

---

# 쓰레기 분류 모델 보고서

빅데이터 7기

분리수거 잘해 조

김재경, 안필주, 이민아, 이주복

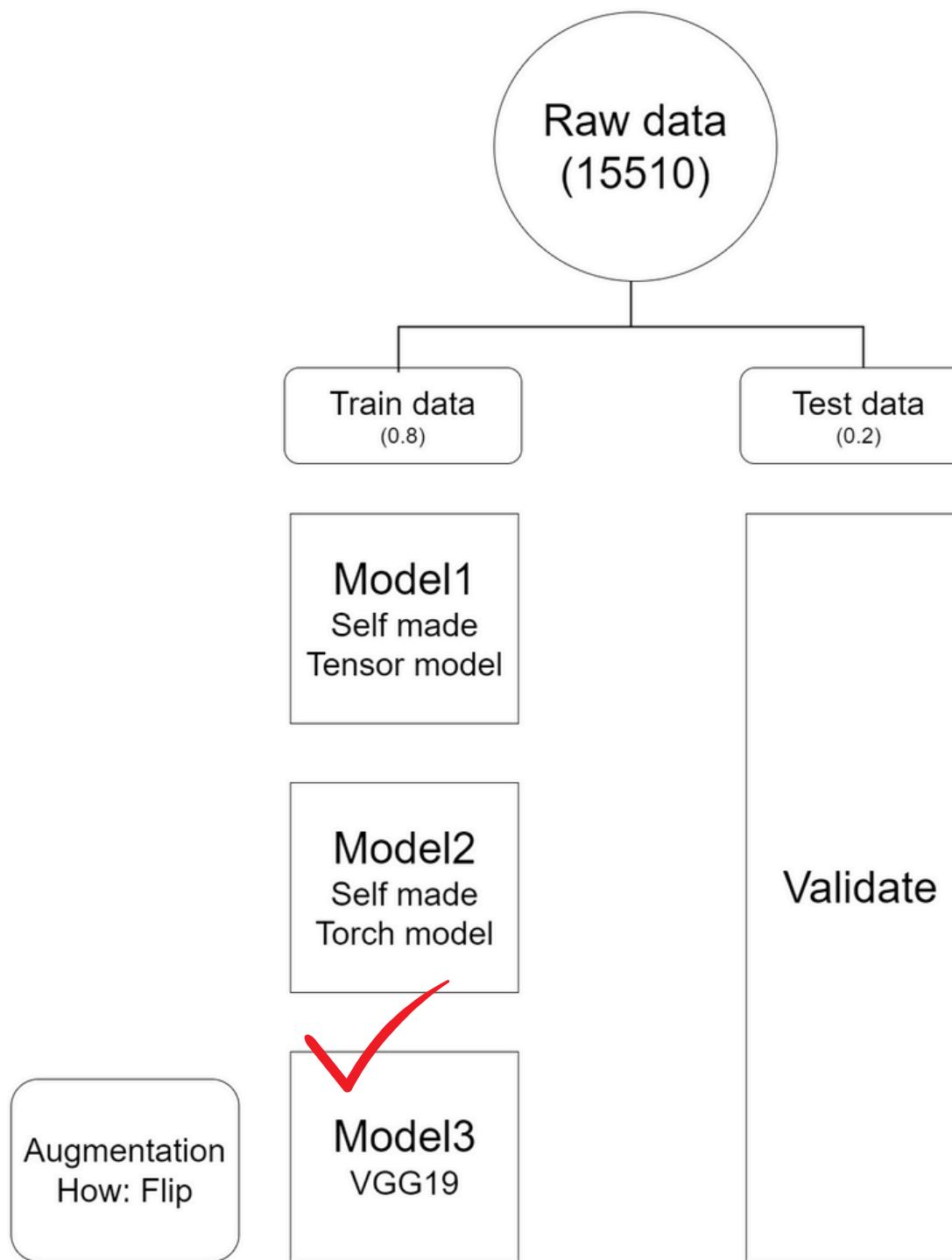
# | 개요

재활용 선별장에서 재활용 쓰레기를 분류할 때, 중량을 이용하여 분류하는 현재 상황에 재활용 분류 모델을 사용하여 분류의 정확성을 높이고자 한다.

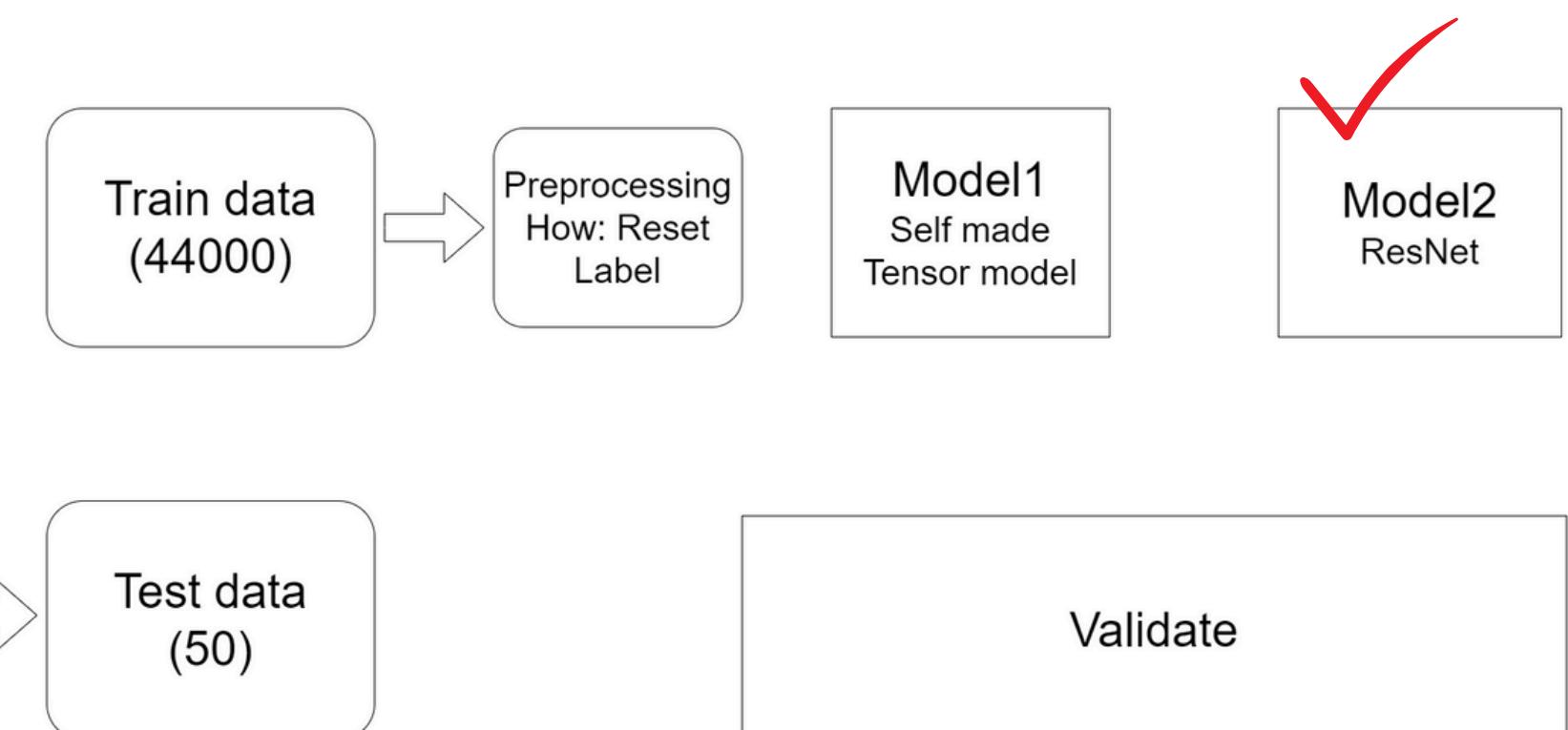


# Process

## Garbage classification



## Clothes classification



# Garbage Classification

- Data

데이터 수: 15510개

Label: 12개

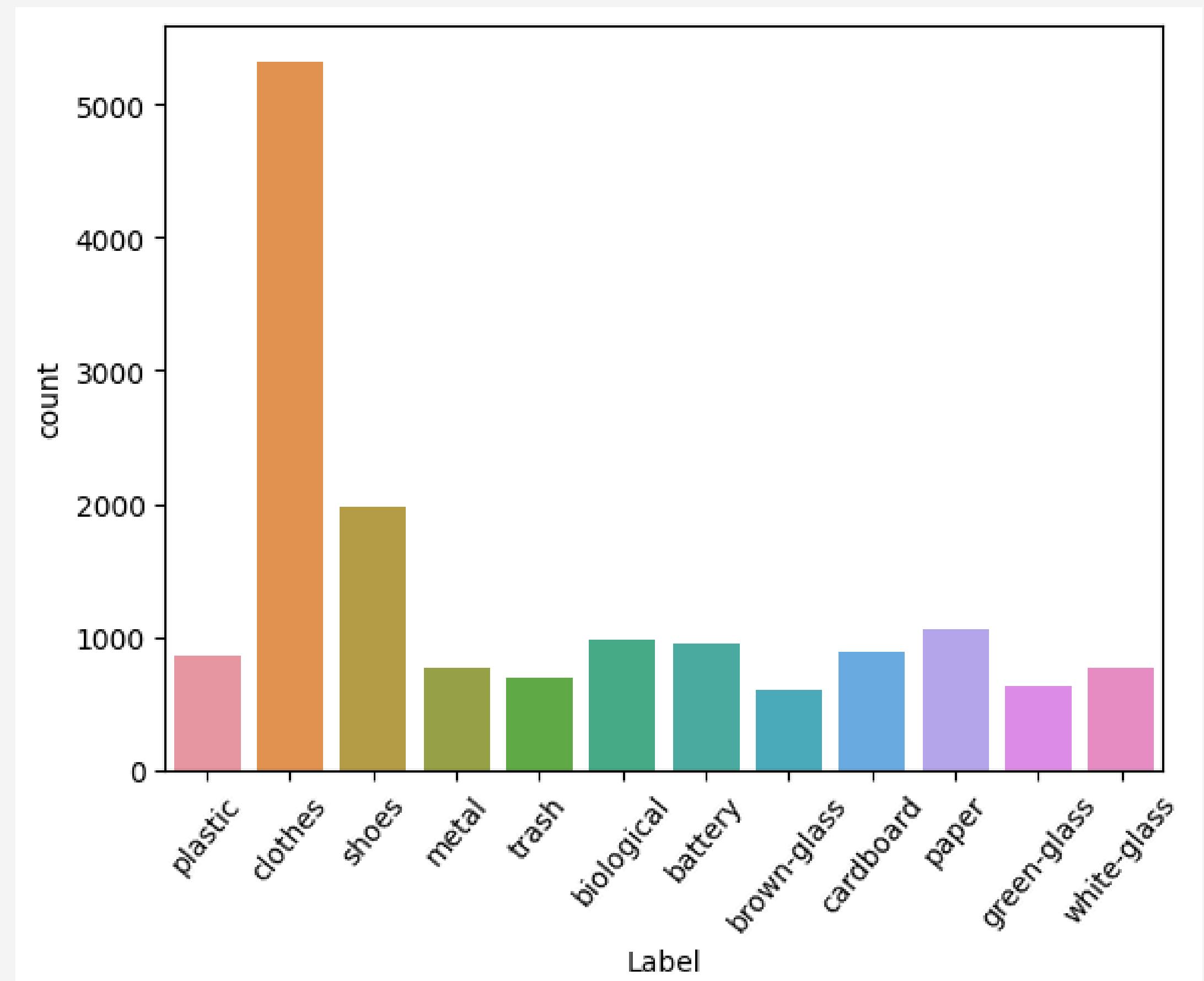
배터리, 식품, 갈색 병, 판지, 의류, 녹색 병, 금속, 종이, 플라스틱, 생필품, 흰색 병



# Garbage Classification

• Label

Label	34.33%
clothes	34.33%
shoes	12.75%
paper	6.77%
biological	6.35%
battery	6.06%
cardboard	5.74%
plastic	5.58%
white-glass	5.00%
metal	4.96%
trash	4.49%
green-glass	4.06%
brown-glass	3.91%



# Garbage Classification

- Model1: Self made Tensor model

## Convolution layer

- Layer: 6
- Activation: Relu
- Padding: Same
- L2 regularizer
- Dropout: 0.2

## Pooling layer

- Maxpooling

## Optimizer

- Adam
- Learning rate: 0.0002

Batch size: 128

Epoch: 100

Early Stopping



Epoch = 34	Loss	Accuracy
Train set	1.40	0.94
Valid set	2.31	0.72

# Garbage Classification

- Model1: Self made Tensor model

## Parameter Tuning

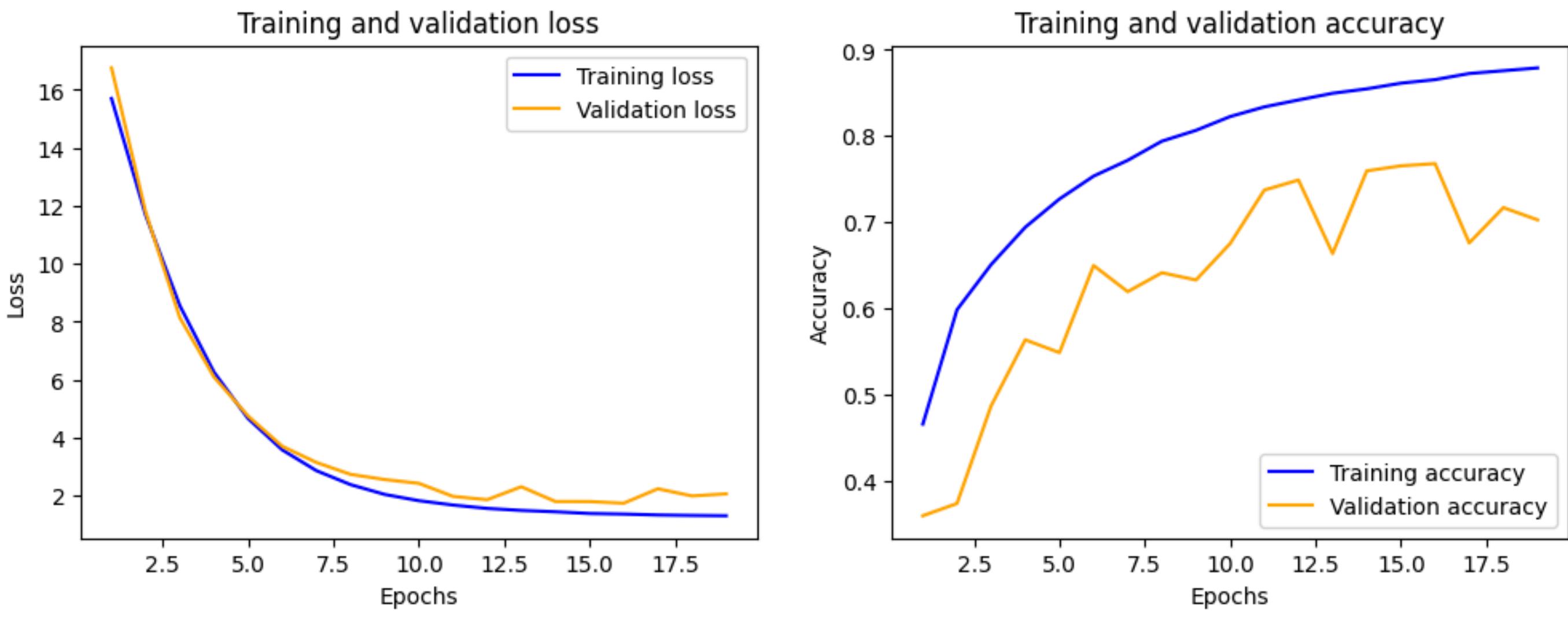
### Dropout 변경

- Layer마다 다르게 적용

- layer1,2: 0.2
- layer3~6: 0.4

### 보완할 내용

- Learning rate 조절
- L2 정규화 수치 변경



Epoch = 18	Loss	Accuracy
Train set	1.32	0.87
Valid set	1.99	0.70

# Garbage Classification

- Model2: Self made Torch model

## Convolution layer

- Layer: 3
- Activation: Relu
- Padding: 2

## Pooling layer

- Maxpooling

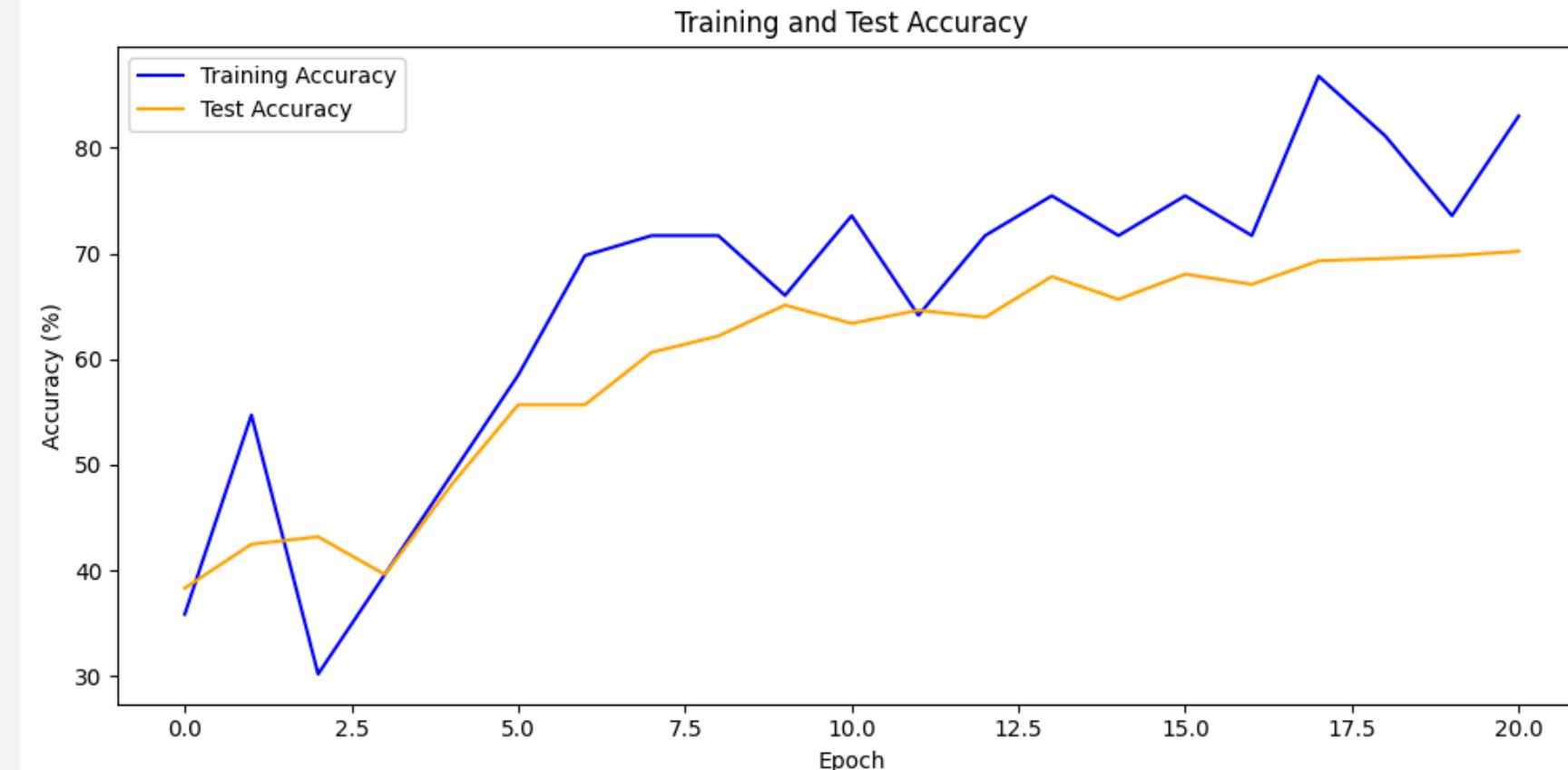
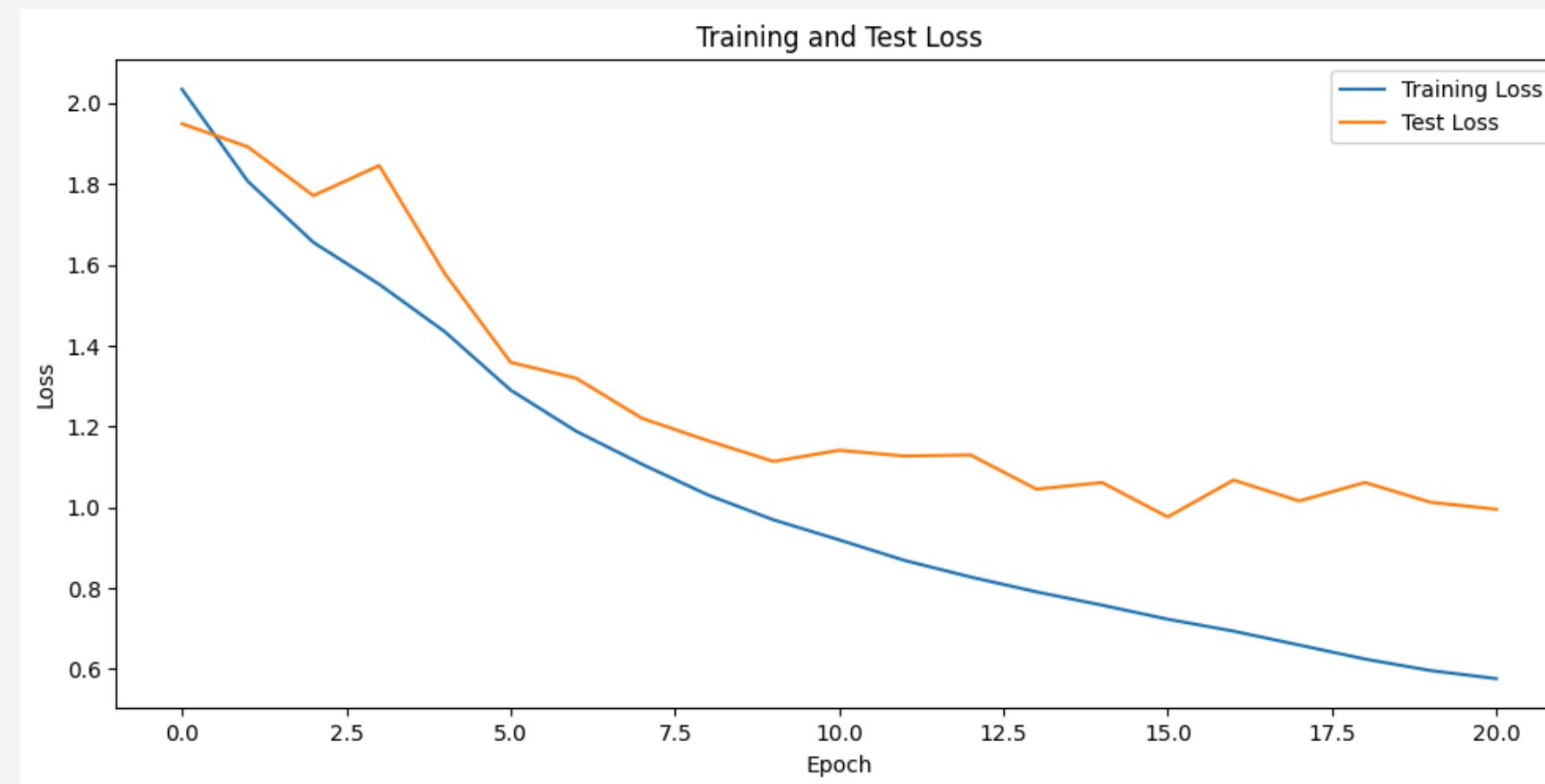
## Optimizer

- SGD
- Learning rate: 0.01
- Momentum: 0.5

Batch size: 64

Epoch: 50

Early Stopping



Epoch = 20	Loss	Accuracy
Train set	0.57	0.81
Valid set	1.03	0.70

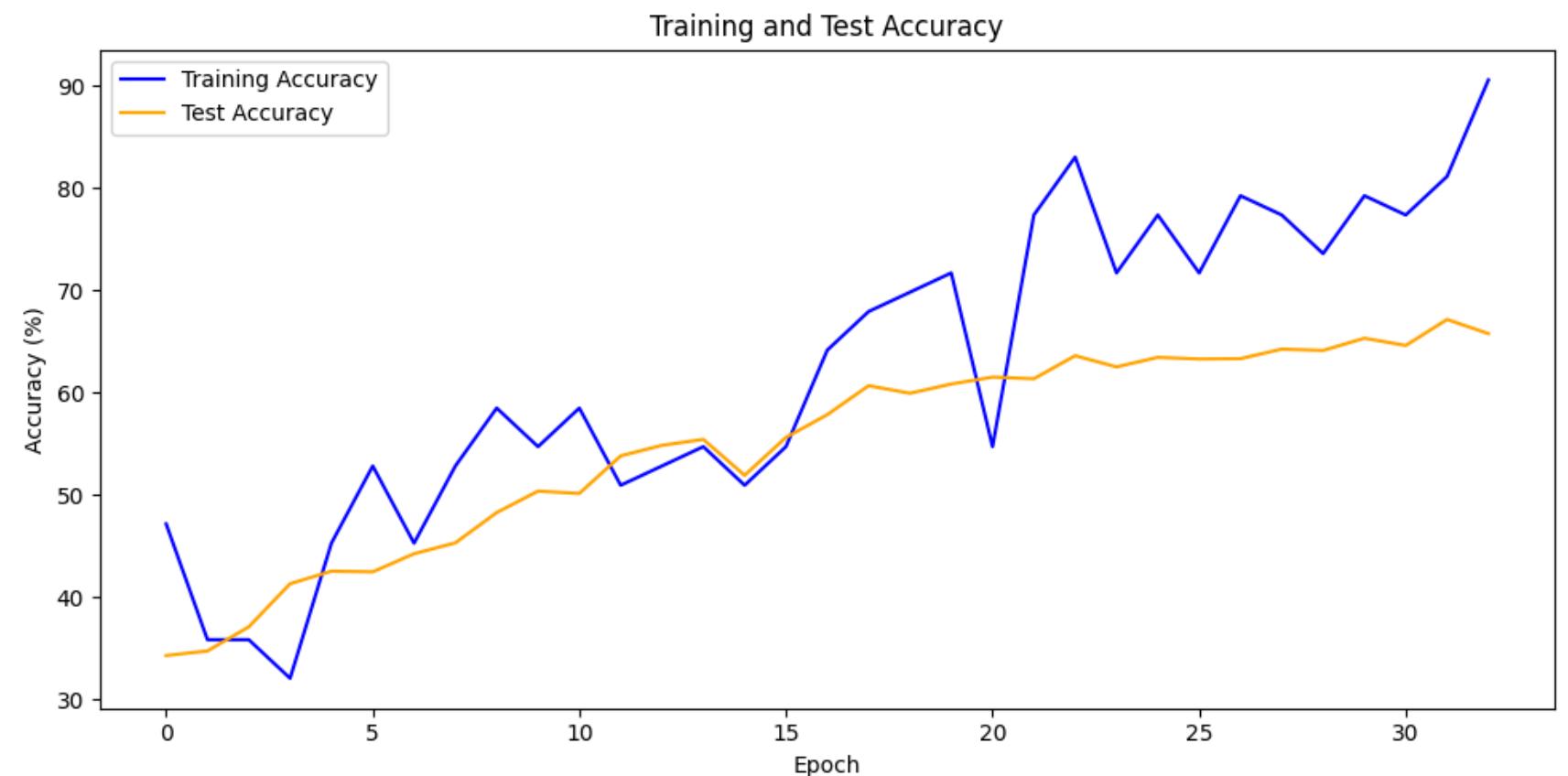
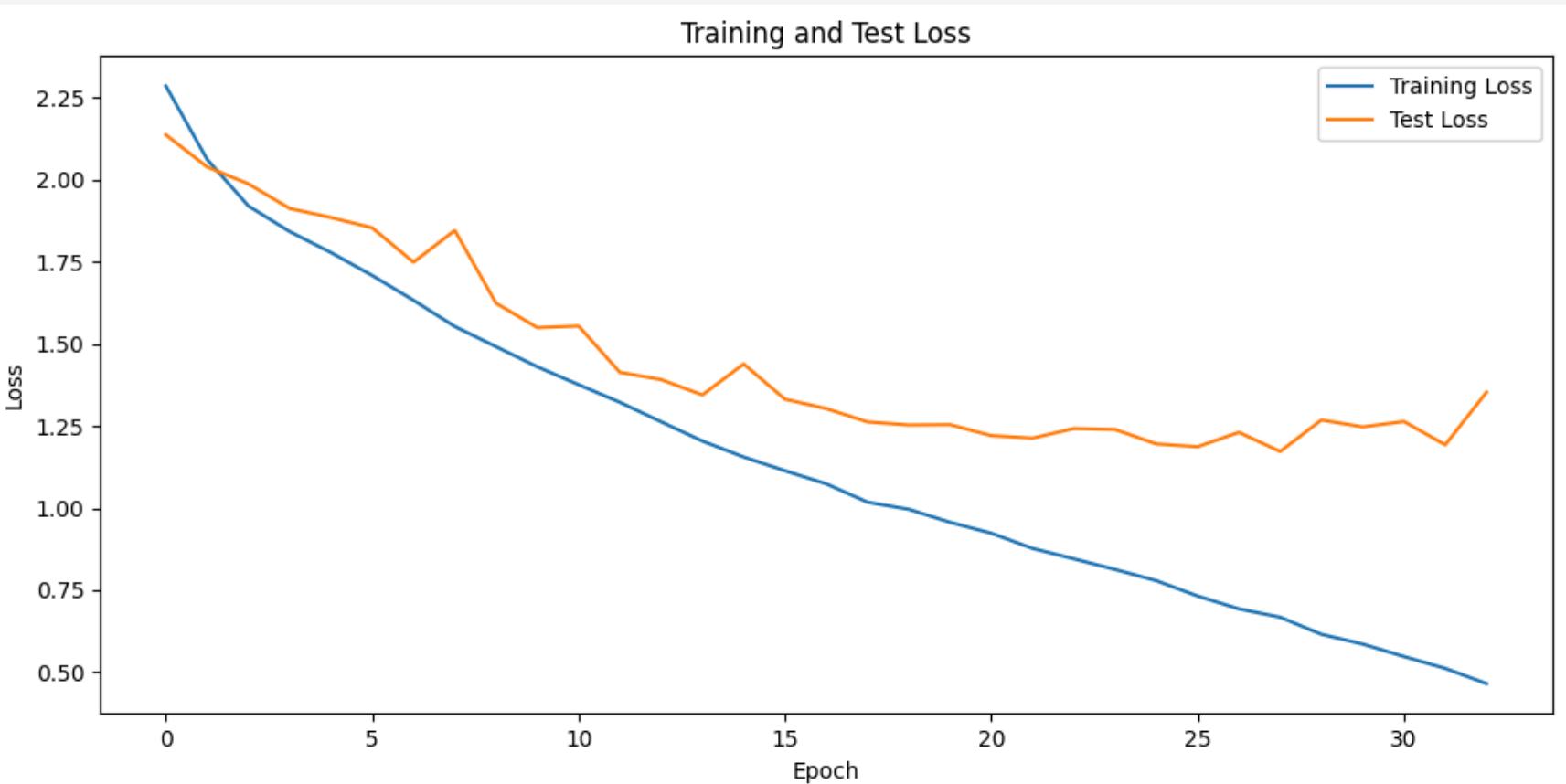
# Garbage Classification

- Model2: Self made Torch model

## Parameter Tuning

## Layer 추가

- layer: 3 -> 5



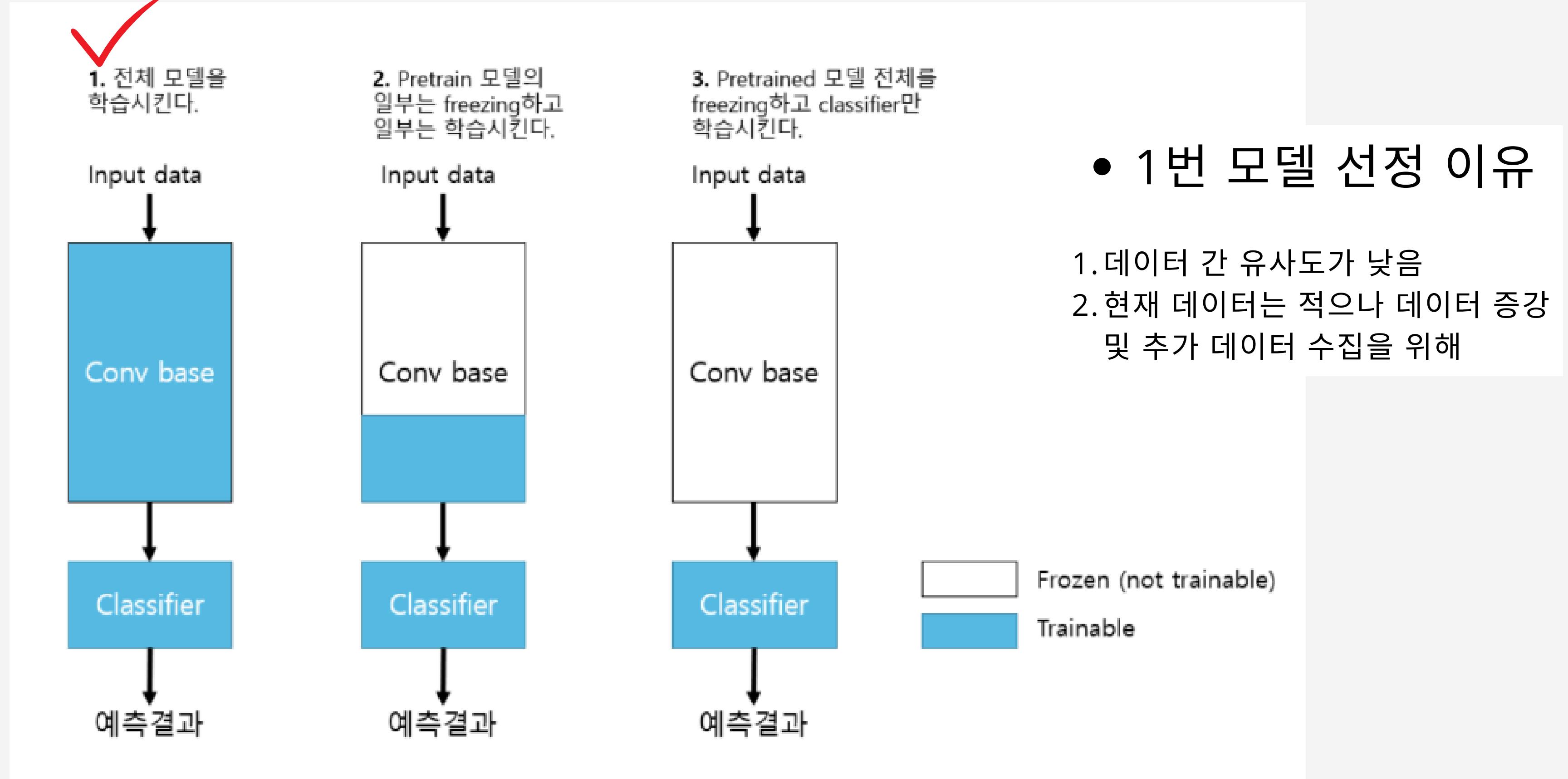
Epoch = 32	Loss	Accuracy
Train set	0.51	0.89
Valid set	1.19	0.67

## 보완할 내용

- Dropout 추가
- L1 or L2 정규화 적용
- Learning rate 조절
- Batch size 조절

# Garbage Classification

- Model3: VGGNet



# Garbage Classification

- Model3: VGGNet

```
=====
Layer (type:depth-idx)      Output Shape     Param #
=====
myVggNet
├─VGG: 1-1      [1, 12]      --
└─Sequential: 2-1      [1, 12]      --
  ├─Conv2d: 3-1      [1, 512, 7, 7]    --
  |  └─ReLU: 3-2      [1, 64, 224, 224]  1,792
  |  └─Conv2d: 3-3      [1, 64, 224, 224]  36,928
  |  └─ReLU: 3-4      [1, 64, 224, 224]  --
  |  └─MaxPool2d: 3-5      [1, 64, 112, 112]  --
  |  └─Conv2d: 3-6      [1, 128, 112, 112]  73,856
  |  └─ReLU: 3-7      [1, 128, 112, 112]  --
  |  └─Conv2d: 3-8      [1, 128, 112, 112]  147,584
  |  └─ReLU: 3-9      [1, 128, 112, 112]  --
  |  └─MaxPool2d: 3-10      [1, 128, 56, 56]  --
  |  └─Conv2d: 3-11      [1, 256, 56, 56]  295,168
  |  └─ReLU: 3-12      [1, 256, 56, 56]  --
  |  └─Conv2d: 3-13      [1, 256, 56, 56]  590,080
  |  └─ReLU: 3-14      [1, 256, 56, 56]  --
  |  └─Conv2d: 3-15      [1, 256, 56, 56]  590,080
  |  └─ReLU: 3-16      [1, 256, 56, 56]  --
  |  └─Conv2d: 3-17      [1, 256, 56, 56]  590,080
  |  └─ReLU: 3-18      [1, 256, 56, 56]  --
  |  └─MaxPool2d: 3-19      [1, 256, 28, 28]  --
  |  └─Conv2d: 3-20      [1, 512, 28, 28]  1,180,160
  |  └─ReLU: 3-21      [1, 512, 28, 28]  --
  |  └─Conv2d: 3-22      [1, 512, 28, 28]  2,359,808
  |  └─ReLU: 3-23      [1, 512, 28, 28]  --
  |  └─Conv2d: 3-24      [1, 512, 28, 28]  2,359,808
  |  └─ReLU: 3-25      [1, 512, 28, 28]  --
  |  └─Conv2d: 3-26      [1, 512, 28, 28]  2,359,808
  |  └─ReLU: 3-27      [1, 512, 28, 28]  --
  |  └─MaxPool2d: 3-28      [1, 512, 14, 14]  --
  |  └─Conv2d: 3-29      [1, 512, 14, 14]  2,359,808
  |  └─ReLU: 3-30      [1, 512, 14, 14]  --
  |  └─Conv2d: 3-31      [1, 512, 14, 14]  2,359,808
  |  └─ReLU: 3-32      [1, 512, 14, 14]  --
  |  └─Conv2d: 3-33      [1, 512, 14, 14]  2,359,808
  |  └─ReLU: 3-34      [1, 512, 14, 14]  --
  |  └─Conv2d: 3-35      [1, 512, 14, 14]  2,359,808
  |  └─ReLU: 3-36      [1, 512, 14, 14]  --
  |  └─MaxPool2d: 3-37      [1, 512, 7, 7]  --
  └─AdaptiveAvgPool2d: 2-2      [1, 512, 7, 7]  --
  └─Sequential: 2-3      [1, 12]      --
    └─Linear: 3-38      [1, 4096]  102,764,544
    └─ReLU: 3-39      [1, 4096]  --
    └─Dropout: 3-40      [1, 4096]  --
    └─Linear: 3-41      [1, 4096]  16,781,312
    └─ReLU: 3-42      [1, 4096]  --
    └─Dropout: 3-43      [1, 4096]  --
    └─Linear: 3-44      [1, 12]  49,164
=====

Total params: 139,619,404
Trainable params: 139,619,404
Non-trainable params: 0
Total mult-adds (Units.GIGABYTES): 19.64
=====

Input size (MB): 0.60
Forward/backward pass size (MB): 118.88
Params size (MB): 558.48
Estimated Total Size (MB): 677.96
=====
```

	└─Sequential: 2-3	[1, 12]	--
	└─Linear: 3-38	[1, 4096]	102,764,544
	└─ReLU: 3-39	[1, 4096]	--
	└─Dropout: 3-40	[1, 4096]	--
	└─Linear: 3-41	[1, 4096]	16,781,312
	└─ReLU: 3-42	[1, 4096]	--
	└─Dropout: 3-43	[1, 4096]	--
	└─Linear: 3-44	[1, 12]	49,164
	Total params:	139,619,404	
	Trainable params:	139,619,404	
	Non-trainable params:	0	
	Total mult-adds (Units.GIGABYTES):	19.64	
	Input size (MB):	0.60	
	Forward/backward pass size (MB):	118.88	
	Params size (MB):	558.48	
	Estimated Total Size (MB):	677.96	

# Garbage Classification

- Model3: VGGNet

## Optimizer

- SGD
- Learning rate: 0.001
- Momentum: 0.5

Batch size: 256

Epoch: 20

Epoch = 5	Loss	Accuracy
Train set	2.13	-
Valid set	1.97	0.40

## Optimizer

- Adam
- Learning rate: 0.0001

Batch size: 256

Epoch: 20

Epoch = 20	Loss	Accuracy
Train set	0.98	-
Valid set	0.90	0.78

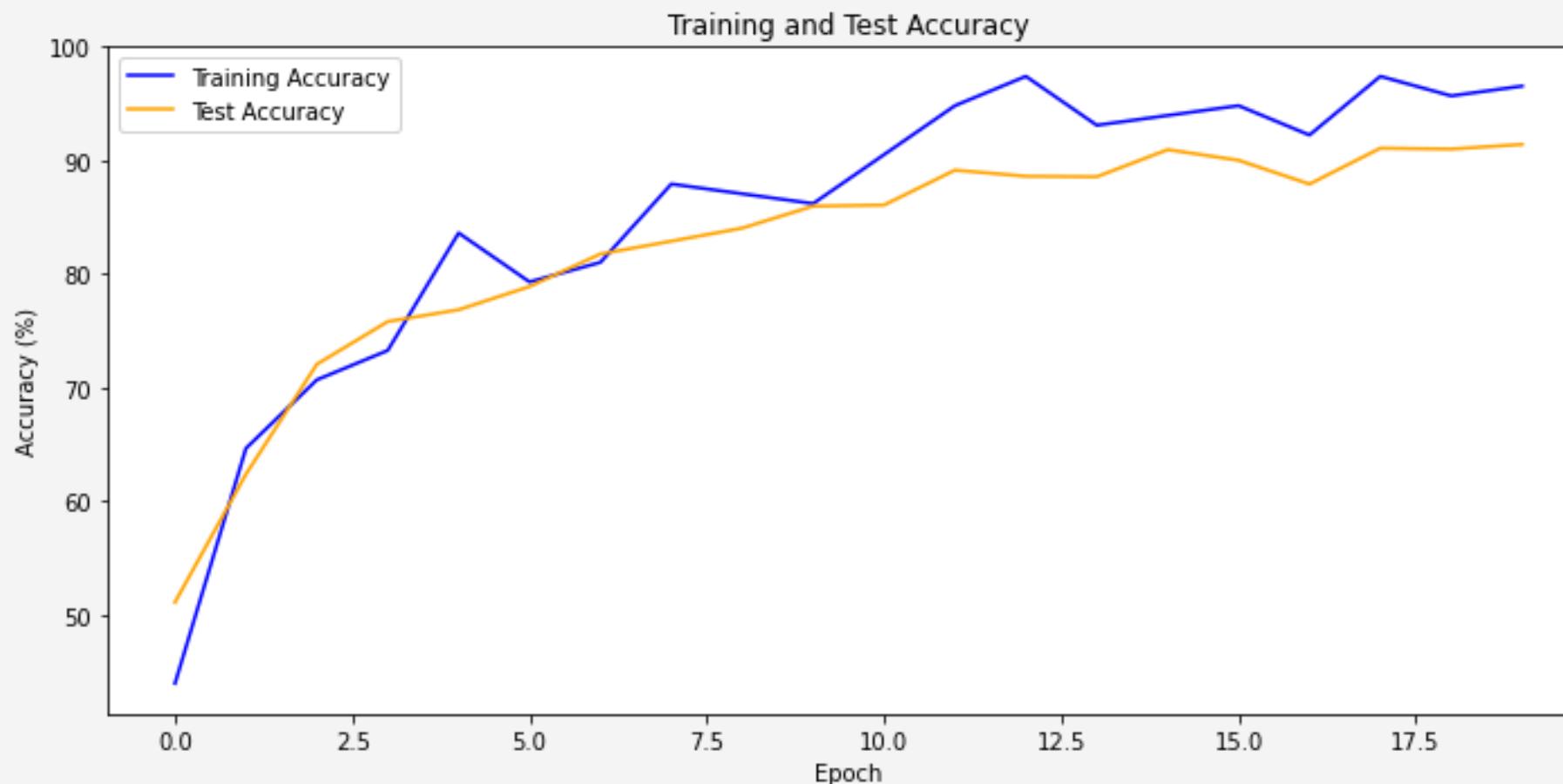
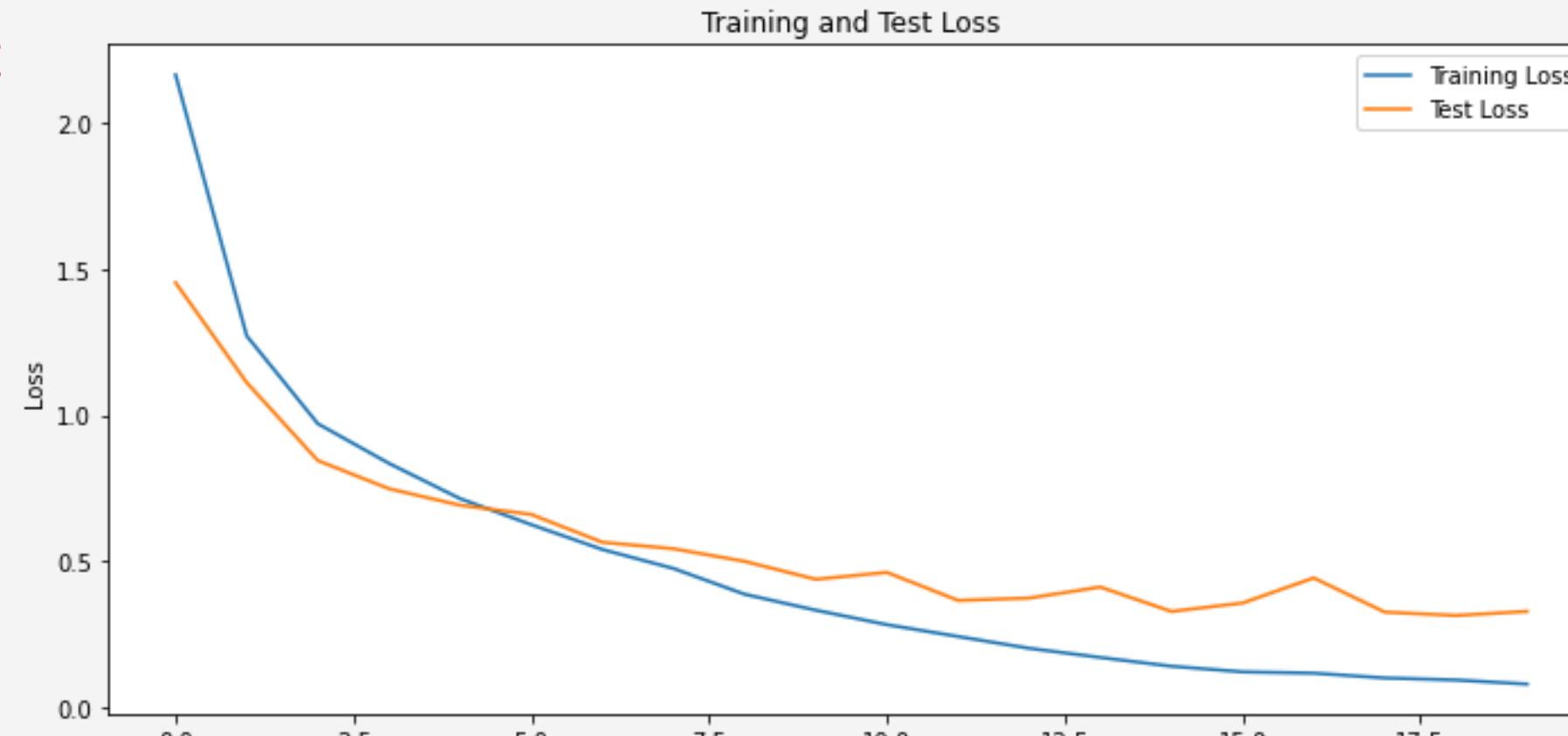
# Garbage Classification

- Model3: VGGNet

## Parameter Tuning

Dropout 추가

- Dropout: 0.5



Epoch = 20	Loss	Accuracy
Train set	0.07	0.95
Valid set	0.32	0.91

# Garbage Classification

- Model3: VGGNet

## Data Augmentation

### Convolution layer

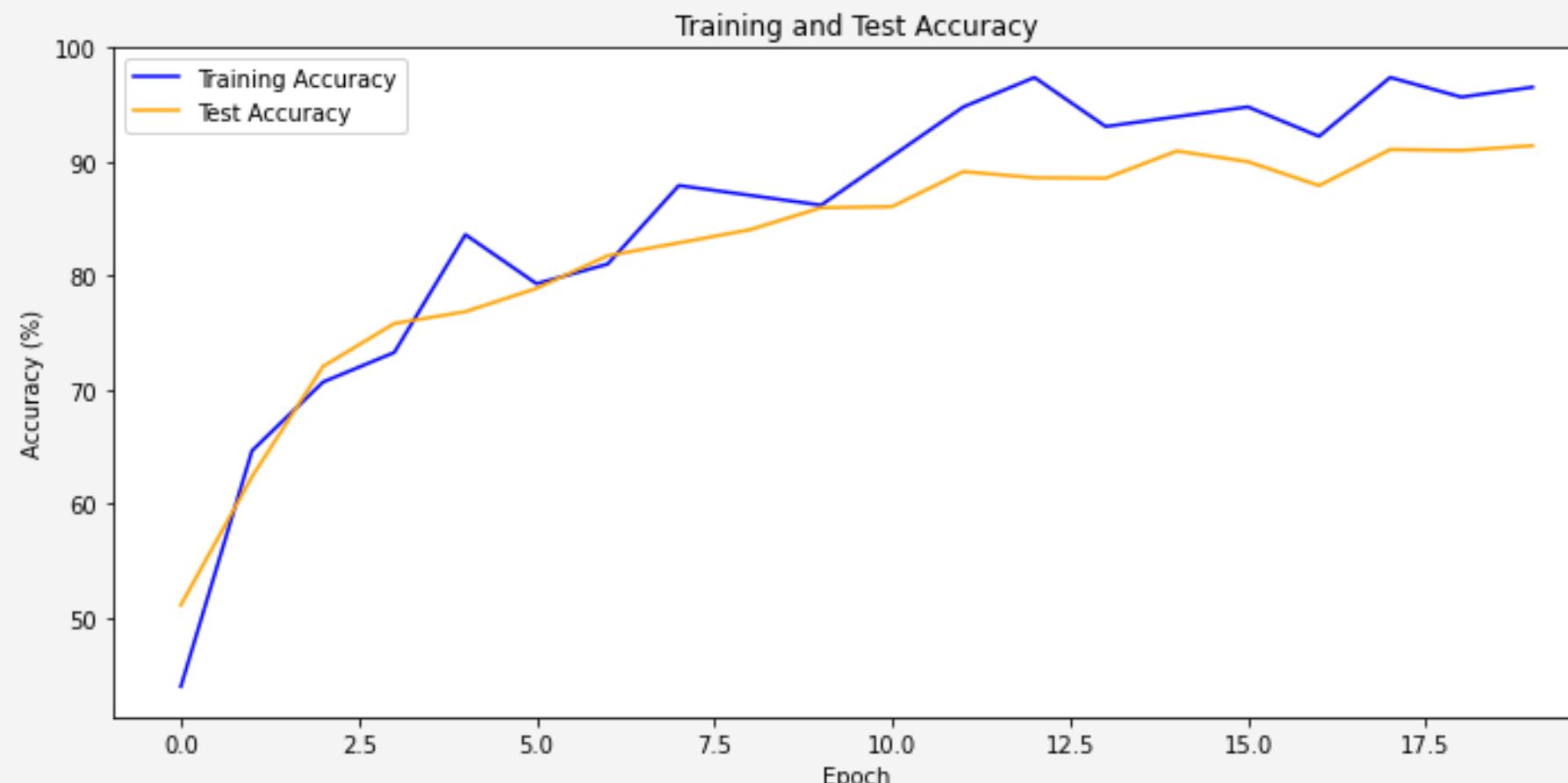
- Dropout: 0.5

### Optimizer

- Adam
- Learning rate: 0.0001

Batch size: 256

Epoch: 20



Epoch = 20	Loss	Accuracy
Train set	0.07	0.96
Valid set	0.32	0.91

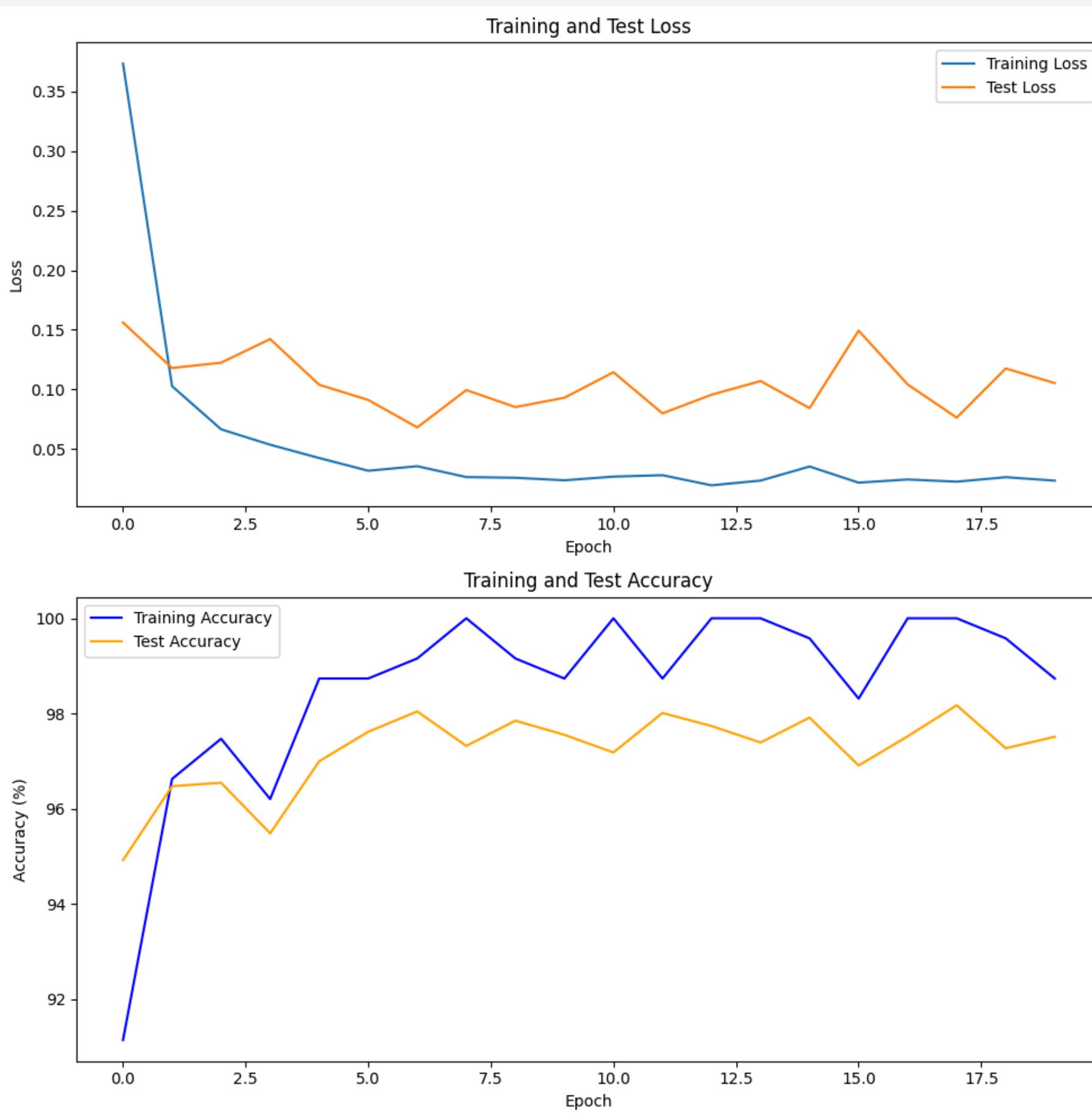
# Garbage Classification

- Model3: VGGNet

## Sequential Tuning

## Sequential Dropout 변경

- Dropout: 0.5 -> 0.3



Epoch = 20	Loss	Accuracy
Train set	0.02	0.98
Valid set	0.12	0.97

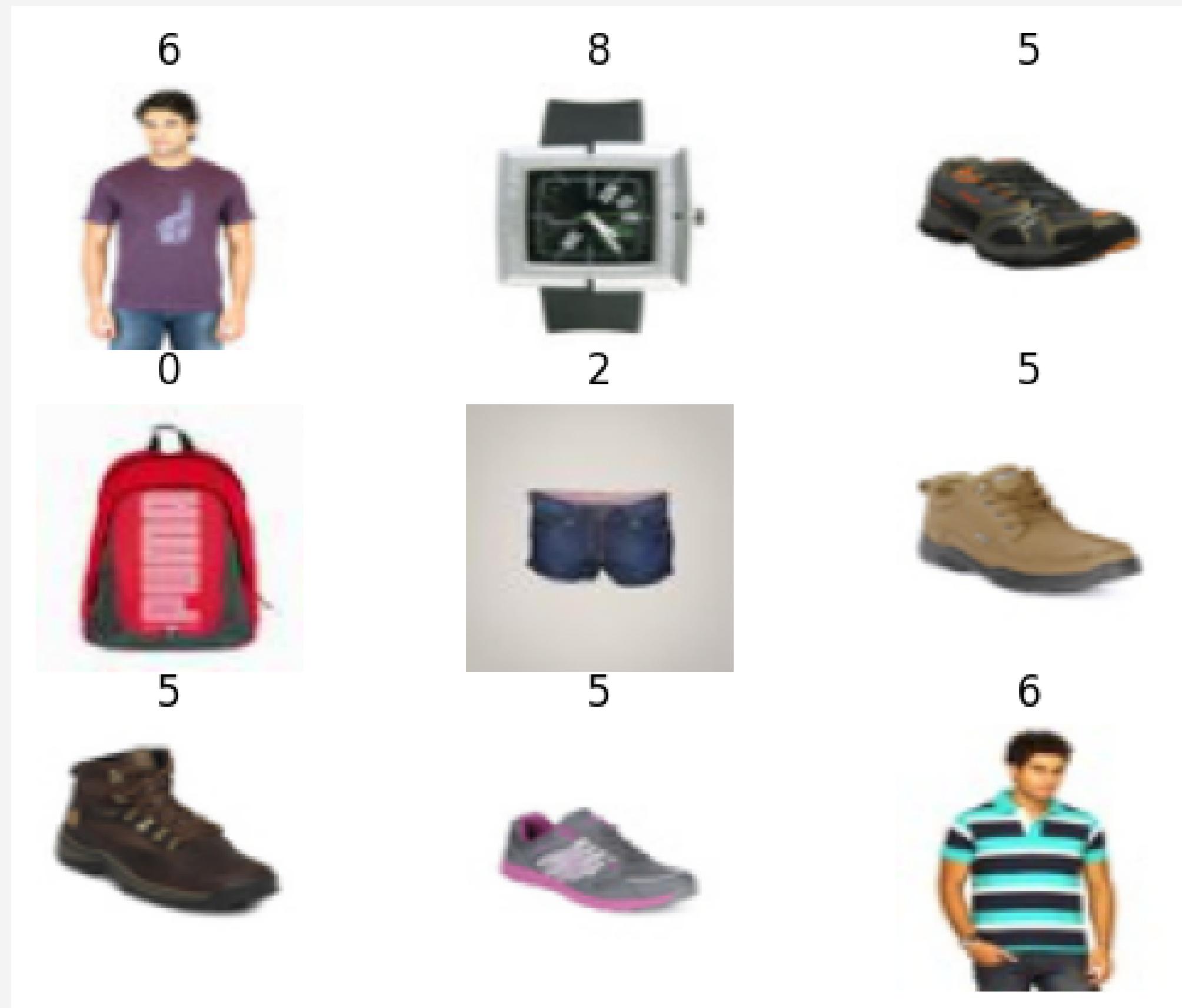
# Clothes Classification

- Data

데이터 수: 44000개

Label: 9개

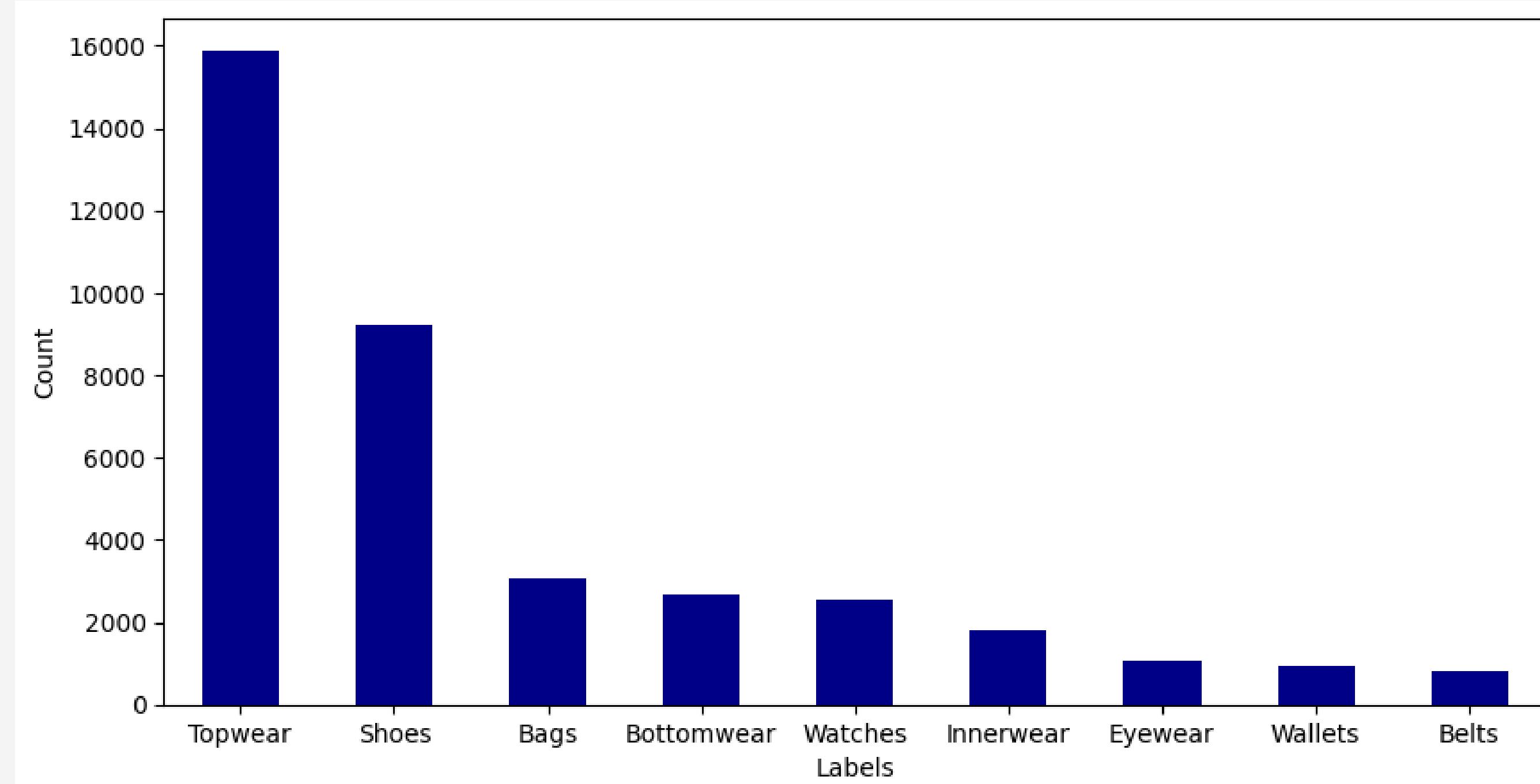
- 0: 가방
- 1: 밸트
- 2: 하의
- 3: 안경
- 4: 속옷
- 5: 신발
- 6: 상의
- 7: 지갑
- 8: 시계



# Garbage Classification

- Label

label	
Topwear	41.77%
Shoes	24.26%
Bags	8.04%
Bottomwear	7.09%
Watches	6.69%
Innerwear	4.76%
Eyewear	2.82%
Wallets	2.45%
Belts	2.13%

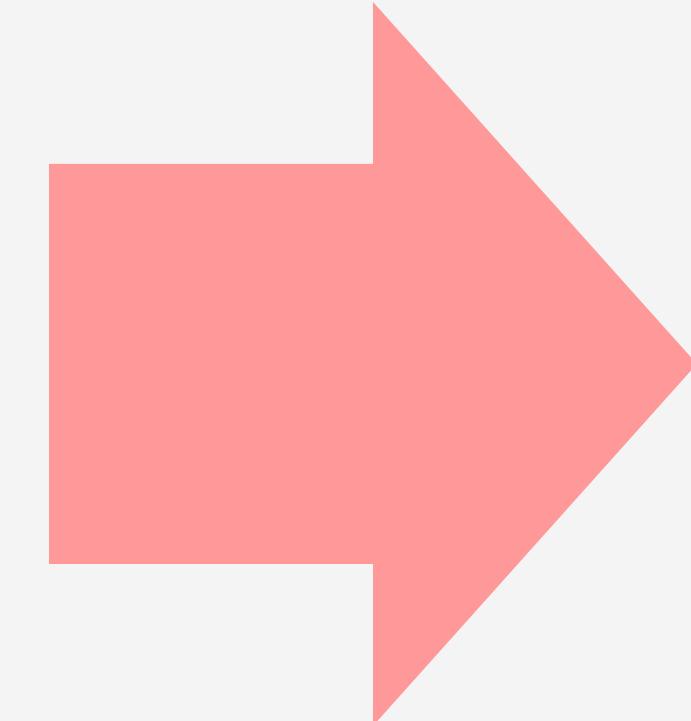


# Clothes Classification

- Preprocessing

## Label(45개)

'Topwear', 'Bottomwear', 'Watches', 'Socks',  
'Shoes', 'Belts', 'Flip Flops', 'Bags', 'Innerwear',  
'Sandal', 'Shoe Accessories', 'Fragrance',  
'Jewellery', 'Lips', 'Saree', 'Eyewear', 'Nails',  
'Scarves', 'Dress', 'Loungewear and Nightwear',  
'Wallets', 'Apparel Set', 'Headwear', 'Mufflers',  
'Skin Care', 'Makeup', 'Free Gifts', 'Ties',  
'Accessories', 'Skin', 'Beauty Accessories',  
'Water Bottle', 'Eyes', 'Bath and Body', 'Gloves',  
'Sports Accessories', 'Cufflinks', 'Sports  
Equipment', 'Stoles', 'Hair', 'Perfumes', 'Home  
Furnishing', 'Umbrellas', 'Wristbands',  
'Vouchers'



## Reseted Label(9개)

'Topwear', 'Bottomwear'  
'Watches', 'Shoes', 'Belts'  
'Bags', 'Innerwear', 'Eyewear',  
'Wallets'

의류를 제외한 품목 제거,  
의류 하위 품목 상위 품목에 추가

# Clothes Classification

- Model1: Self made Tensor model

## Convolution layer

- Layer: 6
- Activation: Relu
- Padding: Same
- L2 regularizer
- Dropout: 0.2

## Pooling layer

- Maxpooling

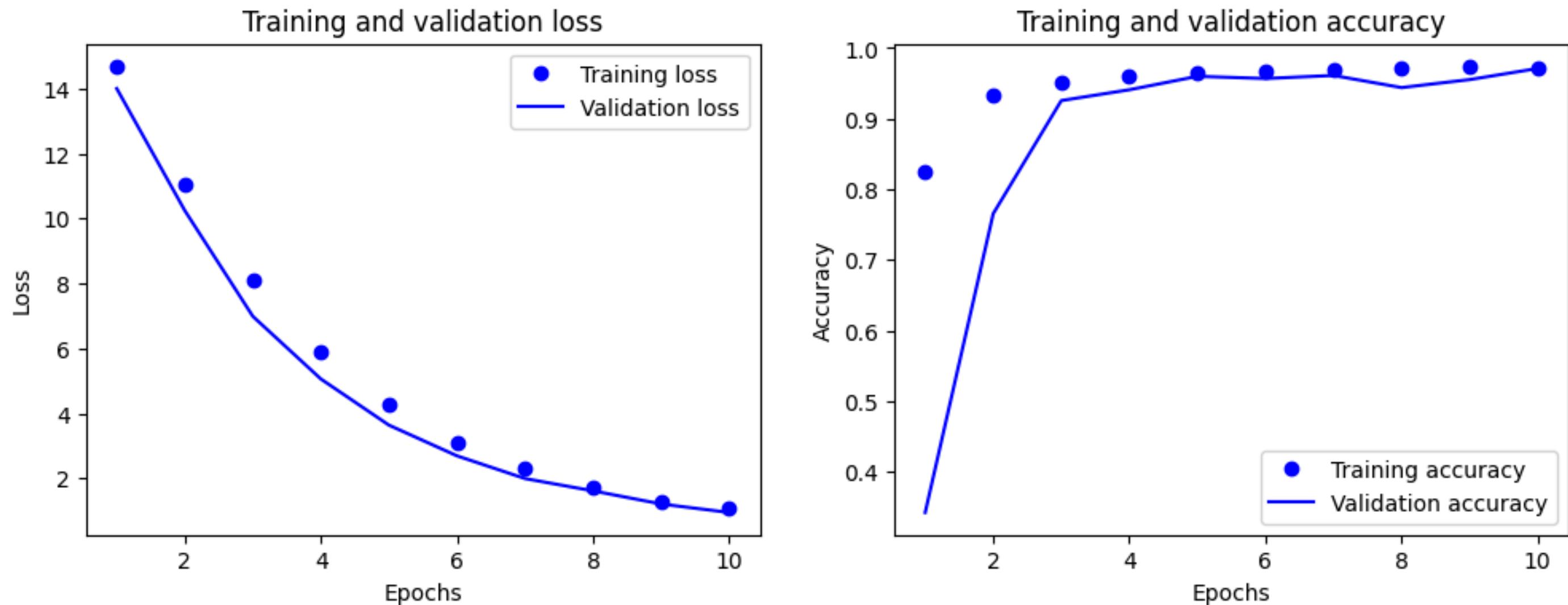
## Optimizer

- Adam
- Learning rate: 0.0002

Batch size: 128

Epoch: 10

Early Stopping



Epoch = 18	Loss	Accuracy
Train set	1.07	0.97
Valid set	0.95	0.97

# Clothes Classification

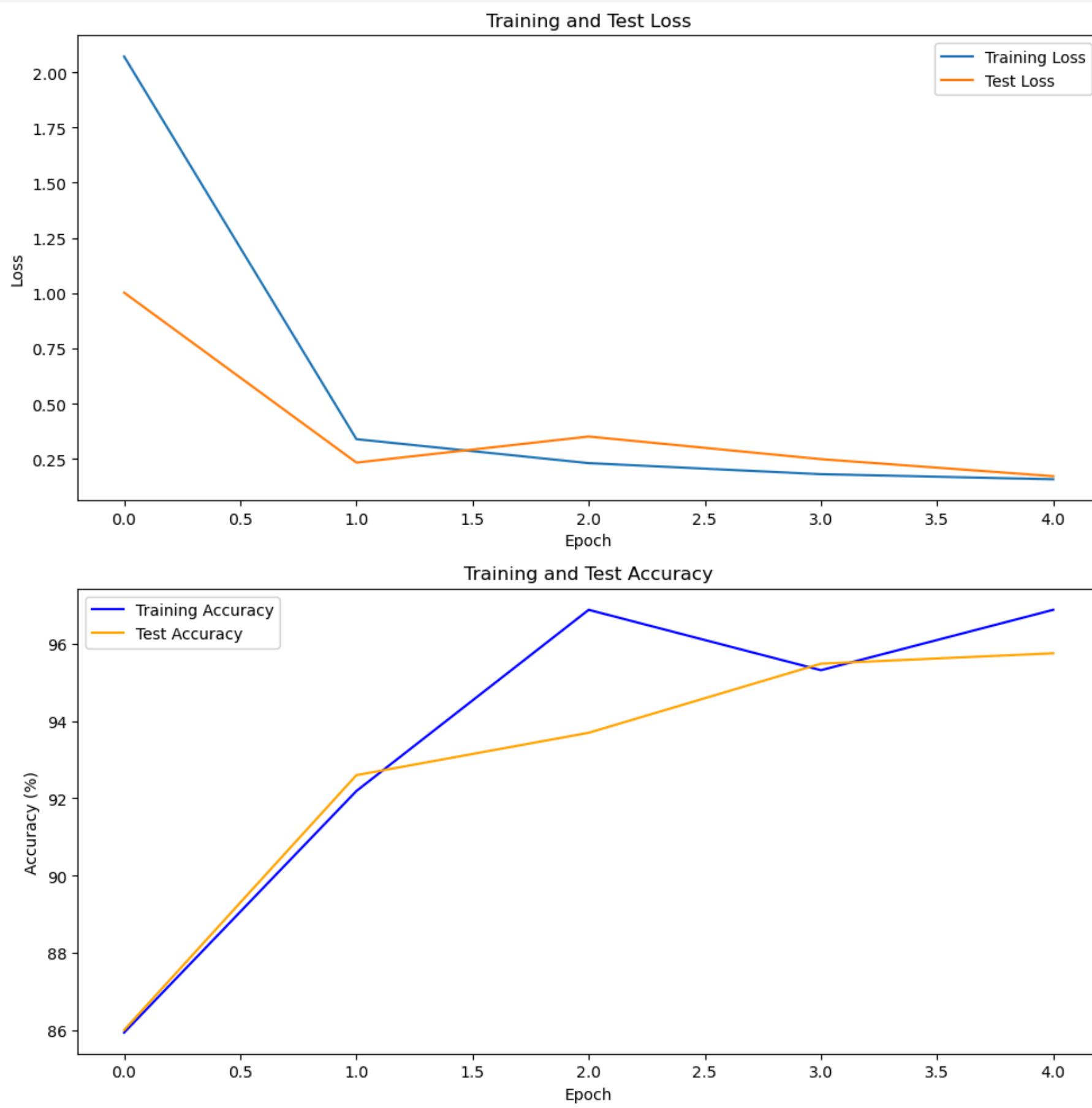
- Model2: ResNet

**Optimizer**

- Adam
- Learning rate: 0.03

**Batch size: 128**

**Epoch: 5**



Epoch = 5	Loss	Accuracy
Train set	0.15	0.97
Valid set	0.17	0.96

## 추가 내용

---

### 모델 보완할 사항

- openCV의 Rotation을 활용하여 데이터를 증강
- VGGNet 모델의 Convolution base frozen

### 추가 고려사항

- 의류 데이터에서 상의가 차지하는 비중이 크므로 데이터를 증강할 때, 상의 데이터를 제외한 나머지 데이터를 증강하여 데이터의 수를 비슷하게 맞추면 모델의 재현성이 좋아지지 않을까?
- 의류 분류 모델에 “의류가 아님”이라는 라벨을 추가하여 쓰레기 분류 모델에서 분류되지 않은 데이터를 의류 분류 모델에서 분류하는 것도 고려해보면 좋을 것 같다.