Image Similarity-Based Fashion Recommendation Web Application Using Angular And Machine Learning.

by

Sanjida Ali Shusmita
22373003

A project submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
M.Engg. in Computer Science and Engineering

Department of Computer Science and Engineering
Brac University
January 2023

Declaration

It is hereby declared that

1. The project submitted is my own original work while completing degree at Brac University.
2. The project does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The project does not contain material that has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. I have acknowledged all main sources of help.

Student's Full Name & Signature:

Sanjida Ali Shusmita

22373003

Approval

The project titled "Image Similarity-Based Fashion Recommendation web application using angular and Machine learning" submitted by

Sanjida Ali Shusmita (22373003)

Of Fall, 2022 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of on January 19, 2023.

Examining Committee:

Supervisor:
(Member)

Moin Mostakim

Senior Lecturer
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Amitabha Chakrabarty

Associate Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi

Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Ethics Statement

This project was carried out in complete compliance with research ethics norms, and the codes and practices set by BRAC University. I have ensured that all my sources have been cited. As the author of this project, I take full responsibility for any ethics code violations.

Abstract

People utilize fashion as a significant form of self-expression for a variety of reasons. It appears to be an essential component of every person's existence in contemporary civilizations, from routine activities to noteworthy moments and events. Since there is a great demand for fashionable goods, the fashion sector is viewed as desirable and lucrative. Although there is a great chance for businesses to engage in industries related to fashion because of the enormous demand, there are also a number of difficulties in meeting the demands of the market. Systems that recommend clothing have been developed to meet these needs. The complex conceptions of this domain and their relevance have been developed, justifying fashion domain-specific traits. Retrieving clothing items, recommending complementary items, outfit recommendations, and capsule wardrobes are the four core functions of image-based fashion recommendation systems. There have been three primary eras and the most recent breakthroughs depicted in an evolutionary trajectory of image-based fashion recommend systems with regard to computer vision advancements. In this project, a CNN-based transfer learning approach for recommending fashion items was implemented. And for visual representation, an angular template was used to show the results. For implementing this, I tried various transfer learning image classification algorithms, among which Resnet50 got the best result. Also, two classifiers were used to improve the performance of the algorithm.

Keywords: Image-Based, Machine Learning, Deep Learning, Transfer learning, Web application, Angular.

Dedication

I would like to dedicate this research to my parents who have brought me to this world and nurtured me to become an adult and I am also thankful to my teachers for their continuous support and valuable guidelines.

Acknowledgement

Firstly, all praise to the Great Allah for whom my project has been completed without any major interruption.
Secondly, many thanks to my respected supervisor Moin Mostakim Sir for his inspiration and guidelines. Even though I did not know if I could do such a good job but my supervisor believed in me, pushed hard for every detail, and with his clear vision and perfect instruction, I was able to complete this project.
And finally to my parents without their support it may not be possible. With their kind support and prayer, we are now on the verge of our post-graduation.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Aims and Objectives

The goal of the project is to develop an intelligent fashion recommender. The system will take an image of any fashion product and will be able to recommend similar fashion products available in the inventory based on a similarity ranking. In addition, it will consider some user-specific data for personalizing purposes. For example, if a user does not like the color yellow, doesn't recommend the product for its color. A customer may upload an image of a fashion item or the image might be selected from an image database, then the recommender system will calculate the similarity ranking and predict some similar fashion items based on similarity ranking. The image dataset will have metadata for the images. This metadata includes the information of gender, age category(men/women/girls/boys), item category, season, year, etc of a certain fashion item. I will include these metadata of a fashion item image to filter age, gender, season, etc based on fashion that are complex(Not possible) to get from the image alone. It will help us to avoid age and gender-related major issues(mistakes) to make the system more accurate. I will use(train) ImageNet based Keras pre-trained models(ResNet, Xception, Inception, NASNetLarge) to extract features of fashion items. Here I will apply cosine similarity on extracted features to get image similarity ranking and will use a triplet of images to get image similarity. There will be the given image, the positive(most similar) image, and the negative(most dissimilar) image, from these three images our model will find the most similar images for a recommendation.

1.2 Project Problem

I will discuss the key difficulties that recommend systems in the fashion industry confront in this part:
Due to the lack of purchase data or the lack of information about the aesthetic appearance of the product in category names, traditional recommend systems like Collaborative Filtering or Content-Based Filtering struggle in the fashion sector [5]. Recent literature, on the other hand, has made use of models that capture a rich representation of fashion items through product images [4], text descriptions or customer reviews [7] [6], or videos [5], which are frequently learned through surrogate tasks like classification or product retrieval. However, in order to generalize effectively across various picture (or text) styles, attribute changes, etc. while learning

product representations from such input data, vast data-sets are required. Furthermore, there is still work to be done in developing a representation that learns which product aspects buyers value the most when assessing fashion products.

1.3 Literature Review

Deldjoo et al.[13]recently published a study on RS that makes use of multimedia content, such as visual, audio, or textual characteristics. This survey looked at a variety of areas, including media streaming for audio and video suggestions, e-commerce for product recommendations, clothing, news and information recommendation, social networking, and more. Although the writers talked about fashion RS as well, they only included a limited selection of works and subjects in this field. Here, we analyze and provide a thorough analysis of key duties, difficulties, and material types encountered in the project.
Additionally, I have located surveys [10][3]where the authors provide an overview of the literature on methods combining computer vision (CV) and/or natural language processing with fashion (NLP). Despite the fact that I believe these works are pertinent to this topic, they continue to be review given here because those systems don't concentrate on RS but rather on other facets of the fashion industry, like posture estimation or text generation from photographs. Additionally, I offer current methods for representing item visual and textual content that are used by RS approaches as additional point of distinction.
The most recent book chapter on fashion RS by Jaradat et al.[8] is possibly the most pertinent piece of research to our current survey. In particular, the authors assert that deep learning represents a turning point in comparison to the traditional methodologies and as a result, the authors analyzed four different tasks that leverage this new methodology. This chapter focuses on discussing the state of the art of fashion recommendation systems. They also included examples, potential issues, and their assessment. The authors specifically concentrated on the size suggestion problem and social media duties in their review.
I studied another paper where they recommends items based on visual similarity using CNN model[11]. In this approach, models are trained on a specific data-set for classification. Their classes are different target classes of fashion items. Features of the images are extracted from the CNN model and get embedding of them. Then calculate the similarity of images applying distance calculating methods(cosine, hamming distance, etc) on extracted features of images from CNN model. Return the most similar images based on the calculated similarity score.
Another paper is recommendations based on embedding and calculating distance[9] Make the layers of ImageNet-based Keras pre-trained model non-trainable and feed images for feature extraction. Get the embedding of all images from the model.Calculate the distance of all images from embedding applying distance calculation methods.Return the most similar images.
I also researched the paper on applying transfer learning to pre-trained models.[2]Apply transfer learning on image classification models with their own data-set. Classify the images based on your own defined classes of images. Return the images of the identified class.
There is also work done on Recommendations based on image triplets image search technique[8].Apply triplets of image for image searching.Find out similar images

based on the prediction score. Return the matched similar images.
In my survey, in addition to examining the current state of the art of the most popular algorithms across a variety of jobs, I delved further to identify the key elements employed by the more contemporary fashion recommended systems. In fact, a thorough debate is made on how the features of the user and the objects themselves might serve as a source for the construction of models that make precise recommendations. A thorough analysis of computer vision methods for improving the interpretation of item images is also covered here because the fashion domain places a strong emphasis on visual characteristics[12].

1.4 Organization Of The Report

The report is structured in the following manner -

Chapter 1 describes the aims and objectives, project problem, literature review, study approaches, and organization of the report structure of my project work.

chapter 2 briefly explains the business understanding and data understanding of my project. In the business understanding section, I explain the business objective, assess the situation, determine project goals, and plan production. In the data understanding section, I discuss the initial data collection process, data description, data exploration, and verification of data quality.

Chapter 3 briefly explains the methodology part of my project which includes the data set description, proposed approach, reasons to choose the approach, and method.

Chapter 4 explains the experiment part in depth, including the training procedure, evaluation criteria, and model comparison.

chapter 5 describes the results and discussion which contains model exploration1, model exploration2 , transfer learning, and model selection.

Chapter 6 contains the visual representation during working on this project.

Chapter 7 presents the conclusion and recommendations for the future while describing the system's weaknesses.

Chapter 2

Business And Data Understanding

In Business understanding and data understanding phase I try to define the goals of the project by understanding the business needs and the data available to us. I iterate between focusing on the business and exploring what data is available. This iteration typically involves specifying the business problem and then exploring if the appropriate data are available to develop a data-driven solution to the problem. During this stage of the project, I spend a great deal of time collecting data from different sources and trying to understand whether it can fulfill our needs.

2.1 Business Understanding

Understanding the project objectives and requirements from a business perspective, then converting this knowledge into a problem definition and a preliminary plan designed to achieve the objectives. This step includes-

2.1.1 Determine Business Objective:

Understand what the customer really wants to accomplish. Understand the objective and constraints that must have to be properly balanced. Uncover important factors that can influence the project outcome.

2.1.2 Assess Situation:

Find detailed facts about all of the resources, constraints, assumptions, and other factors that should be considered in determining the data analysis goal and project plan.

2.1.3 Determine Project Goals:

Specify the project goals or project objectives in technical terms. That is what the project will do from a business perspective.

2.1.4 Produce Project Plan:

Description of the intended plan for achieving the project goals and thereby achieving the business goals. The plan will specify the steps to be performed during the rest of the project, including the initial selection of tools and techniques.

2.2 Data Understanding

Collecting data and proceeds with activities to become familiar with the data, identifying data quality, discovering first insights into the data, and/or detecting interesting subsets to form hypotheses regarding hidden information. This step includes-

2.2.1 Collect Initial Data:

Acquire the data listed in the project resources. If the data source is multiple then integrate the data. Use the necessary tools to load and understand the data.

2.2.2 Describe Data:

Examine the “gross” or “surface” properties of the acquired data

2.2.3 Explore Data:

Distribute the key attributes, and relationships between pairs or small numbers of attributes. Explore properties of significant sub-populations and do simple statistical analyses. It may address the project goals.

2.2.4 Verify Data Quality:

Examine the quality of the data, addressing questions such as: Is the data complete (does it cover all the cases required)? Is it correct, or does it contain errors, and if there are errors, how common are they? Are there missing values in the data? If so, how are they represented, where do they occur, and how common are they?

Chapter 3

Methodology

3.1 Data set Description

I have collected a data set from Kaggle.It contains about 44424 fashion product images and metadata of all the images in a CSV file. From 44424 data, 5 images were not found, so we have 44419 images. The CSV file is well-filled with corresponding values in each column. It has 10 columns of features. They are id, gender,master-Category, sub-Category, article type, base Color, season, year, usage and product display name. The images are very clean and have a nice and clear background. Most of the images are 2400x1800 in size. Some of them are 1600x1200. The link for the data is https://www.kaggle.com/datasets/paramaggarwal/fashion-product-images-dataset .
Model was trained in 2 ways for 2 different purposes:

3.1.1 Inventory Items(known) Recommendation:

In this case, I train the model for the image feature extraction to get embedding of them, so the whole data-set was used to train the model with no different train, test, and validation split of the data set.

3.1.2 Unknown(new/no filter) Item Recommendation:

In this case, I train Keras pre-trained model ResNet50 with total image = 7905 and total class=15. We split train, test, and validation as test=0.2%, validation = 2% of (total-test), and train = total-test-validation. Here we freeze 170 layers of ResNet50.

3.2 Proposed Approach

Briefly describe the approach. Emphasize why I selected this approach among different competing alternatives. I make two types of recommendations in my system:
(1)For Inventory Images
(2) For New(outside image uploaded by user) Images.
In my project, I intended to use the keras pre-trained model for feature extraction and used the cosine similarity method for calculating the similarity of the images. I stored the features in a variable.

3.2.1 For Inventory Images:

I calculate the cosine similarity of the embedding of the images' features to get the images to get the top most similar images in order.

3.2.2 For Unknown Images:

First, I classify the image to get it's subcategory, then I use another classifier to get its type(gender and article type) for filtering. Then I use the keras model to get the feature embedding and apply cosine similarity to get the topmost images from the filtered ones. I mainly do 2 part classification to reduce the inference time. At first, when I was not using any classifiers, it took about 43 seconds. Then I added the subcategory classifier to reduce the number of images. Then the system took 24 seconds for the results. Then I added a 2ND classifier to reduce the number of images further. Then it was giving results within seconds. For inventory images, I get the recommendations within seconds. Additional filters can also be provided for better recommendations.

3.3 Reasons to choose the Approach:

1. I chose Keras pre-trained model as I have only 44k data. Transfer learning helps get the expected accuracy level get the expected level of accuracy with a low amount of data.
2. Pre-trained model performs better in feature extraction and embedding.
3. I used two classifiers to reduce the time complexity of similarity calculation to make the system real-time.

3.4 Method:

3.4.1 System Overview:

How different components interact.I add some diagrams so it is easier to get an overall picture.
In my project, I have two types of recommendations. They are for inventory images and non inventory inventory images. First, I used the Resnet50 model for feature extraction. Then for inventory images, I calculate the similarity with cosine similarity with all the images and show the best results. I also add some filters. If filters are added then the number of images is reduced. For unknown images, I add two classifiers for subcategories and derived type features. And then I extract the image features using Resnet50 Models. Then I calculate the similarity of the images using cosine similarity and show the results. The following images briefly show how the different components interact with each other.

3.4.2 Neural Network Architecture/Machine Learning Approach:

For inventory image recommendations, I got the embedding of all images from the features that were extracted with the Resnet50 model. The feature extraction

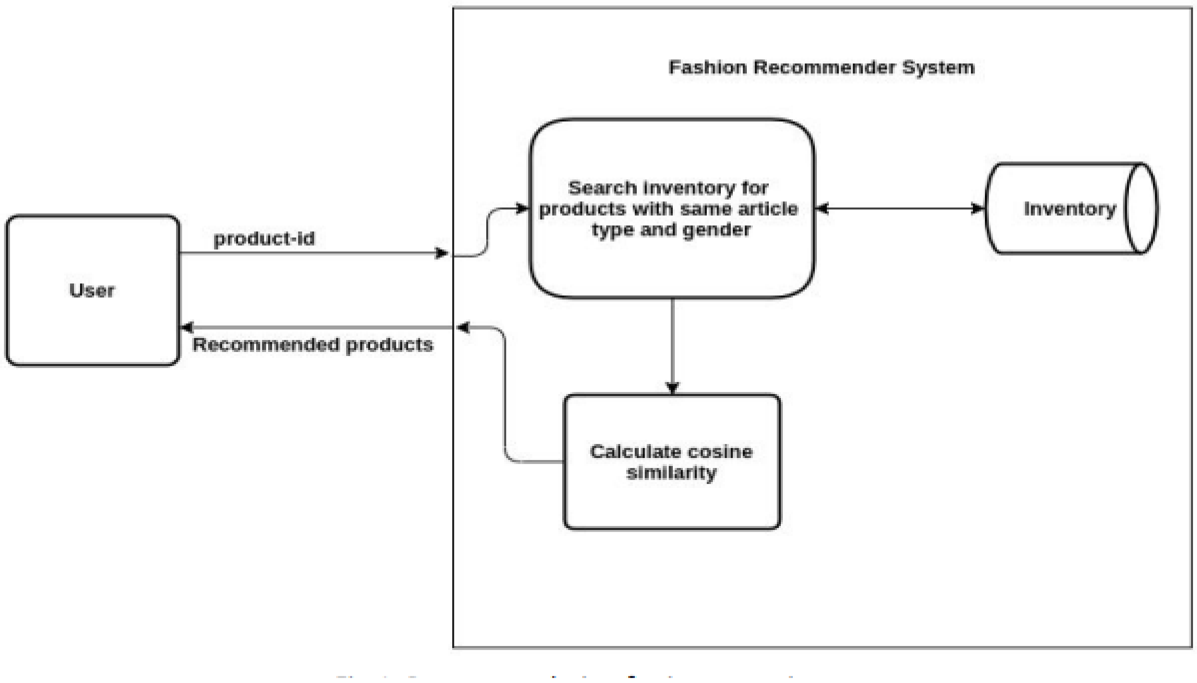

Fashion Recommender System
Search inventory for products with same article type and gender
Inventory
User
product-id
Recommended products
Calculate cosine similarity

Figure 3.1: Recommendation for Inventory Items

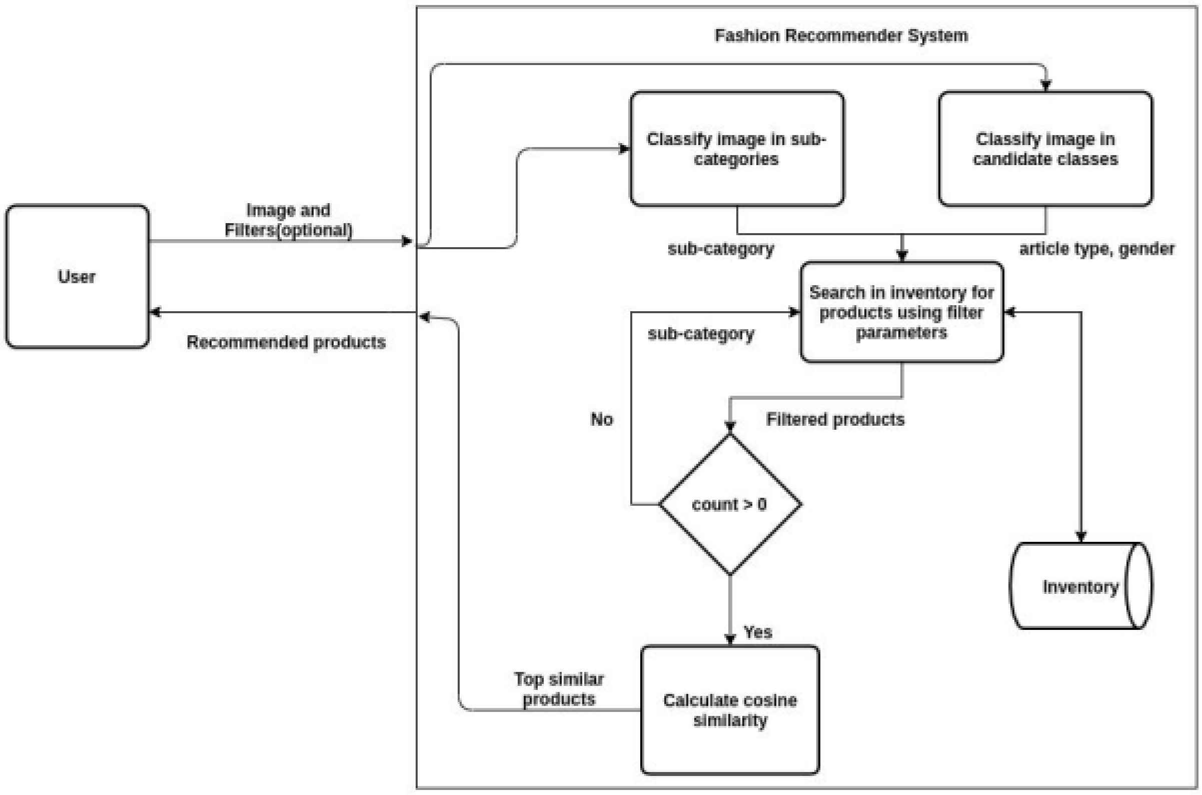

Fashion Recommender System
Classify image in sub-categories
Classify image in candidate classes
User
Image and Filters(optional)
Recommended products
sub-category
article type, gender
Search in inventory for products using filter parameters
sub-category
No
Filtered products
count > 0
Inventory
Yes
Top similar products
Calculate cosine similarity

Figure 3.2: Recommendations for Unknown Items (search by image)

method is explained in the Feature Extraction section. I then stored the embedding. Users can choose features to filter out the results.Though this is optional. If a user chooses to add filters then the system calculates the cosine similarity of the embedding of all the filtered images to get the top similar images as recommendations.If a user chooses not to add filters then the system calculates the cosine similarity of the embedding among 44419 images to get the top similar images as recommendations. For example, suppose a user chooses Blue as the base Color for filtering and there are 5000 image of base Color Blue and chooses an item, then the system will calculate the cosine similarity of the items embedding with the 5000 images embedding with the base Color Blue and gives the top similar images as recommendations. For unknown images, I first classify the image by its subcategory, then by its derived Type which is a combined feature added by us and then extracted the features and used those features for embedding and calculate the cosine similarity with sub Category and derived Type as default filters. Other filters may be added if the user chooses some. The filters work as it does for the inventory images. Our derived Type feature is composed of gender and article Type. An example for the derived Type may be "Men T-shirt". Suppose a user uploads an image and it's subcategory is top wear. Let's assume that the Top wear subcategory has 12000 images. Then the derived type classifier is run on the 12000 images. Let's also assume that the image is classified as a Men T-shirt and this feature has 800 images. Then the system will calculate the cosine similarity of the image's 5 embedding that was found from its feature with the 800 image's embedding and gives the top similar images as recommendations. It will be explained in detail in the next section. From the features mentioned in the Data set, i use subcategory as a feature the for one classification and also use gender and article Type combined as derived Type as a feature for the 2nd classification. For subcategory classification, I used the keras classifier model, Resnet50. For derived Type classifiers, we calculate the similarity with a "candidate image" of each derived Type. There are 280 unique derived Types. So we just need to compare the image with 280 images. For this candidate image selection at first we extract all images features for embedding. Then we calculate cosine similarity with all the images. After that we calculate summation of all similarity scores for every image in every derived Type group. Then for the 280 groups, we select an image of every group with the highest score as that group's candidate. When we classify an unknown item, for article type model to classify it against subcategory. Then we classify the item further with the candidate classifier. Here cosine similarity of the unknown item is calculated against all candidate items of that subcategory. From the highest similarity score, we find the closest derived Type for the unknown item and this is the derived Type of the known item detected by the candidate classifier. Afterward, I calculate the similarity of the unknown items against that identified derived type group and recommend the most similar items for the unknown item.

3.4.3 Feature Extraction:

For feature extraction, I used the Resnet50 pre-trained model on Image Net. I just used the layers that are used for feature extraction. We froze all layers of the model and used GlobalMaxPooling for feature extraction. Then get embedding of all images from these extracted features from the model.

3.4.4 Augmentation:

For Inventory image recommendations, I did not use any image augmentation technique as it's not needed here. For Unknown(without any filtering parameter) image recommendations, I used model-level default augmentation which is performed online(on the fly) during model training for subcategory classification.

3.4.5 Prepossessing:

The fashion product image data set consists of 44419 images with metadata. These metadata describe product id, product type, gender, etc. I delete incomplete data and derive some new features(columns) for computational, performance, and accuracy for the predicted similar products for a recommendation. I prepossess images online while feeding those to Keras model that is done by the model and resize images to fit the model.

Chapter 4

Experiment:

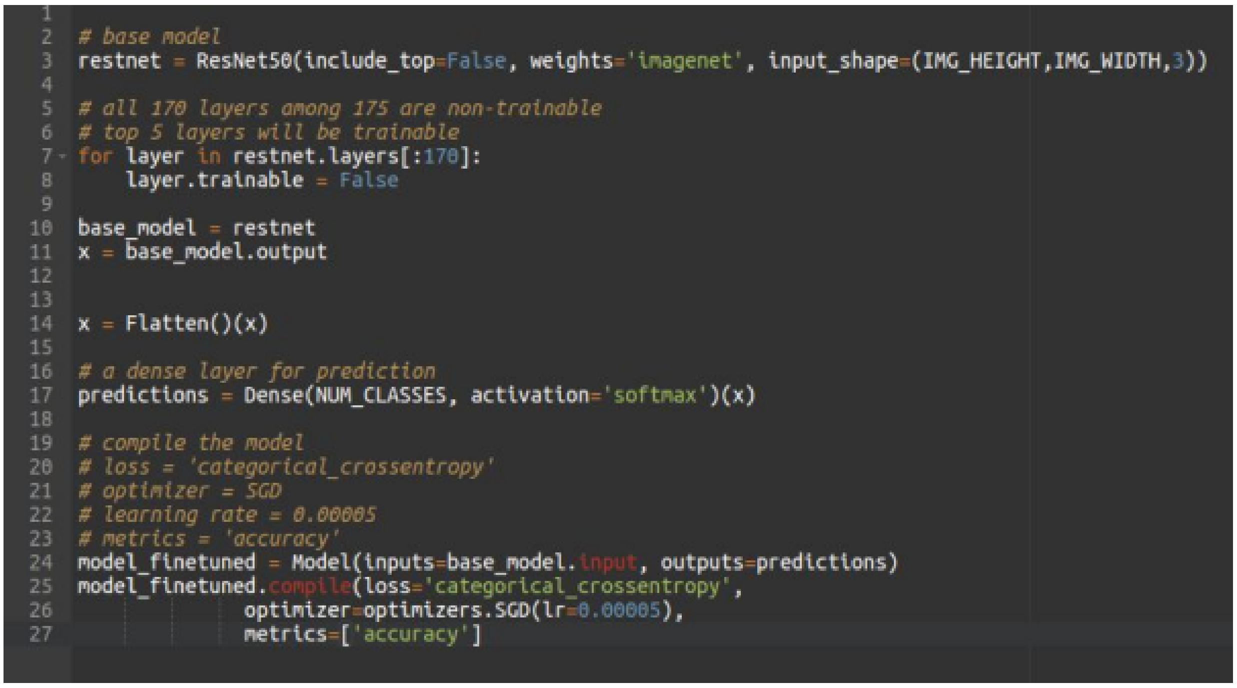
In this section, I provide details of how experiments were conducted, the results obtained and their implications. The following subsections must be included in all reports.

4.1 Training Procedure:

For feature extraction model selection, I experimented with images of article type with 500 or more images. The number of images we used was 34719. I used slightly smaller data than the original because when I took all the images memory ran out. I used pairwise cosine similarity for all the images. This gave us a 34719 X 34719 similarity matrix. For a single image, I checked the top 10 most similar products and compared their article type and gender. If it matches the current product's article type and gender then it's a good recommendation. I did experiments on Inception, Xception, NasnetLarge, InceptionResnet, Resnet50, Resnet50V2, Resnet152 as they give 7 exceptionally good results for image similarity detection. I took the top 10 recommendations with error thresholds of 0, 0.1, 0.2. For selecting a proper pre-trained keras model for image classification we also had done some experiments. I compared the performance of several models including ResNet50 and Xception. I also ran the same model with different hyperparameters such as learning rate, batch size, data augmentation, etc. Also took different amounts of data to train, and changed the class and number of classes. All those experiments showed that our selected model ResNet50 gives the best result on classifying the images into sub-categories with training and validation accuracy of 90% and 85% respectively. The description of the trained model and its hyper-parameter are pictured below

4.2 Evaluation Criteria:

In a recommendation system, no wrong prediction should be allowed even if some good recommendations are omitted. So I needed to be precise with the recommendation. I also measured the performance of the feature extractor model after calculating the similarity between products. Then I took the top 10 similar products into consideration and compared their article type and gender. If they are equal I counted it as a true positive. I calculated accuracy with this value. The formula is
Accuracy = Count(True Positive) / 10



```
# base model
restnet = ResNet50(include_top=False, weights='imagenet', input_shape=(IMG_HEIGHT,IMG_WIDTH,3))

# all 170 layers among 175 are non-trainable
# top 5 layers will be trainable
for layer in restnet.layers[:170]:
    layer.trainable = False

base_model = restnet
x = base_model.output


x = Flatten()(x)

# a dense layer for prediction
predictions = Dense(NUM_CLASSES, activation='softmax')(x)

# compile the model
# loss = 'categorical_crossentropy'
# optimizer = SGD
# learning rate = 0.00005
# metrics = 'accuracy'
model_finetuned = Model(inputs=base_model.input, outputs=predictions)
model_finetuned.compile(loss='categorical_crossentropy',
                        optimizer=optimizers.SGD(lr=0.00005),
                        metrics=['accuracy']
```

Figure 4.1: ResNet50 customization for sub-category classification

So the error of the model for the current product is-

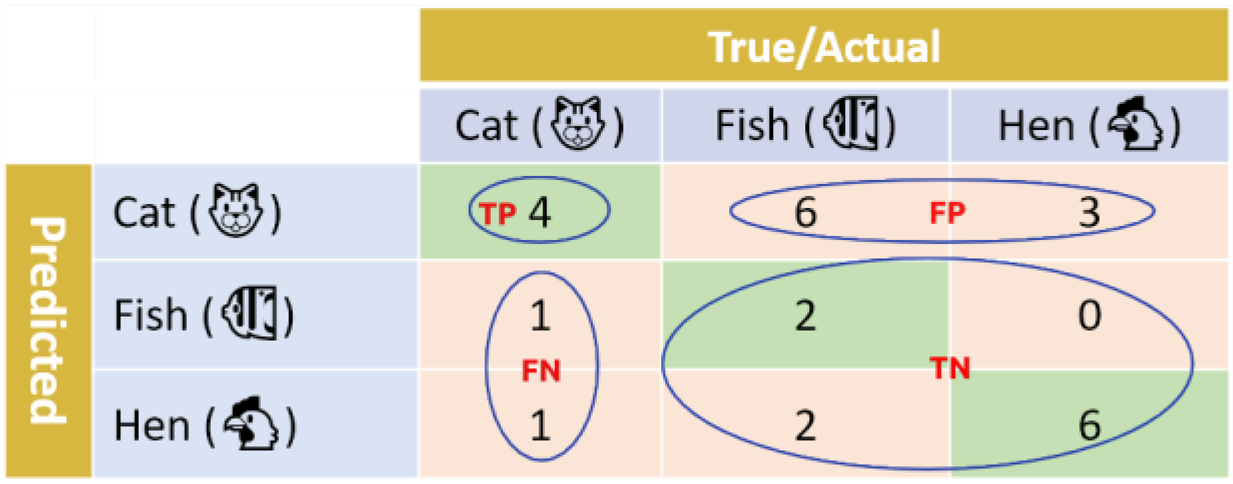
Error = 1 - Accuracy

If the error of the current model for that particular product is less or equal to the threshold then count it as TRUE POSITIVE. Following this step for every other selected(34719) product I calculated TRUE POSITIVE. Then calculated the model performance as below
Performance = Count(True Positive) / 34719
I mainly focused on the article type and gender as our evaluating features. If any of the recommendations on gender or article type is wrong then that recommendation is wrong. For the sub-category classifier model, I considered the train, validation, and test accuracy. Train and validation accuracy(as well as a loss) are calculated when the model is trained. After running the trained model on test data, I calculated train accuracy manually. For all three cases, the formula is same,
Accuracy = Count(True Prediction) / Total sample .

4.2.1 Model Accuracy Metric:

For calculating accuracy, I used the accuracy algorithm that was built into the model. For accuracy the metric is:
Accuracy = (TP+TN)/(TP+TN+FP+FN).
1.True Positive, or TP, are cases with positive labels which have been correctly classified as positive.
2. True Negative, or TN, are cases with negative labels which have been correctly classified as negative.
3. False Positive, or FP, are cases with negative labels which have been incorrectly classified as positive.

4. False Negative, or FN, are cases with positive labels which have been incorrectly classified as negative. In my case,
5. If a T-shirt is correctly classified as a T-shirt, then it is true positive or TP.
6. If a Pant is classified as T-shirt, then for a T-shirt it is false positive or FP.
7. If a Pant is classified as anything other than a T-shirt, then for a T-shirt it is true negative or TN.
8. If a T-shirt is classified as anything other than a T-shirt, then for T-shirt it is false negative or FN. So basically the accuracy we calculated is, the percentage of correctly predicted classes. Mathematically,
Accuracy = Correct Prediction / Total Samples.



		True/Actual		
		Cat	Fish	Hen
Predicted	Cat	TP 4	6	3
	Fish	1	2	0
	Hen	1	2	6

Figure 4.2: Model Accuracy Metric

4.3 Model Comparison:

4.3.1 Data Description:

The data is composed of 44419 images with sizes varying from 2400X1800 and 1600X1200. As the models work with images with varying sizes, I chose a standard size for resizing that can be used by all the models. The standard size I used is 600X450 as it is of the same ratio as the original except for NasnetLarge which only takes images of size 331X331. For time issues I am not using all the data. I am using images with article types that have 500 or more images for all the models except NasnetLarge.For NasnetLarge, i am using images with article types that have 650 or more for memory issues in Kaggle. With that filtering I will use 34719 images for all other models and for NasnetLarge I will use 31284 images.

4.3.2 Accuracy Metric:

I can use 3 accuracy metrics:
1. Accuracy
2. Precision
3. Recall

As this is a recommendation system, so I can not have wrong predictions even if some good recommendations are omitted. So i need to be precise with the recommendation. So I use precision.

4.3.3 Features Used For Calculating:

I am mainly focusing on the article type and gender as it would be embarrassing if I recommended a pair of heels when a man's shoe is given as a reference item or if I recommended a shirt when a watch is given. So I am emphasizing article type and secondly gender.

4.3.4 Comparison:

Table 4.1: ***Xception TopN 10***

Feature	***Tolerance***	***Performance(%)***
article Type	0.2	80.96
article Type	0.1	75.56
article Type	0	66.38
gender	0.2	75.98
gender	0.1	68.27
gender	0	56.40
gender and article Type	0.2	63.16
gender and article Type	0.1	54.46
gender and article Type	0	42.41

Table 4.2: ***InceptionV3 TopN 10***

Feature	***Tolerance***	***Performance(%)***
article Type	0.2	78.15
article Type	0.1	71.85
article Type	0	61.81
gender	0.2	75.04
gender	0.1	66.62
gender	0	53.77
gender and article Type	0.2	60.23
gender and article Type	0.1	51.04
gender and article Type	0	38.41

4.3.5 Decision:

As we can clearly see from the above plots, if I use the Resnet152 model for feature extraction, it gives the best results for all the important features chosen but the size of this model is 232MB. So it will take up more space in the memory. That will be a problem as I need to keep 44k embedding for getting a faster recommendation for unknown images so memory issues will arise. But we can also see from the above

Table 4.3: ***ResNet50 TopN 10***

Feature	***Tolerance***	***Performance(%)***
article Type	0.2	82.21
article Type	0.1	76.73
article Type	0	67.84
gender	0.2	81.02
gender	0.1	74.24
gender	0	63.67
gender and article Type	0.2	67.50
gender and article Type	0.1	58.92
gender and article Type	0	47.03

Table 4.4: ***ResNet152 TopN 10***

Feature	***Tolerance***	***Performance(%)***
article Type	0.2	82.99
article Type	0.1	77.44
article Type	0	68.66
gender	0.2	81.39
gender	0.1	74.75
gender	0	63.94
gender and article Type	0.2	68.48
gender and article Type	0.1	59.98
gender and article Type	0	47.80

Table 4.5: ***Resnet50V2 TopN 10***

Feature	***Tolerance***	***Performance(%)***
article Type	0.2	81.97
article Type	0.1	76.17
article Type	0	66.78
gender	0.2	79.32
gender	0.1	71.74
gender	0	60.26
gender and article Type	0.2	65.50
gender and article Type	0.1	56.42
gender and article Type	0	44

Table 4.6: ***InceptionResnet TopN 10***

Feature	***Tolerance***	***Performance(%)***
article Type	0.2	78.27
article Type	0.1	71.93
article Type	0	62.78
gender	0.2	73.73
gender	0.1	65.49
gender	0	53.68
gender and article Type	0.2	59.01
gender and article Type	0.1	49.99
gender and article Type	0	38.61

Table 4.7: ***NasnetLarge TopN 10***

Feature	***Tolerance***	***Performance(%)***
article Type	0.2	82.71
article Type	0.1	77.40
article Type	0	68.48
gender	0.2	75.81
gender	0.1	66.86
gender	0	54.14
gender and article Type	0.2	64.91
gender and article Type	0.1	56.12
gender and article Type	0	43.75

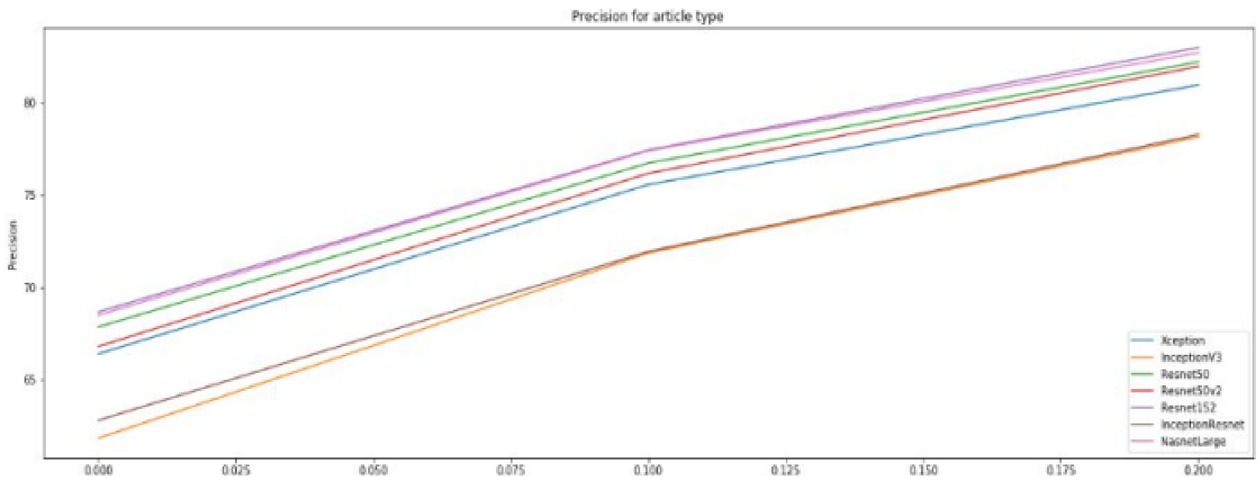


Figure 4.3: Precision-Tolerance For Article-Type

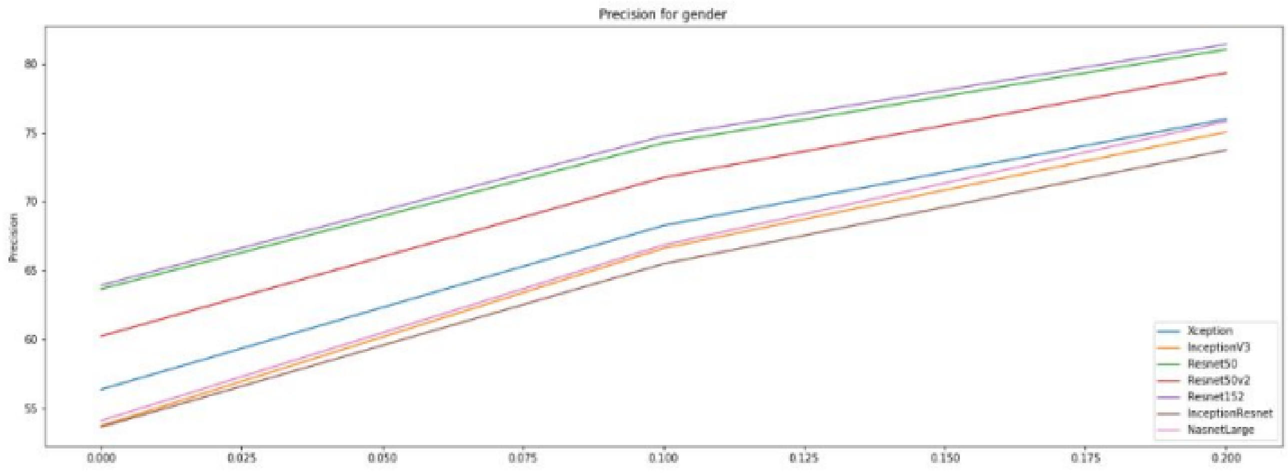


Figure 4.4: Precision-Tolerance For Gender

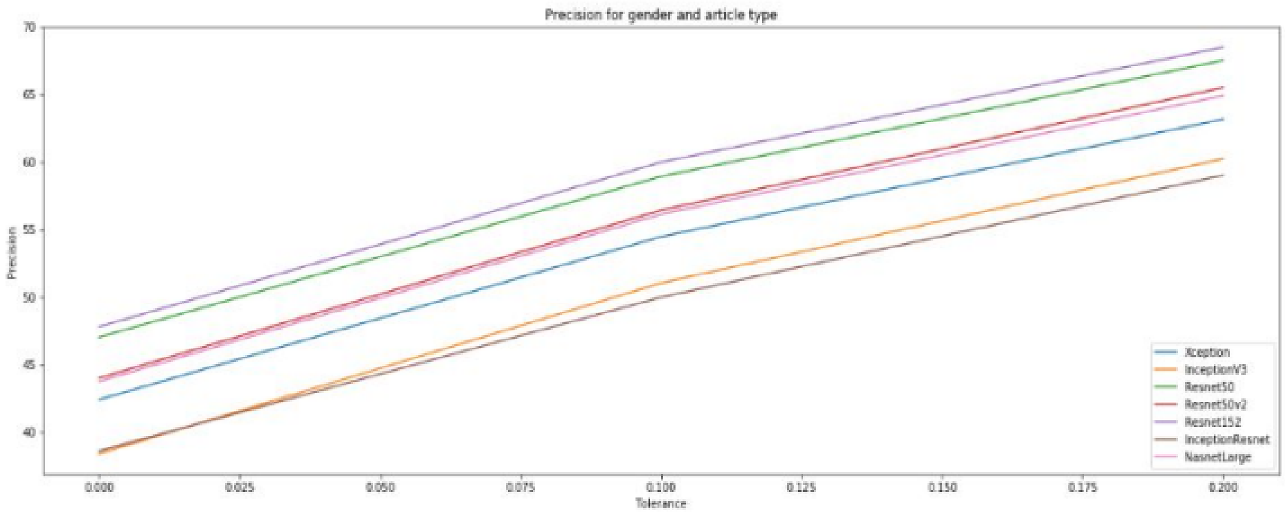


Figure 4.5: Precision-Tolerance For Both Article-Type And Gender

plots that Resnet50 is a close second but its size is just 98MB which is almost 125% lesser than Resnet50. So, I am using Resnet50 for feature extraction.

Chapter 5

Result And Discussion

5.1 Model Exploration 1

5.1.1 ResNet50

Data Augmentation:
rotation range=30
zoom range=0.15
width shift range=0.2
height shift range=0.2
shear range=0.15
horizontal flip=True
fill mode="nearest"

Table 5.1: ***Model Description for ResNet50***

Result No	***Trainable Layers***	***Dropout***	***Dense***	***Dropout***	***Dense(Prediction)***
1	4 upper layers	0.2	512	0.2	512
2	4 upper layers	0.2	512	0.2	512
3	4 upper layers	0.2	512	0.2	512
4	all layers=false	0.2	512	0.2	512
5	4 upper layers	0.2	512	0.2	512
6	4 upper layers	0.2	512	0.2	512
7	4 upper layers	0.2	512	0.2	512

Table 5.2: ***ResNet50***

Result No	***Class Column***	***Classes***	***Batch size***	***Learning Rate***
1	master category	4	32	0.00001
2	master category	4	10	0.00001
3	master category	4	5	0.00001
4	master category	4	10	0.00001
5	master category	4	10	0.00001
6	master category	4	10	0.00001
7	gender	5	10	0.00001
7	baseColour	11	10	0.00001

Table 5.3: ***Accuracy And Results for ResNet50***

No	***Train Accuracy***	***Train Loss***	***V Accuracy***	***V Loss(avg)***	***Test Accuracy***
1	71%	0.7822	72%	1.10	71%
2	76%	0.6653	79%	0.69	78.91%
3	75%	0.65	80%	0.78	78.91%
4	75%	0.69	80%	0.36	79.8%
5	81%	0.52	77%	0.37	76%
6	77.5%	0.62	79.9%	0.34	79%
7	46.7%	1.35	42%	1.3	41.4%

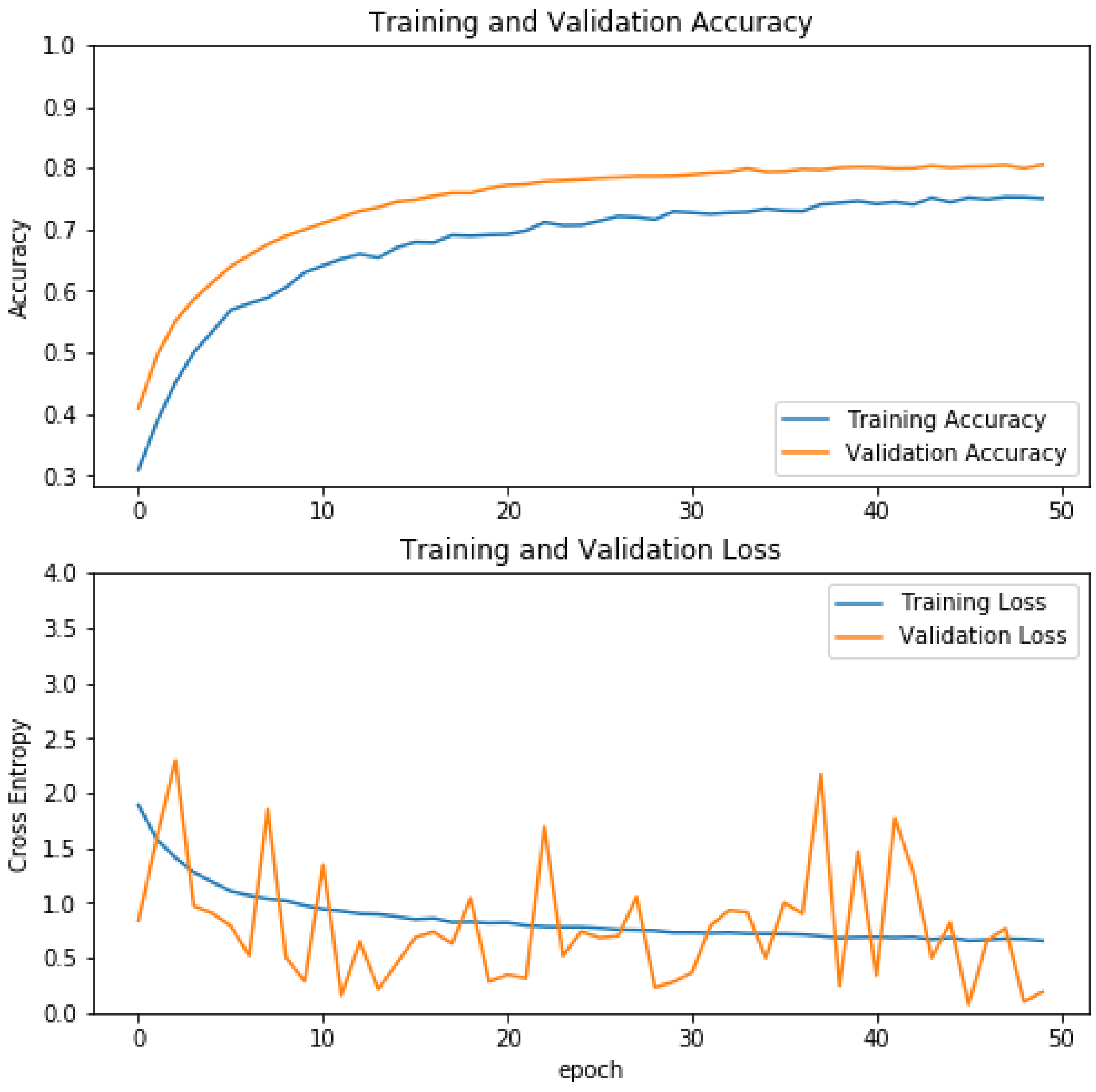


Figure 5.1: ResNet50 model training and validation history for batch size-5

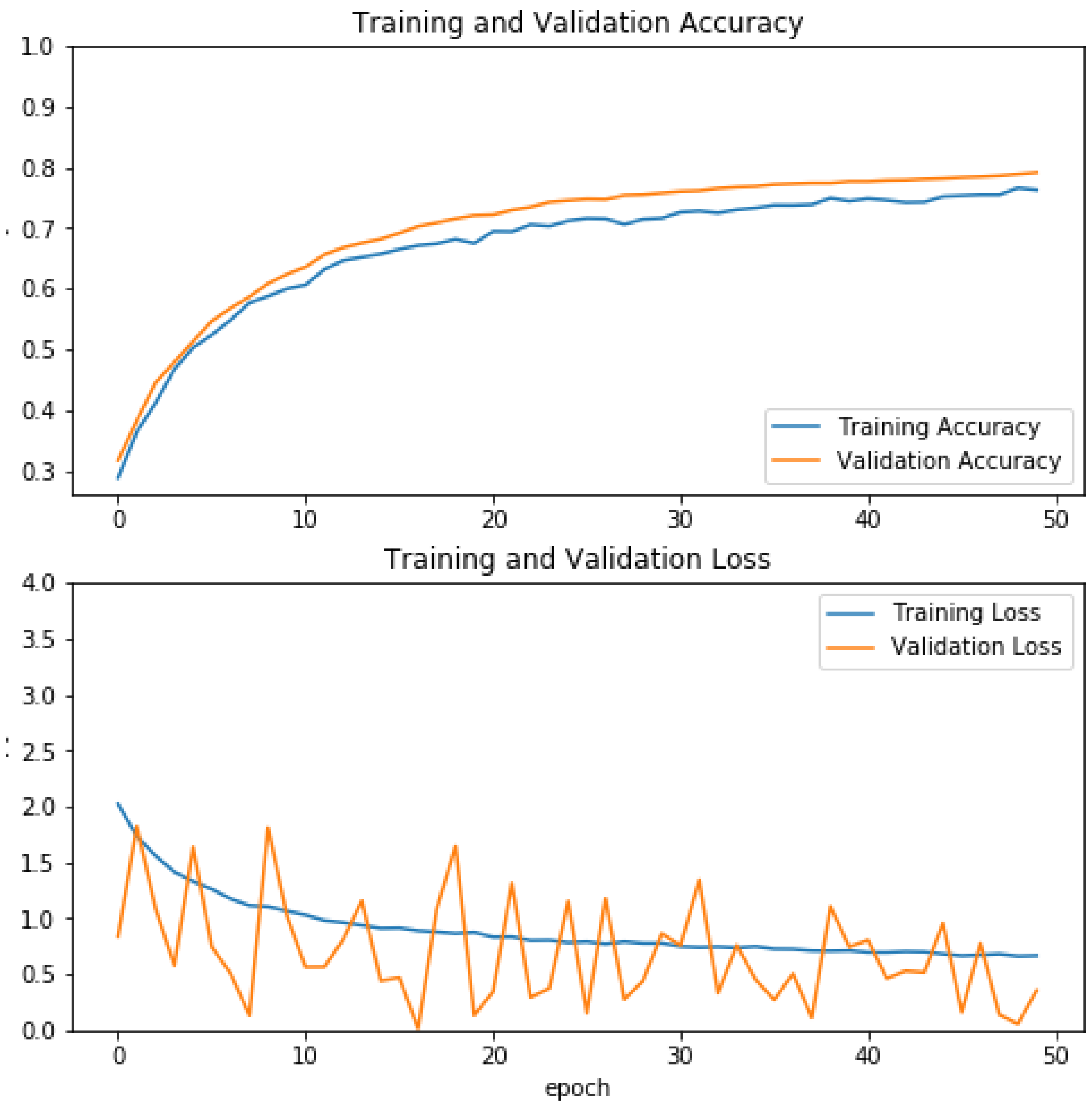


Figure 5.2: ResNet50 model training and validation history for batch size-10

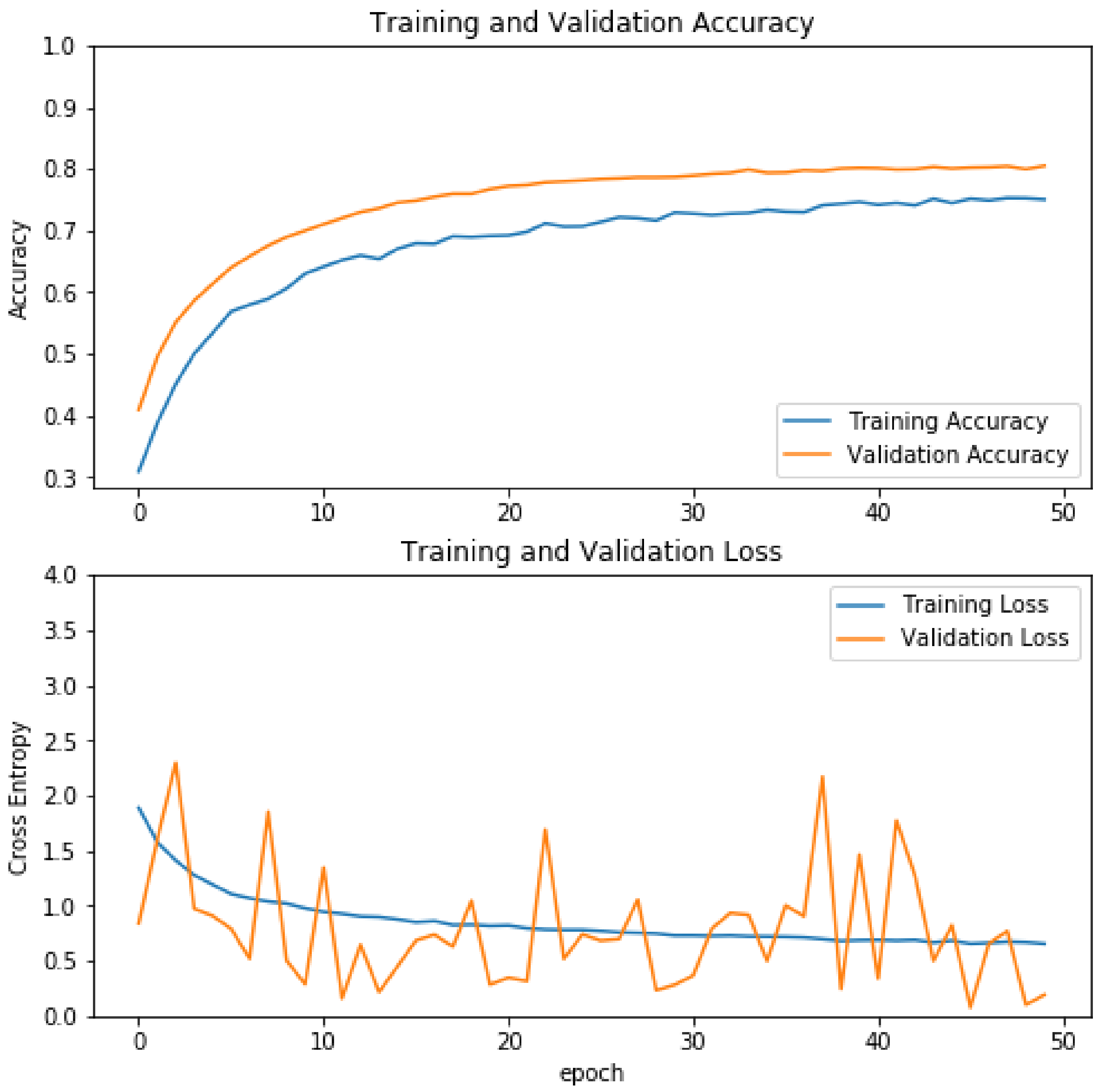


Figure 5.3: ResNet50 model training and validation history for batch size-5

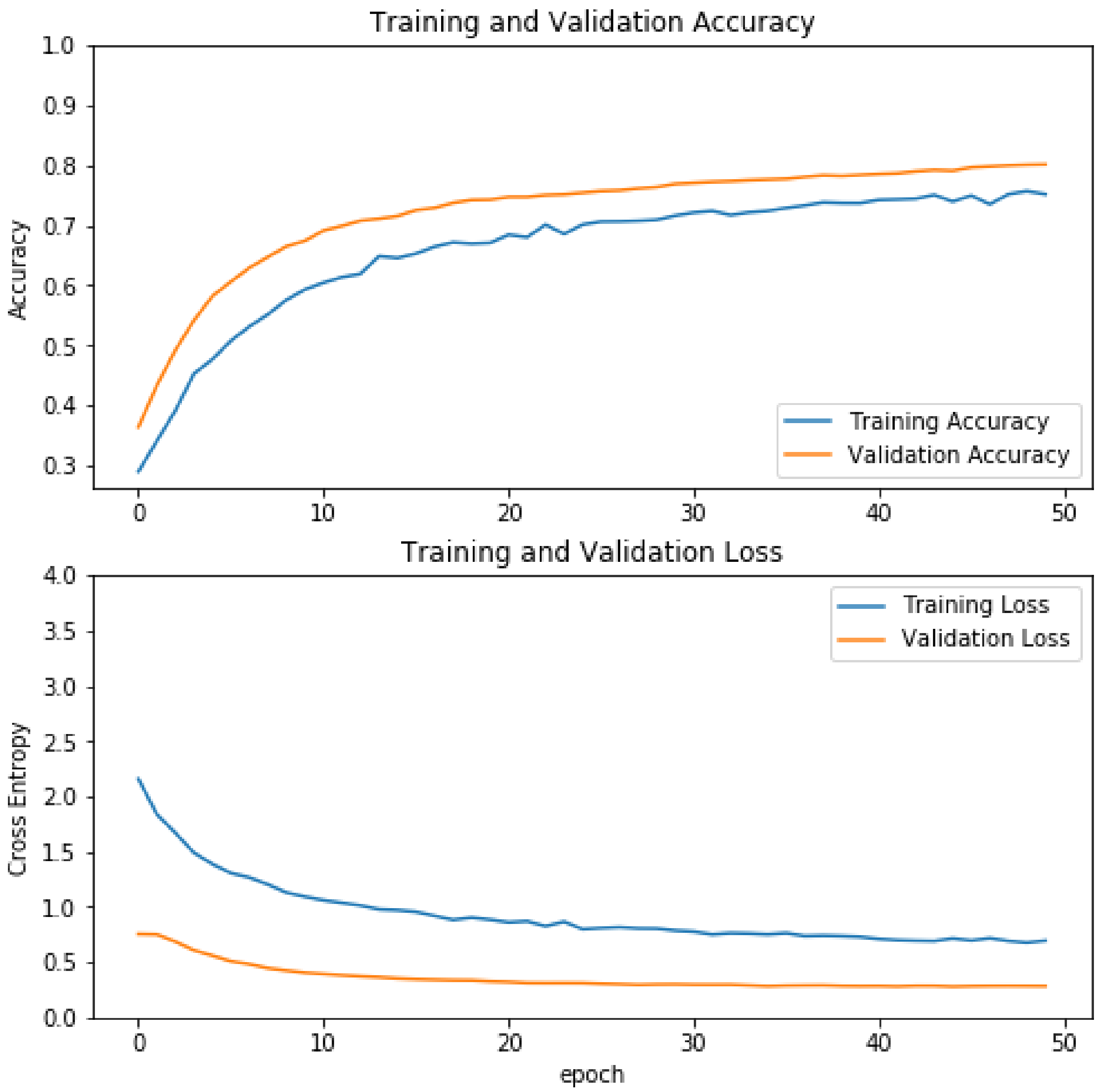


Figure 5.4: ResNet50 model training and validation history for batch size-10 and shuffle=False for validation data

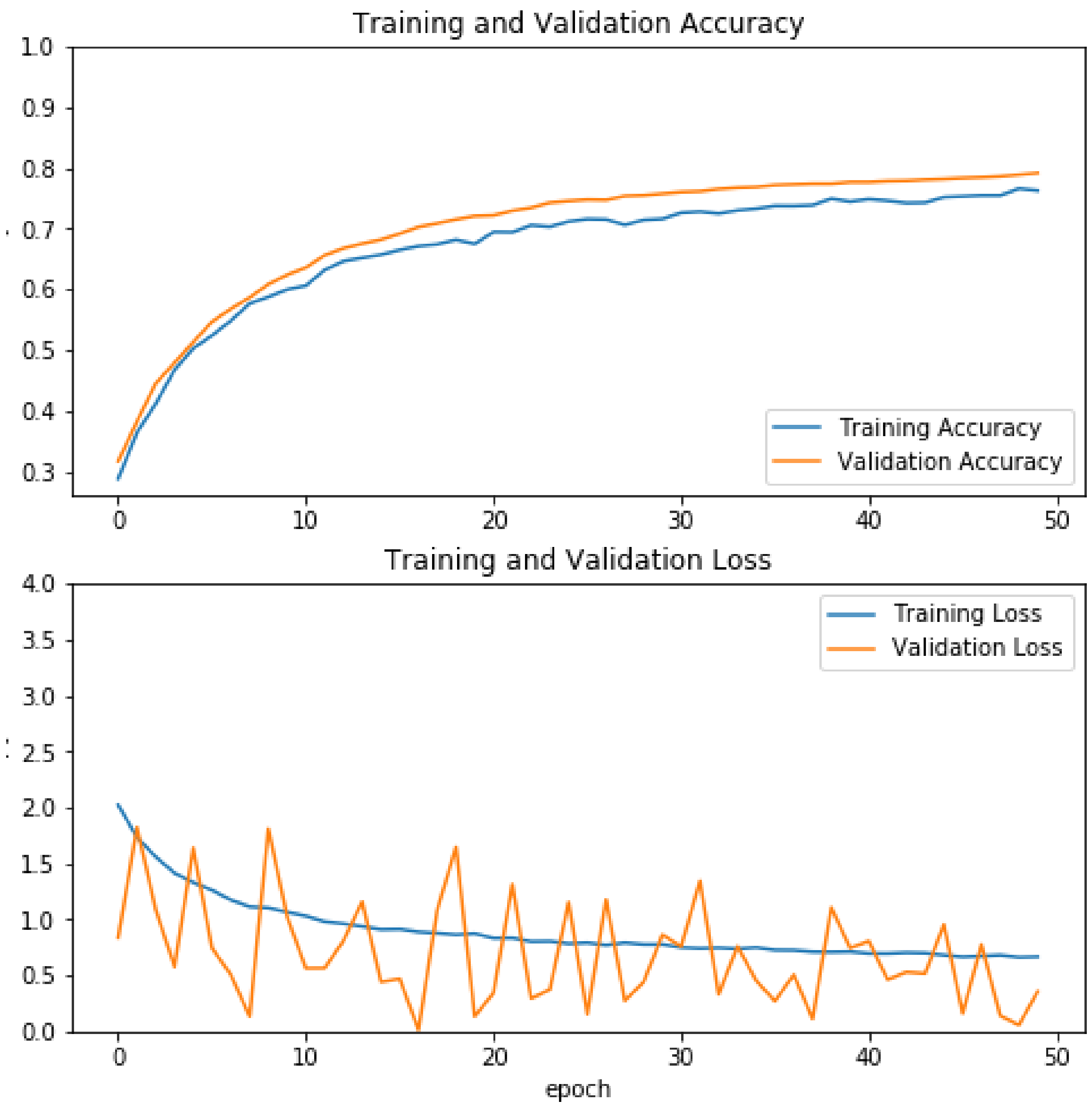


Figure 5.5: ResNet50 model training and validation history for batch size-10 and removing dropouts

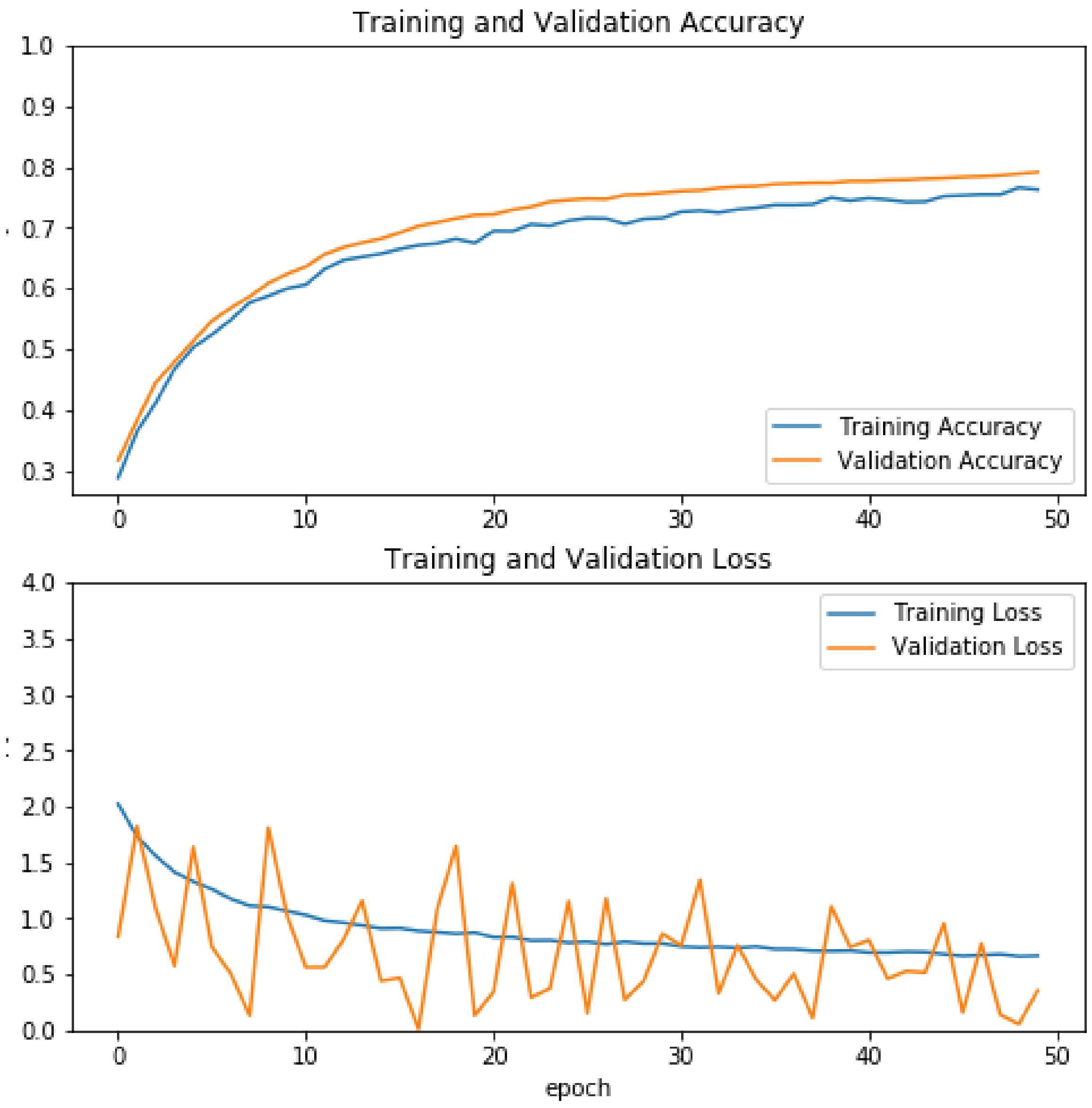


Figure 5.6: ResNet50 model training and validation history for batch size-10, adding a dense layer and a dropout after it

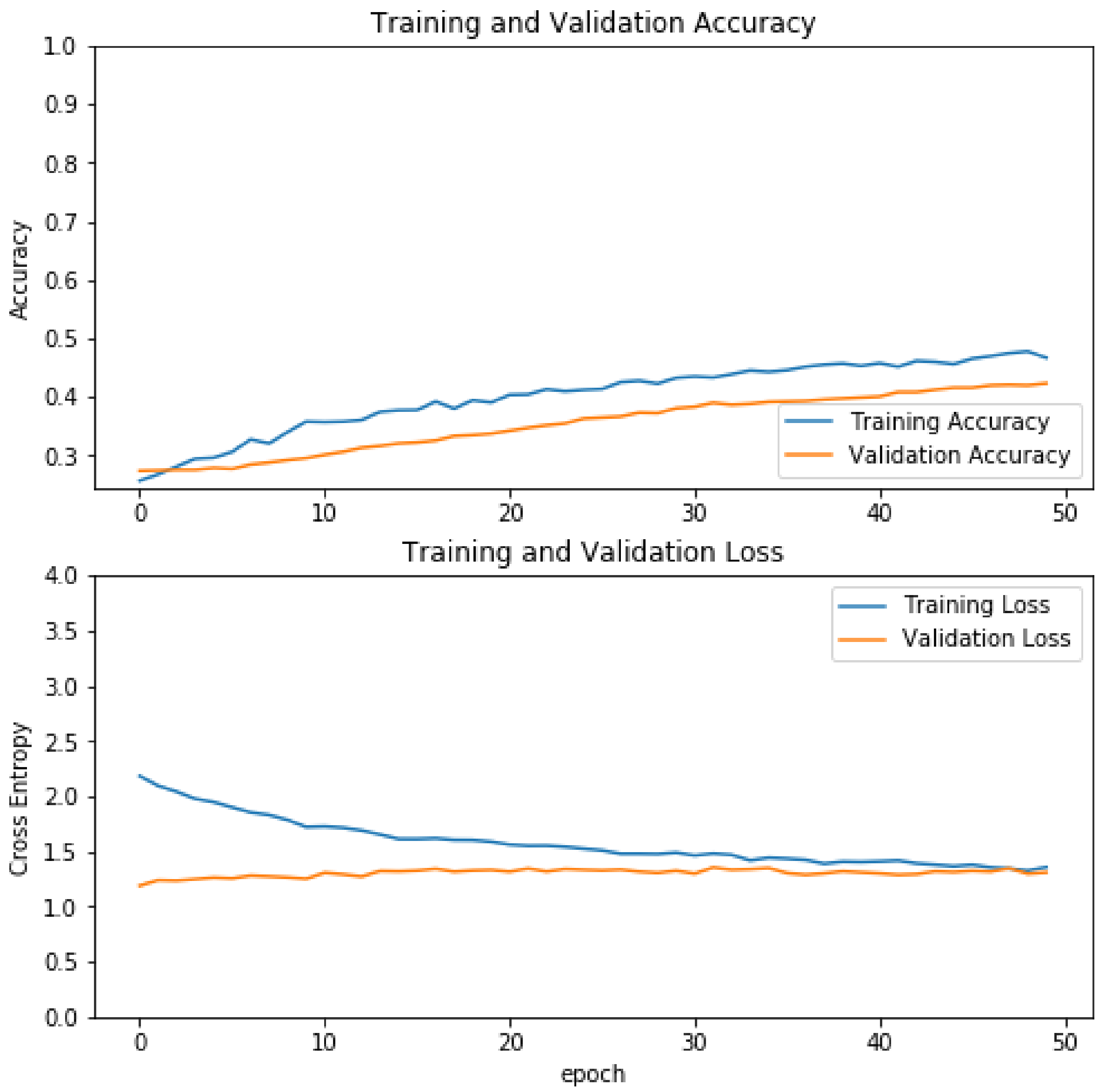


Figure 5.7: ResNet50 model training and validation history for gender classification

5.1.2 Xception:

Data Augmentation:
rotation range=30
zoom range=0.15
width shift range=0.2
height shift range=0.2
shear range=0.15
horizontal flip=True
fill mode=”nearest”

Table 5.4: ***Model Description for Xception***

Result No	***Trainable Layers***	***Dropout***	***Dense***	***Dropout***	***Dense(Prediction)***
8	all layers=false	0.2	512	0.2	512
9	all layers=false	0.2	512	0.2	512
10	all layers=false	0.2	512	0.2	512
11	all layers=false	0.2	512	0.2	512
12	4 upper layers	0.2	512	0.2	512
13	4 upper layers	0.2	512	0.2	512
14	4 upper layers	0.2	512	0.2	512

Table 5.5: ***Xception***

Result No	***Class Column***	***Classes***	***Batch size***	***Learning Rate***	***Optimizer***
8	subcategory	5	25	0.00001	SGD
9	master category	6	25	0.00001	SGD
10	subcategory	5	25	0.00001	RMSProp
11	subcategory	6	25	0.00001	RMSProp
12	subcategory	4	10	0.00001	SGD
13	master category	4	10	0.00001	SGD
14	gender	5	10	0.00001	RMSProp
14	baseColour	11	10	0.00001	RMSProp

Table 5.6: ***Accuracy And Results for Xception***

No	***Train Accuracy***	***Train Loss***	***V Accuracy***	***V Loss(avg)***	***Test Accuracy***
8	28.58%	1.55	18.41%	20.56	17.76%
9	45.28%	1.59	17.32%	27.76	16.56%
10	86.39%	0.476	19.39%	151.188	17.36%
11	83.54%	0.56	18.02%	106.4	16.9%
12	34.86%	1.52	20.04%	25.57	21.36%
13	77.5%	0.62	79.9%	0.34	79%
14	46.7%	1.35	42%	1.3	41.4%

Notes:
1. If dropout is increased, training accuracy degrades. It will help when the model

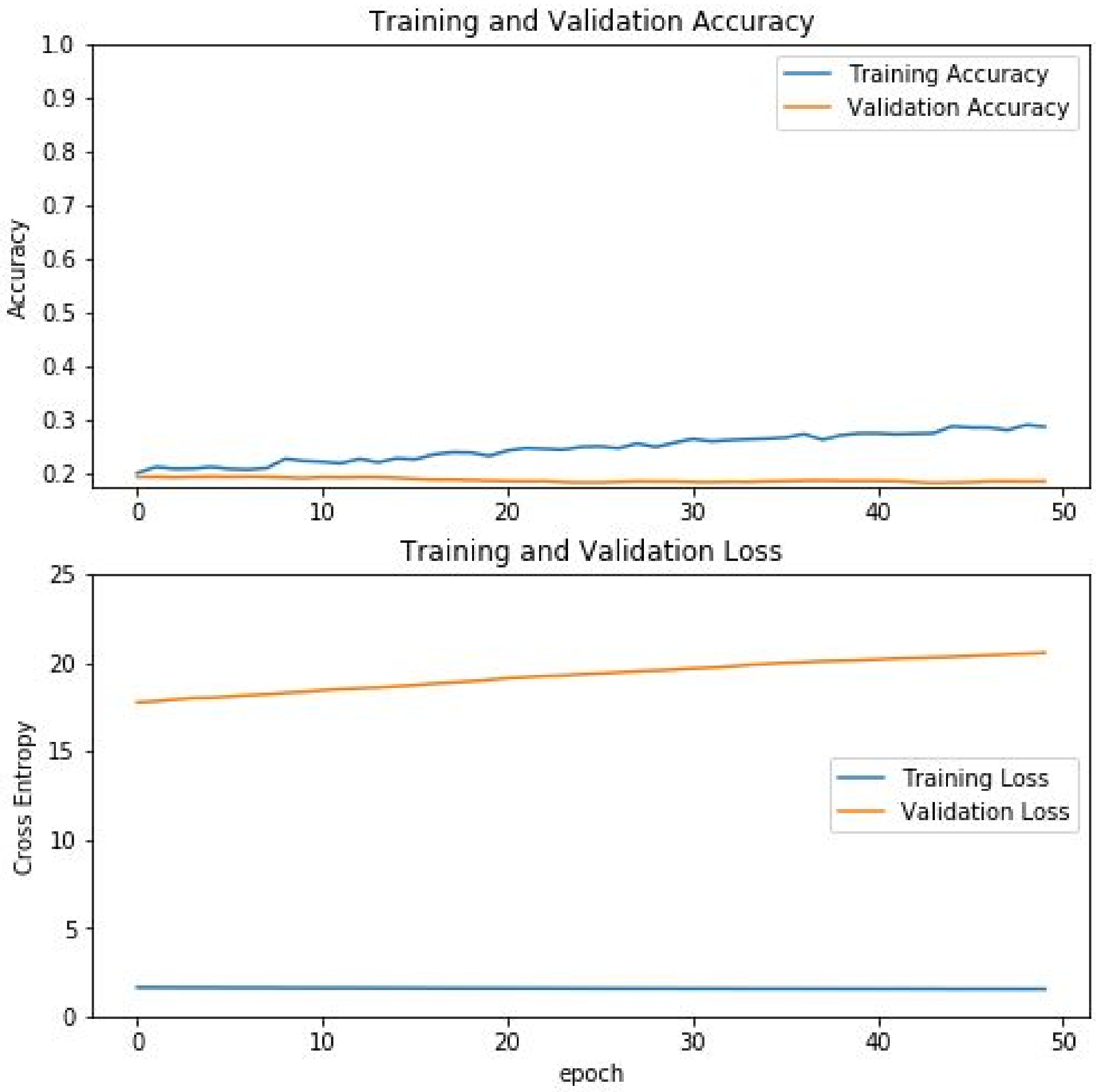


Figure 5.8: Xception model training and validation history for batch size-32

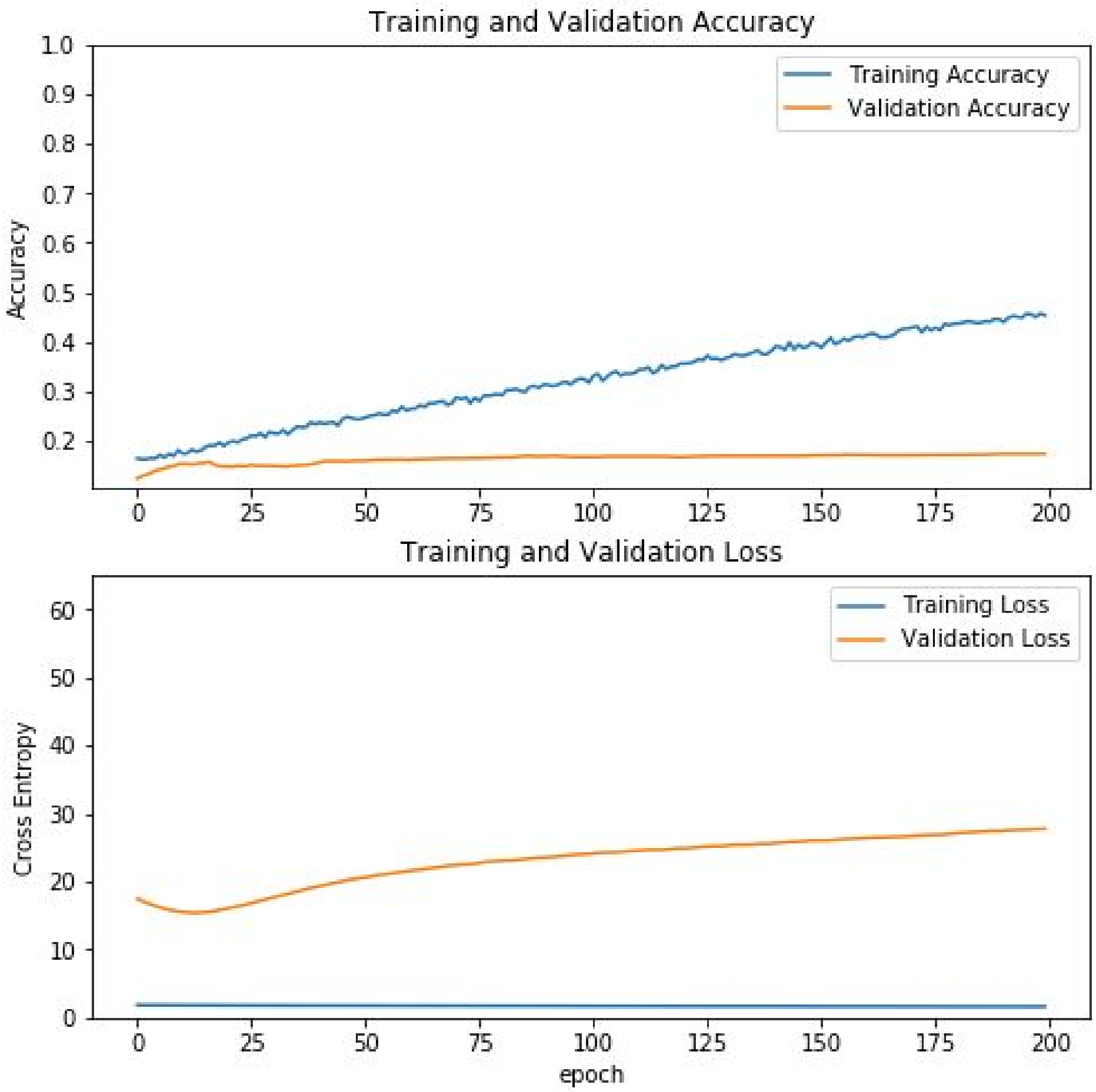


Figure 5.9: Xception model training and validation history for batch size-10

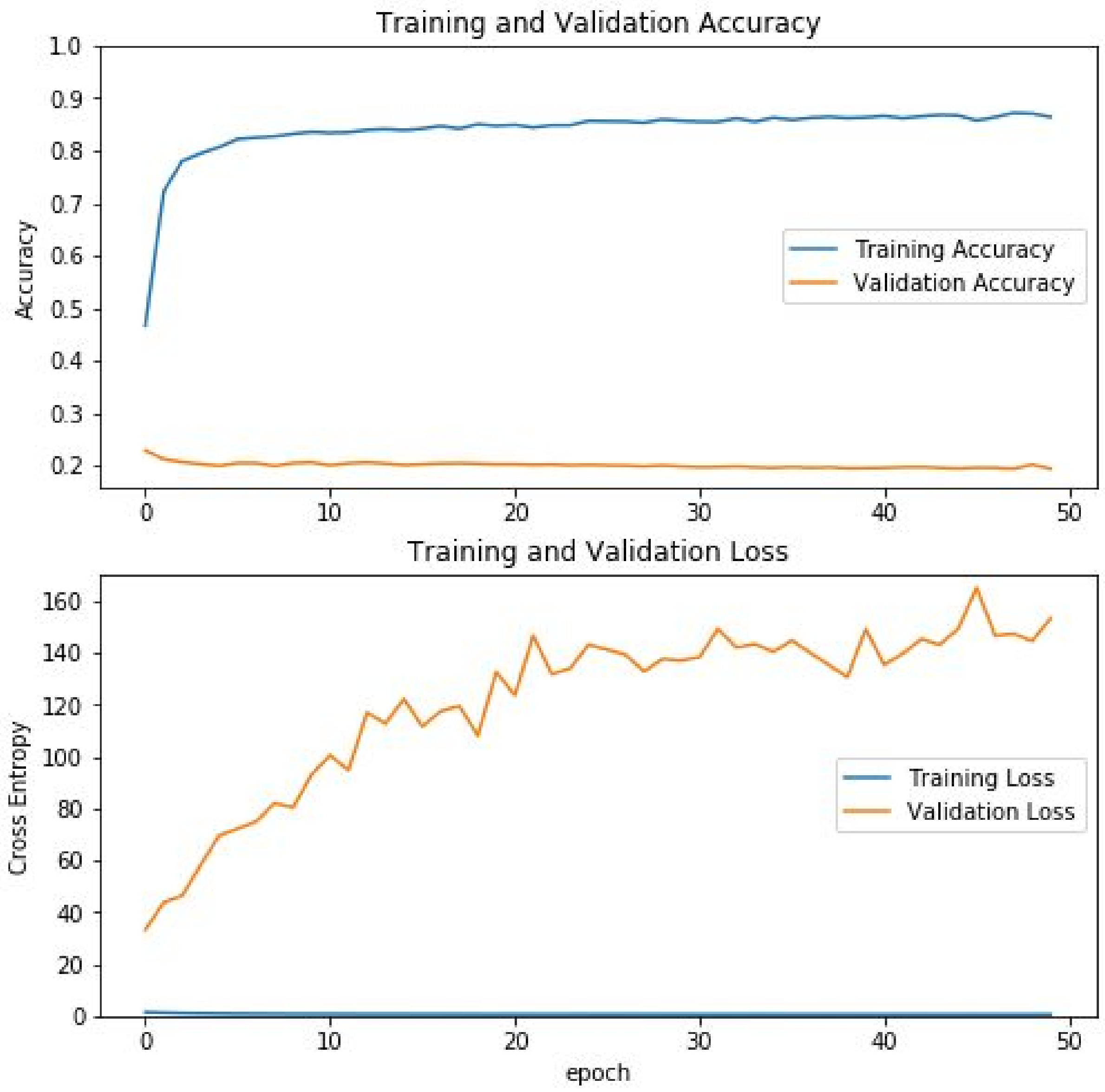


Figure 5.10: Xception model training and validation history for batch size-5

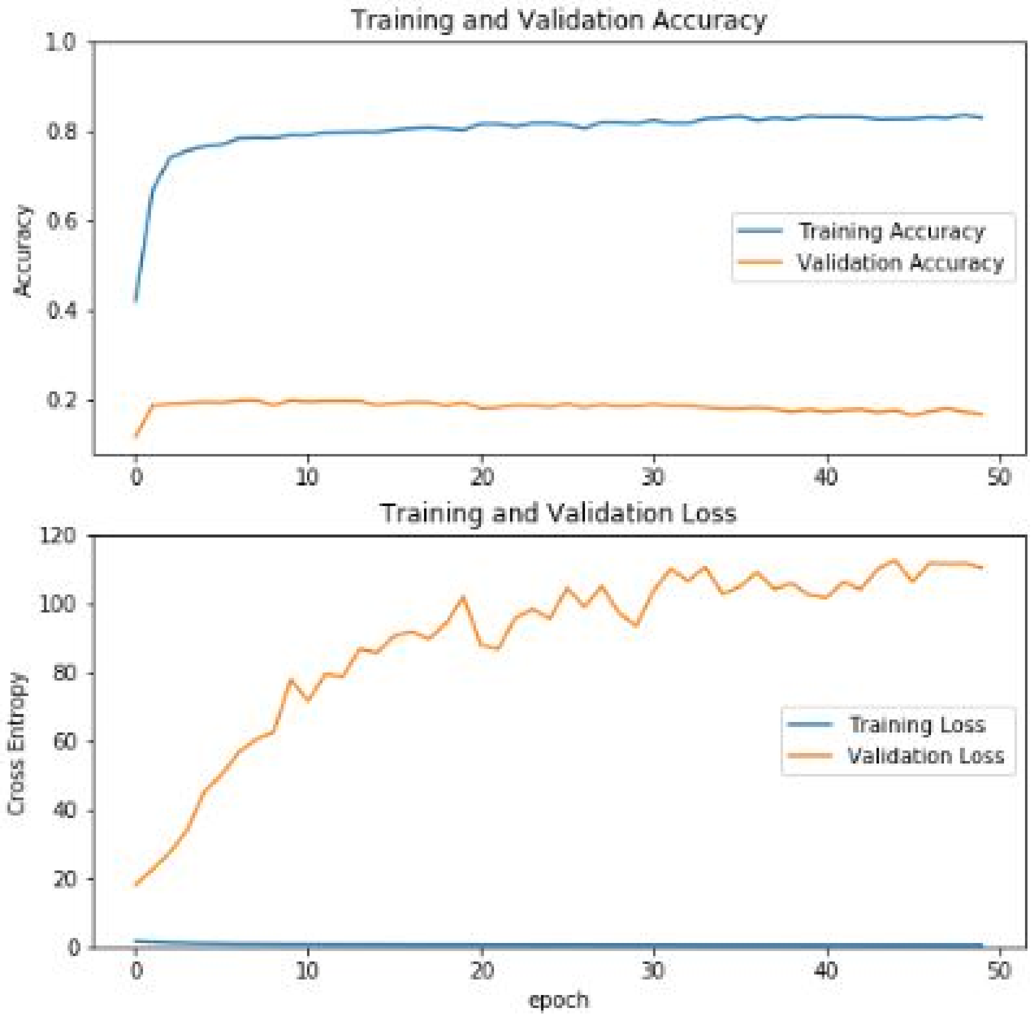


Figure 5.11: Xception model training and validation history for batch size-10 and shuffle=False for validation data.

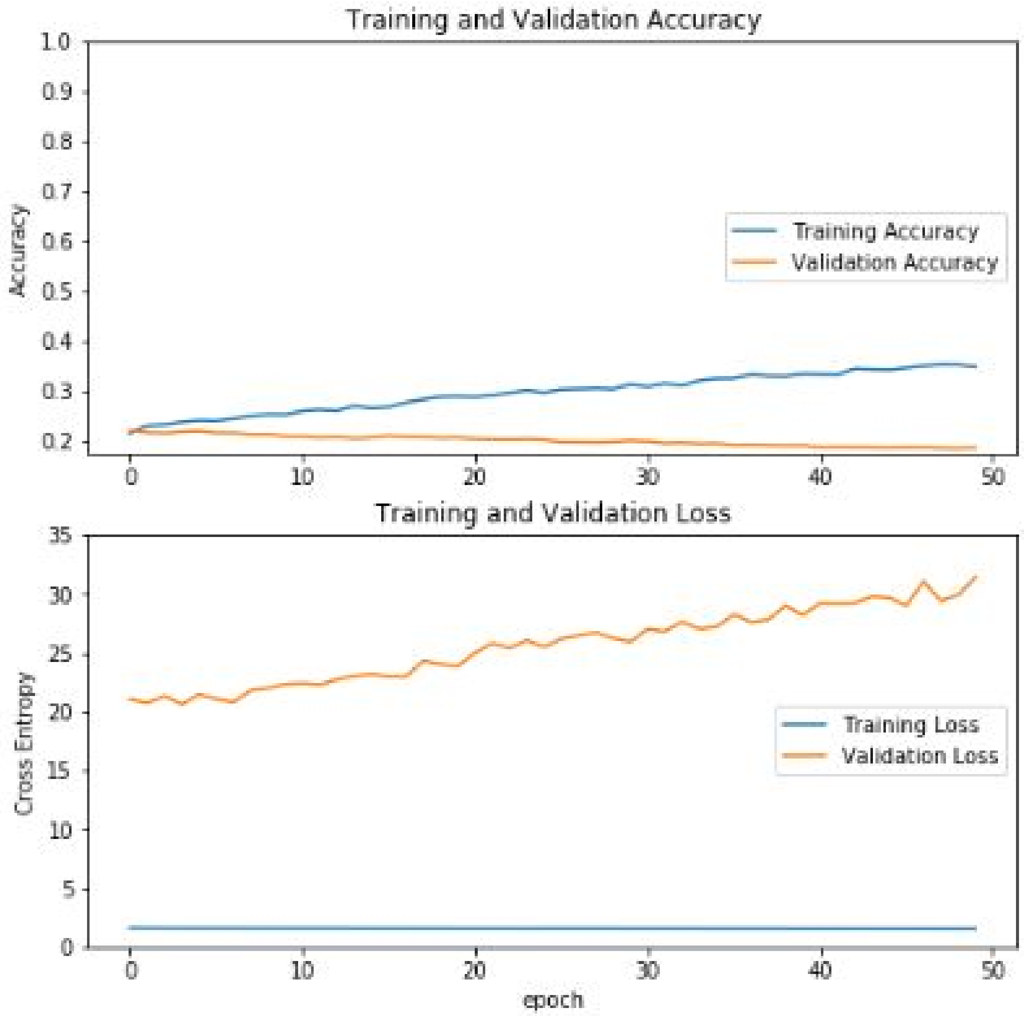


Figure 5.12: Xception model training and validation history for batch size-10 and removing dropouts.

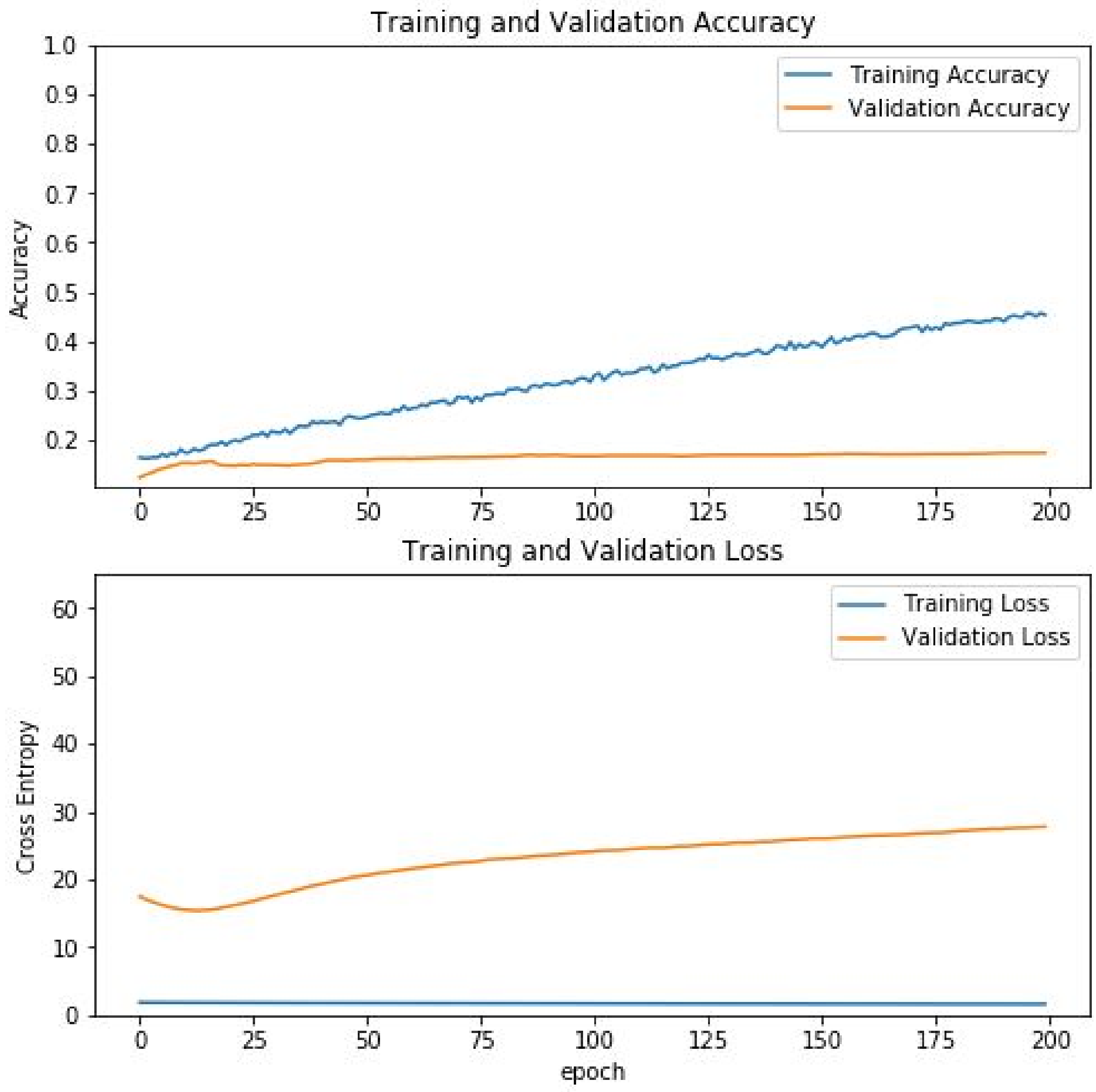


Figure 5.13: Xception model training and validation history for batch size-10, adding a dense layer and a dropout after it

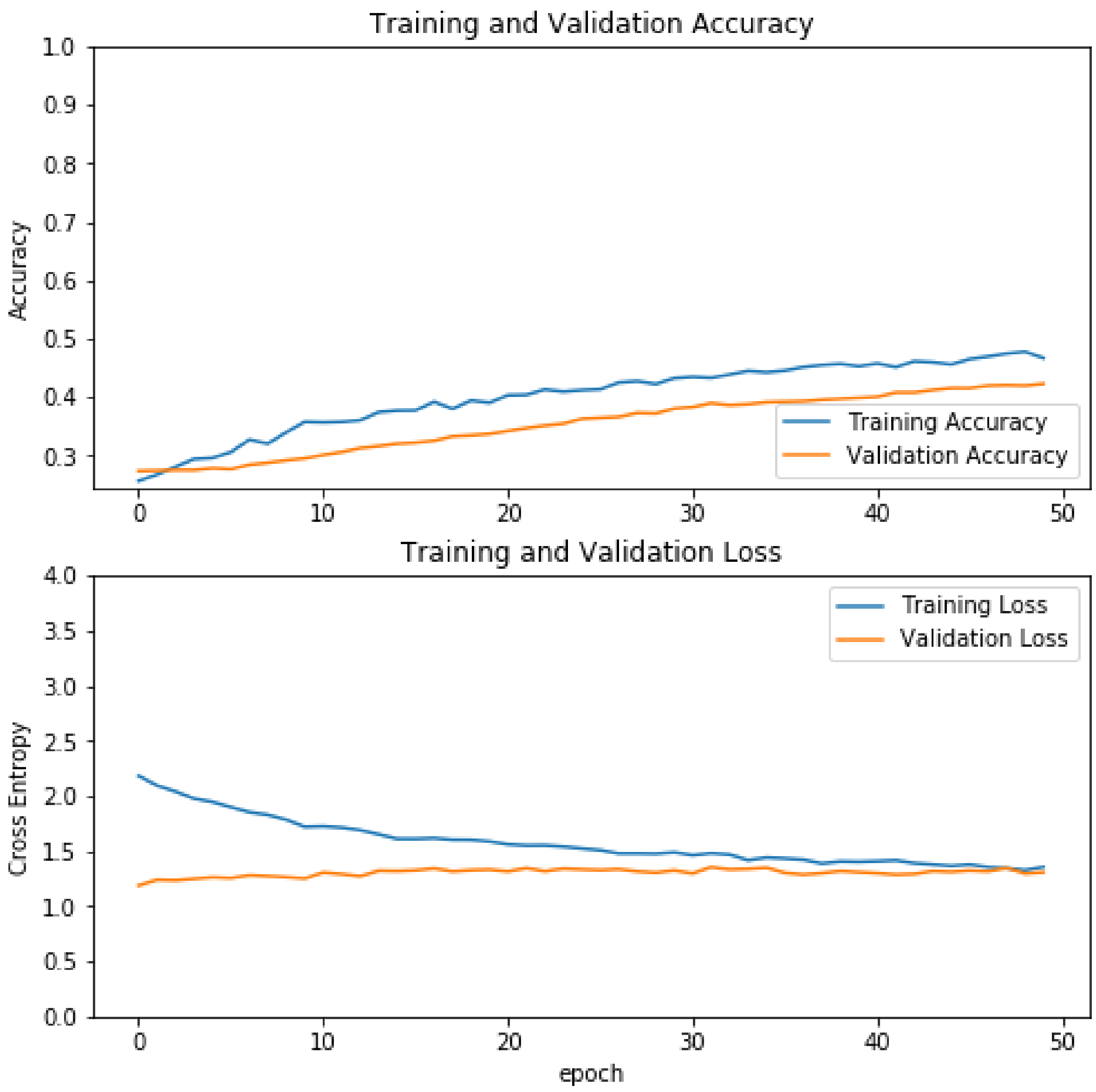


Figure 5.14: Xception model training and validation history for gender classification

is over-fitted.
2. If validation accuracy increases over each epoch and validation loss fluctuate then check if validation data is being shuffled or not. If validation data is being shuffled every time (shuffle=True for imageDataGenerator) then turn it off.
3. When dropout is added to the model, the validation accuracy might be higher than training accuracy. Because validation takes place on the actual model.
4. If the number of data per class is very low or the difference of train data of two classes is high then the accuracy degrades. There should be an equal number of train data in each class or at least the difference must have to be very low.

5.2 Model Exploration 2

5.2.1 ResNet50

Data Augmentation:
rotation range=30
zoom range=0.15
width shift range=0.2
height shift range=0.2
shear range=0.15
horizontal flip=True
fill mode="nearest"

Table 5.7: ***Model Deimage data generatort50***

Result No	***Trainablethe Layers***	***Dense***	***Dropout***	***Dense(Prediction)***
1	25 upper layers			512
2	25 upper layers	512	0.2	512
3	25 upper layers			512

Table 5.8: ***ResNet50***

Result No	***Class Column***	***Classes***	***Batch size***	***Learning Rate***
1	baseColour	7	10	0.0001
2	baseColour	7	10	0.00005
3	baseColour	7	10	0.001

5.3 Transfer Learning

Pre-trained model's performance analysis Trained using 5000 different product images and checked similarity ranking. Following is the output of different models

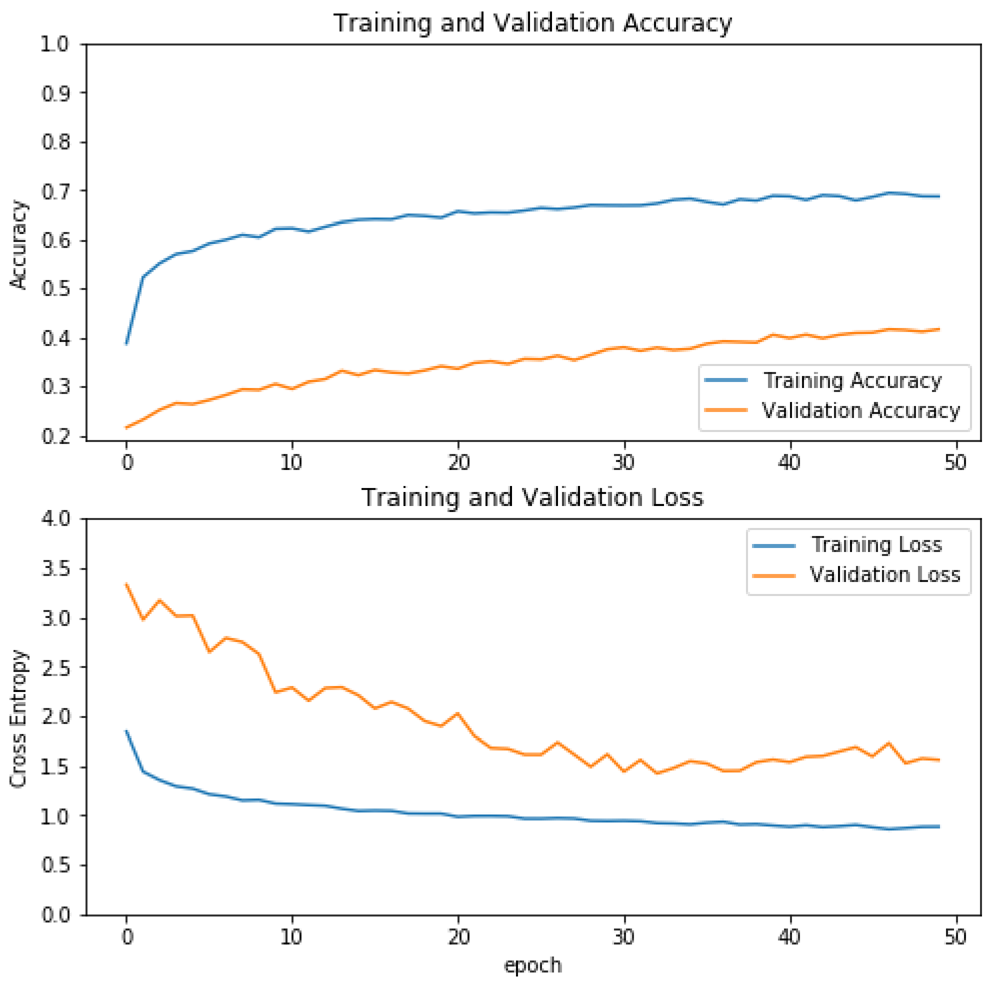


Figure 5.15: Accuracy and loss graph for ResNet50 model with 25 fine tuned upper layers

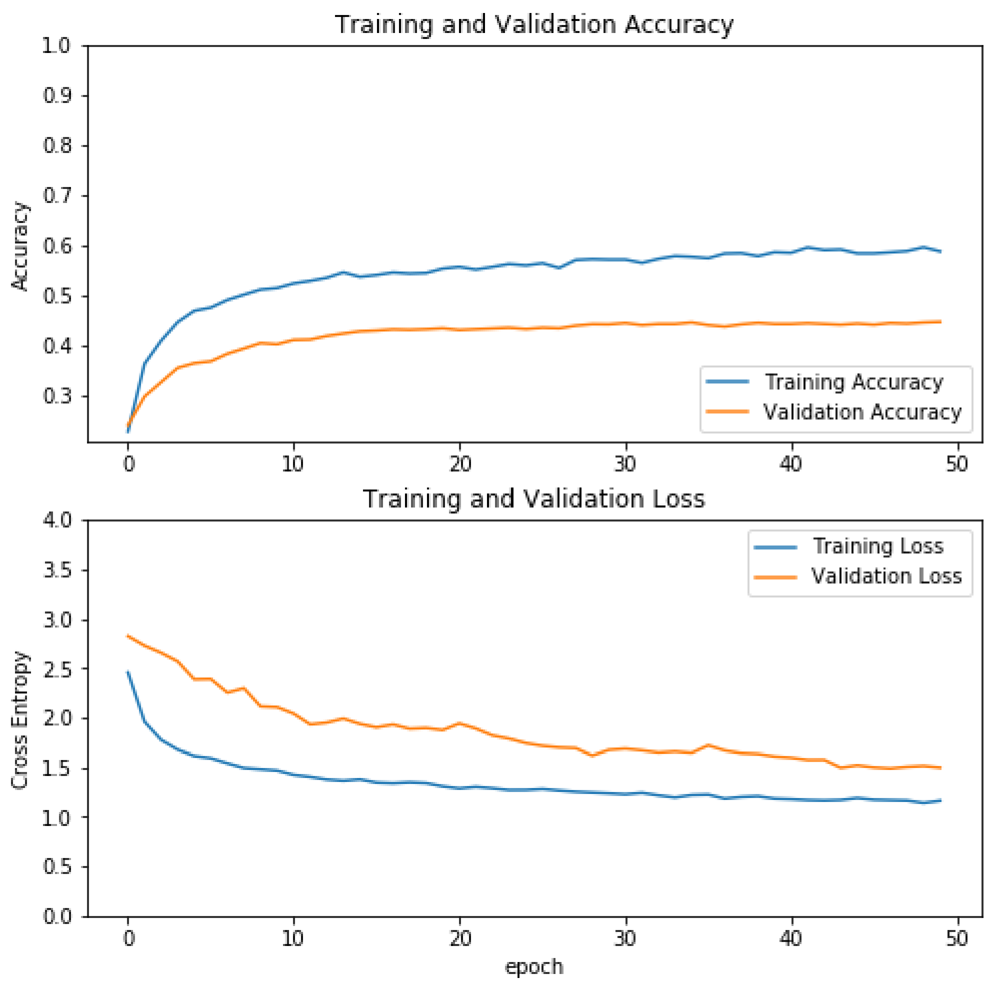


Figure 5.16: : Accuracy and loss graph for ResNet50 model with 25 fine tuned upper layers, a fully connected dense layer and a dropout.

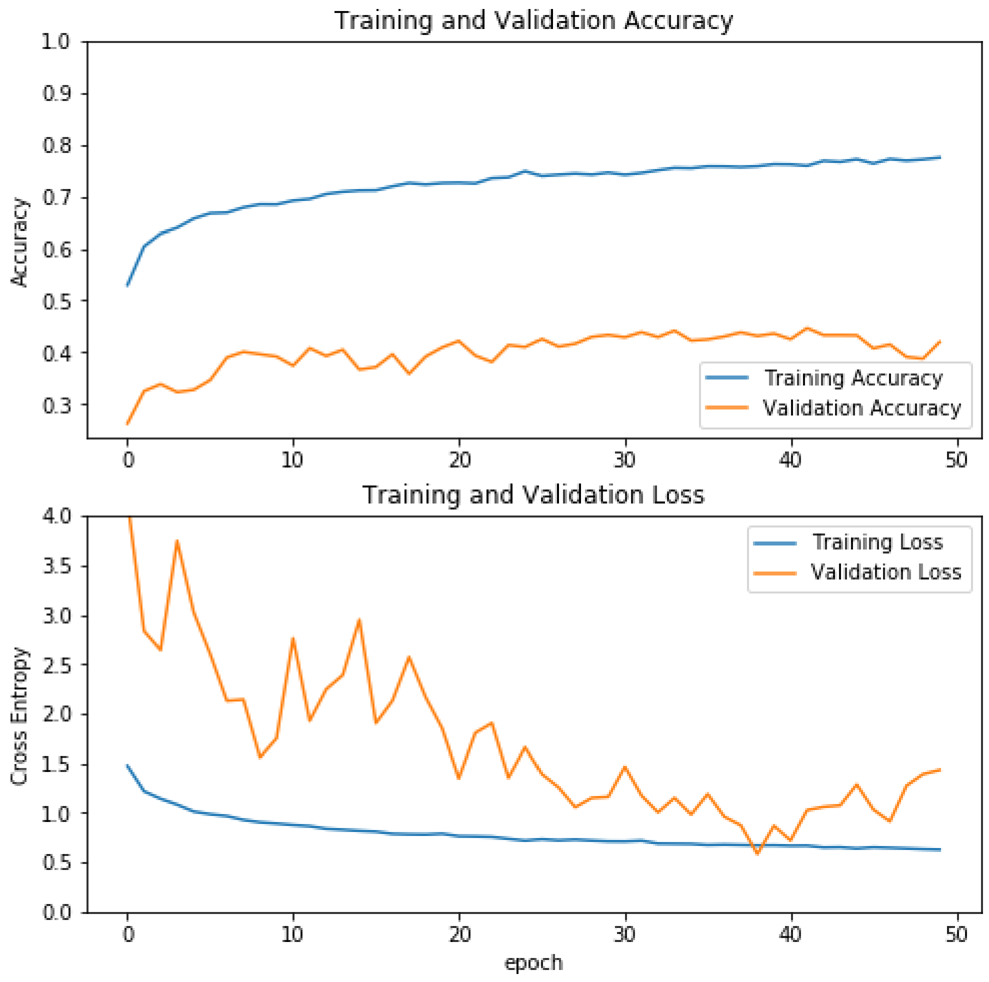


Figure 5.17: Accuracy and loss graph for ResNet50 model with 25 fine tuned upper layers and no custom layers on top of it

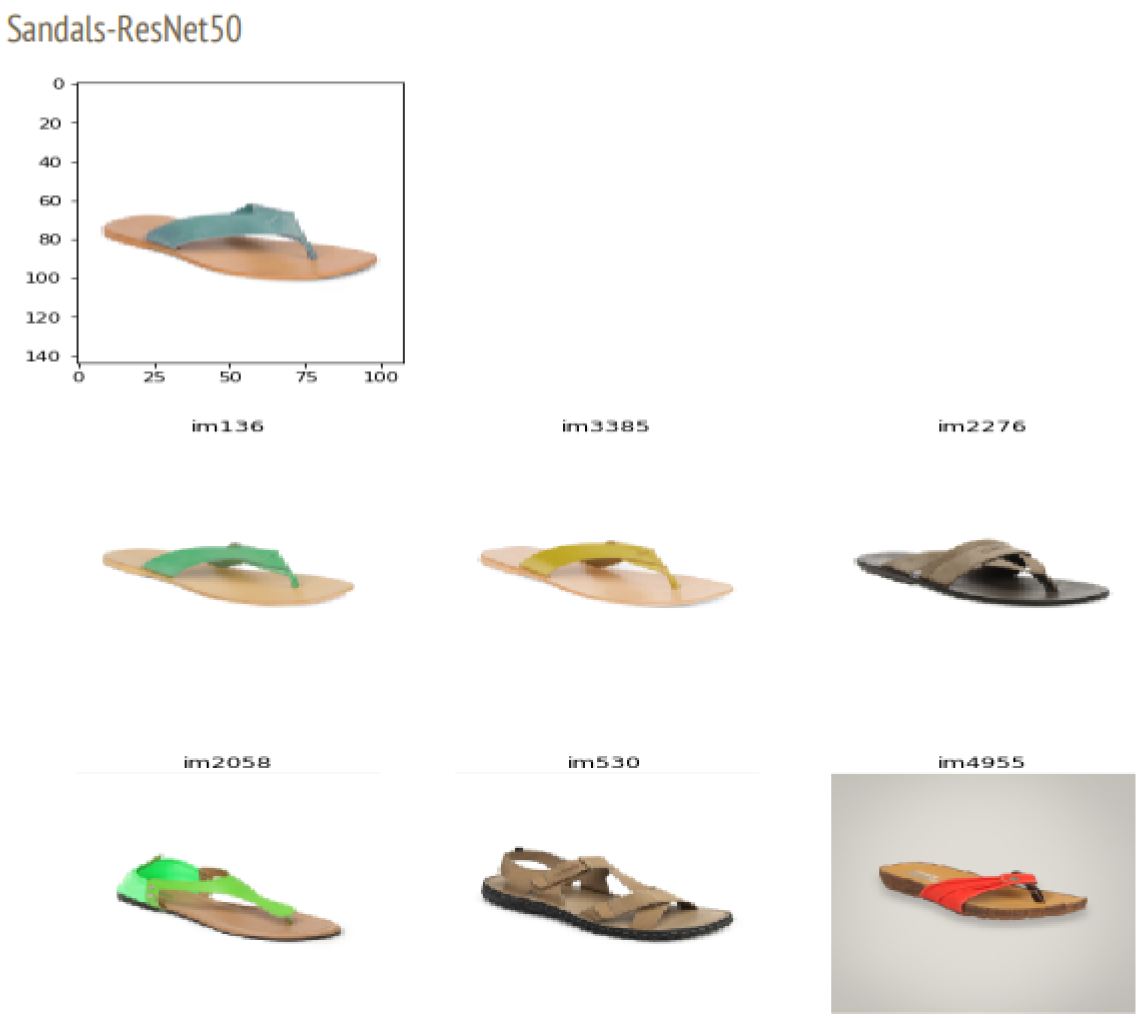
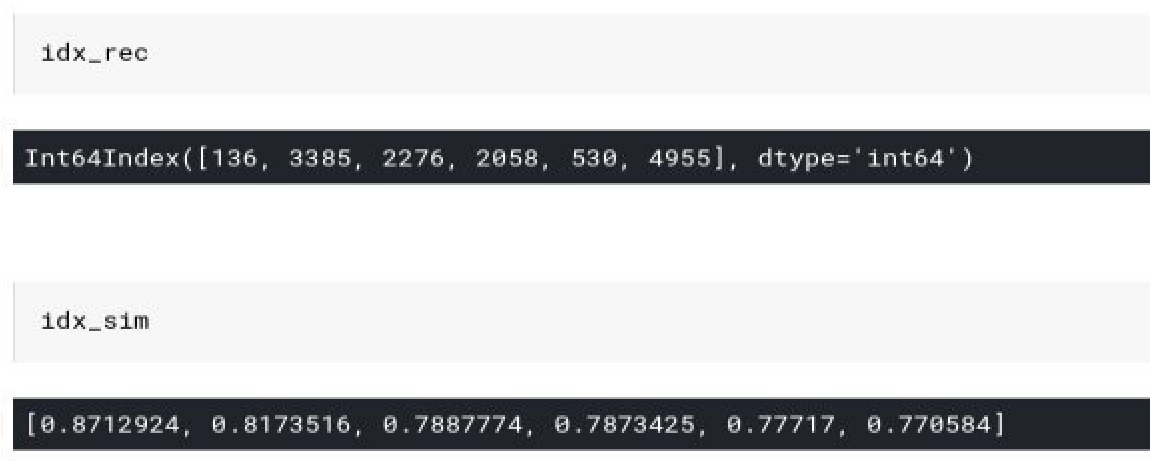


Figure 5.18: Query image of sandal and suggested similar products



```
idx_rec
```

```
Int64Index([136, 3385, 2276, 2058, 530, 4955], dtype='int64')
```

```
idx_sim
```

```
[0.8712924, 0.8173516, 0.7887774, 0.7873425, 0.77717, 0.770584]
```

Figure 5.19: Similar ids indices and corresponding similarity value for each sandal

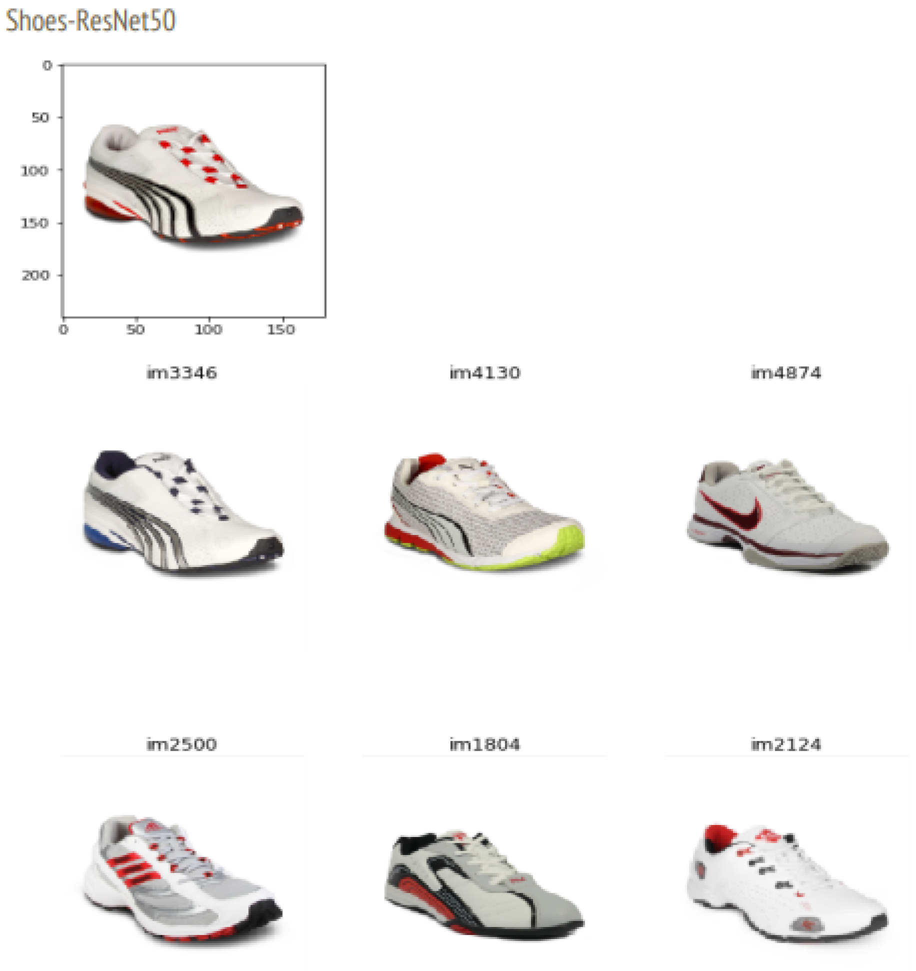
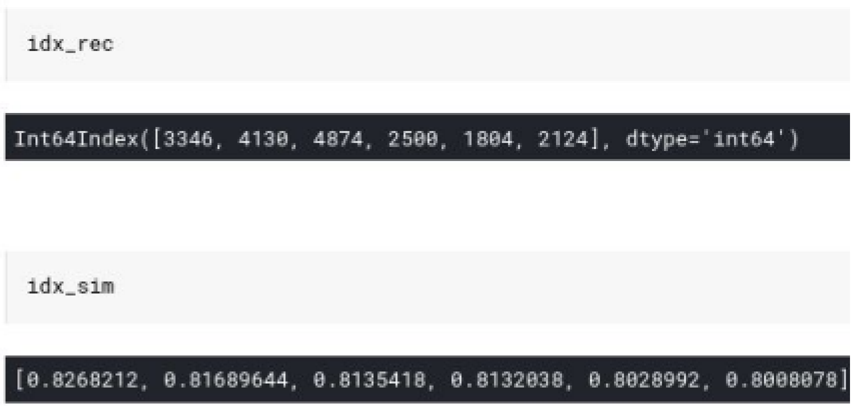


Figure 5.20: Query image of sport shoe and suggested similar shoes



```
idx_rec
```

```
Int64Index([3346, 4130, 4874, 2500, 1804, 2124], dtype='int64')
```

```
idx_sim
```

```
[0.8268212, 0.81689644, 0.8135418, 0.8132038, 0.8028992, 0.8008078]
```

Figure 5.21: Similar ids indices and corresponding similarity value for each shoe

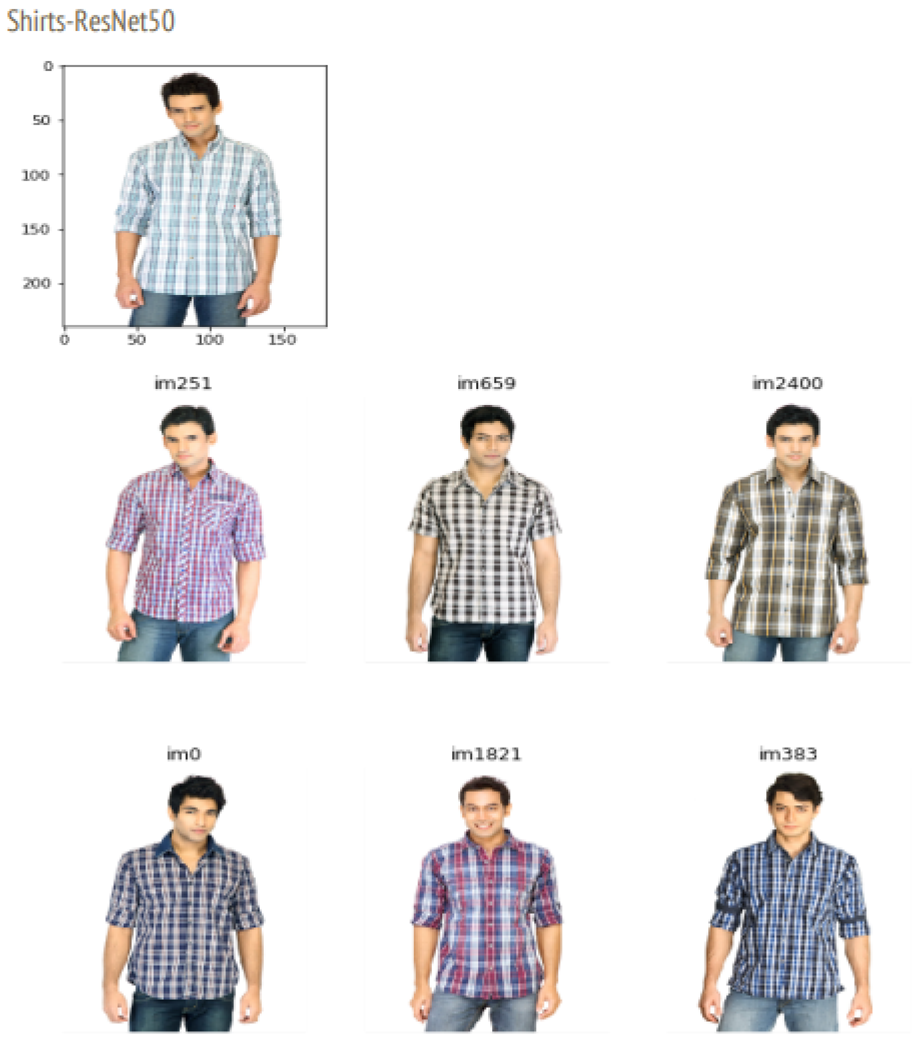
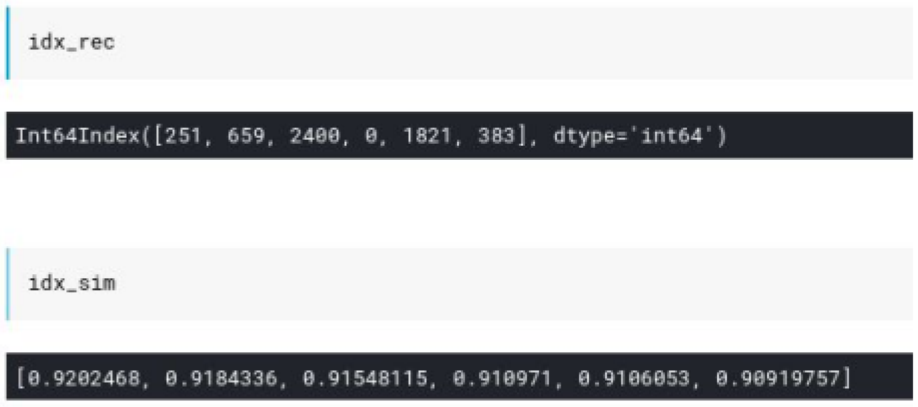


Figure 5.22: Query image of shirt and suggested similar shirts



```
idx_rec
```

```
Int64Index([251, 659, 2400, 0, 1821, 383], dtype='int64')
```

```
idx_sim
```

```
[0.9202468, 0.9184336, 0.91548115, 0.910971, 0.9106053, 0.90919757]
```

Figure 5.23: Similar ids indices and similarity values of corresponding shirts

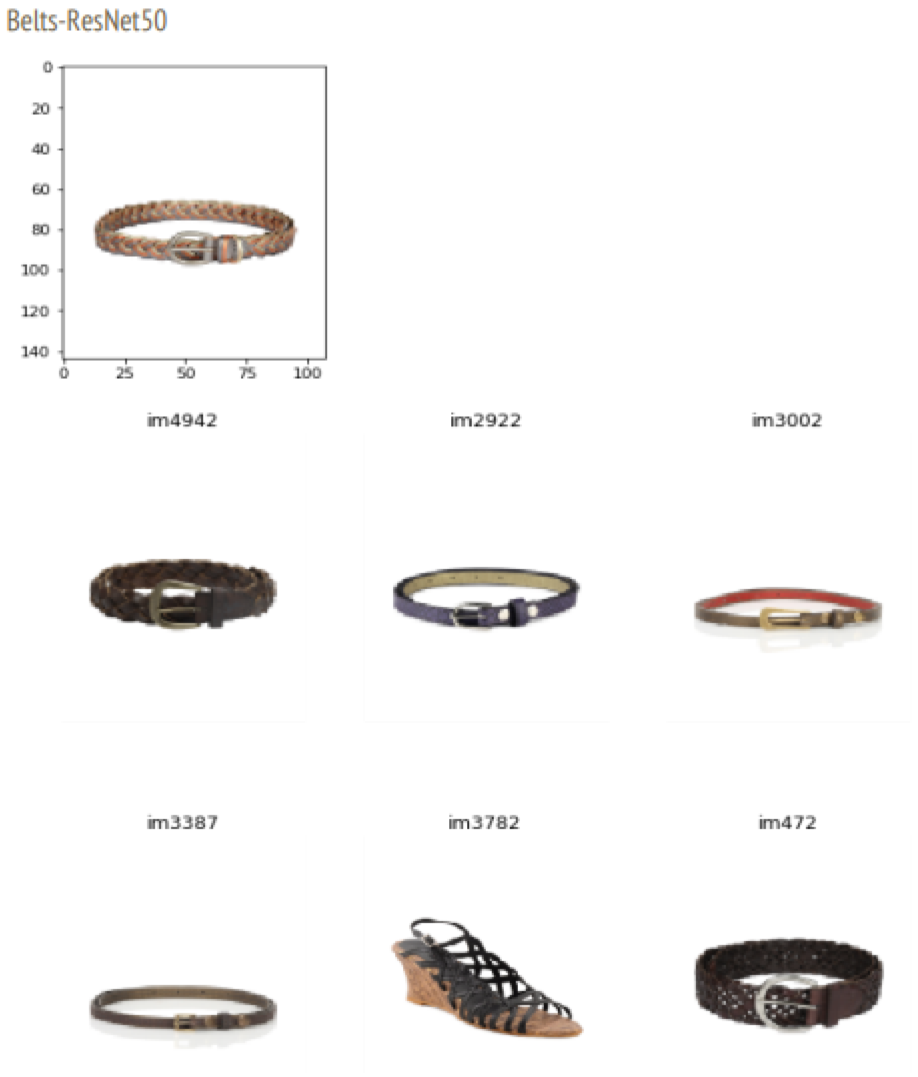
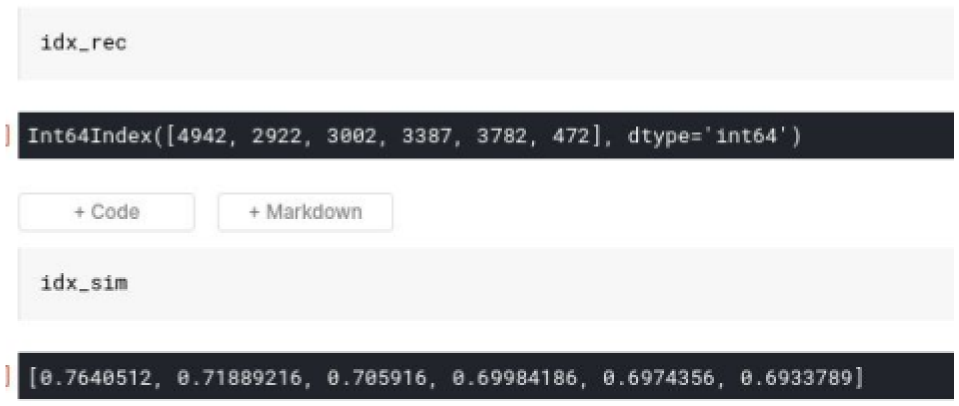


Figure 5.24: Query image of belt and suggested visually similar products



```
idx_rec
```

```
Int64Index([4942, 2922, 3002, 3387, 3782, 472], dtype='int64')
```

```
idx_sim
```

```
[0.7640512, 0.71889216, 0.705916, 0.69984186, 0.6974356, 0.6933789]
```

Figure 5.25: Similar product's ids and similarity values of corresponding shirts

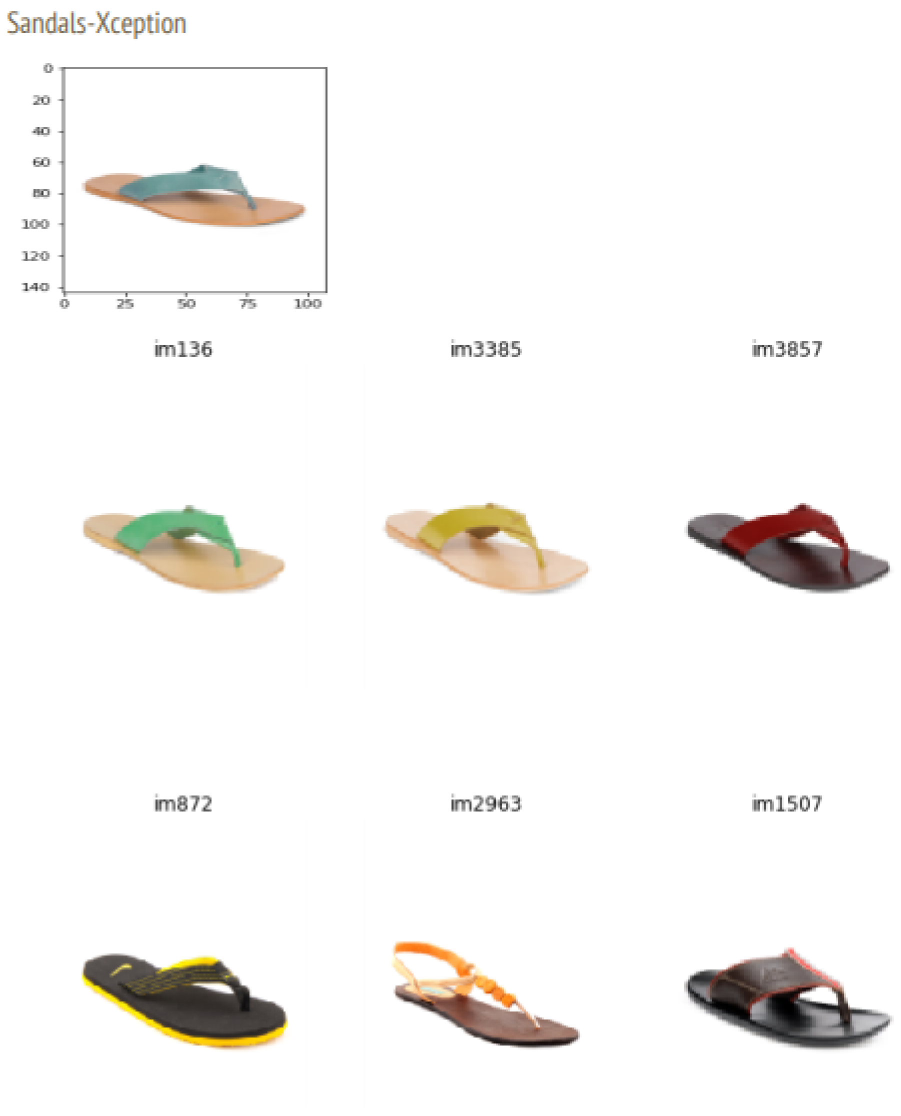
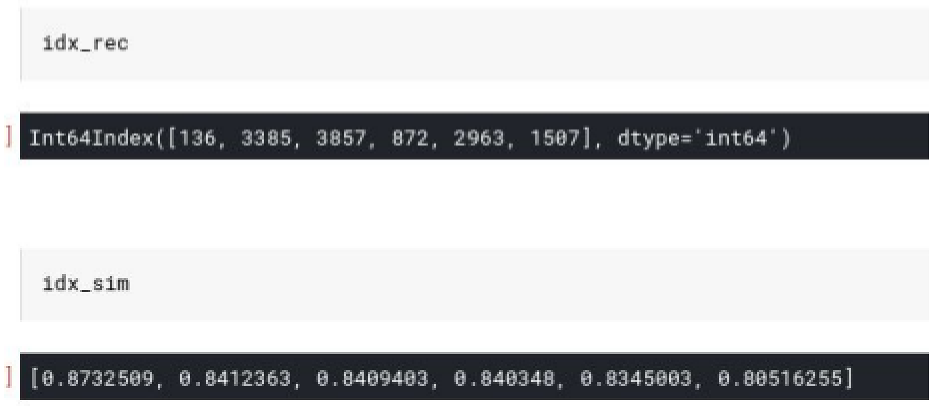


Figure 5.26: Query image of sandal and suggested similar products



```
idx_rec
```

```
Int64Index([136, 3385, 3857, 872, 2963, 1507], dtype='int64')
```

```
idx_sim
```

```
[0.8732509, 0.8412363, 0.8409403, 0.840348, 0.8345003, 0.80516255]
```

Figure 5.27: Similar ids indices and corresponding similarity value for each sandal

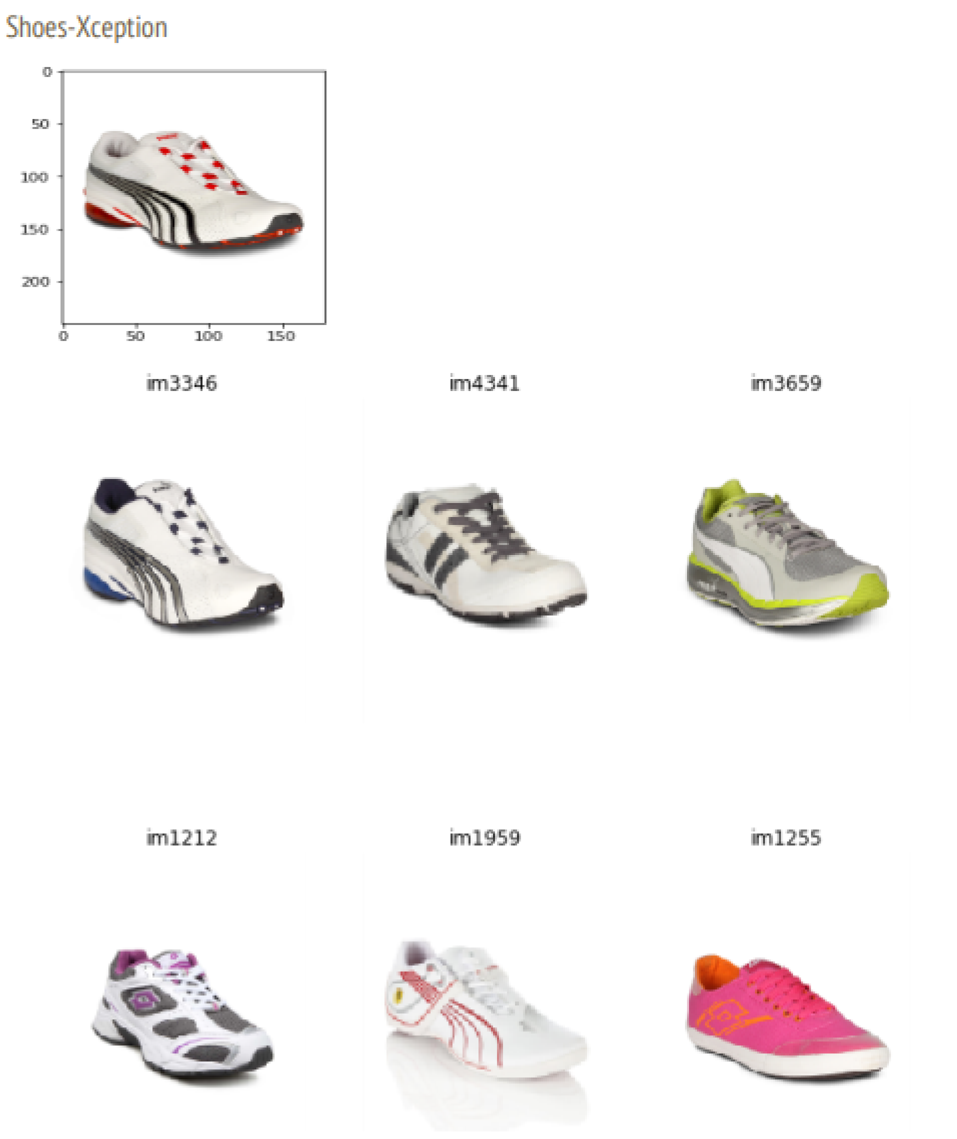
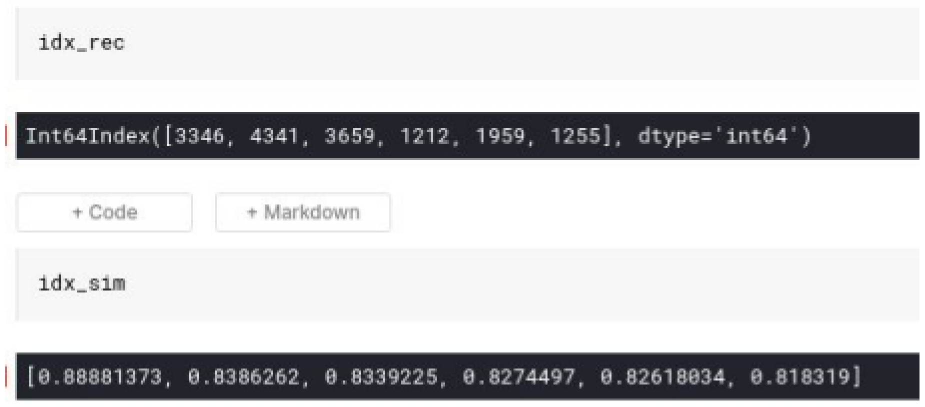


Figure 5.28: Query image of sport shoe and suggested similar shoes



```
idx_rec
```

```
Int64Index([3346, 4341, 3659, 1212, 1959, 1255], dtype='int64')
```

```
idx_sim
```

```
[0.88881373, 0.8386262, 0.8339225, 0.8274497, 0.82618034, 0.818319]
```

Figure 5.29: Similar ids indices and corresponding similarity value for each shoe

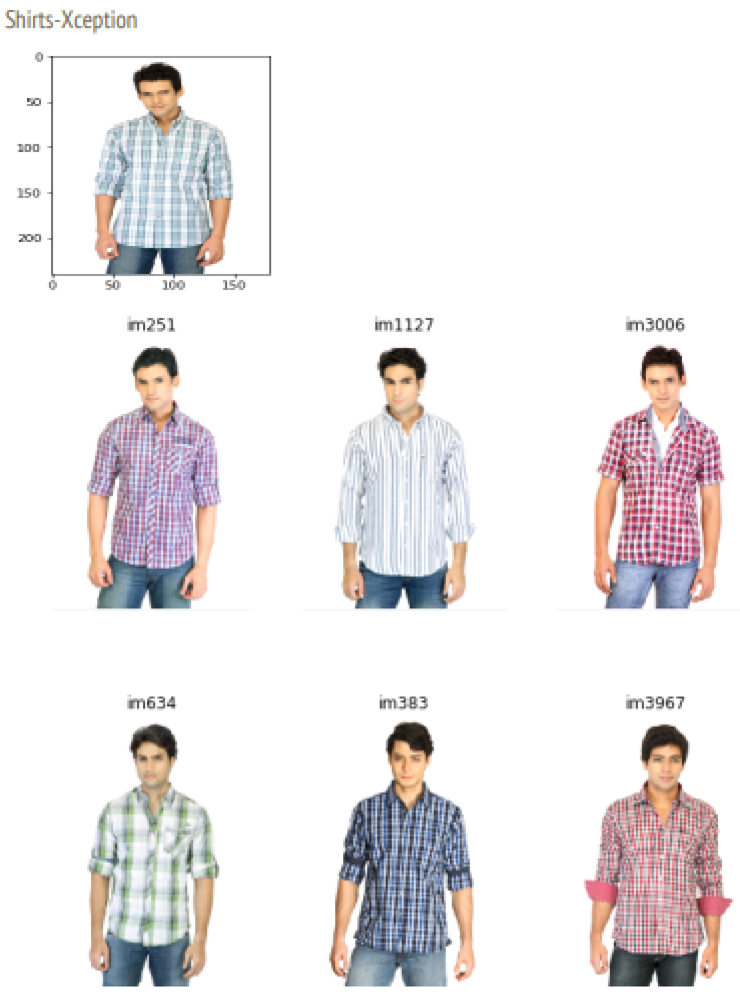
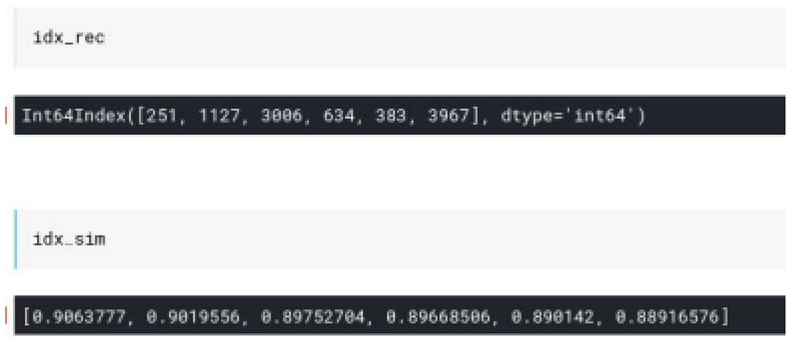


Figure 5.30: Query image of shirt and suggested similar shirts



```
idx_rec
```

```
Int64Index([251, 1127, 3006, 634, 383, 3967], dtype='int64')
```

```
idx_sim
```

```
[0.9063777, 0.9019556, 0.89752704, 0.89668506, 0.890142, 0.88916576]
```

Figure 5.31: Similar ids indices and similarity values of corresponding shirts

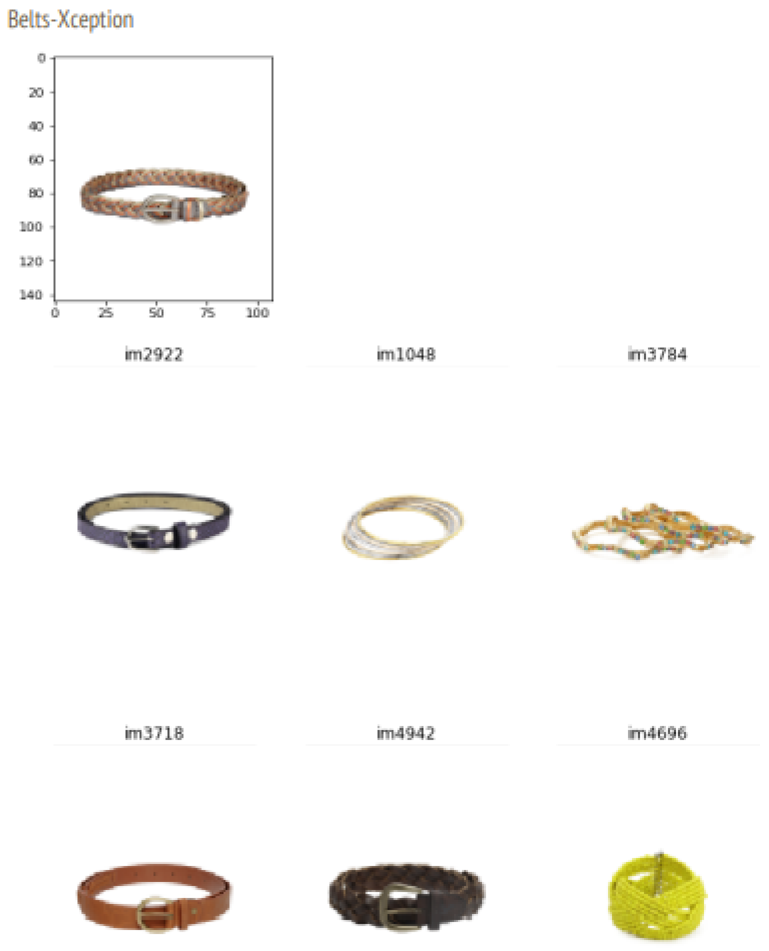
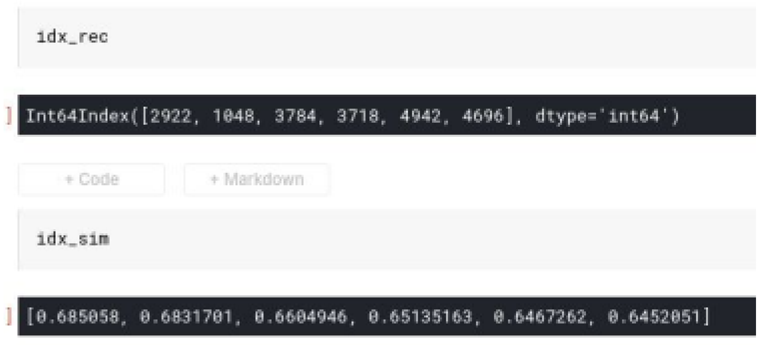


Figure 5.32: Query image of belt and suggested visually similar products



```
idx_rec
```

```
Int64Index([2922, 1048, 3784, 3718, 4942, 4696], dtype='int64')
```

```
idx_sim
```

```
[0.685058, 0.6831701, 0.6604946, 0.65135163, 0.6467262, 0.6452051]
```

Figure 5.33: Similar product's ids and similarity values of corresponding shirts

5.3.1 ResNet50

5.3.2 Xception

5.3.3 ResNet50 vs Xception

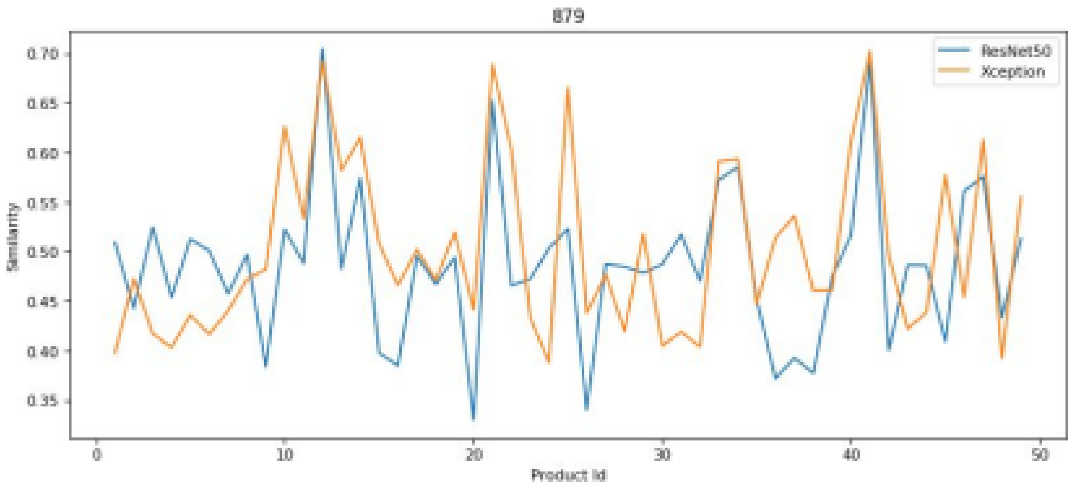


Figure 5.34: Similarity between product-879 and first 50 other products

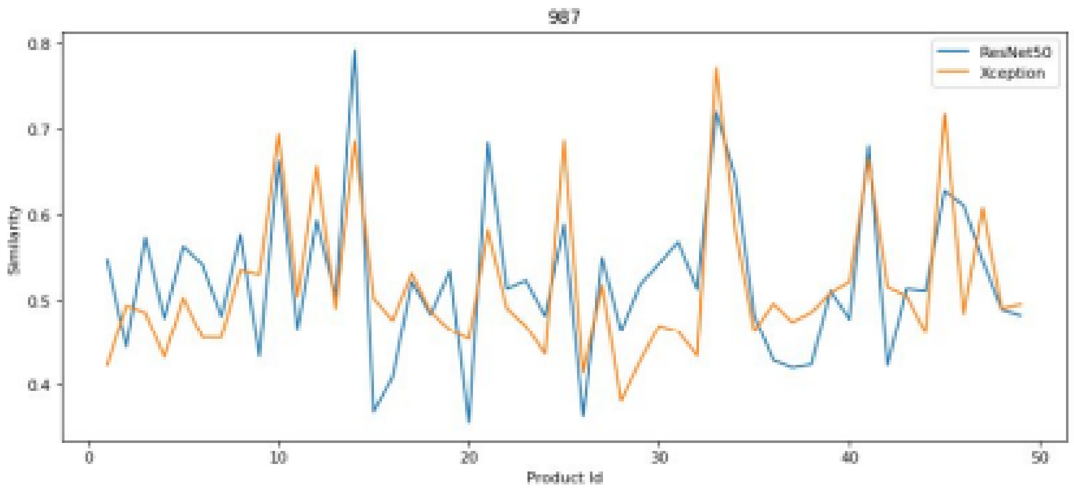


Figure 5.35: Similarity between product-987 and first 50 other products

5.4 Model Selection

For model selection based on accuracy of image(no filter parameter) recommendation, I got the best result for Resnet50 and Resnet 152. The following table shows the results for both:
As we can clearly see from the above plots if I used the Resnet152 model for feature extraction. it gives the best results for all the important features chosen but the size

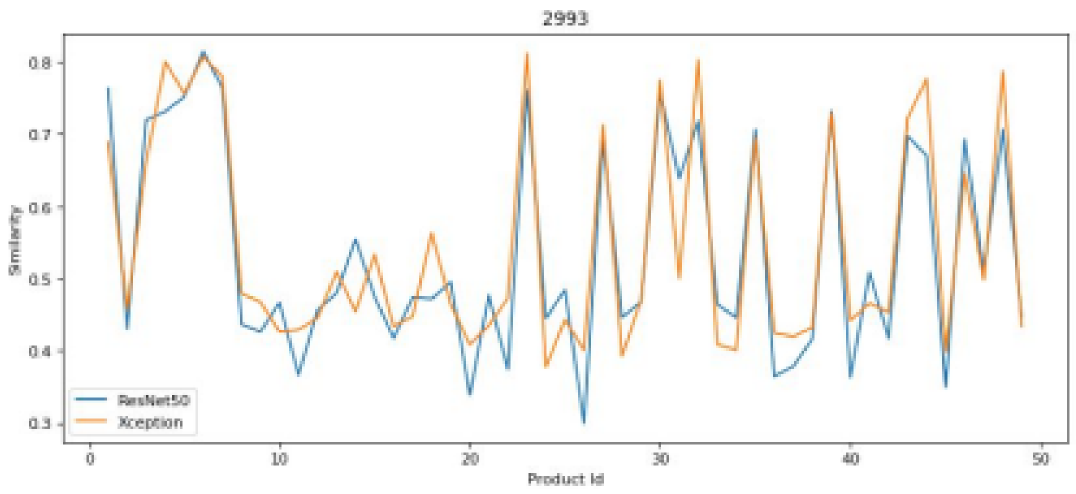


Figure 5.36: Similarity between product-2993 and first 50 other products

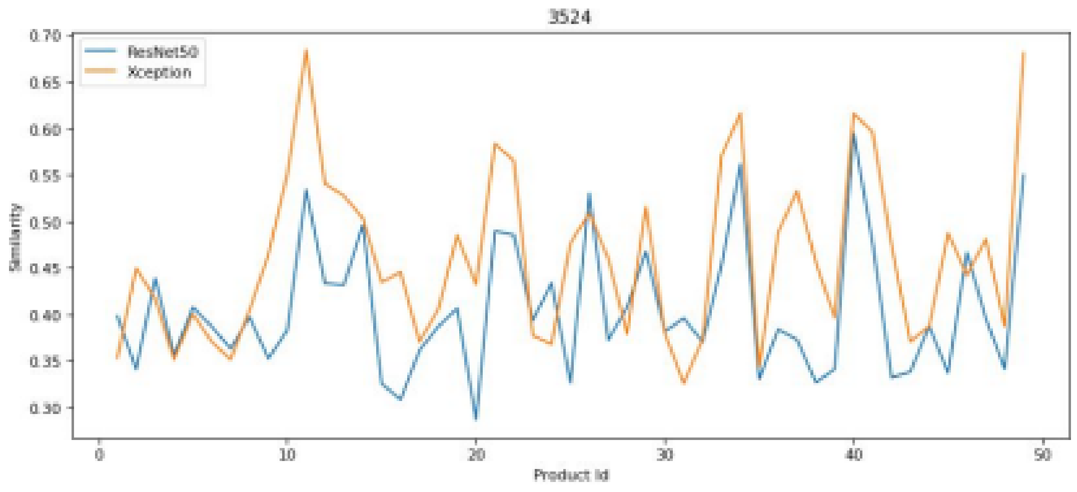


Figure 5.37: Similarity between product-3524 and first 50 other products

Table 5.9: ***Best Result for ResNet50 TopN 10***

Feature	***Tolerance***	***Performance(%)***
article Type	0.2	82.21
article Type	0.1	76.73
article Type	0	67.84
gender	0.2	81.02
gender	0.1	74.24
gender	0	63.67
gender and article Type	0.2	67.50
gender and article Type	0.1	58.92
gender and article Type	0	47.03

Table 5.10: ***Best Result for ResNet152 TopN 10***

Feature	***Tolerance***	***Performance(%)***
article Type	0.2	82.99
article Type	0.1	77.44
article Type	0	68.66
gender	0.2	81.39
gender	0.1	74.75
gender	0	63.94
gender and article Type	0.2	68.48
gender and article Type	0.1	59.98
gender and article Type	0	47.80

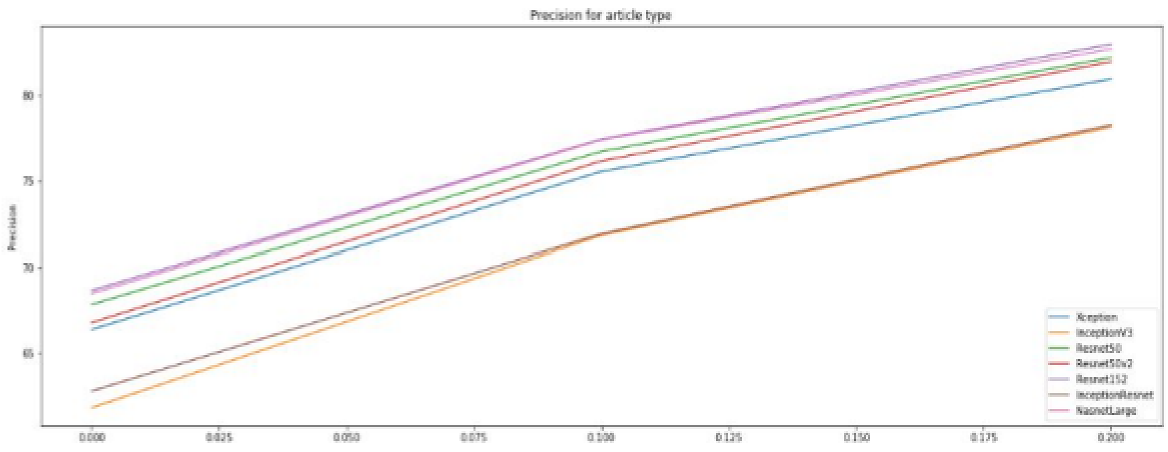


Figure 5.38: Feature extractor model comparison for article type

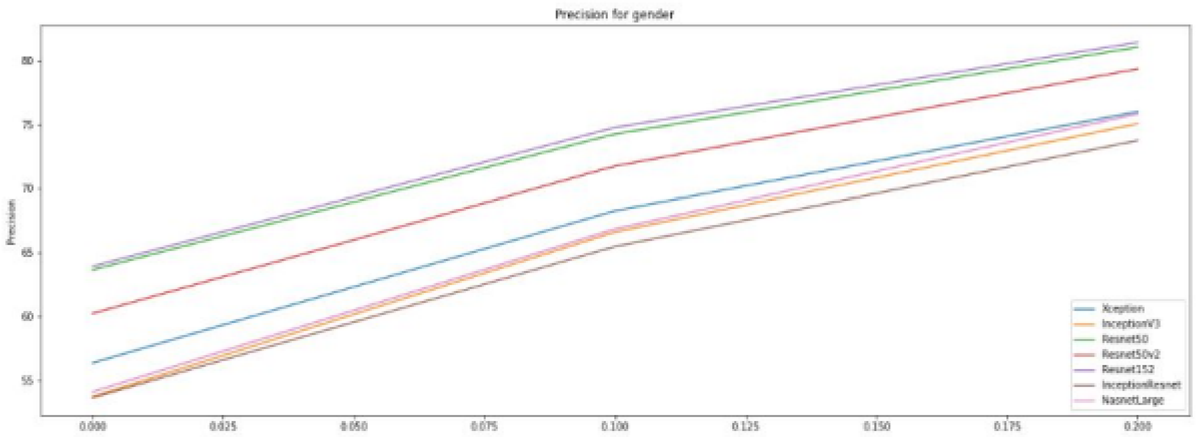


Figure 5.39: Feature extractor model comparison for gender

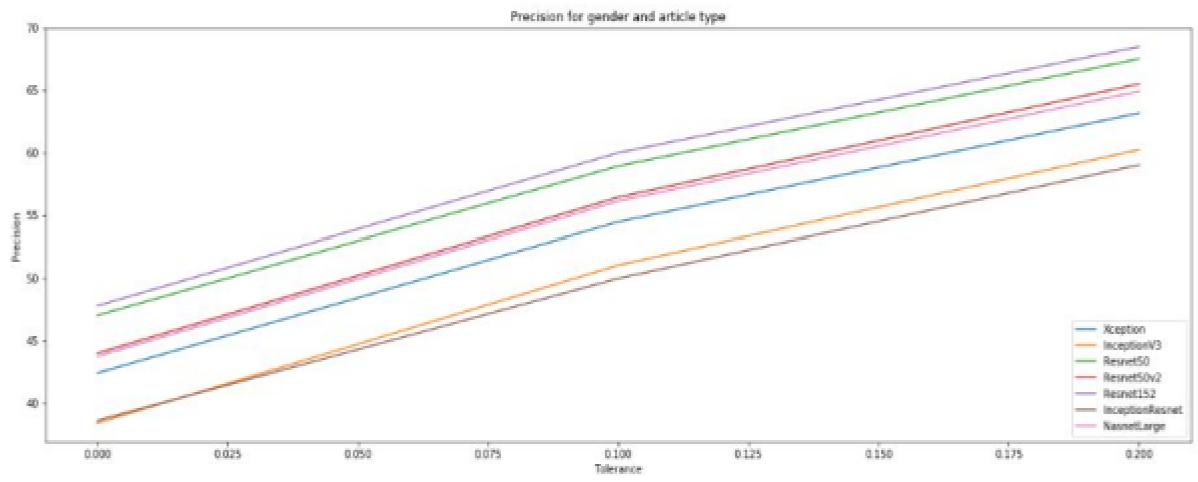


Figure 5.40: Feature extractor model comparison for article type and gender

of this model is 232MB. On the other hand, we can also see that Resnet50 is close enough to Resnet152 but its size is just 98MB, which is almost 125% lesser than Resnet50. So Resnet50 is lighter and faster while extracting features from images. So, I used Resnet50 for feature extraction.

5.4.1 Classifier(for subCategory):

Table 5.11: ***Classifier(for subCategory)***

Train accuracy	90%
Train loss	0.27
Validation accuracy	85%
Validation loss	0.16
Test accuracy	86.46% (1367 / 1581)

5.4.2 Classifier(for articleType/derivedType):

After an image is classified in subcategory, I classify it further in the next level for identifying its derived type; that is gender and articleType. In this case, use a candidate-based classifier that works on calculating similarity among candidate images of every group. Here I also take Precision as an accuracy metric. That is: if any of the recommended items is wrong by gender or article type, then the whole recommendation is treated as wrong. In this case for 5000 images, Precision: 75%.

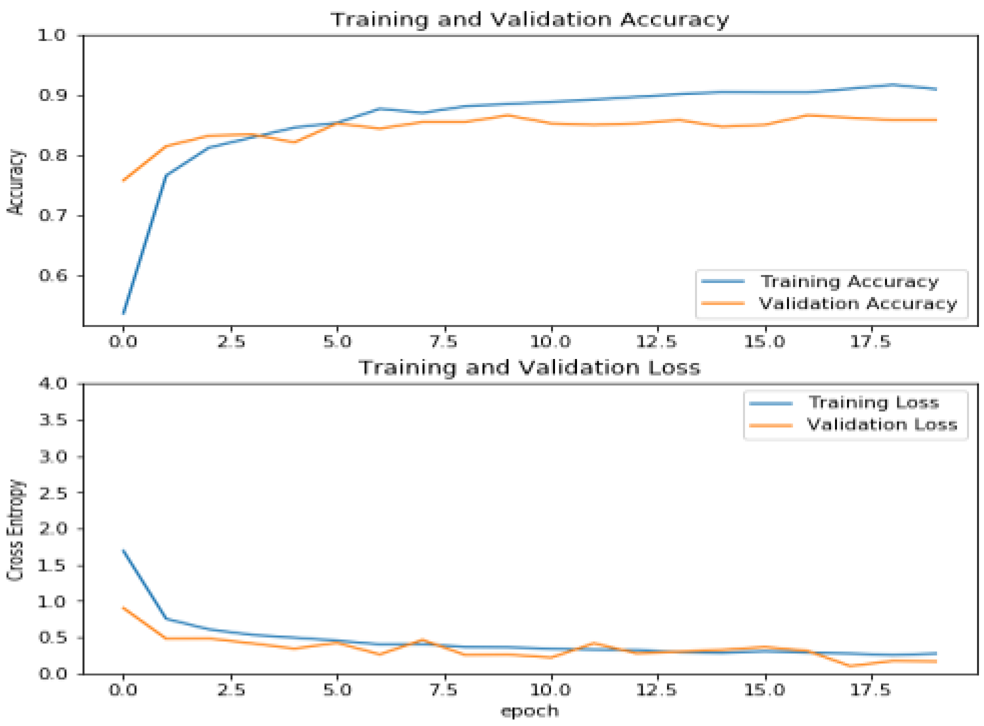


Figure 5.41: Training history of custom classifier on sub-category

Chapter 6

Visual Representation

After training the model with the best model with the two classifiers, I saved the model. The embeddings were also saved in a file. I used these files for the API implementation. For generating API Flask library was used in this project. After making the API, I displayed the result in a frinto Angular template named “Fashion House”. In this template, we can checkout the most popular fashion items of each category in the homepage. If we click on one of them top recommendation in that category is shown in a different page. There is a search option also for unknown images in which we can upload an image optionally. Imagine if we like a certain dress not in inventory, then we can click a photo of the liked fashion product and upload it, then the system would recommend us images similar to the fashion product we like, we can also customize the search with various filters.

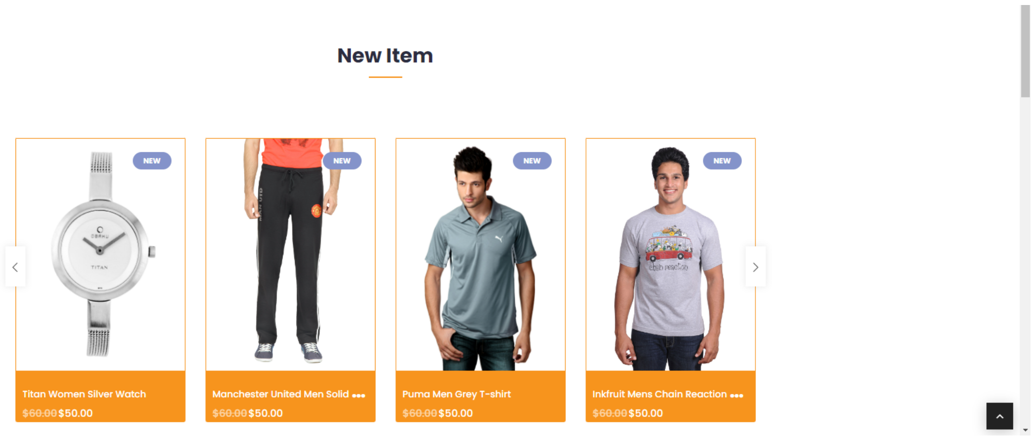


Figure 6.1: Top rated Item shown on the Homepage

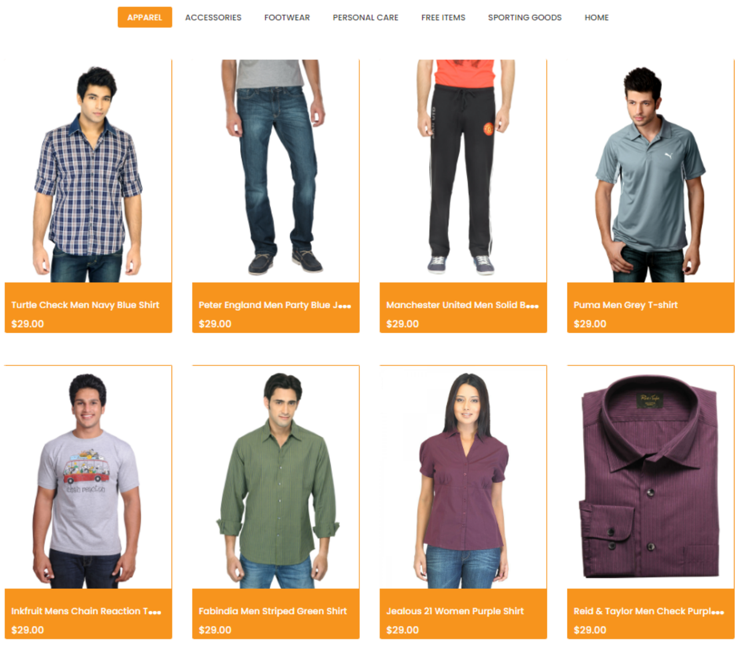

APPAREL
ACCESSORIES
FOOTWEAR
PERSONAL CARE
FREE ITEMS
SPORTING GOODS
HOME
Turtle Check Men Navy Blue Shirt
$29.00
Peter England Men Party Blue J•••
$29.00
Manchester United Men Solid B•••
$29.00
Puma Men Grey T-shirt
$29.00
Inkfruit Mens Chain Reaction T•••
$29.00
Fabindia Men Striped Green Shirt
$29.00
Jealous 21 Women Purple Shirt
$29.00
Reid & Taylor Men Check Purpl•••
$29.00

Figure 6.2: Top rated apparel shown on the Homepage

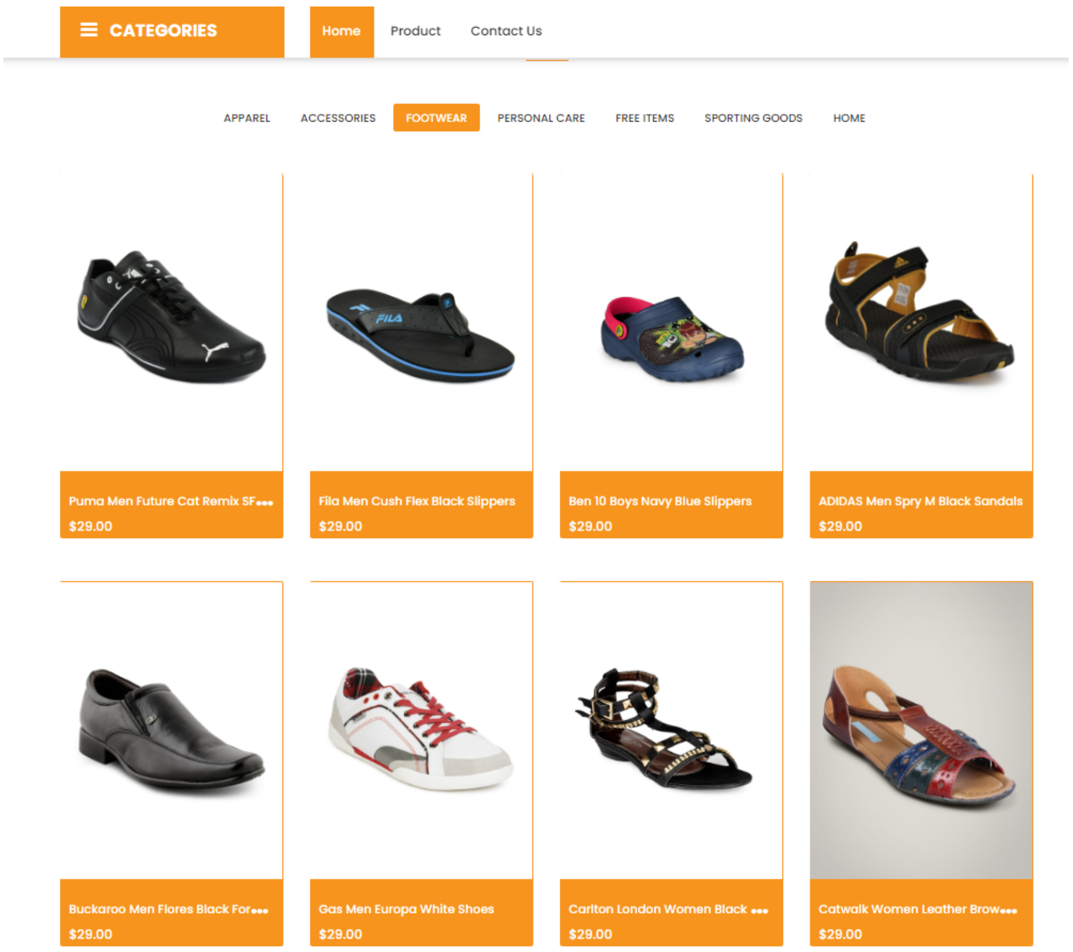

CATEGORIES
Home
Product
Contact Us
APPAREL
ACCESSORIES
FOOTWEAR
PERSONAL CARE
FREE ITEMS
SPORTING GOODS
HOME
Puma Men Future Cat Remix SF•••
$29.00
Fila Men Cush Flex Black Slippers
$29.00
Ben 10 Boys Navy Blue Slippers
$29.00
ADIDAS Men Spry M Black Sandals
$29.00
Buckaroo Men Flores Black For•••
$29.00
Gas Men Europa White Shoes
$29.00
Carlton London Women Black •••
$29.00
Catwalk Women Leather Brow•••
$29.00

Figure 6.3: Top rated footwear shown on the Homepage

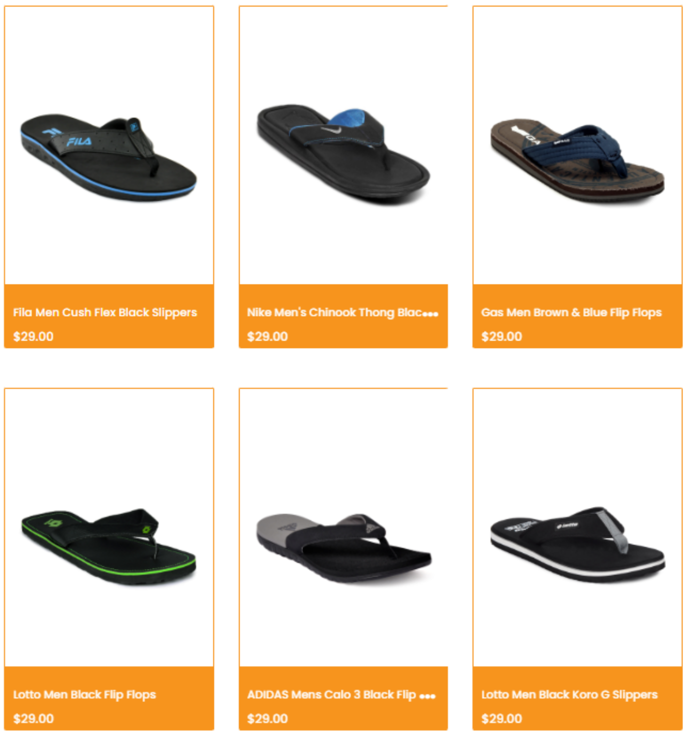
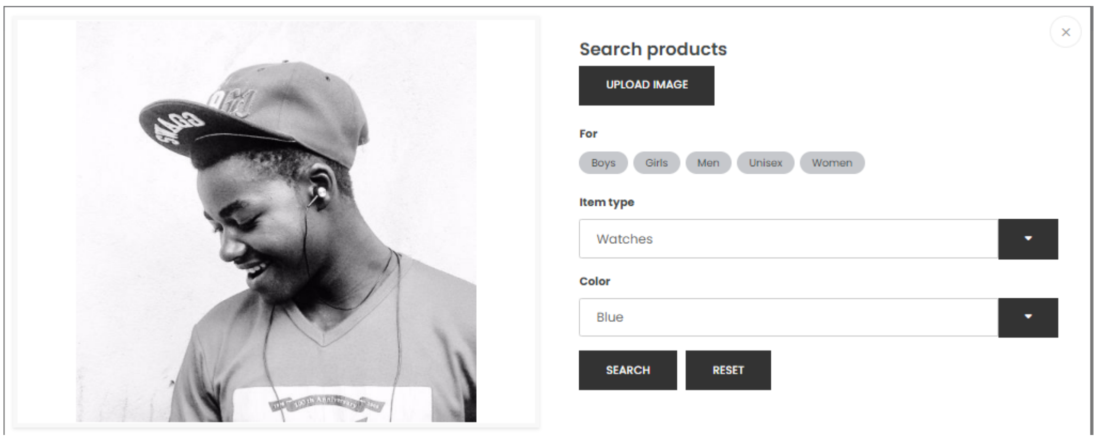

Fila Men Cush Flex Black Slippers
$29.00
Nike Men's Chinook Thong Blac...
$29.00
Gas Men Brown & Blue Flip Flops
$29.00
Lotto Men Black Flip Flops
$29.00
ADIDAS Mens Calo 3 Black Flip ...
$29.00
Lotto Men Black Koro G Slippers
$29.00

Figure 6.4: Top rated footwear list after selecting a men black flipflops



Item type

Watches

Color

Blue

Figure 6.5: The search modal for searching items

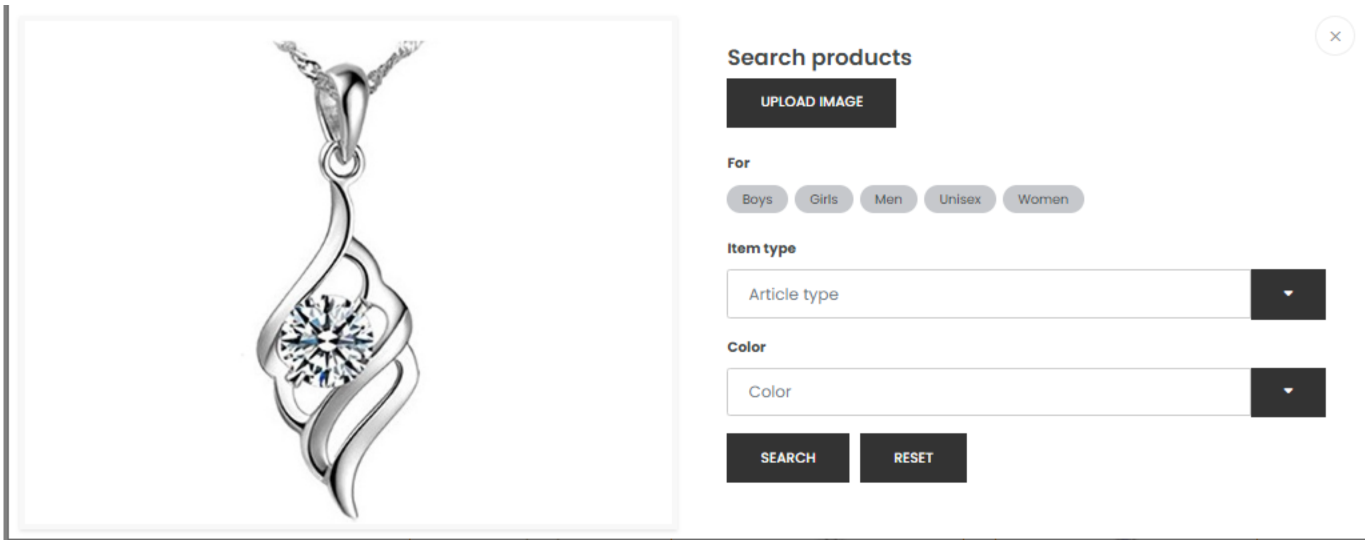


Figure 6.6: The search modal after selecting an unknown image of an earring

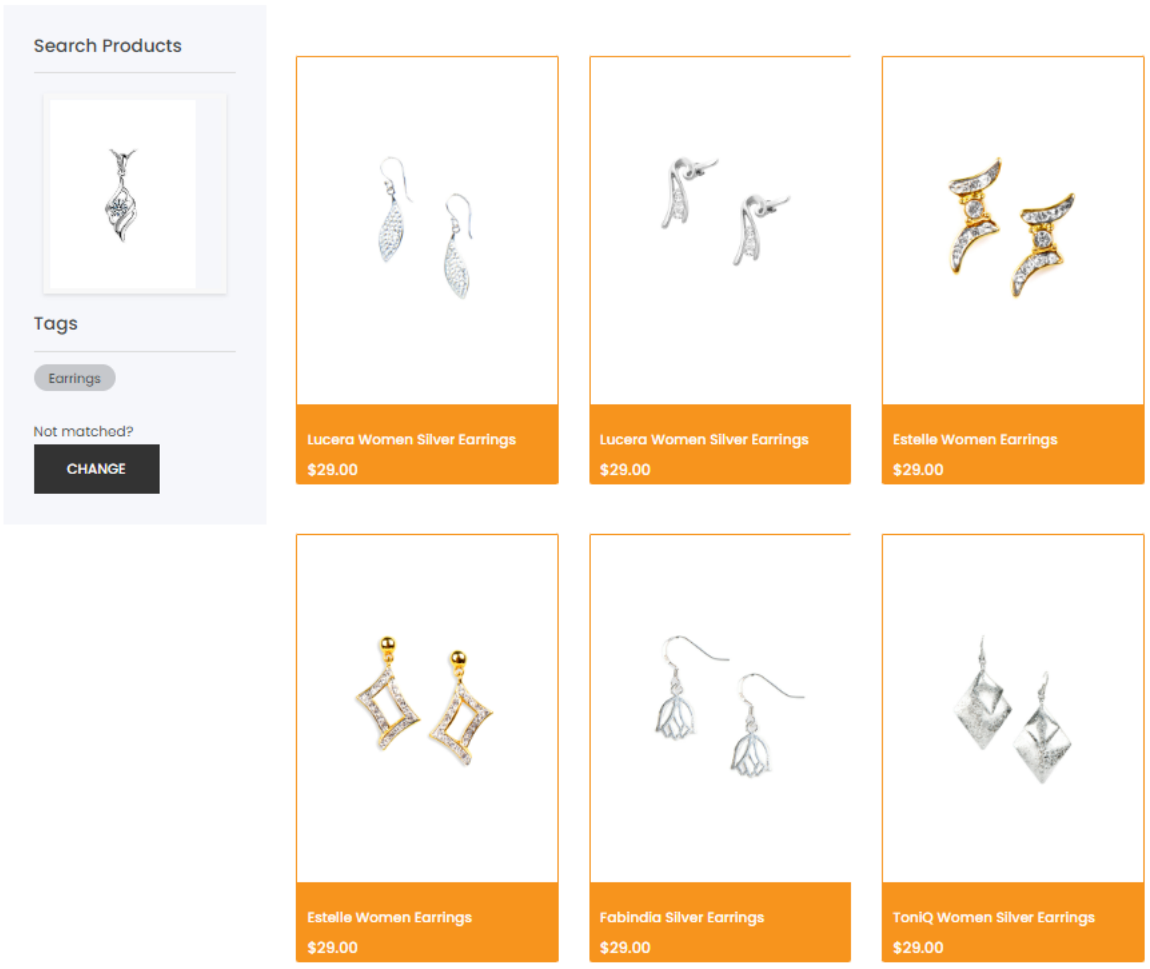


Figure 6.7: Top similar earrings after searching for an unknown image

Chapter 7

Conclusion And Future Work:

7.1 Conclusion:

The majority of the current recommended systems' attention is given to making similarity-based recommendations[1]. I implemented Resnet50 Pretrained model embeddings for comparing the similarity scores. I used cosine similarity for comparison. The more the cosine similarity score, the more similar the items are. I also used two classifiers for better performance. With two classifiers, the searching is almost instant for all the product barring internet speed. Then I used this model and also embedding for making the API. For that, I used the Flask library. Then I displayed its result through an angular UI. With it, we can search for two types of items. They have known items and unknown items.

7.2 Future Work:

7.2.1 Limitations:

1. I only built, tested, and ran it on PC. Mobile platforms are not considered here.
2. There might be some way to improve accuracy and inference time for unknown item recommendations.

7.2.2 Future Research:

1. Build it for mobile platforms.
2. I have planned to improve accuracy and inference time for an unknown item recommendations.
3. Extract current fashion trends from social media and apply them to fashion recommendation.

Bibliography

[1] M. Vartak and S. Madden, “Chic: A combination-based recommendation system,” in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, 2013, pp. 981–984.

[2] E. V. De Melo, E. A. Nogueira, and D. Guliato, “Content-based filtering enhanced by human visual attention applied to clothing recommendation,” in *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*, IEEE, 2015, pp. 644–651.

[3] 2016. [Online]. Available: http://www.ijcee.org/index.php?m=content&amp;c=index&amp;a=show&amp;catid=81&amp;id=1034.

[4] R. He and J. McAuley, “Vbpr: Visual bayesian personalized ranking from implicit feedback,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, Feb. 2016. DOI: 10.1609/aaai.v30i1.9973. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/9973.

[5] N. Zhang, Z. Zhu, X. Lian, and D. Liu, *Web-Age Information Management*, B. Cui, Ed. 17th International Conference, WAIM 2016, 2016, vol. 9658.

[6] K. Zhao, X. Hu, J. Bu, and C. Wang, *Deep style match for complementary recommendation*, 2017. DOI: 10.48550/ARXIV.1708.07938. [Online]. Available: https://arxiv.org/abs/1708.07938.

[7] X. Chen, H. Chen, H. Xu, *et al.*, 2019. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3331184.3331254.

[8] N. Dokoohaki, *Fashion Recommender Systems*. Springer, 2020.

[9] S. Chakraborty, M. S. Hoque, N. Rahman Jeem, M. C. Biswas, D. Bardhan, and E. Lobaton, “Fashion recommendation systems, models and methods: A review,” in *Informatics*, MDPI, vol. 8, 2021, p. 49.

[10] W.-H. Cheng, S. Song, C.-Y. Chen, S. C. Hidayati, and J. Liu, “Fashion meets computer vision: A survey,” *ACM Computing Surveys (CSUR)*, vol. 54, no. 4, pp. 1–41, 2021.

[11] M. KHALID, M. KEMING, and T. HUSSAIN, “Design and implementation of clothing fashion style recommendation system using deep learning,” *Rom J Inform Technol Autom Control*, vol. 31, no. 4, pp. 123–136, 2021.

[12] Y. Deldjoo, F. Nazary, A. Ramisa, *et al.*, “A review of modern fashion recommender systems,” *arXiv preprint arXiv:2202.02757*, 2022.

[13] V. W. Aneli, Y. Deldjo, T. D. Noja, D. Malitesta, and F. A. Merra, “Sigir ’21: Proceedings of the 44th international acm sigir conference on research and development in information retrieval,” in *A Study of Defensive Methods to Protect Visual Recommendation Against Adversarial Manipulation of Images*, pp. 1094–1103.