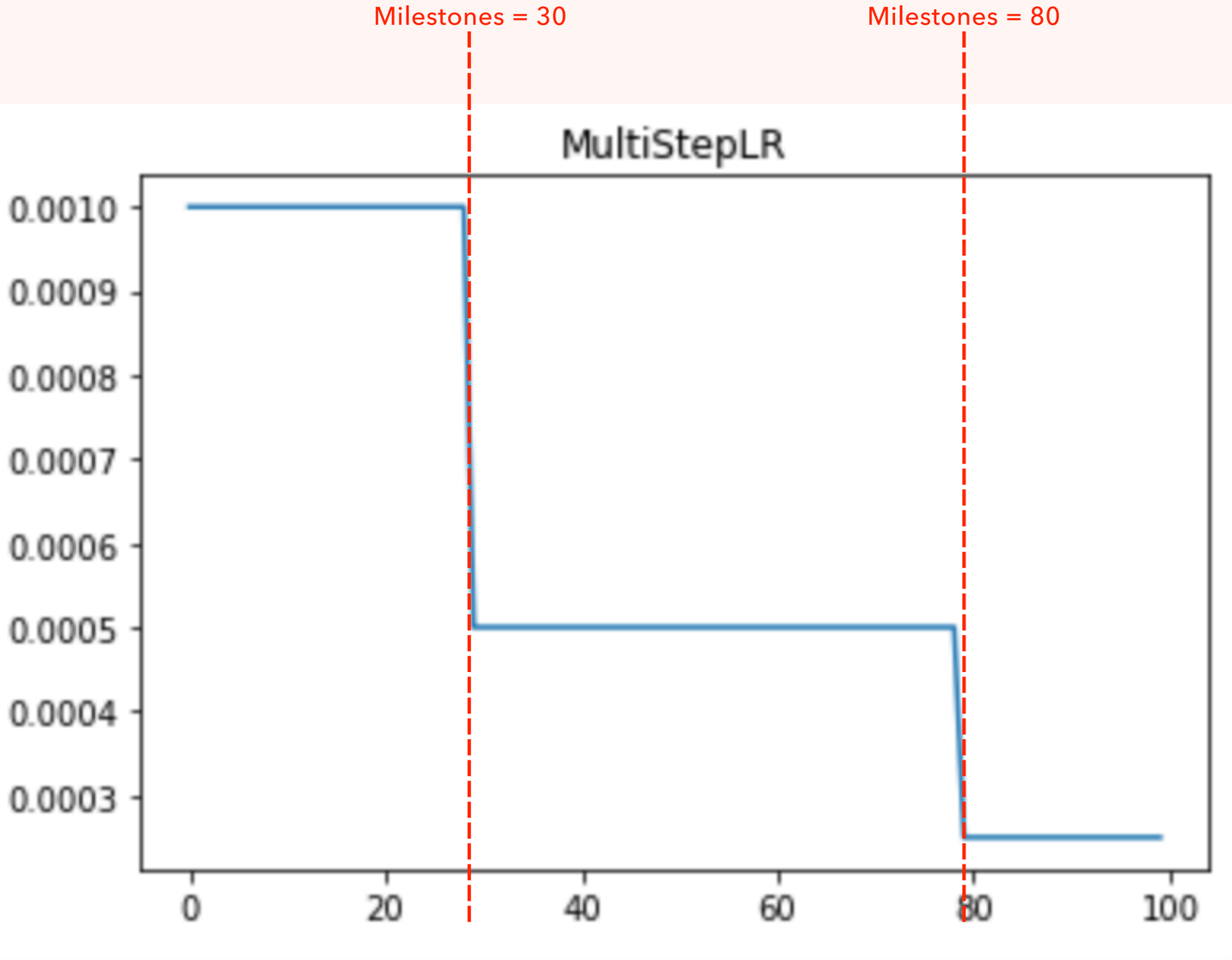
gamma : 학습률을 감소시킬 비율



스케줄러 - MultiStepLR

Milestones : 학습률을 감소시킬 epoch지점을 설정
(그래프 기준 30, 80에서 학습률 감소)

gamma : 학습률을 감소시킬 비율



5.결과

5. 결과 : ResNet 실행 표(하이스코어)

ResNet - high score(0.38 down / 0.86 up)

모델	스케줄러	배치사이즈	Test Loss	Test Accuracy	전처리기
ResNet50	ReduceLR (lr = 0.001)	32	0.380	0.858	
ResNet101	ReduceLR (lr = 0.001)	16	0.377	0.864	horiflip(p=0.3)
	ReduceLR (lr = 0.01)	16	0.384	0.860	horiflip(p=0.3)
	Cosine(10) (Lr = 0.0001)	16	0.373	0.857	Resize, RandomAutocontrast(p =0.4), ToTensor
	Cosine(10) (Lr = 0.0001)	32	0.383	0.863	***Best*** Resize, RandomAutocontrast(p=0.4), ToTensor
ResNet152	ReduceLR (lr = 0.001)	16	0.375	0.865	horiflip(p=0.3)
	ReduceLR (lr = 0.001)	16	0.361	0.865	horiflip(p=0.3), + fc: 3 layers
	ReduceLR (lr = 0.001)	16	0.363	0.864	vertiflip, horiflip, rotate, + fc: 3 layers
	ReduceLR (lr = 0.0001)	16	0.371	0.860	vertiflip, horiflip, rotate, + fc: 3 layers
	ReduceLR (lr = 0.001)	32	0.361	0.845	horiflip(p=0.3)
	ReduceLR (lr = 0.001)	32	0.361	0.871	vertiflip, horiflip, rotate

스케줄러 5가지 시도 : 하이스코어 거르고 난 결과에서 남은 스케줄러가 101에서는 리듀스와 코사인, 152에서는 리듀스

5. 결과 : ResNet 실행 표(하이스코어)

ResNet - high score(0.38 down / 0.86 up)

모델	스케줄러	배치사이즈	Test Loss	Test Accuracy	전처리기
ResNet50	ReduceLR (lr = 0.001)	32	0.380	0.858	
ResNet101	ReduceLR (lr = 0.001)	16	0.377	0.864	horiflip(p=0.3)
	ReduceLR (lr = 0.01)	16	0.384	0.860	horiflip(p=0.3)
	Cosine(10) (Lr = 0.0001)	16	0.373	0.857	Resize, RandomAutocontrast(p =0.4), ToTensor
	Cosine(10) (Lr = 0.0001)	32	0.383	0.863	***Best*** Resize, RandomAutocontrast(p=0.4), ToTensor
ResNet152	ReduceLR (lr = 0.001)	16	0.375	0.865	horiflip(p=0.3)
	ReduceLR (lr = 0.001)	16	0.361	0.865	horiflip(p=0.3), + fc: 3 layers
	ReduceLR (lr = 0.001)	16	0.363	0.864	vertiflip, horiflip, rotate, + fc: 3 layers
	ReduceLR (lr = 0.0001)	16	0.371	0.860	vertiflip, horiflip, rotate, + fc: 3 layers
	ReduceLR (lr = 0.001)	32	0.361	0.845	horiflip(p=0.3)
	ReduceLR (lr = 0.001)	32	0.361	0.871	vertiflip, horiflip, rotate

5. 결과 : ResNet 실행 표(하이스코어)

ResNet - high score(0.38 down / 0.86 up)

모델	스케줄러	배치사이즈	Test Loss	Test Accuracy	전처리기
ResNet50	ReduceLR (lr = 0.001)	32	0.380	0.858	
ResNet101	ReduceLR (lr = 0.001)	16	0.377	0.864	horiflip(p=0.3)
	ReduceLR (lr = 0.01)	16	0.384	0.860	horiflip(p=0.3)
	Cosine(10) (Lr = 0.0001)	16	0.373	0.857	Resize, RandomAutocontrast(p =0.4), ToTensor
	Cosine(10) (Lr = 0.0001)	32	0.383	0.863	***Best*** Resize, RandomAutocontrast(p=0.4), ToTensor
ResNet152	ReduceLR (lr = 0.001)	16	0.375	0.865	horiflip(p=0.3)
	ReduceLR (lr = 0.001)	16	0.361	0.865	horiflip(p=0.3), + fc: 3 layers
	ReduceLR (lr = 0.001)	16	0.363	0.864	vertiflip, horiflip, rotate, + fc: 3 layers
	ReduceLR (lr = 0.0001)	16	0.371	0.860	vertiflip, horiflip, rotate, + fc: 3 layers
	ReduceLR (lr = 0.001)	32	0.361	0.845	horiflip(p=0.3)
	ReduceLR (lr = 0.001)	32	0.361	0.871	vertiflip, horiflip, rotate

5. 결과 : ResNet 실행 표(하이스코어)

ResNet - best score

모델	스케줄러	배치사이즈	Test Loss	Test Accuracy	전처리기
ResNet101	ReduceLR (lr = 0.001)	16	0.377	0.864	horiflip(p=0.3)
	Cosine(10) (Lr = 0.0001)	16	0.373	0.857	Resize, RandomAutocontrast(p =0.4), ToTensor
ResNet152	ReduceLR (lr = 0.001)	32	0.361	0.871	vertiflip, horiflip, rotate