# Fatching

## : Fashion Recommendation Service

- Final Presentation -

# Objective and Motivation

**“To find the best outfit with the item I HAVE !!!”**

- Most fashion applications on the market do not have recommended services.
- Even if there is some recommendation service, it recommends items suitable for the updated items in the shopping mall.
- We need a service to search what clothes go well with the item we have.

# Progress

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Jaehyuk | | Developing Advanced Search Page | | | | | | | Developing Result Page | | | | | |
| Juwon | | Developing Main Page | | | | | | | Developing Result Page | | | | | |
| Hankyu | | DB/Server Setting | | | | | Connecting Server - App | | | | Connecting Server - Model | | | |
| Kyunghyun | | Developing Classification Model | | | | | | Developing Recommendation Model | | | | | | |
| Bomin | | | | Developing Recommendation Model | | | | | | | | | Preparing Final Report | |

# Final Design (1)

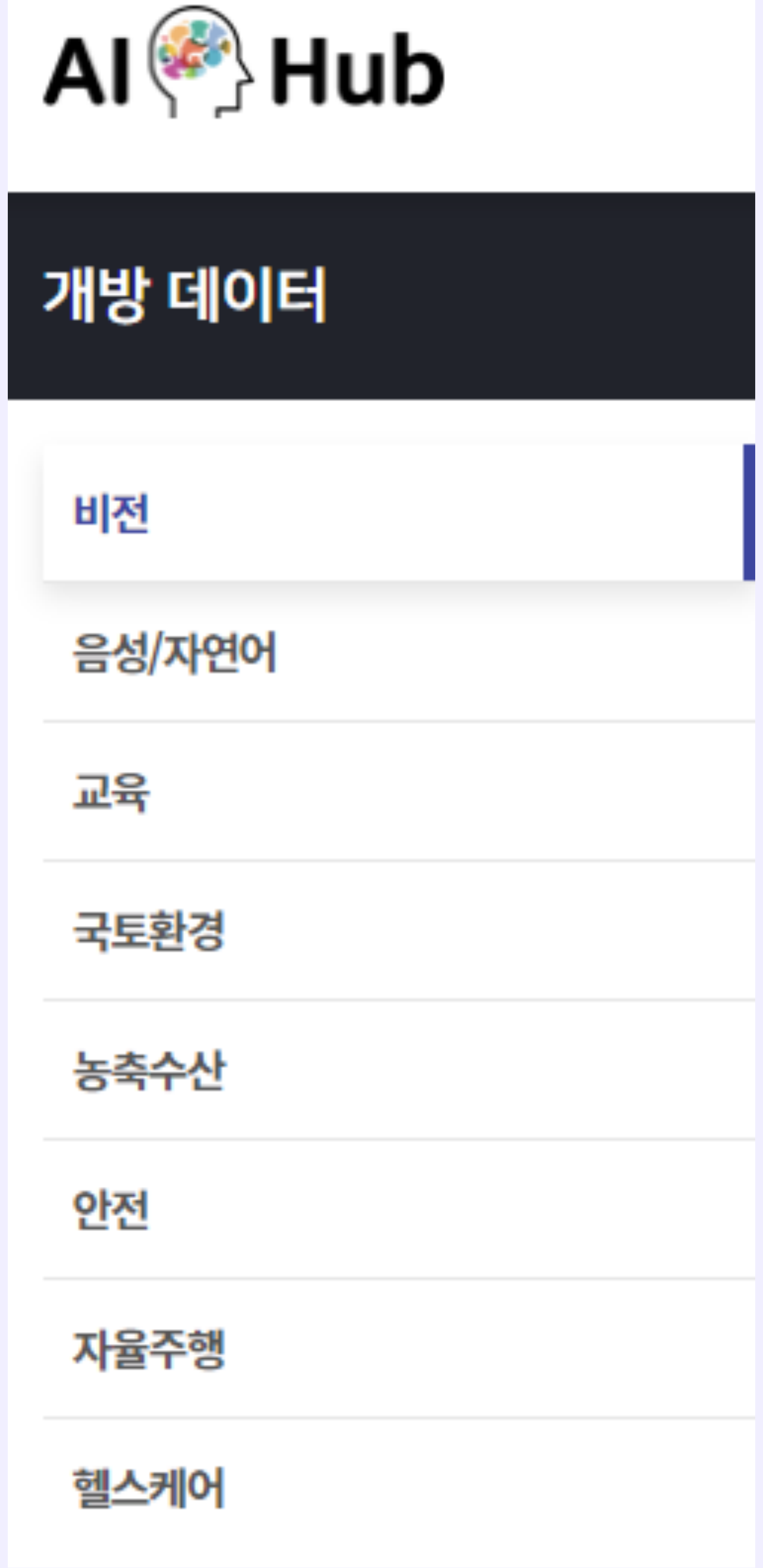
## : Back-End

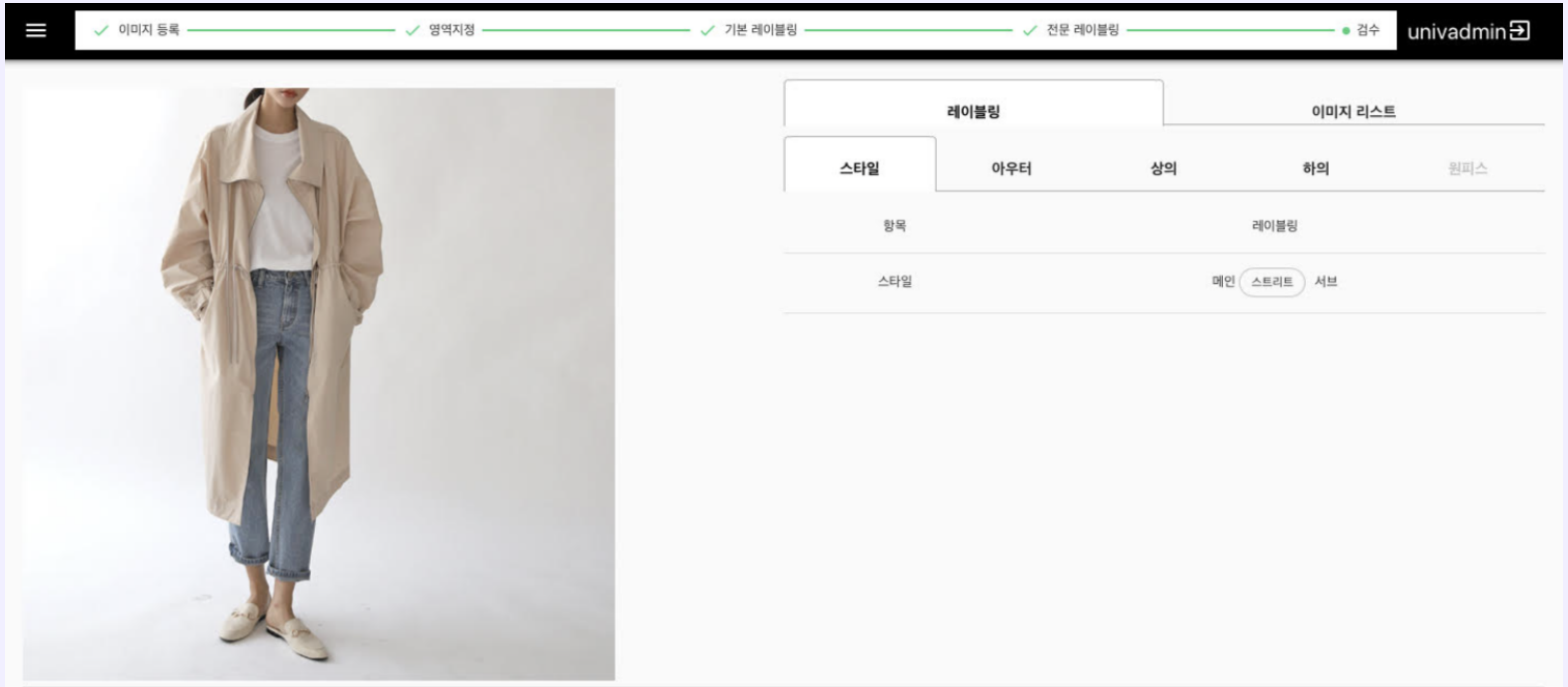
1. Fashion Image Classification Model

2. Fashion Recommendation Model

1. Fashion image classification model

# Dataset

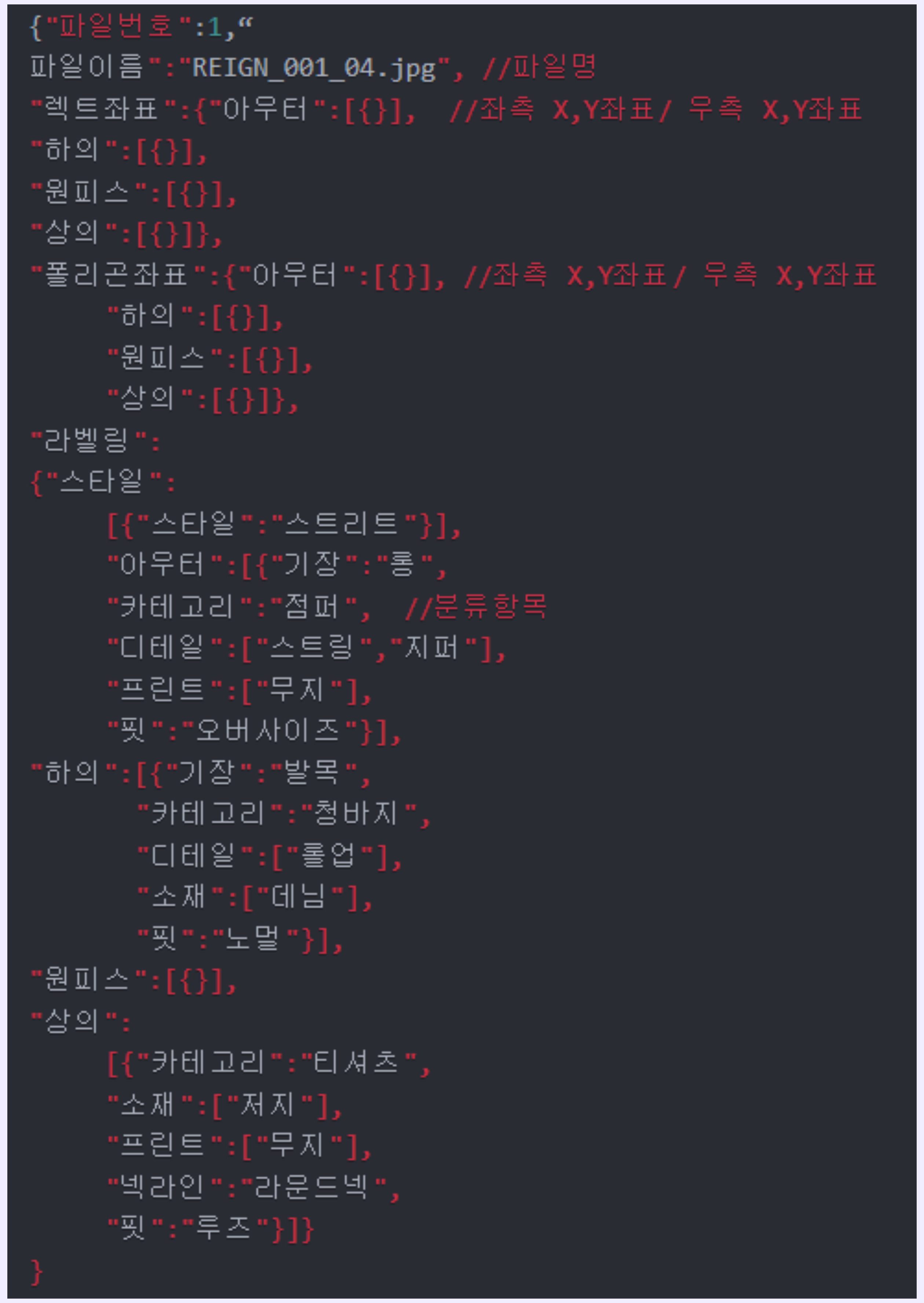




| 항목 | 레이블링 |
|---|---|
| 스타일 | 메인 스트리트 서브 |

# 1. Fashion image classification model

## Dataset



```
{"파일번호":1,“
파일이름":"REIGN_001_04.jpg", //파일명
"렉트좌표":{"아우터":[{}],  //좌측 X,Y좌표/ 우측 X,Y좌표
"하의":[{}],
"원피스":[{}],
"상의":[{}]},
"폴리곤좌표":{"아우터":[{}], //좌측 X,Y좌표/ 우측 X,Y좌표
     "하의":[{}],
     "원피스":[{}],
     "상의":[{}]},
"라벨링":
{"스타일":
     [{"스타일":"스트리트"}],
     "아우터":[{"기장":"롱",
     "카테고리":"점퍼",  //분류항목
     "디테일":["스트링","지퍼"],
     "프린트":["무지"],
     "핏":"오버사이즈"}],
"하의":[{"기장":"발목",
       "카테고리":"청바지",
       "디테일":["롤업"],
       "소재":["데님"],
       "핏":"노멀"}],
"원피스":[{}],
"상의":
     [{"카테고리":"티셔츠",
     "소재":["저지"],
     "프린트":["무지"],
     "넥라인":"라운드넥",
     "핏":"루즈"}]}
}
```

Input data

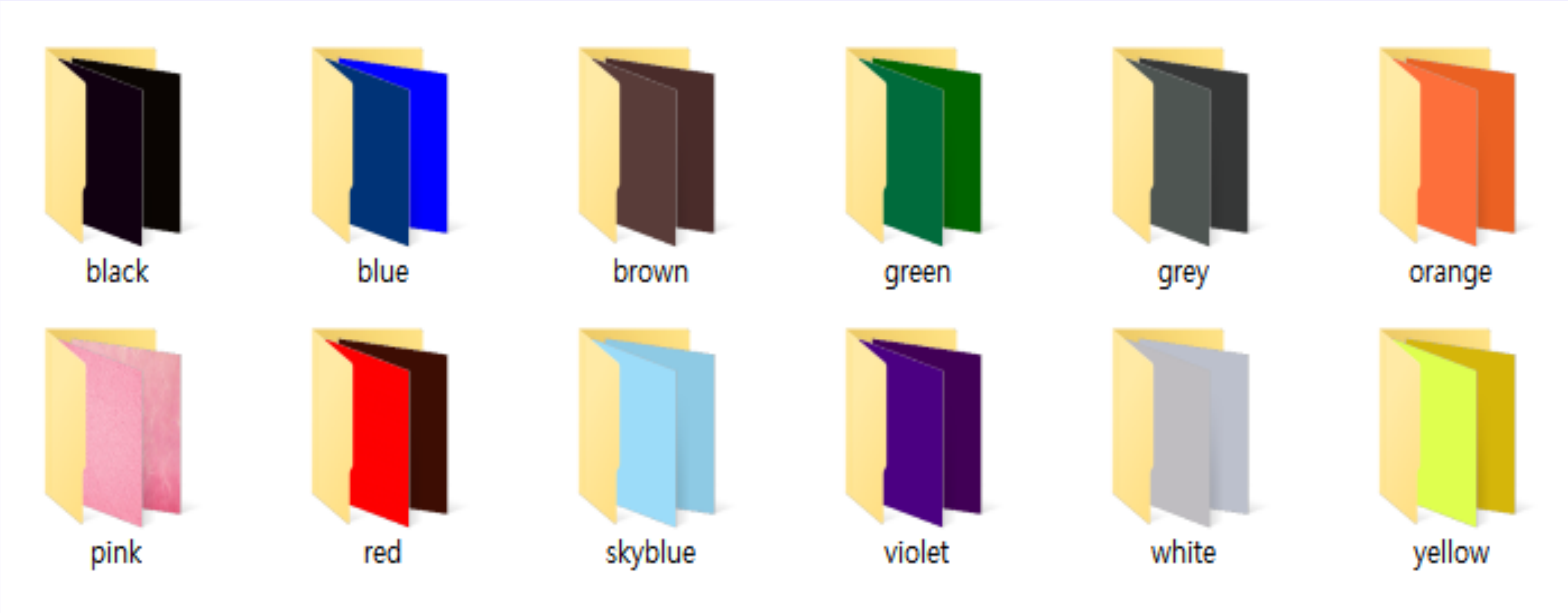
⇨

1. Color
2. Category
3. Length
4. Fit

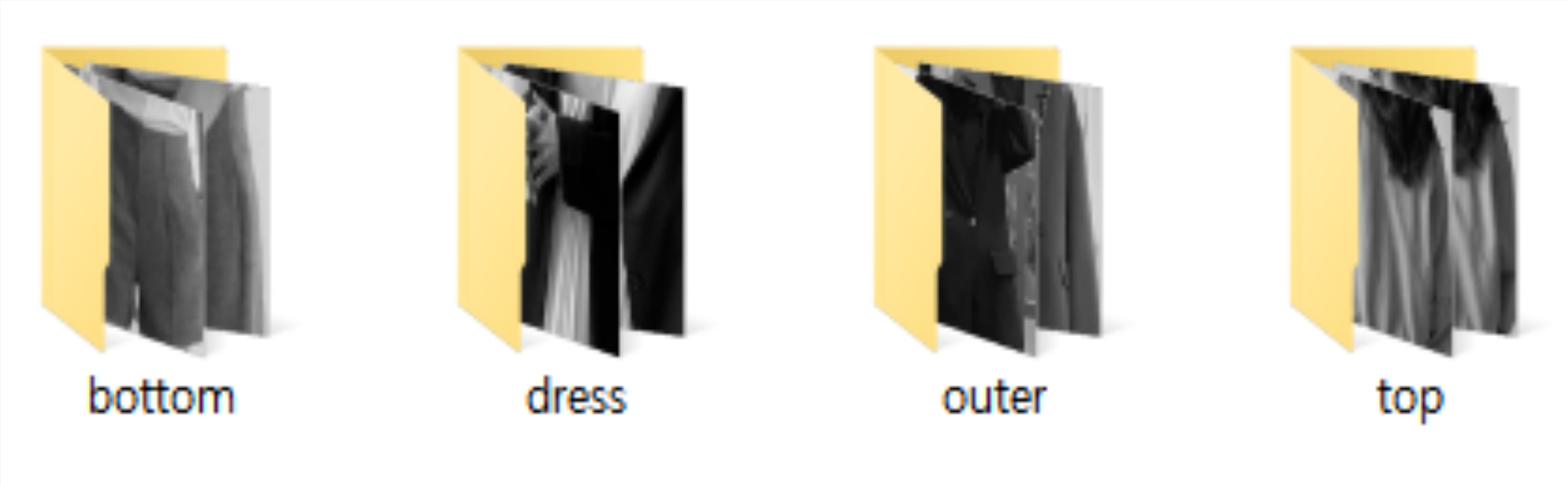
1. Fashion image classification model

# Dataset

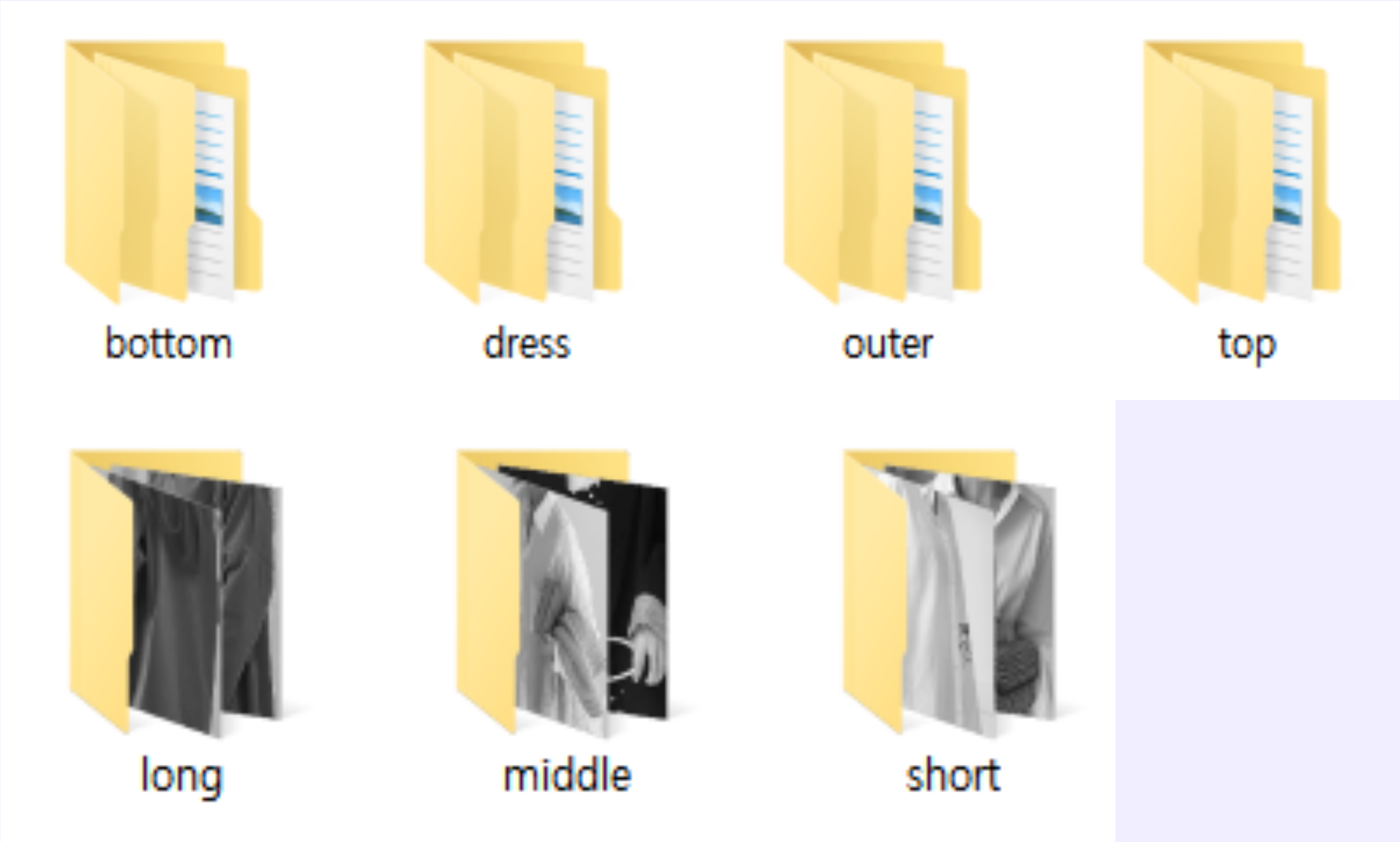
## 1. Color



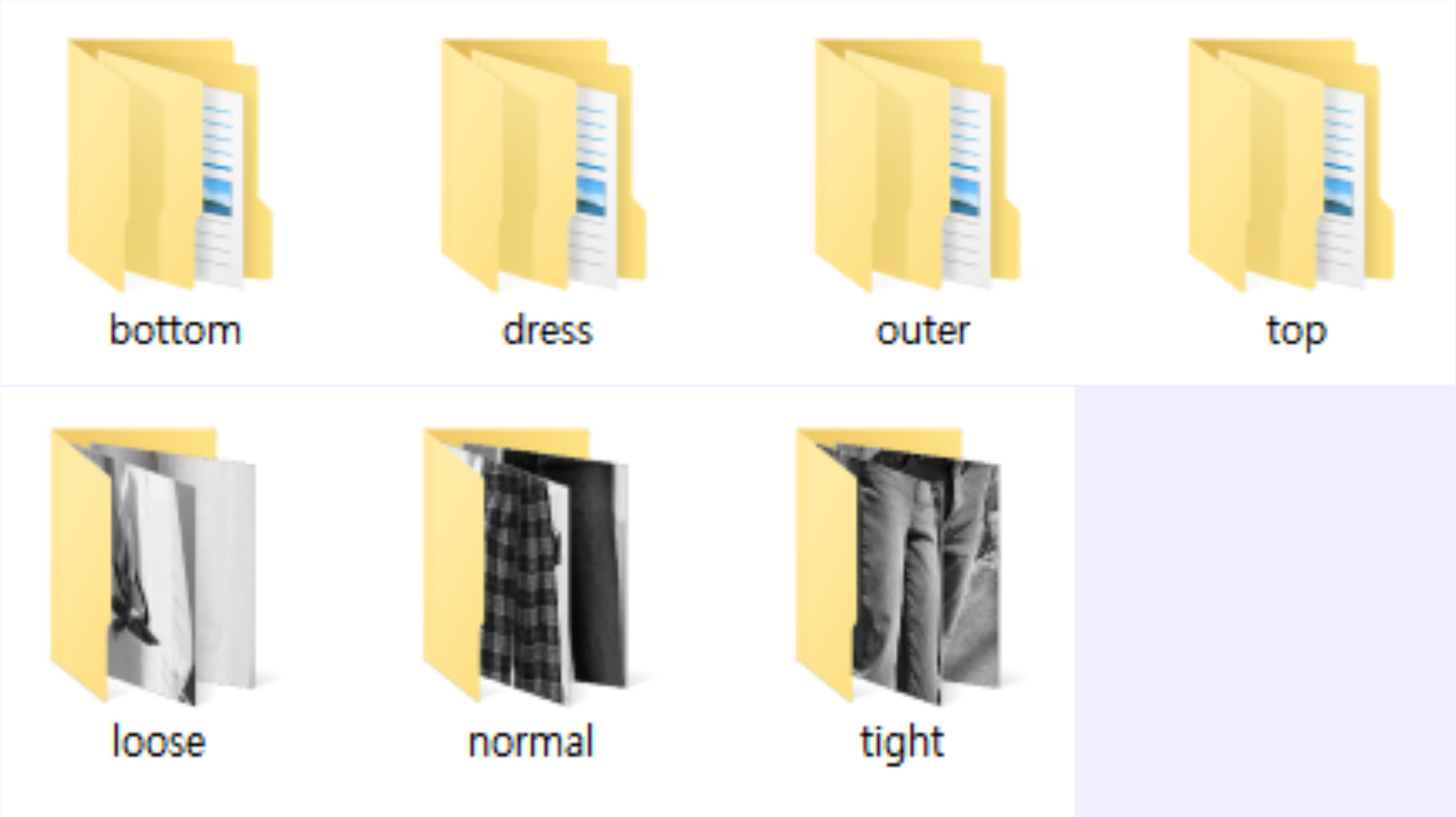
## 2. Category



## 3. Length



## 4. Fit



# 1. Fashion image classification model

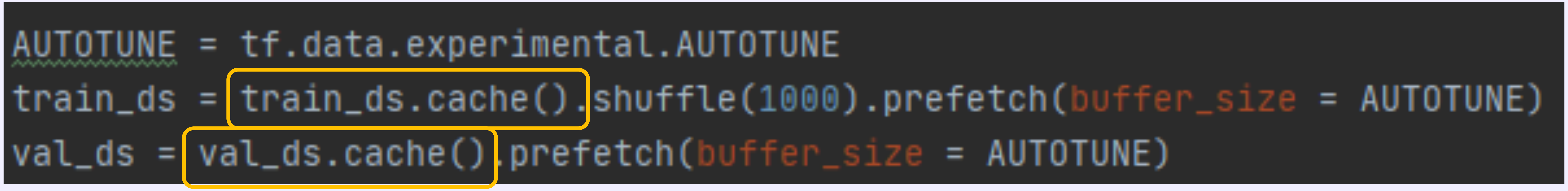
## Model

TensorFlow ™

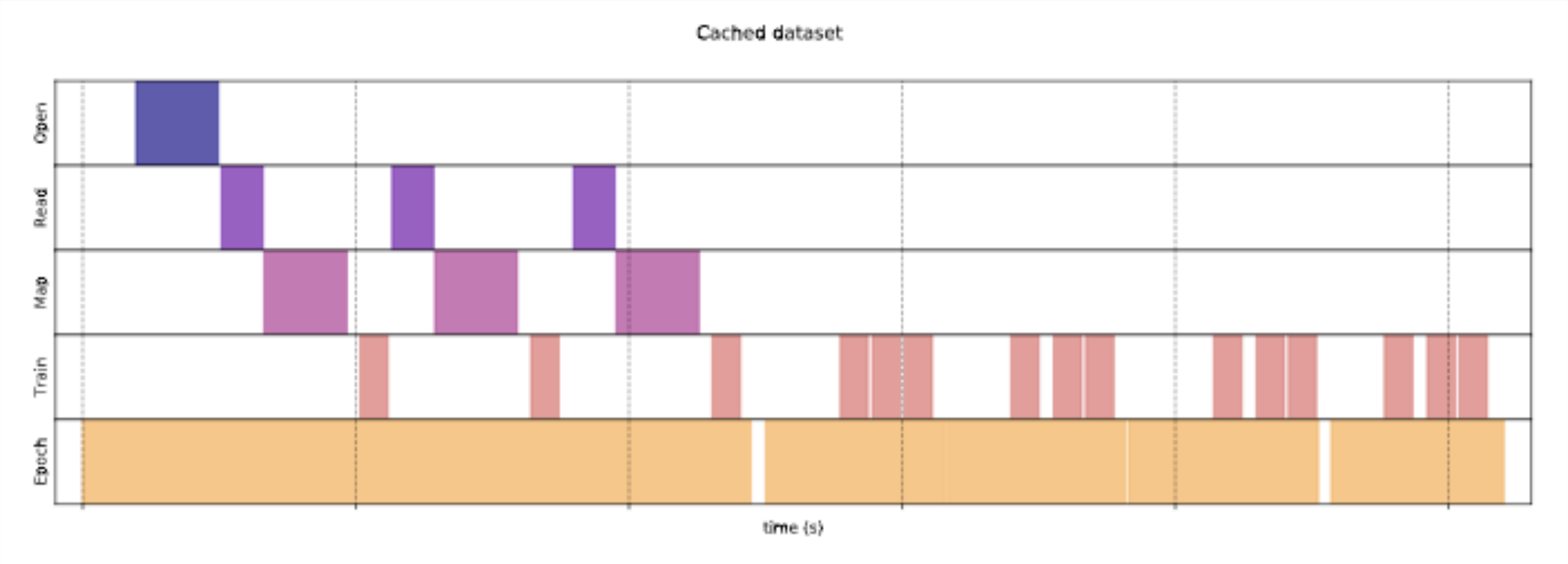
```
batch_size = 32
train_ds = tf.keras.preprocessing.image_dataset_from_directory(data_dir, validation_split = 0.2, subset = "training",
                                                               seed = 123, image_size = (img_height, img_width), batch_size = batch_size)
val_ds = tf.keras.preprocessing.image_dataset_from_directory(data_dir, validation_split = 0.2, subset = "validation",
                                                             seed = 123, image_size = (img_height, img_width), batch_size = batch_size)
```

```
model.compile(optimizer = 'adam', loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits = True), metrics = ['accuracy'])
epochs = 15
history = model.fit(train_ds, validation_data = val_ds, epochs = epochs)
```

# Model



```
AUTOTUNE = tf.data.experimental.AUTOTUNE
train_ds = train_ds.cache().shuffle(1000).prefetch(buffer_size = AUTOTUNE)
val_ds = val_ds.cache().prefetch(buffer_size = AUTOTUNE)
```

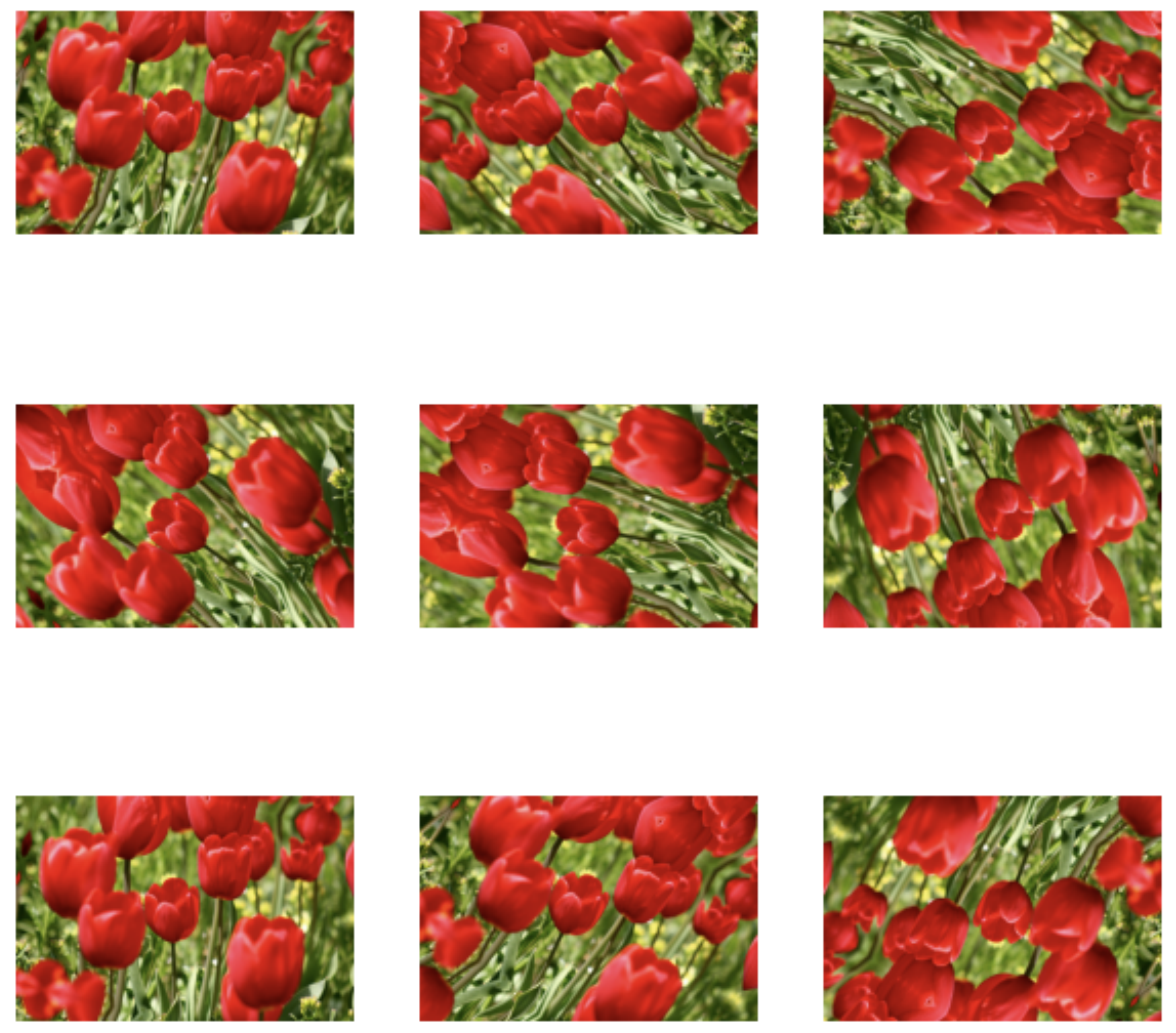


# Model

1. Fashion image classification model

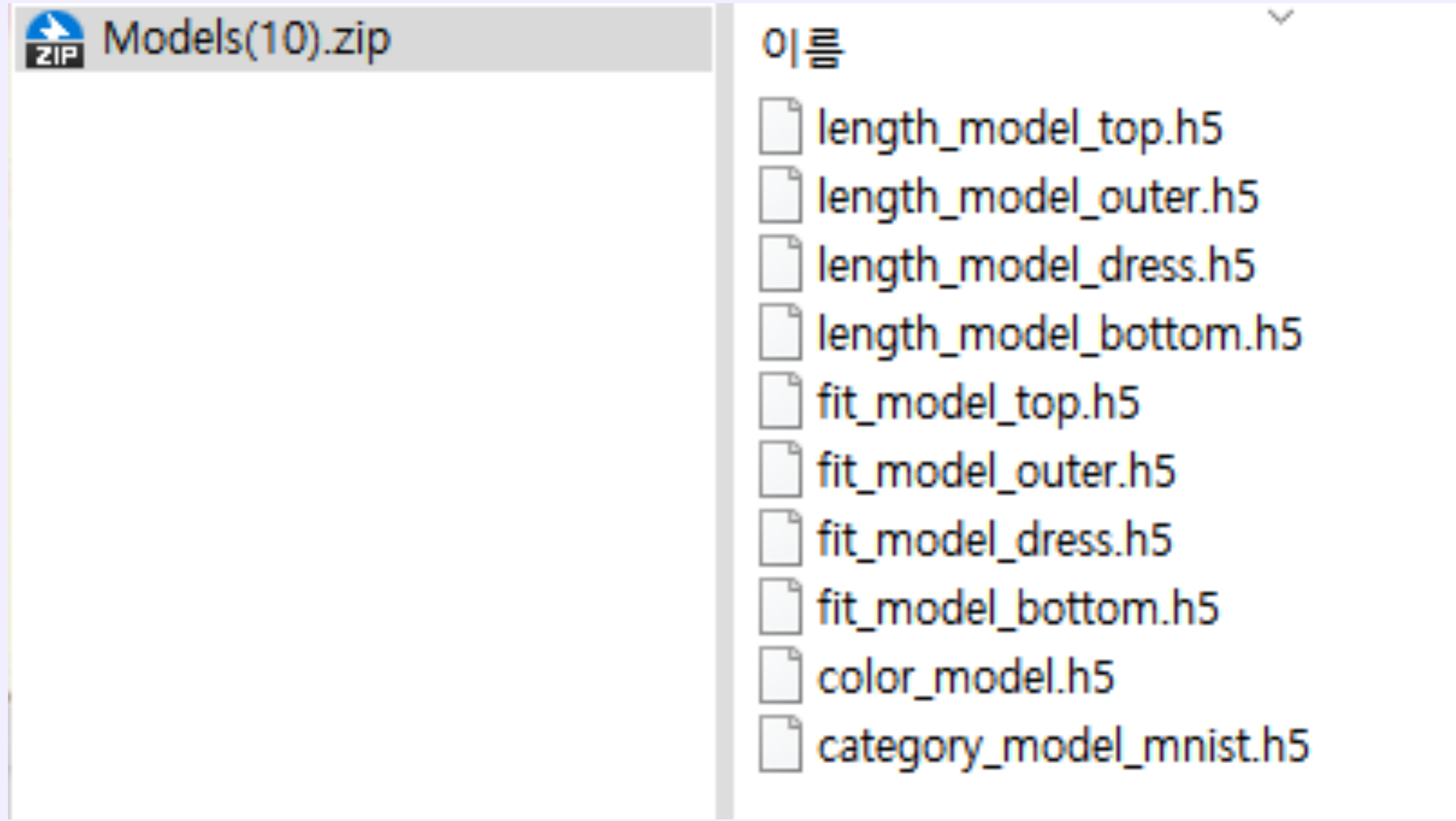
```
data_augmentation = keras.Sequential([layers.experimental.preprocessing.RandomFlip("horizontal",
                    input_shape = (img_height, img_width, 3)), layers.experimental.preprocessing.RandomRotation(0.1),
                    layers.experimental.preprocessing.RandomZoom(0.1)])
```

Example image



# Model

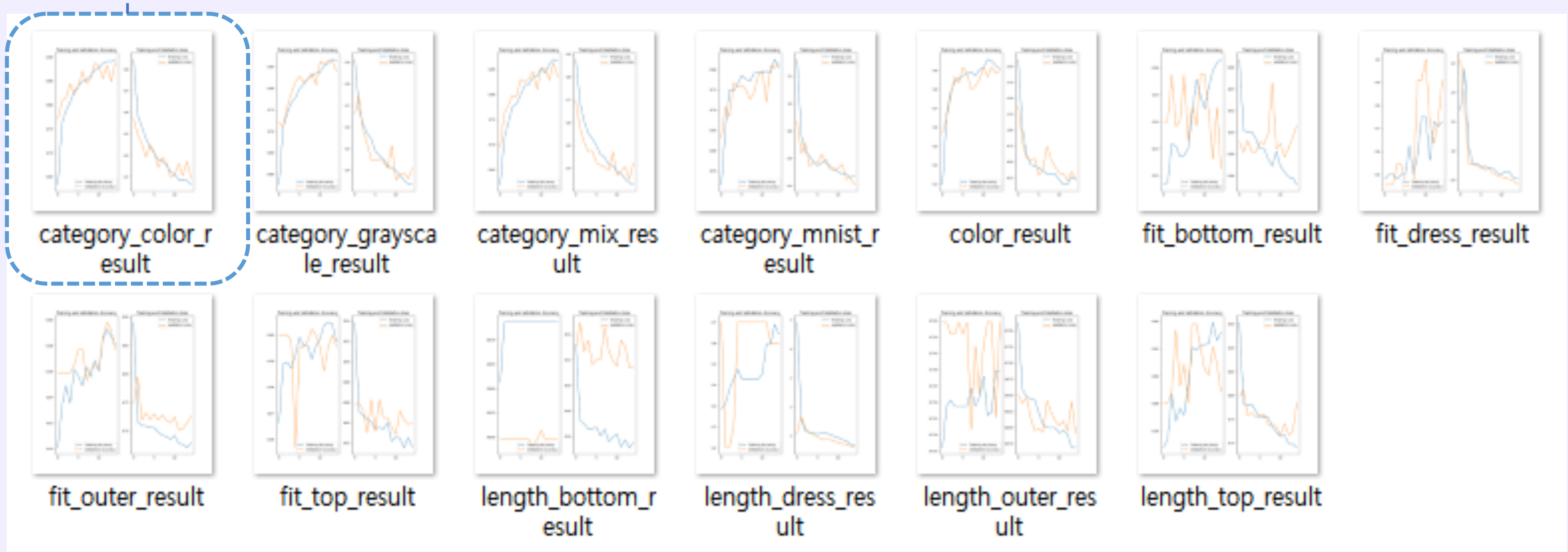
## 1. Fashion image classification model



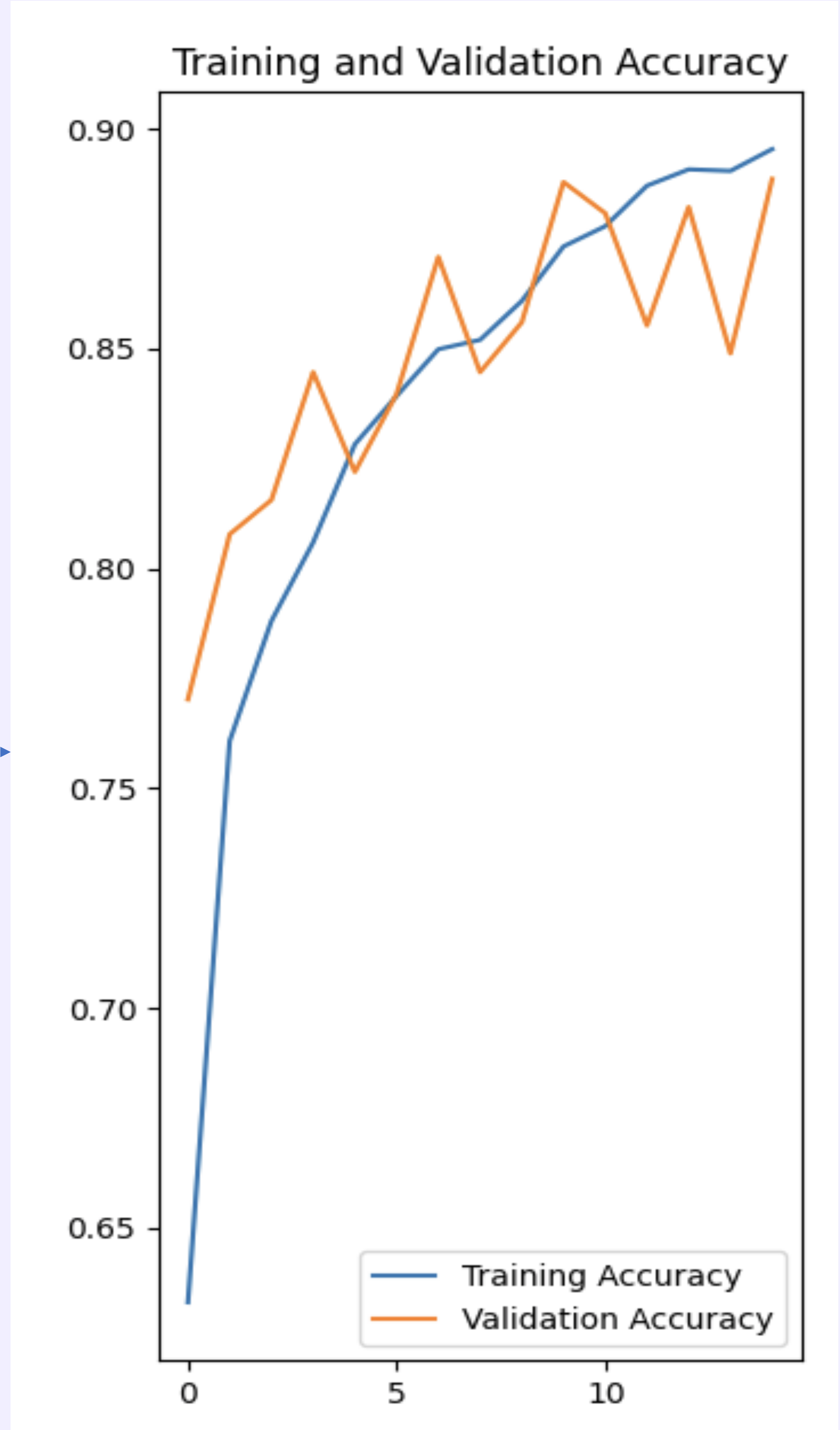
Models(10).zip

이름

- length_model_top.h5
- length_model_outer.h5
- length_model_dress.h5
- length_model_bottom.h5
- fit_model_top.h5
- fit_model_outer.h5
- fit_model_dress.h5
- fit_model_bottom.h5
- color_model.h5
- category_model_mnist.h5

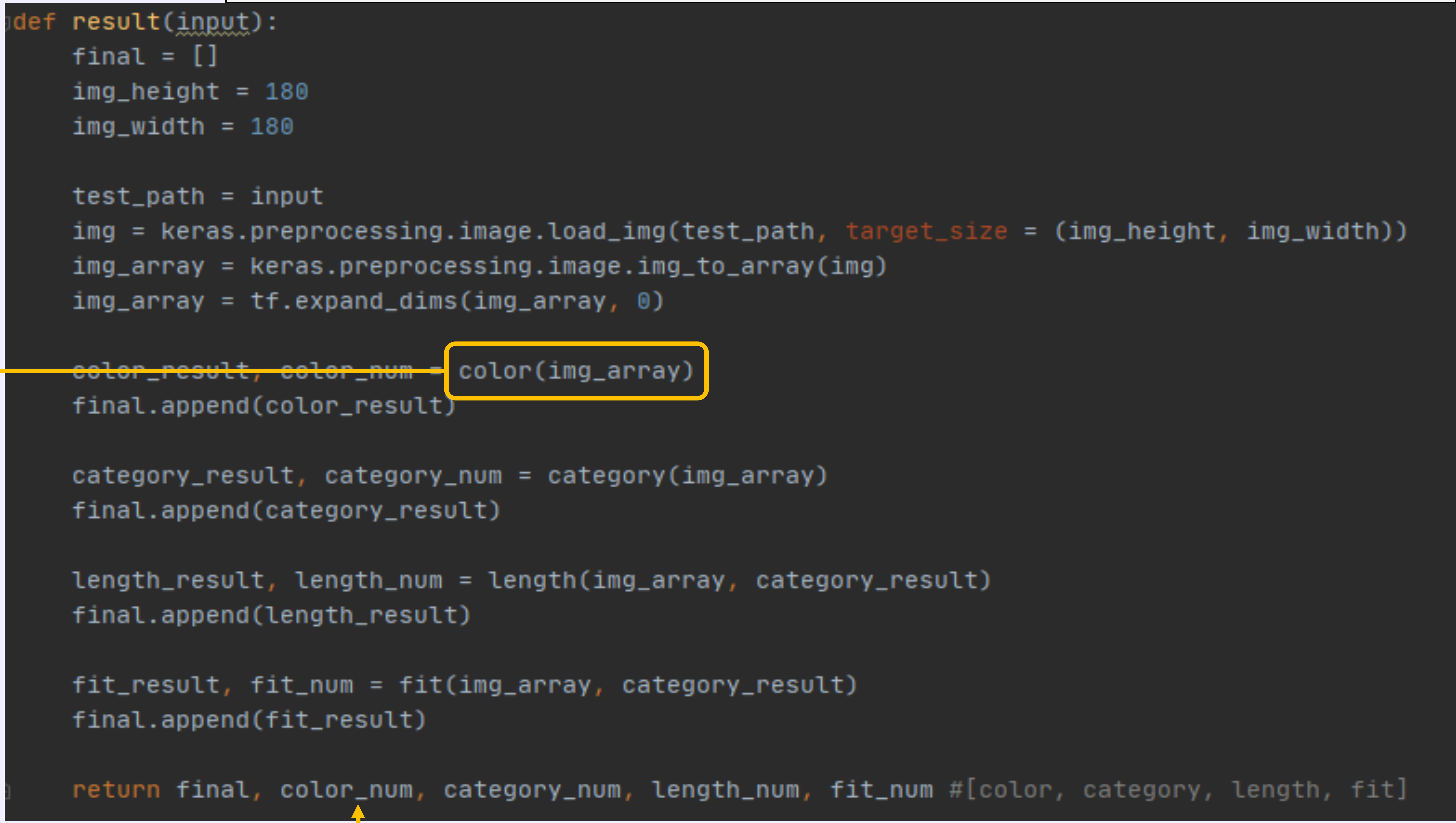


- category_color_result
- category_grayscale_result
- category_mix_result
- category_mnist_result
- color_result
- fit_bottom_result
- fit_dress_result
- fit_outer_result
- fit_top_result
- length_bottom_result
- length_dress_result
- length_outer_result
- length_top_result



## 1. Fashion image classification model

# Input & Output



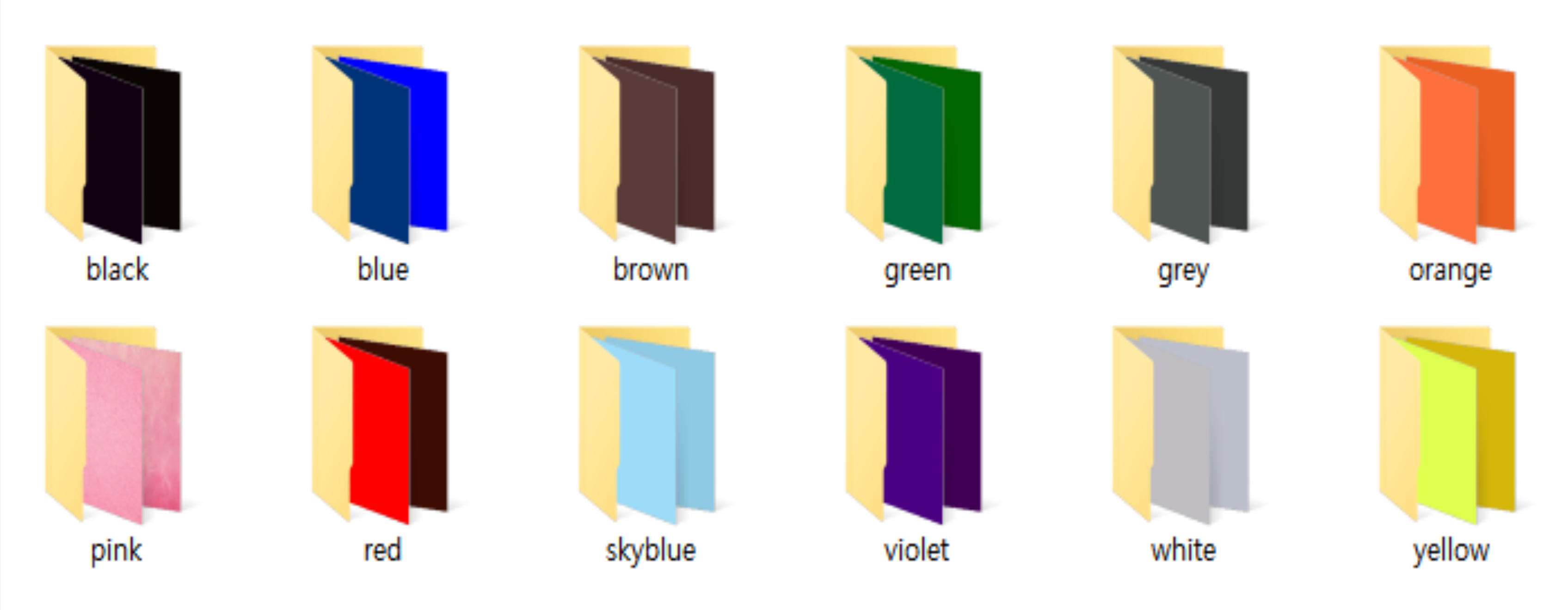
```
def result(input):
    final = []
    img_height = 180
    img_width = 180

    test_path = input
    img = keras.preprocessing.image.load_img(test_path, target_size = (img_height, img_width))
    img_array = keras.preprocessing.image.img_to_array(img)
    img_array = tf.expand_dims(img_array, 0)

    color_result, color_num = color(img_array)
    final.append(color_result)

    category_result, category_num = category(img_array)
    final.append(category_result)

    length_result, length_num = length(img_array, category_result)
    final.append(length_result)

    fit_result, fit_num = fit(img_array, category_result)
    final.append(fit_result)

    return final, color_num, category_num, length_num, fit_num #[color, category, length, fit]
```

```
def color(img_array):
    model = keras.models.load_model("C:/Users/khcho/Desktop/capstone/color_model.h5")
    predictions = model.predict(img_array)
    score = tf.nn.softmax(predictions[0])
    class_names = ['black', 'blue', 'brown', 'green', 'grey', 'orange', 'pink', 'red', 'skyblue', 'violet', 'white', 'yellow']

    return(class_names[np.argmax(score)], np.argmax(score))
```

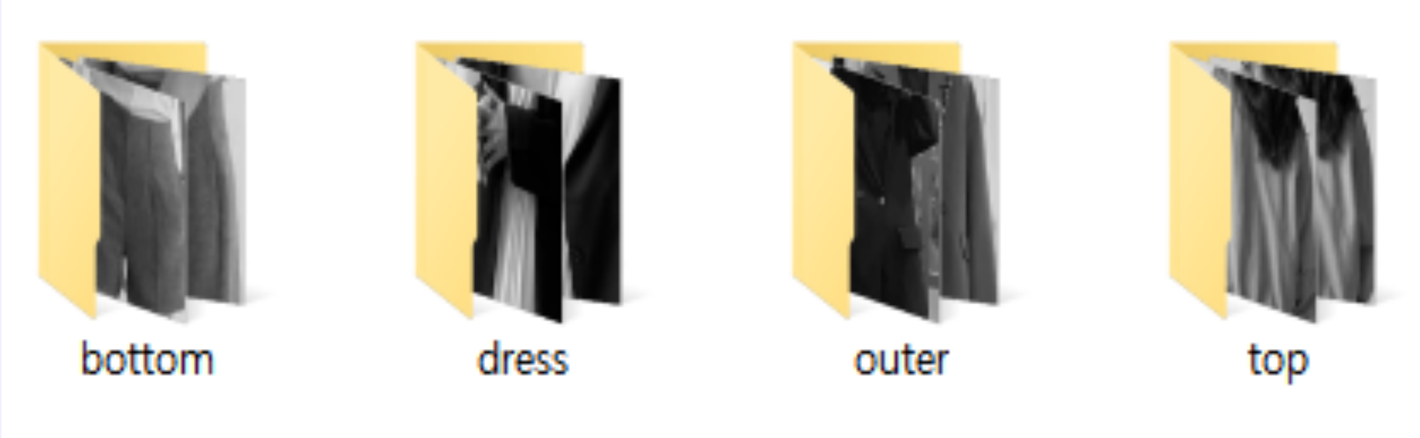
2. Fashion recommendation model

# Dataset

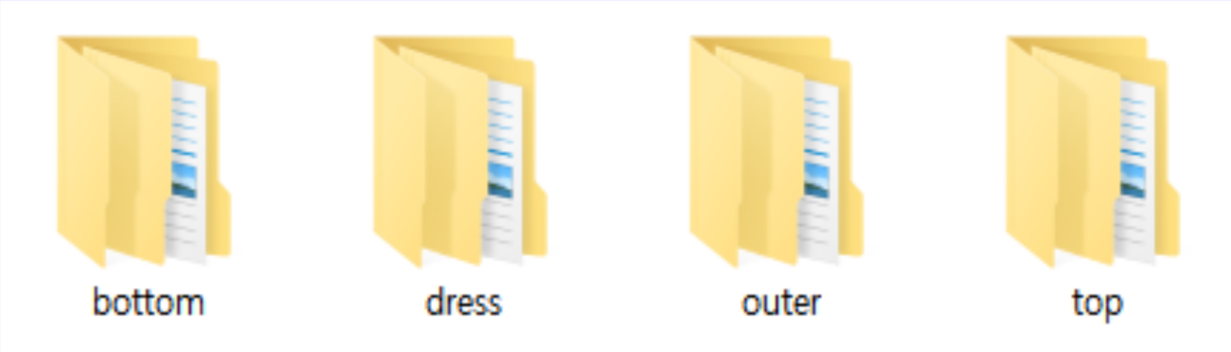
## 1. Color

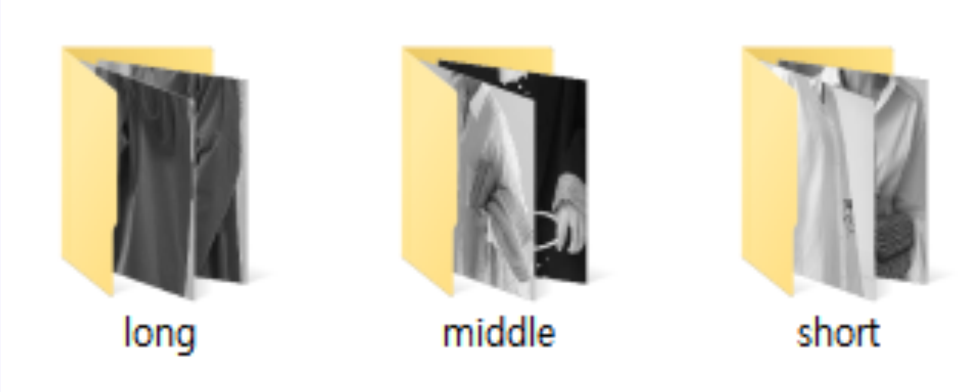


## 2. Category

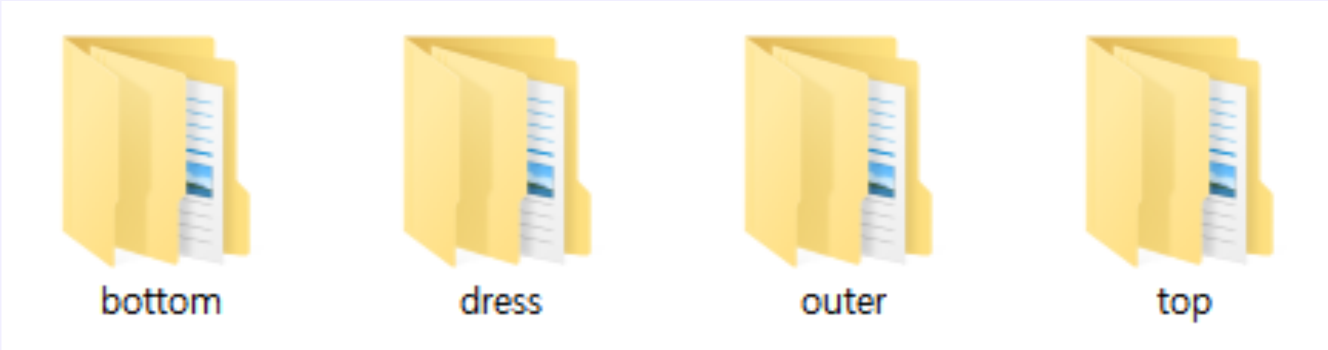


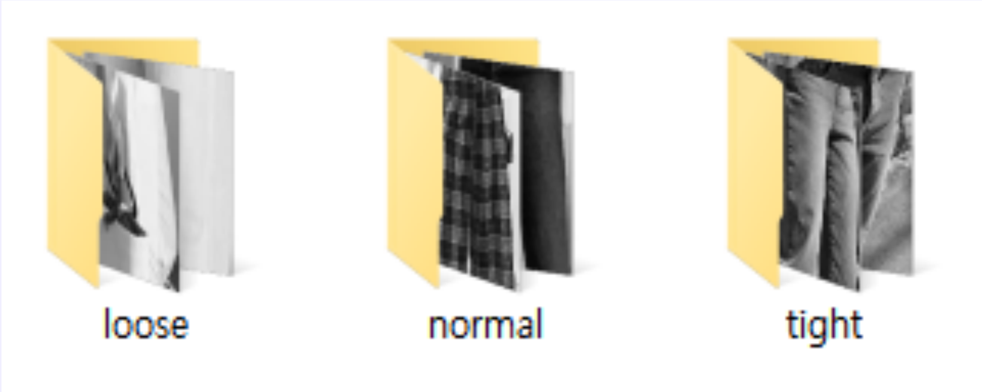
## 3. Length





## 4. Fit





Color(12) * Category(4) * Length(3) * Fit(4) = 432

# Algorithm

- 432*432 zero matrix

  → If the items match each other, add 1.

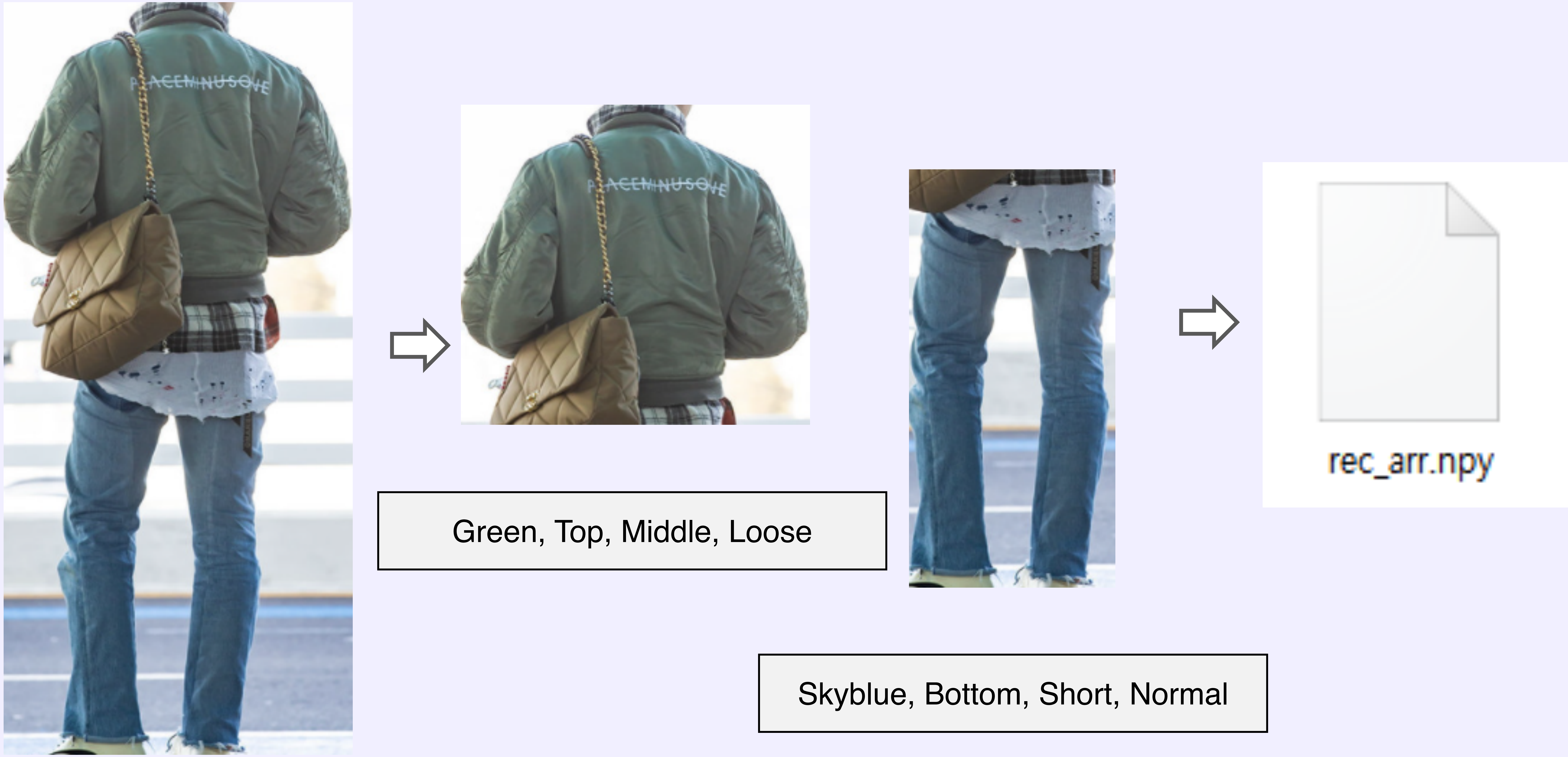
| | | Top, yellow | | | | | | Bottom, black | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Top, yellow | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bottom, black | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Find the fashion vector that matches most.

  → Find the picture of the item in data.

| | | | | | | Bottom, blue | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 5 | 2 | 2 | 3 | 1 |
| Top, yellow | 0 | 0 | 0 | 0 | 0 | 5 | 4 | 5 | 2 | 3 |
| | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 3 | 4 | 4 |
| | 0 | 0 | 0 | 0 | 0 | 2 | 2 | 1 | 2 | 1 |
| | 0 | 0 | 0 | 0 | 0 | 1 | 3 | 1 | 2 | 2 |
| | 5 | 5 | 1 | 2 | 1 | 0 | 0 | 0 | 0 | 0 |
| | 2 | 4 | 2 | 2 | 3 | 0 | 0 | 0 | 0 | 0 |
| | 2 | 5 | 3 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| | 3 | 2 | 4 | 2 | 2 | 0 | 0 | 0 | 0 | 0 |
| | 1 | 3 | 4 | 1 | 2 | 0 | 0 | 0 | 0 | 0 |

# Model

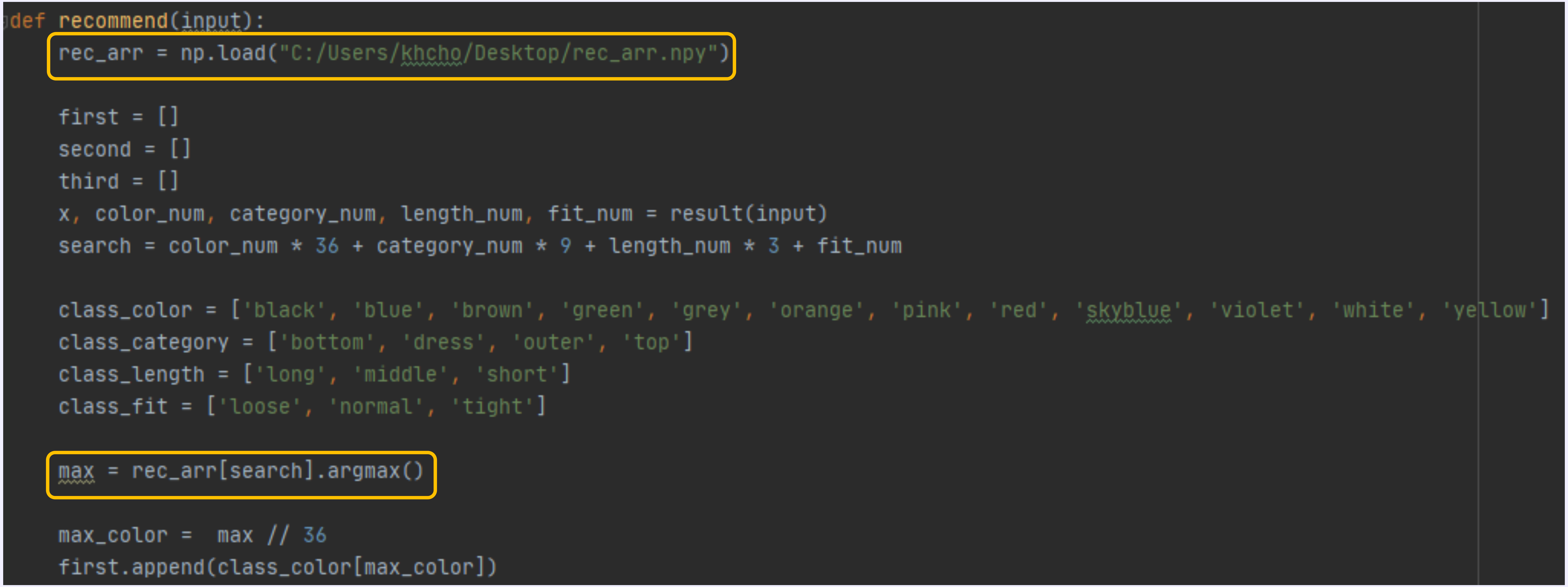


Green, Top, Middle, Loose

Skyblue, Bottom, Short, Normal

2. Fashion recommendation model

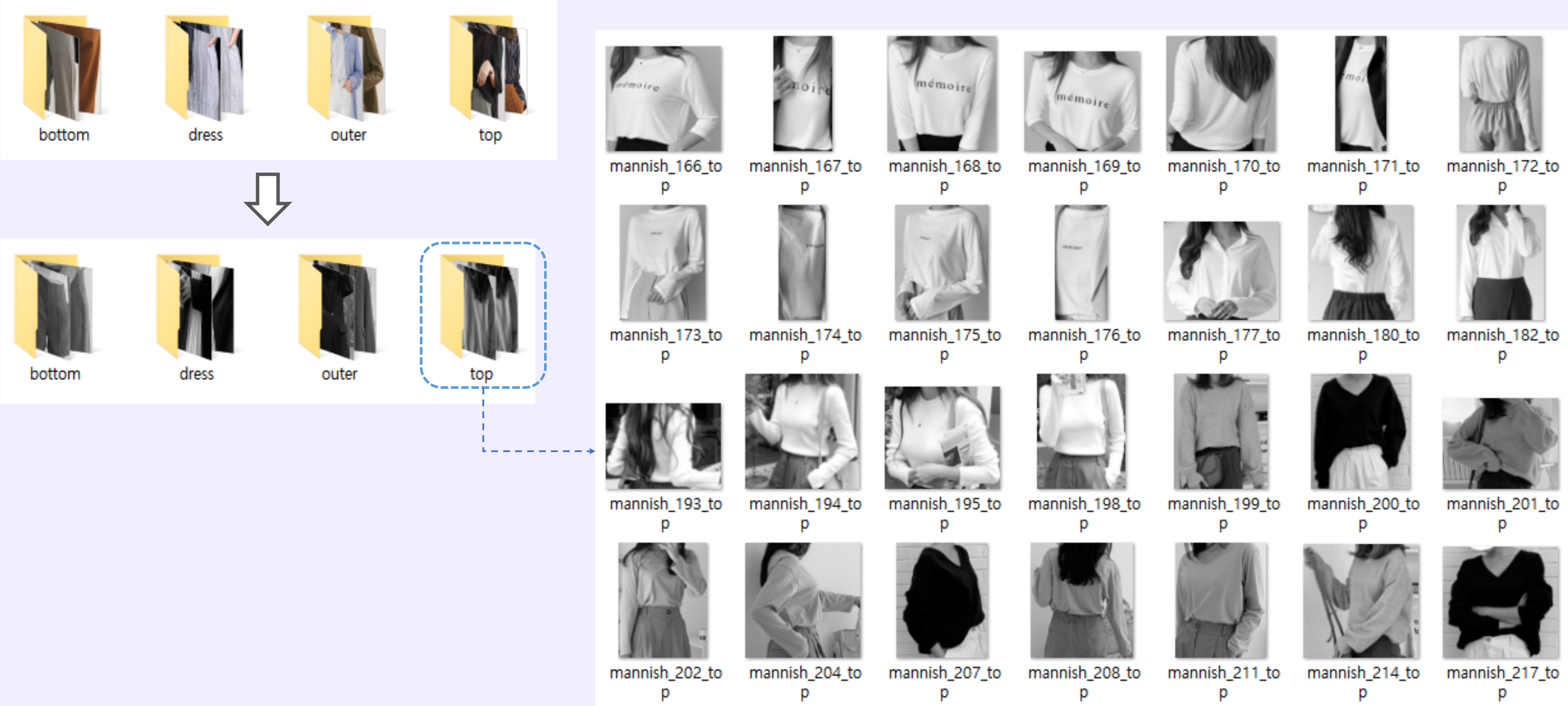
# Model



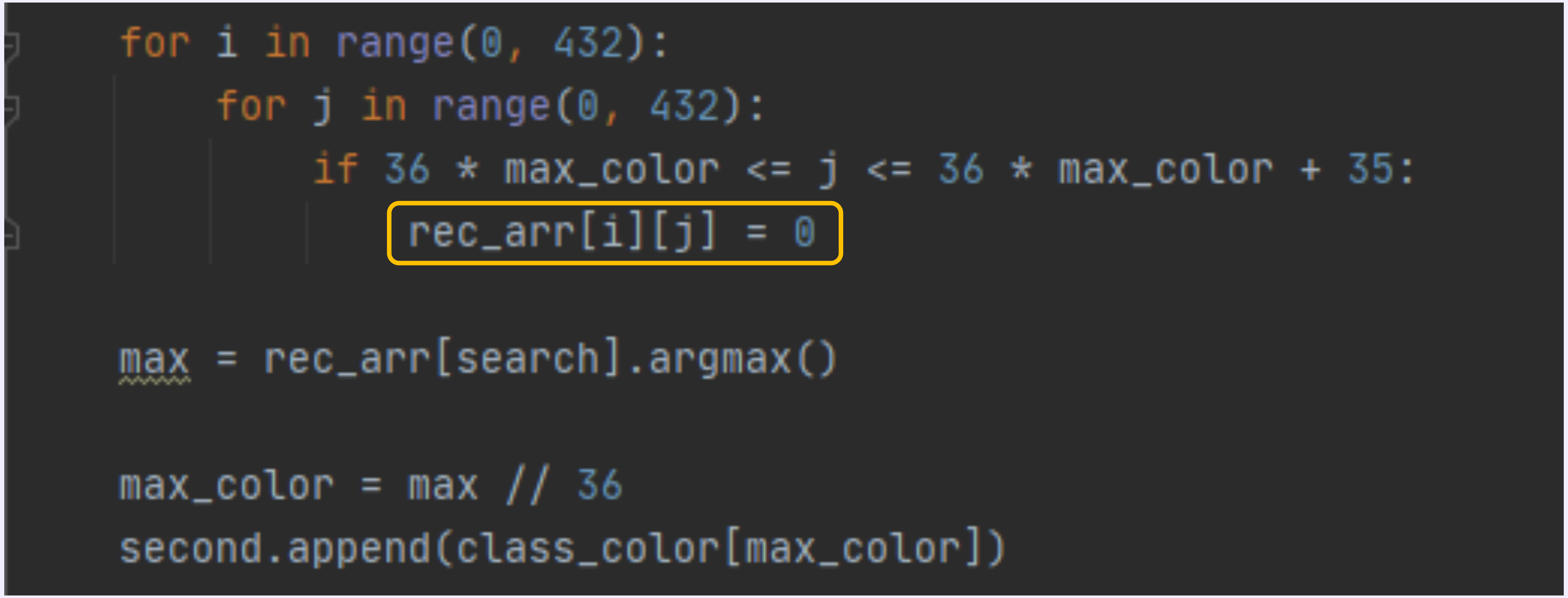
```python
def recommend(input):
    rec_arr = np.load("C:/Users/khcho/Desktop/rec_arr.npy")

    first = []
    second = []
    third = []
    x, color_num, category_num, length_num, fit_num = result(input)
    search = color_num * 36 + category_num * 9 + length_num * 3 + fit_num

    class_color = ['black', 'blue', 'brown', 'green', 'grey', 'orange', 'pink', 'red', 'skyblue', 'violet', 'white', 'yellow']
    class_category = ['bottom', 'dress', 'outer', 'top']
    class_length = ['long', 'middle', 'short']
    class_fit = ['loose', 'normal', 'tight']

    max = rec_arr[search].argmax()

    max_color =  max // 36
    first.append(class_color[max_color])
```

1. Fashion image classification model

# Challenges



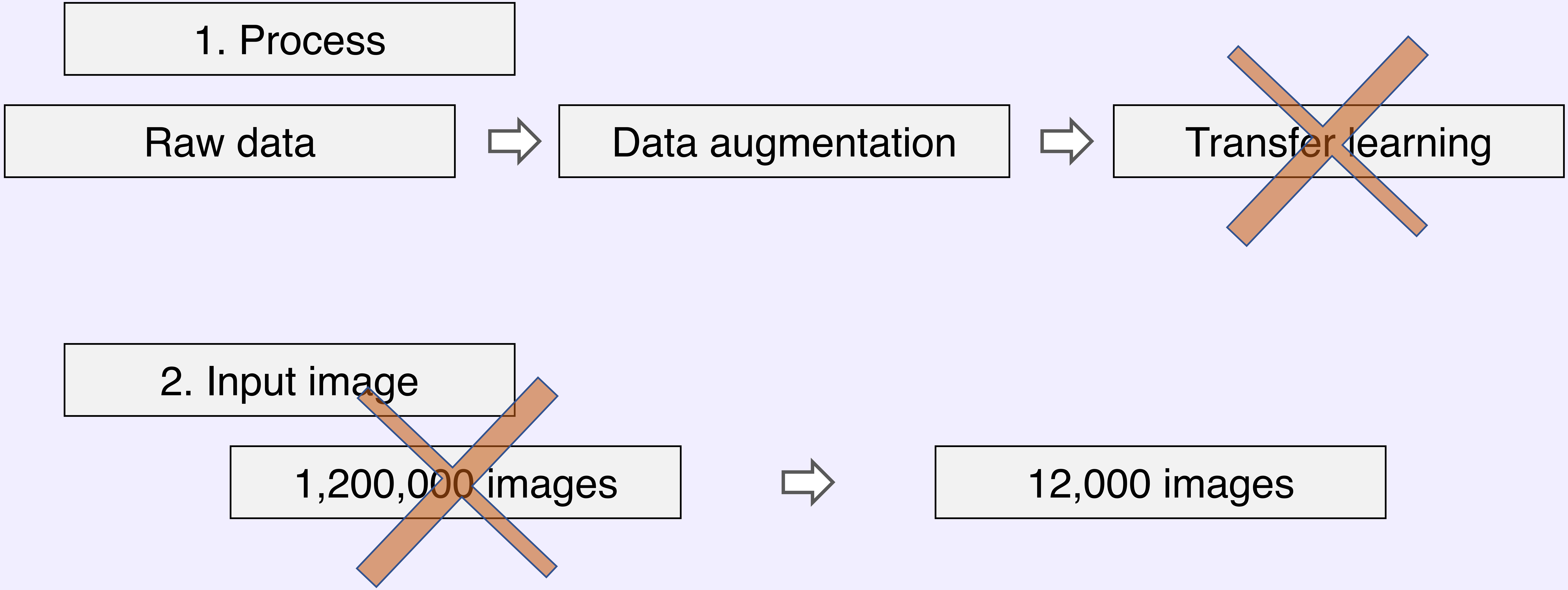
# Challenges



```
for i in range(0, 432):
    for j in range(0, 432):
        if 36 * max_color <= j <= 36 * max_color + 35:
            rec_arr[i][j] = 0

max = rec_arr[search].argmax()

max_color = max // 36
second.append(class_color[max_color])
```

1. Fashion image classification model

# Limitations



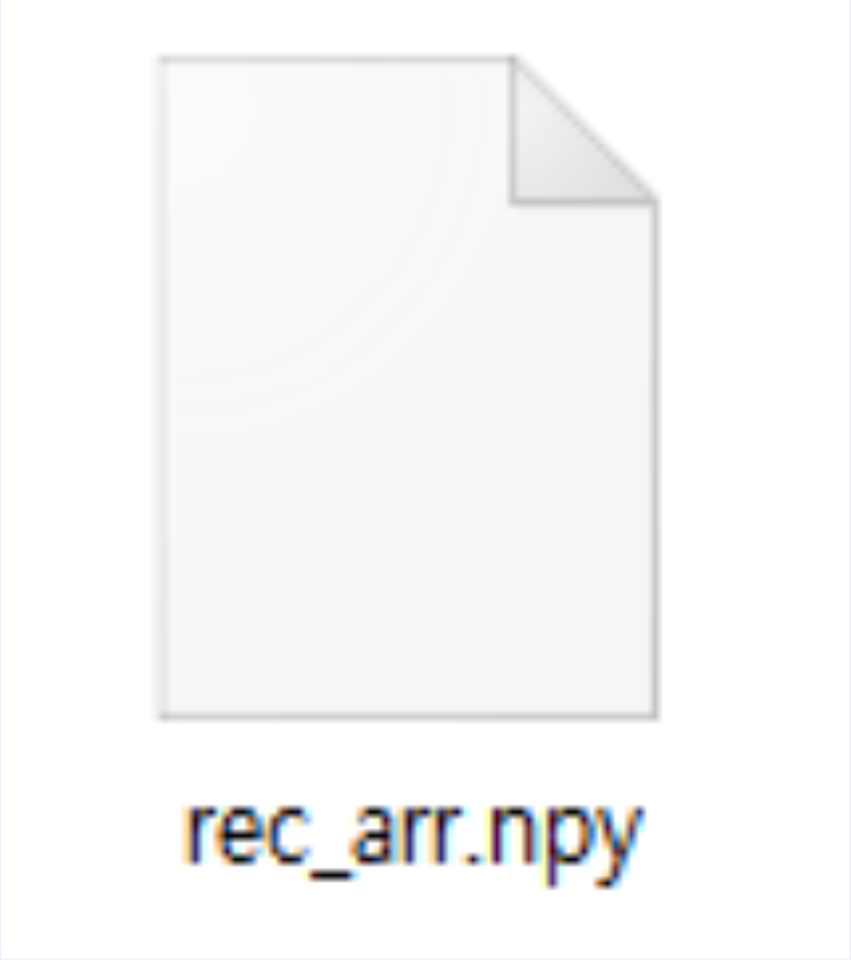
1. Process

Raw data → Data augmentation → ~~Transfer learning~~

2. Input image

~~1,200,000 images~~ → 12,000 images

# Limitations



# Final Design (2)

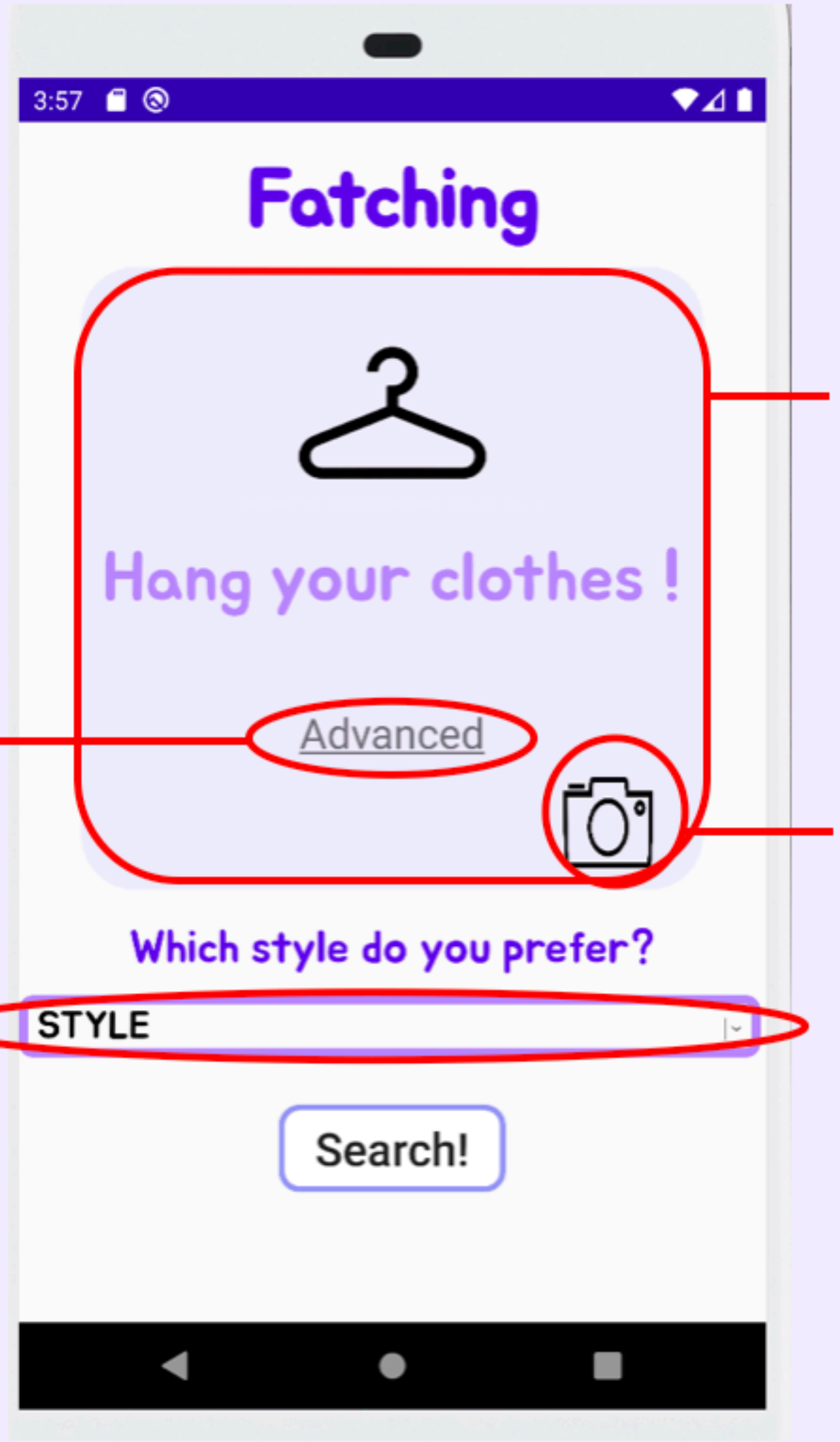
## : Front-End

1. Main Page

2. Advanced Search Page

3. Result page

# Main Page



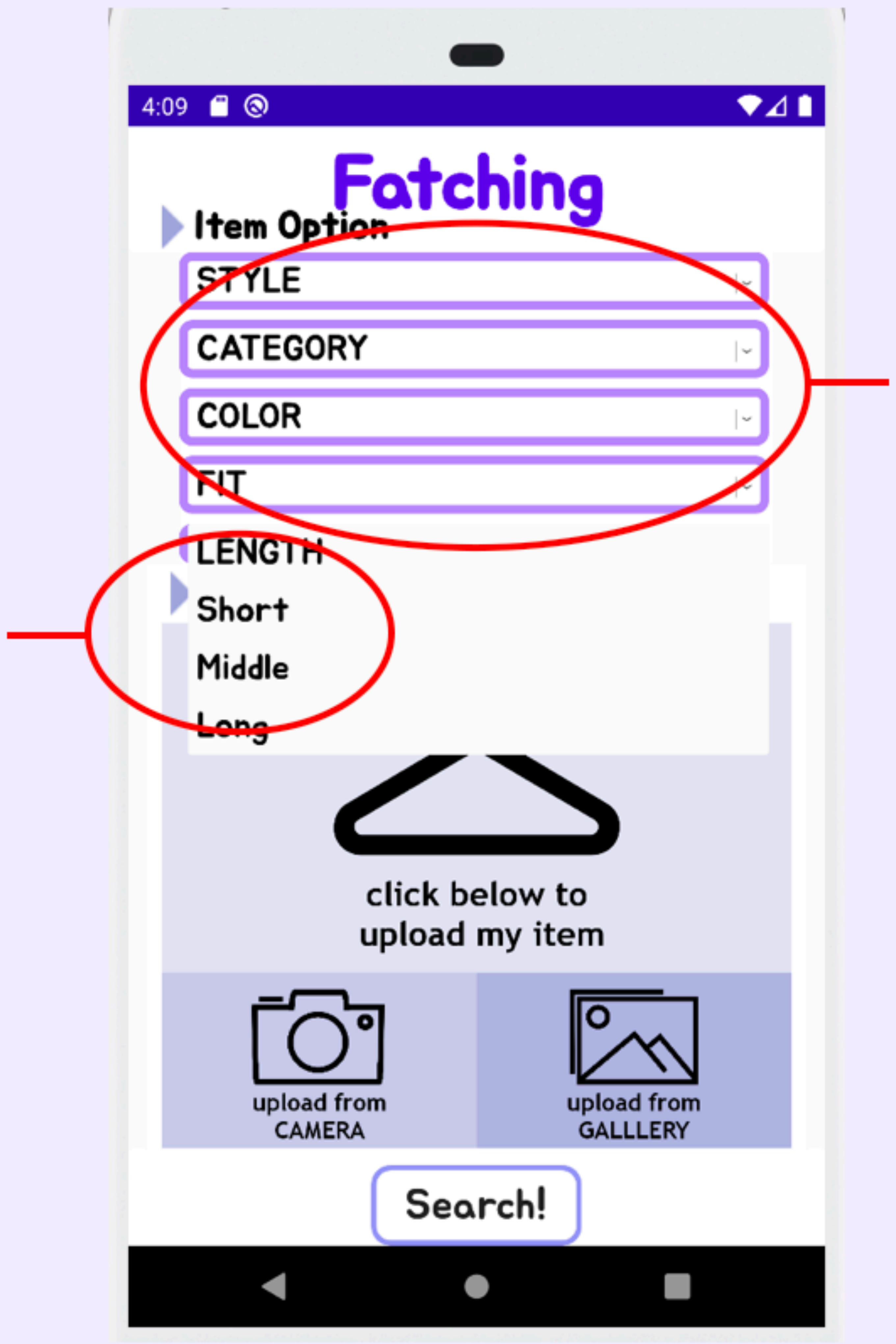
Upload your Fashion Item from the gallery

Move to advanced page

Upload your Fashion Item through camera

Choose a style that you want to recommend

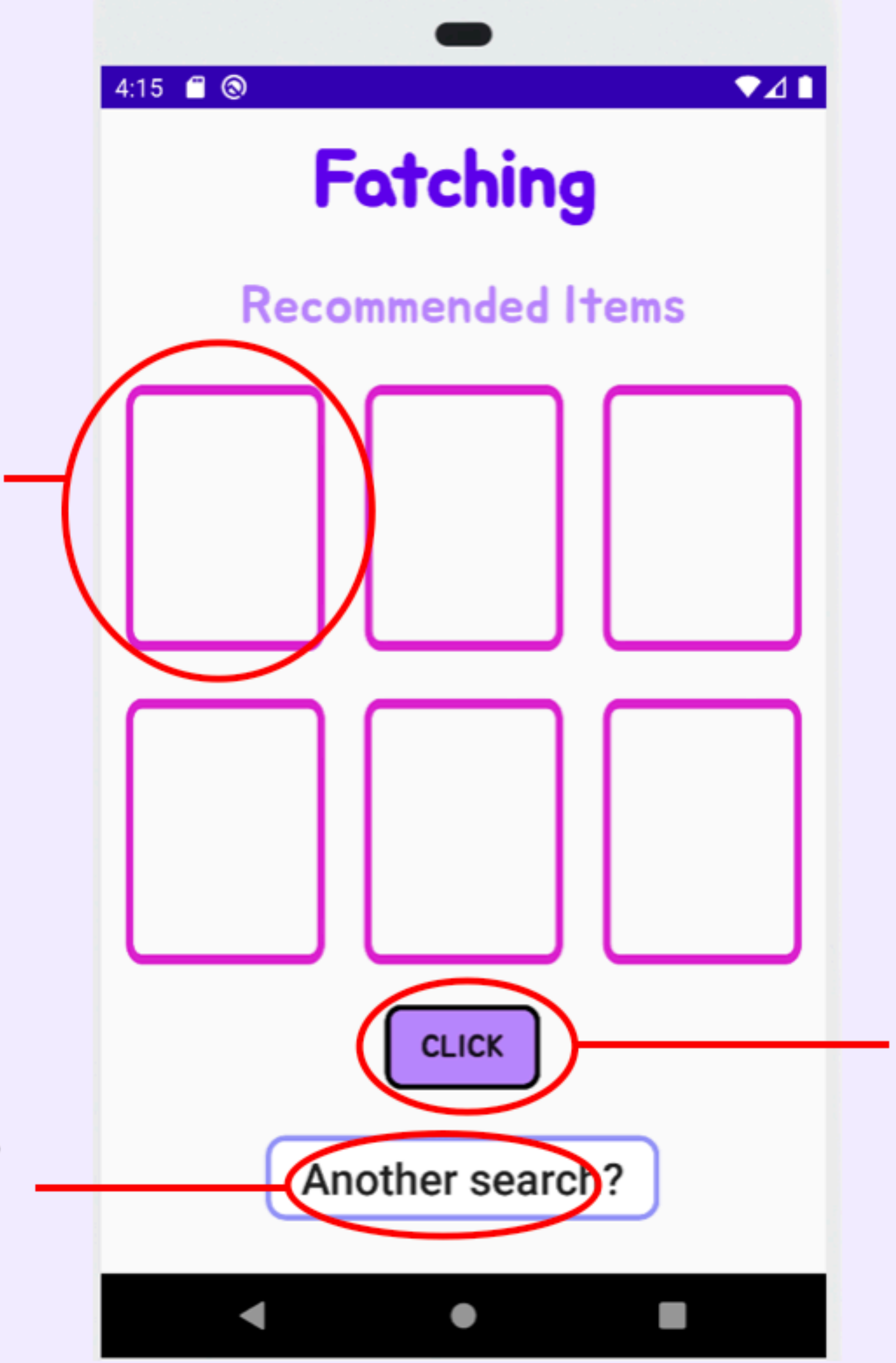
# Advanced Search Page



Choose 5 detailed options

Ex) Length options

# Result Page



ImageView that represents recommended items

Button for bringing in recommended items from the server

Go back to main page to search another item

# Final Implementation

## : Demo Video

# Thank you!