Athens University of Economics & Business
MSc Business Analytics
Course: Machine Learning & Content Analytics

Project
SHOP THE LOOK

Professors: H. Papageorgiou | G. Perakis
Students: S. Despotis P2822111 | T. Papailiou P2822128

August 2022

Contents

1. Introduction

1.1 Our Project Idea: “Shop The Look”

Every day, consumers waste a lot of time searching for clothes, especially when they know exactly what they want. Since fashion is about the look – combining multiple items together – we thought it would be a great idea to give users a faster tool to search for outfits online, rather than typing multiple keywords into search engines. In our case, the adage “A picture is worth a thousand words" was valid, since the images were the “key” solution to our problem. Due to the fact that people are familiar with snapping pictures (e.g., for their social media) and also pictures contain a wide variety of information, we focused on providing an easy to use solution to users: “Snap and Shop”. We believe that our new tool will provide many advantages for customers and online businesses.

Specifically, for customers, the “Shop The Look” tool, will give a personalized user experience, with a fully automated process of matching scenes to products. Having an app that recognizes the clothes of a photo and that searches for similar products in online stores, will be more practical and time efficient. Secondly, by pointing out relevant alternatives based on what they are looking at, customers could be inspired, and discover a much bigger variety of similar outfits. Thirdly, many times users know what they want, but they cannot find it online. By providing the user with that information, it will create more enjoyable customer journeys.

For online fashion businesses, the new tool will increase conversion and basket size. When shoppers are better engaged with their brand and its personalized style suggestions, they will likely also increase the frequency of purchases. Secondly, enabling consumers to buy the entire look in the picture with one click, removes the pain of complex human processes and their maintenance. So, companies could save time and costs from every party that is involved in apparel purchases. Thirdly, the companies can leverage huge amounts of data from users and create detailed reports (e.g., sales performance, user queries) and create personalized landing pages or campaigns.

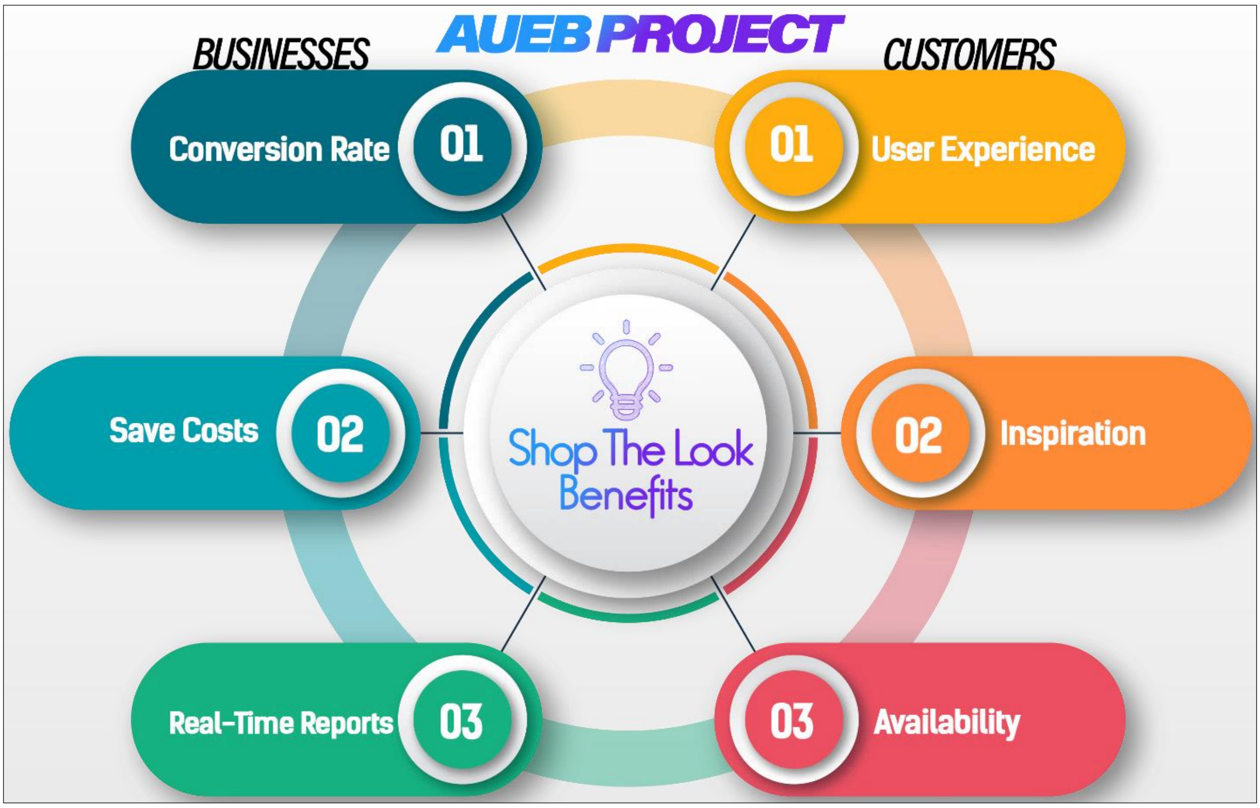


Figure 1-1 Benefits for Business and Customers

1.2 Our Vision & Goals

Our mission is to help both customers and digital fashion businesses grow, by providing them an easy to use solution with a visual search system that fully automates the process of matching products to scenes. Having that as a baseline, we had to utilize the latest machine learning tools and build an efficient algorithm that collects results that are most similar to the uploaded pictures. Building such an algorithm poses great challenges, such as how to deal with variations in image quality, lighting, background, different human poses, gender identification and finding the right product in a large database in real-time.

Our vision is to be the best visual search in the fashion industry. In the future, it would be advantageous to combine our "Shop The Look" algorithm with AR technology to provide instant interactive experience solutions or size help features with a "Fit Quiz" to help the shoppers find the right fit for clothing & shoes without measurements or size charts.

Figure 1-2 Our Vision for the Shop The Look

Finally, the COVID-19 pandemic has accelerated the shift towards a more digital world and triggered changes in online shopping behaviors that are likely to have lasting effects. This means that for many brands, their e-commerce strategy must be fitted to the rising digitalization and personalization of consumers' shopping habits and demands. Therefore, our solution follows the trends of our era, where ecommerce is continuously growing. The following research from online businesses Glami.gr and Skroutz.gr show that consumers' interest in online fashion has been increasing in the last few years. The average consumption price for Skroutz fashion's products for 28 December 2021 to 28 March 2022 is presented below:

Γενική κατηγορία	Προϊόν	Μέση τιμή κατανάλωσης
1. Μόδα & Αξεσουάρ	Ανδρικά μπουφάν	68€
	Γυναικείες τσάντες	65€
	Ανδρικά φούτερ	43€

Reference: Skroutz.gr
28 December 2021 - 28 March 2022

Figure 1-3 Skroutz Average Consuption Price

Figure 1-4 Fashion Ecommerce Review for 2021 by Glami

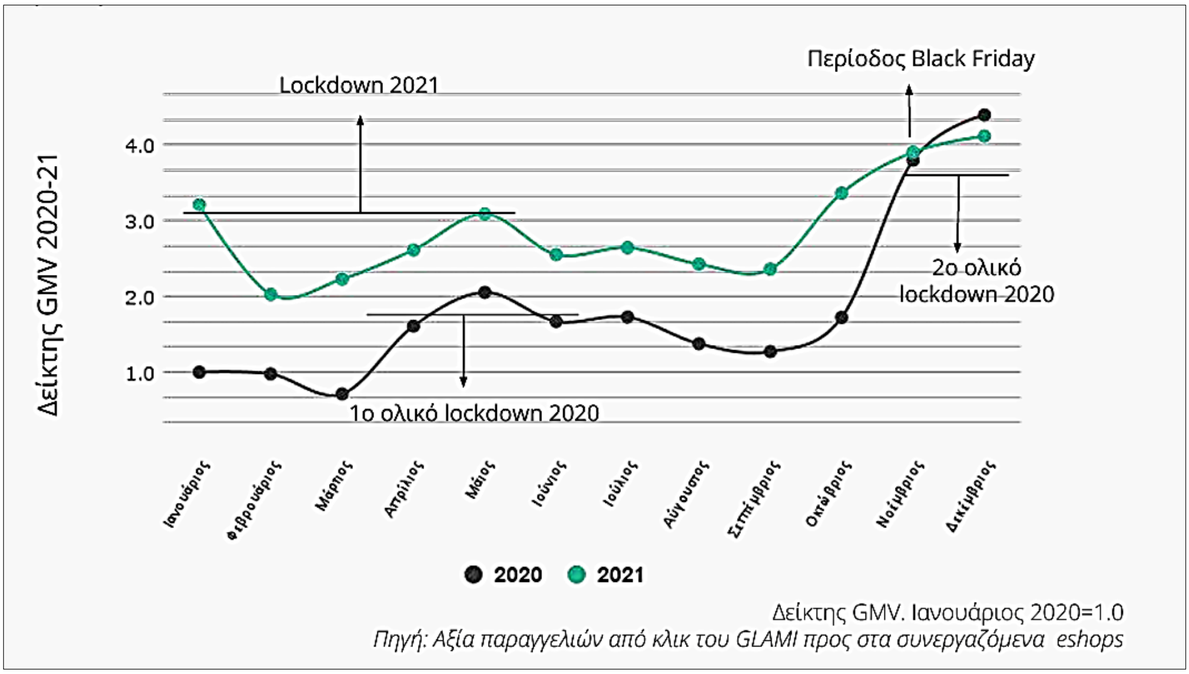


Figure 1-5 GMV Metric for 2021 by Glami

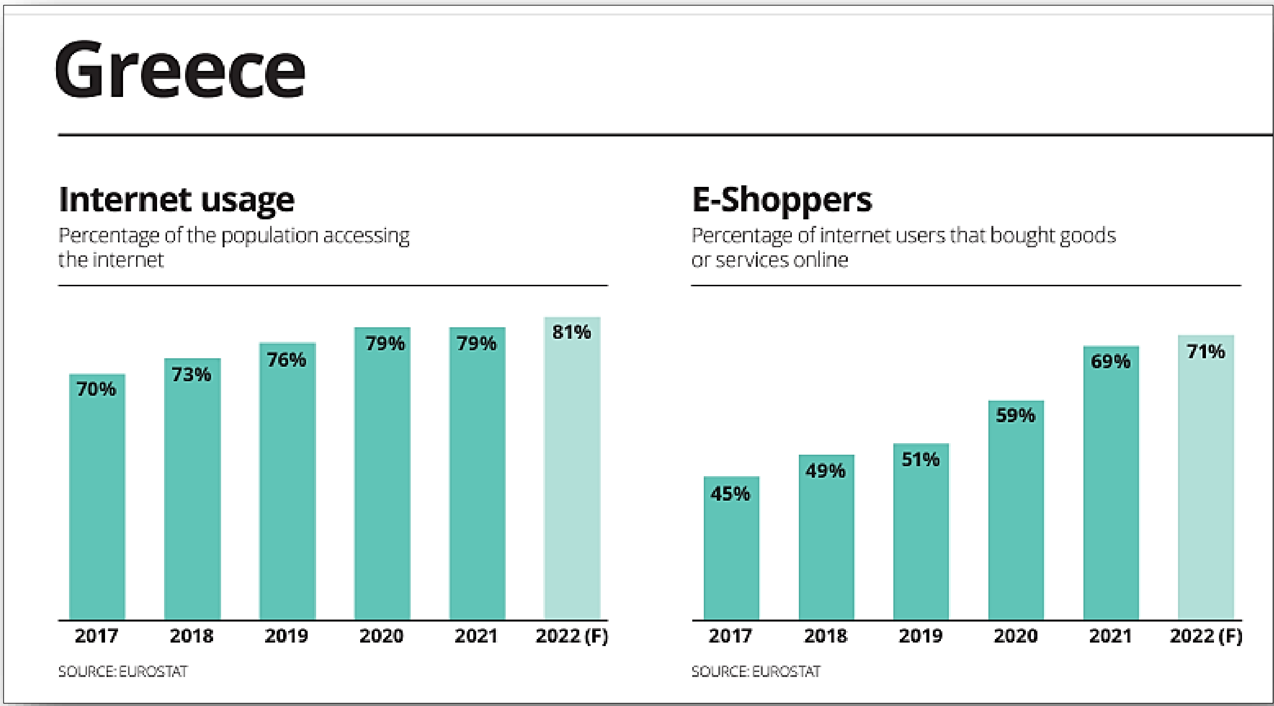


Figure 1-6 Ecommerce in Greece by Eurostat

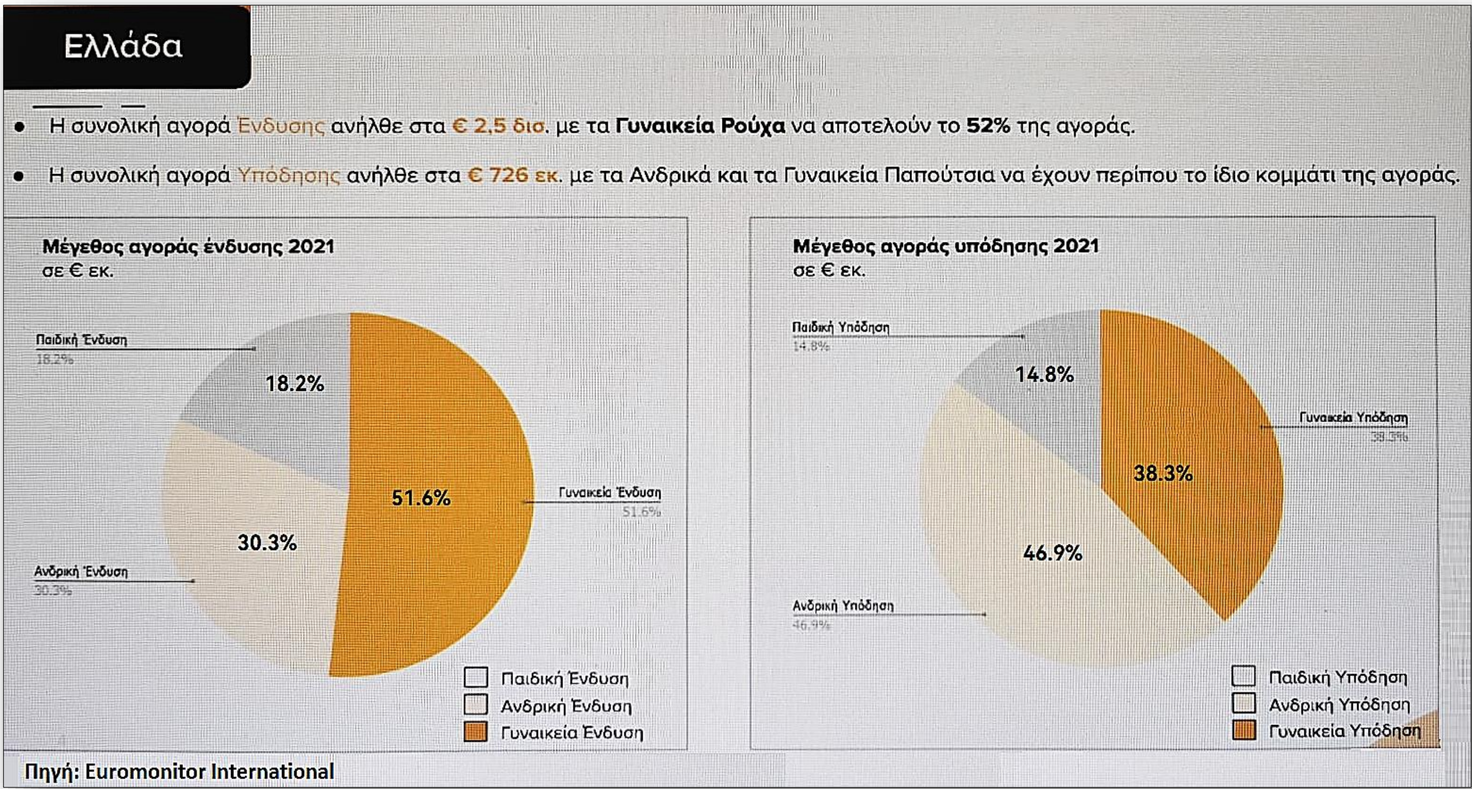


Figure 1-7 Digital Fashion in Greece by Euromonitor International

2. Methodology

2.1 Proposed Recommendation System

As we saw in the previous section our proposed solution (Complete the Look: Scene-based Complementary Product Recommendation, n.d.). To achieve the desired result, we had to combine different modules with different tasks (Buy Me That Look: An Approach for Recommending Similar Fashion Products, n.d.). We will begin by determining the gender (Gender Classification) of the person in the query image. We used the dataset (Men / Women Classification Kaggle Dataset , n.d.) from Kaggle, which contained a considerable number of images for men and women. ResNET50, which is a model of CNN (Convolutional Neural Network) was used for the 2-class classification task. Next, considering the gender was determined, we used the Yolo Object Detection module (Version No.5) using a dataset from (Roboflow Dataset, n.d.), to identify objects from these specific categories: top-wear, bottom-wear and footwear. We used that module, in order to identify the objects in the images and localize the objects with bounding boxes. Once the bounding boxes are obtained for a certain object, that part of the image is cropped and the embeddings for the cropped image are obtained, which will help in identifying similar products from the catalogue/database of products whose semantic embeddings are already known. Last but not least, we trained three Siamese models for generating the embeddings for the catalogue of images, in order to automatically recommend new products. The directory structure that we will use for this project is visualized in the schema below.

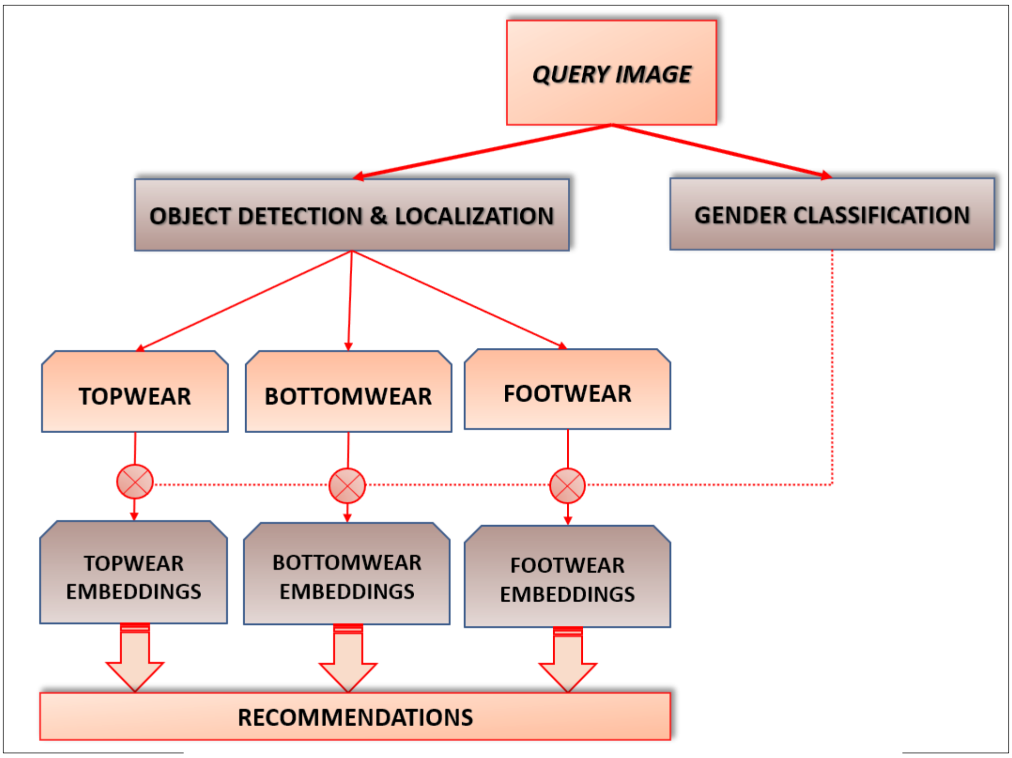


Figure 2-1 Shop The Look Directory Structure

2.2 Data Collection

In order to make our Fashion Recommendation Engine perform properly, we had to find a database consisting of fashion products to recommend. Since we both work at Skroutz.gr (E-commerce company), we retrieved a sample fashion database from our company. Specifically, we used Skroutz's database to query images using specific item categories (women's topwear, women's footwear, women's bottomwear, men's topwear, men's footwear, men's bottomwear) and later we saved the product details ('product_id', 'description', 'brand','gender', 'category', 'category_name') for the results obtained. This way we can categorize products into various sections which would make it easier while recommending items. Last but not least, we uploaded the database to Google Drive, and we imported the drive to our Collab project.

product_id	description	brand	gender	category	category_name	code	path
30098097	A353 Βραδινό Γυναικείο Top Μαύρο	Awama	women	topwear	Γυναικείες Μπλούζες	wtw	women_topwear/wtw_30098097.jpeg
32265292	A288 Off-Shoulder Γυναικείο Top Ροζ	Awama	women	topwear	Γυναικείες Μπλούζες	wtw	women_topwear/wtw_32265292.jpeg
29431617	1666 Αμάνικη Γυναικεία Μπλούζα Γκρι	Philosophy Wear	women	topwear	Γυναικείες Μπλούζες	wtw	women_topwear/wtw_29431617.jpeg
35677304	1813 Μακρυμάνικο Γυναικείο Βαμβακερό Πουλόβερ ...	Philosophy Wear	women	topwear	Γυναικείες Μπλούζες	wtw	women_topwear/wtw_35677304.jpeg
31340241	12200264 Γυναικείο Βαμβακερό Πουλόβερ Blue Iolite	Jack & Jones	women	topwear	Γυναικείες Μπλούζες	wtw	women_topwear/wtw_31340241.jpeg

Table 2.1 Skroutz Database Sample

2.3 Exploratory Data Analysis

Now that we've gathered the database/collection from Skroutz.gr, let's do some exploratory analysis on it. The dataset consists of 9600 fashion products in various categories, displayed in the following image.

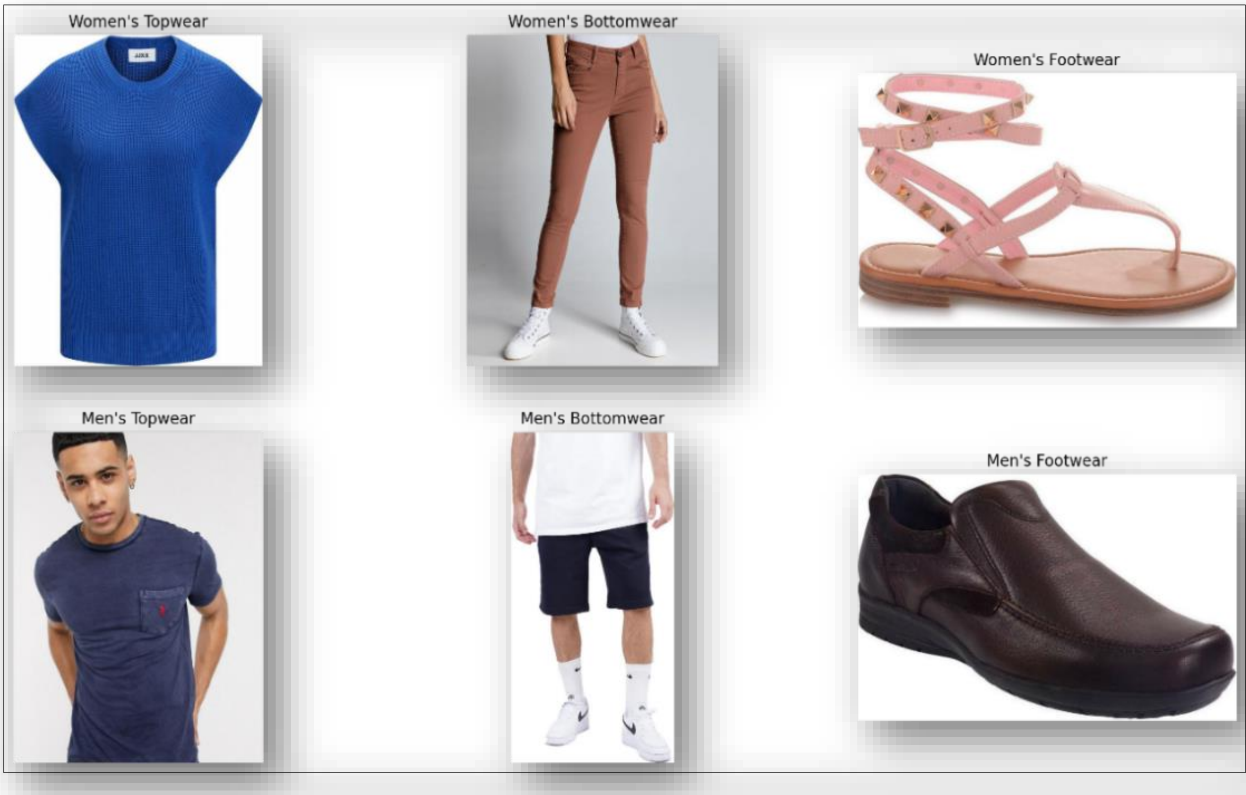


Table 2.2 Skroutz Database Fashion Categories

The fashion items are almost equally split among the two genders, where we have 5000 men's products and 4600 women's items. The plot below shows the product split for men and women.

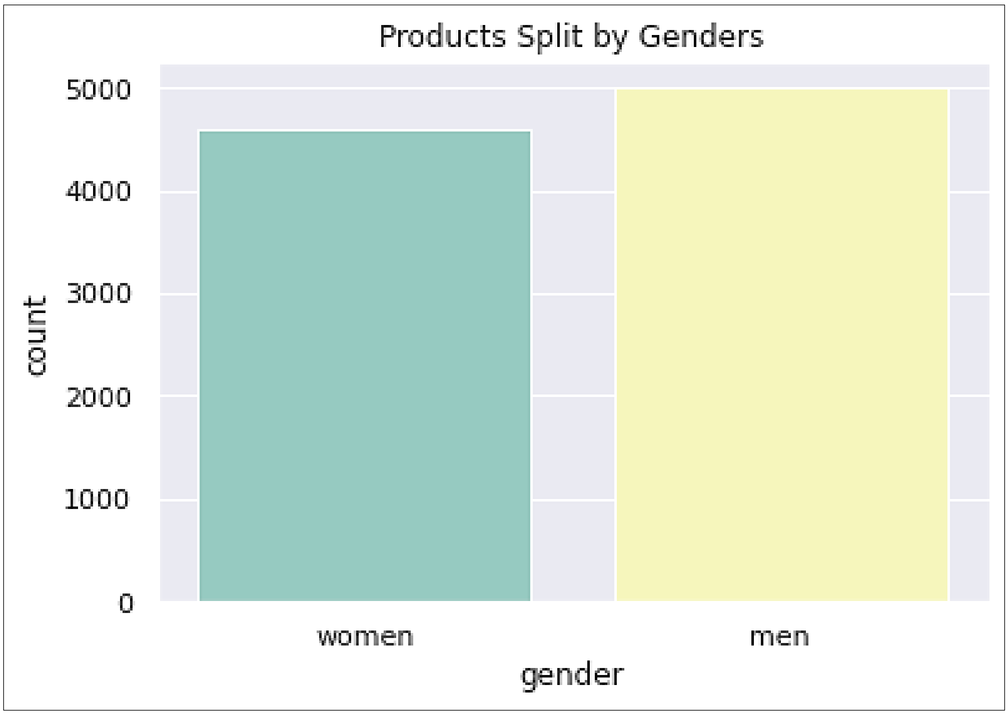


Table 2.3 Products Split by Genders

Bellow we can se the item distribution across fashion categories. Specifically, we have Top wear ~ 4000, Bottom wear ~ 3600 and Footwear ~ 2000.

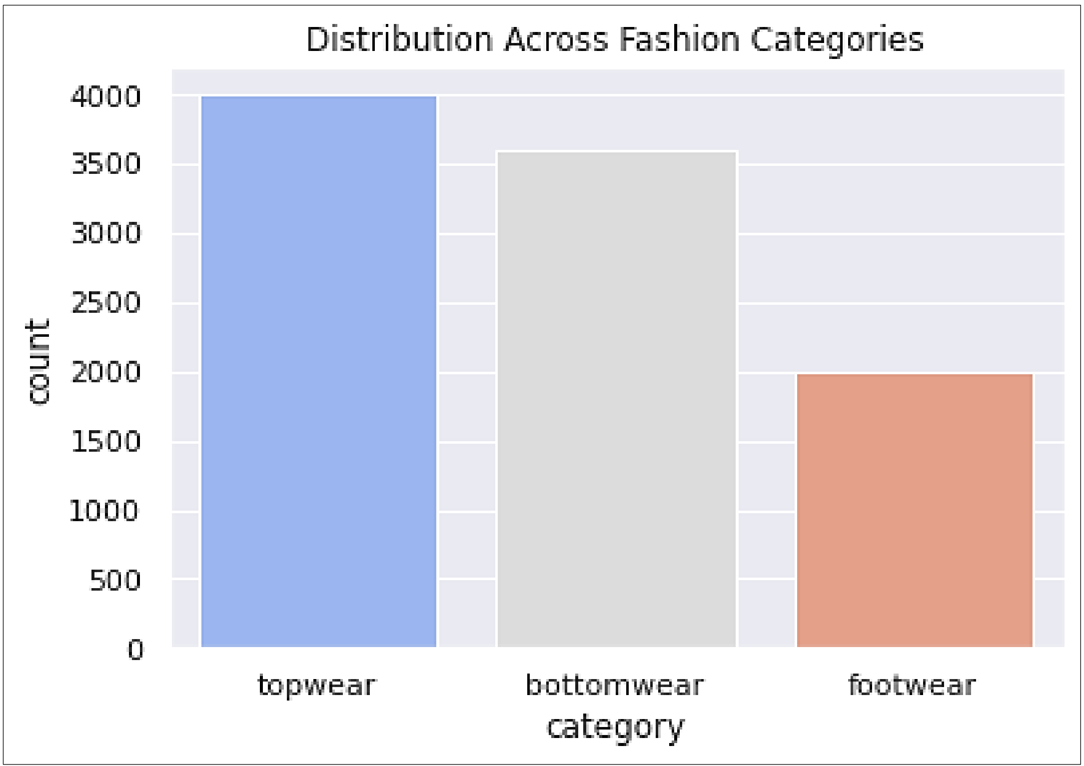


Figure 2-2 Bar Plot with Distribution Across Fashion Categories

Next, we can observe the distribution across product's categories in the dataset. Men's bottom wear has 1000 items each, whereas Women's and Men's top wear have 400 products each.

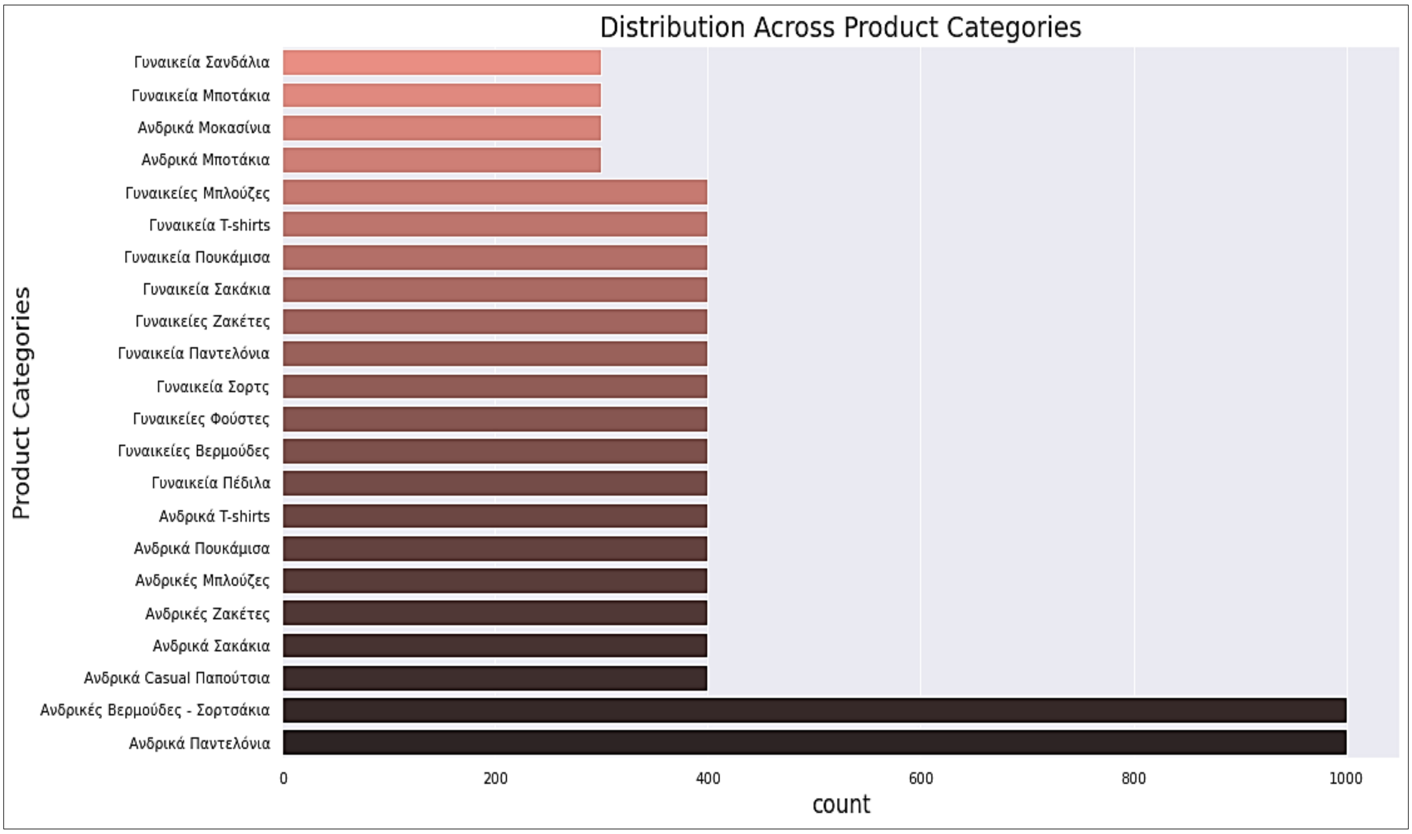


Figure 2-3 Bar Plot with Distribtuion Across Product Categories

As far as the brands, we can see that Jack & Jones and Ralph Lauren are the most common brands in the dataset, with over 400 products each one.

Top 25 Most Common Brands in the Dataset	
Jack & Jones	491
Ralph Lauren	403
BodyTalk	365
Pepe Jeans	343
Edward Jeans	315
Famous Shoes	240
Sogo	186
Body Action	170
Tom Tailor	162
Adidas	161

Table 2.4 Top 25 Most Common Brands in the Dataset

2.4 Gender Classification

As we mentioned earlier, our first module in the Fassion Recommendation System was the Gender Classification. This process determines the person's gender e.g. male or female that appears in the query image. Therefore, this process allows us to generate and query embeddings for subset of our catalogue and shows only relevant reccomendations to the user based on his/her gender. So, we used (Men / Women Classification Kaggle Dataset , n.d.) from Kaggle platform[1]. This dataset contains 3354 pictures (jpg) of men (1414 files) and women (1940 files) with full body and upper body images for both genders. The training dataset of full body images of models wearing various fashion items, was necessary since we would have similar query images.

[1] https://www.kaggle.com/datasets/playlist/men-women-classification

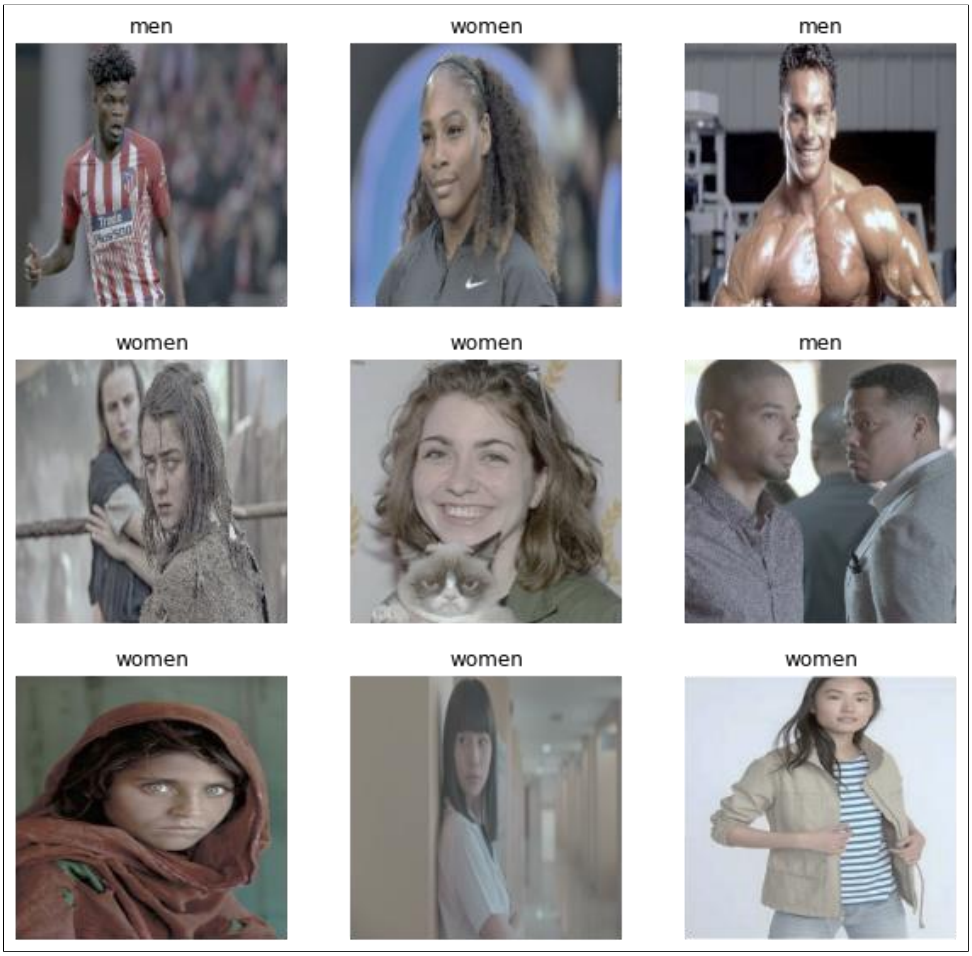


Figure 2-4 Sample of Kaggle Dataset "Men/Women Classification

To classify the “Man/Woman” dataset, we used a convolutional neural network named “ResNet-50”. It’s worth mentioning that we experimented with different ResNet models such that “ResNet-101” and “ResNet-152”, but the ResNet-50 combined with our hyperparameter tuning gave better results. Furthermore, ResNet-50 is a convolutional neural network that is 50 layers deep. You can load a pretrained version of the network trained on more than a million images from the “ImageNet” database. The network has an image input size of 224-by-224. Therefore, we used ResNet50 backbone pretrained with 'ImageNet' weights attached to a fully connected network that outputs the probability scores for the two classes. Due to the fact that we would use natural images (*like* “ImageNet’s”) and the model was used to extract relevant features, which can be used to classify the input, we decided to freeze all the layers from our convolutional neural network. Also, since the task was simple, we wanted to decrease computation time and avoid overfitting problems.

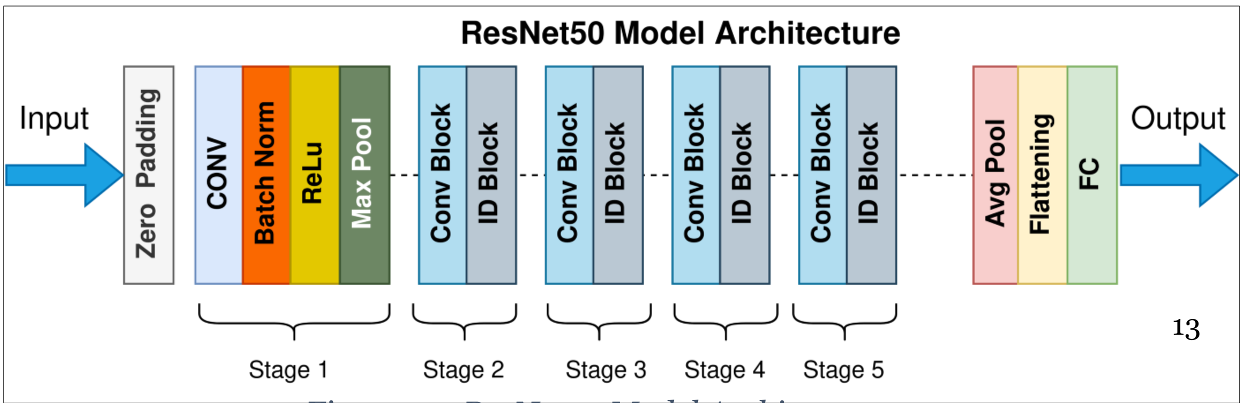


Figure 2-5 ResNet50 Model Architecture

As mentioned before in order to achieve better results in terms of loss and accuracy we experimented with the hyperparameters. Specifically, at first, we had high overfitting during resnet50 transfer learning and we made changes to the dropout, batch size, optimizer and learning rate. The final parameters we used:

- **Number of training epochs**: 100 (number of complete passes through the training dataset)
- **Batch size**: 32 (number of samples processed before the model is updated)
- **Learning rate**: 0.001 (controls how much to change the model in response to the estimated error each time the model weights are updated)
- **Dropout rate**: 0.6 (dropout is only used during the training of a model and is a powerful regularization technique)
- **Target shape:** (224, 224)
- **Optimizer:** SGD

From the bellow plot and table for the model used for classification we can draw useful conclusions.

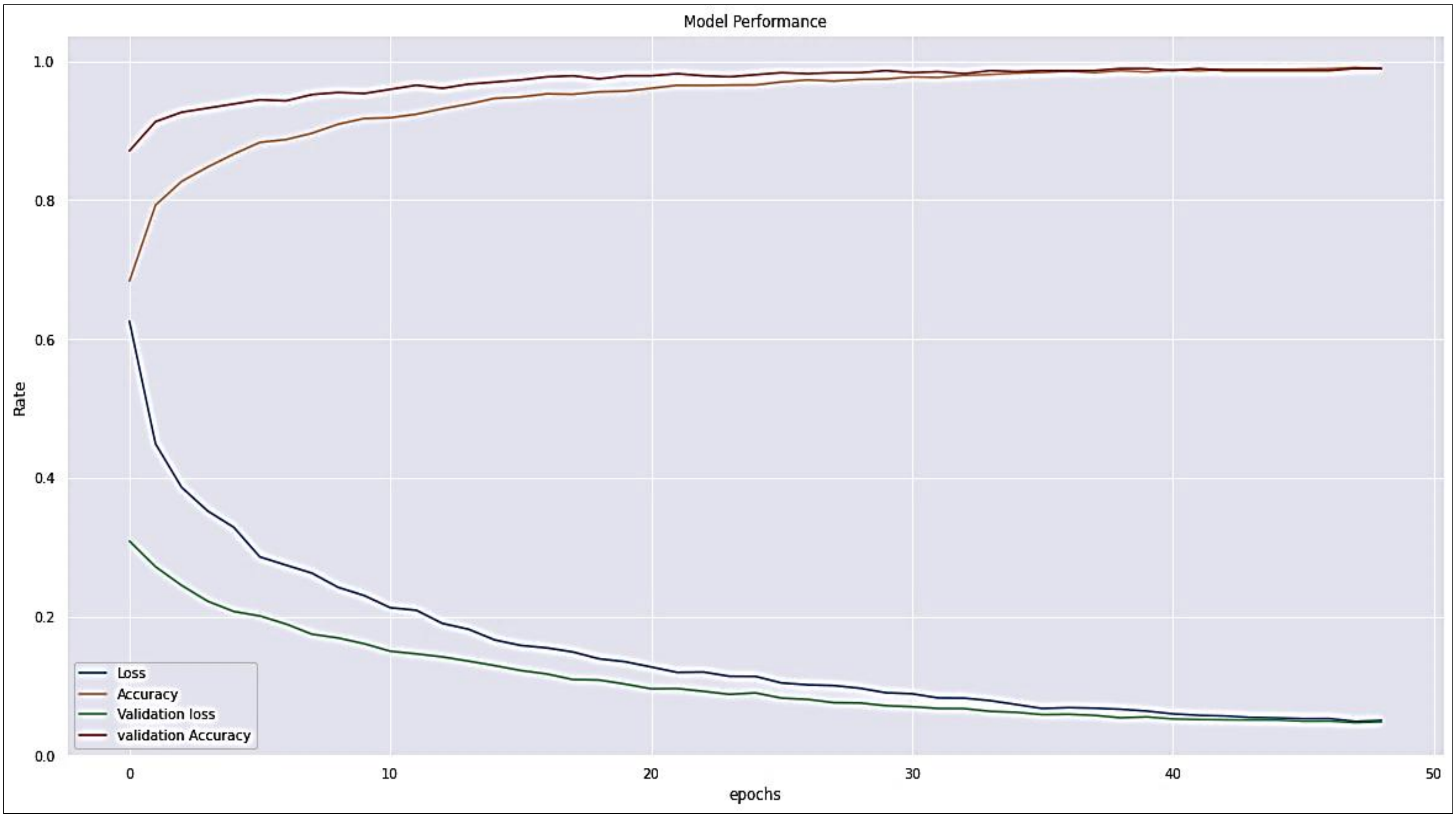


Figure 2-6 The loss and accuracy of our deep neural network while training and validating

Model Performance	
Training Accuracy ResNet	0.99
Training Loss ResNet	0.06
Validation Accuracy ResNet	0.99
Validation Loss ResNet	0.05

Table 2.5 Model Performance Metrics

We can see that the training and validation errors are close to each other. As the learning continues, they are both decreasing, so our model is still getting better at extrapolating. Moreover, our model was able to learn more specifics on the training data and do better there, which is normal because all information relevant for the training data will not be relevant for the validation set. However, the training loss rate (0.06) and the validation loss (0.05) are close. In terms of metric accuracy, both achieve the rate of 0.99.

Next, we proceed by experimenting with our model performance. We have chosen 32 images of the validation set to check and we plotted a confusion matrix to compare the actual values to the predicted values.

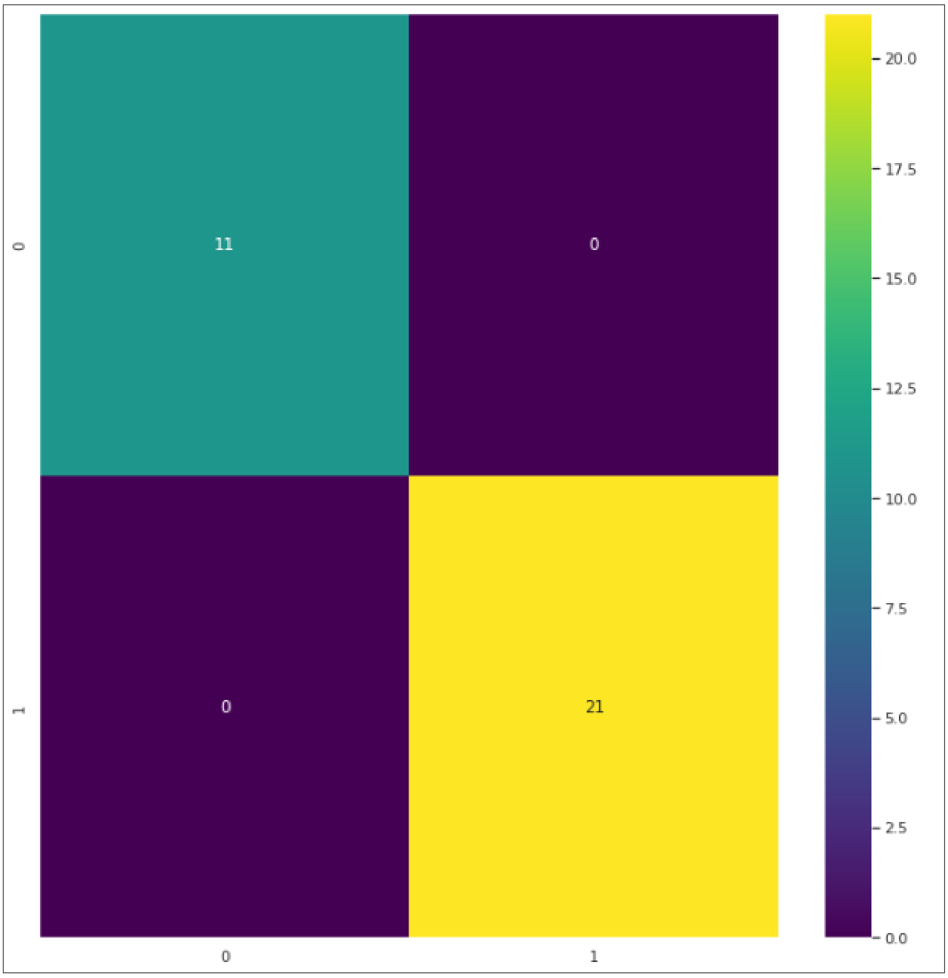


Figure 2-7 Confusion Matrix to Compare Predicted Values for Random Batch

As we can see, our model in terms of predictions for a random images batch is performing well. This process was repeated with many other batches in order to have a general “view” of our model. We will now print a sample of the model’s predictions in order to be sure with images and verify is the predictions are accurate.

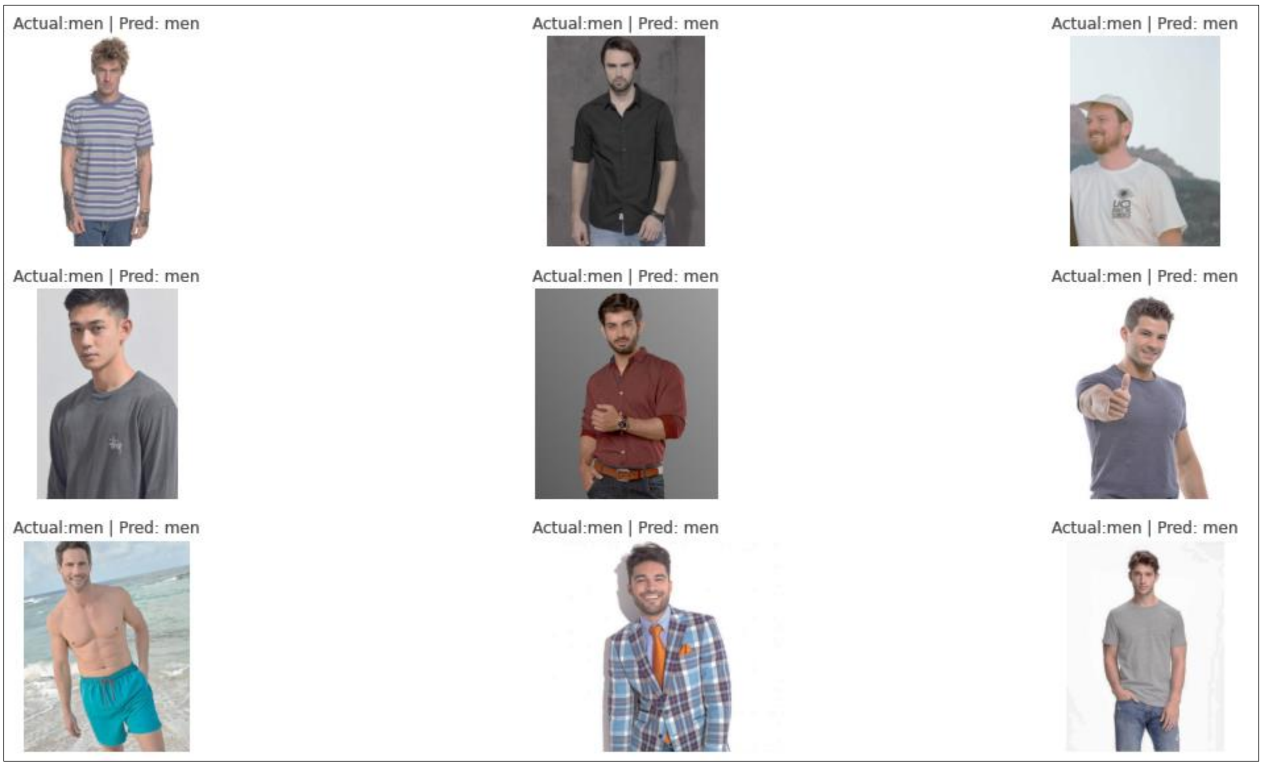


Figure 2-8 Predictions Results for Gender (Actual vs Predicted)

2.5 Object Detection and Localization

In order to recommend similar products, we'll first need to detect all the products in a given image, and for generating the embeddings we'll need to crop out the detected product from an image.

Object detection is one of the most interesting aspects of Computer Vision. Most object detection systems have some kind of trade-off between inference speed and detection accuracy. In order to evaluate our object detection model, we'll use a metric called 'mAP' or Mean Average Precision (Breaking Down Mean Average Precision (mAP), n.d.). For object detection whether a detection is considered correct or not depends on the IoU (Intersection over Union) threshold. The IoU score for a predicted bounding box and actual bounding box is defined as following:

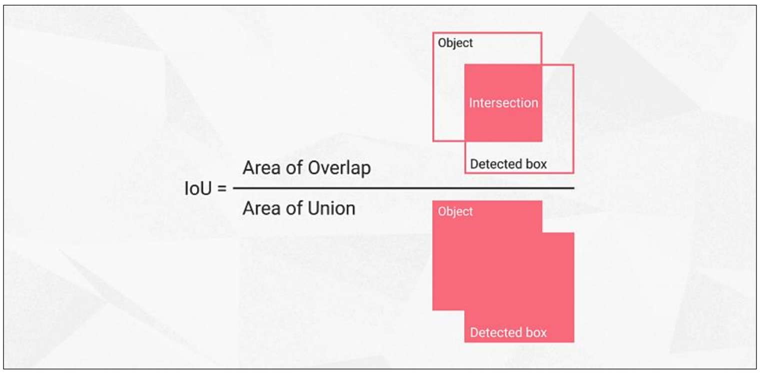


Figure 2-9 IoU Score for a predicted bounding box and actual bounding box

For a pre-defined IoU threshold we can define if a detection was accurate or not if the IoU is greater than the thresold. Based on the detections Precision and Recall is defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \ \& \ \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- Where TP = True Positive,
- FP = False Positive & FN = False Negative

The average precision is calculated by using the Area Under the Precision and Recall Curve.

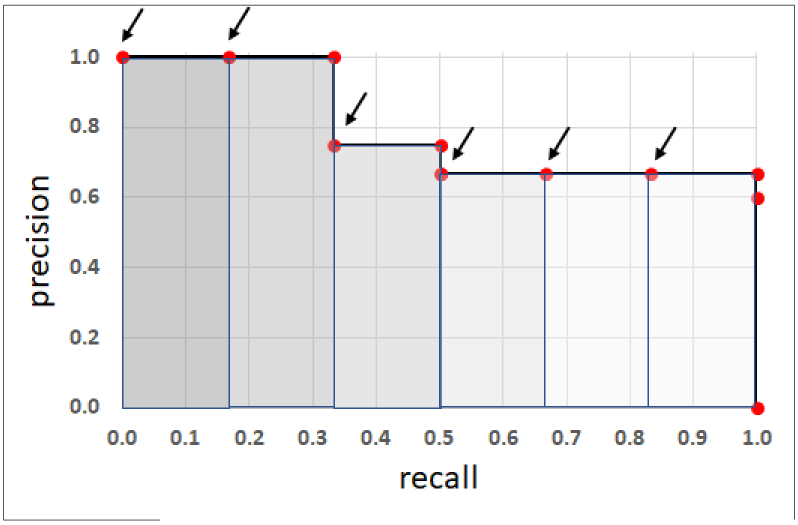


Figure 2-10 Average Precision

The Mean Average Precision is calculated by taking mean of the Average Precision values over different values of threshold, for example in the COCO Primary Challenge the mAP was calculated by averaging the AP scores over a range of IoU thresholds from 0.5 to 0.95 with a step size of 0.05 and finally taking the mean of the AP scores over all the classes (Breaking Down Mean Average Precision).

Now we need to choose among various methods for object detection to apply to our solution. (Object detection: speed and accuracy comparison, n.d.) compares various object detection model's performance against the MS COCO Dataset that contains 80 classes. We'll look at the Inference speed and mAP scores. Below is a plot showing the highest and lowest Frames Per Second (FPS) values reported in the respective papers.

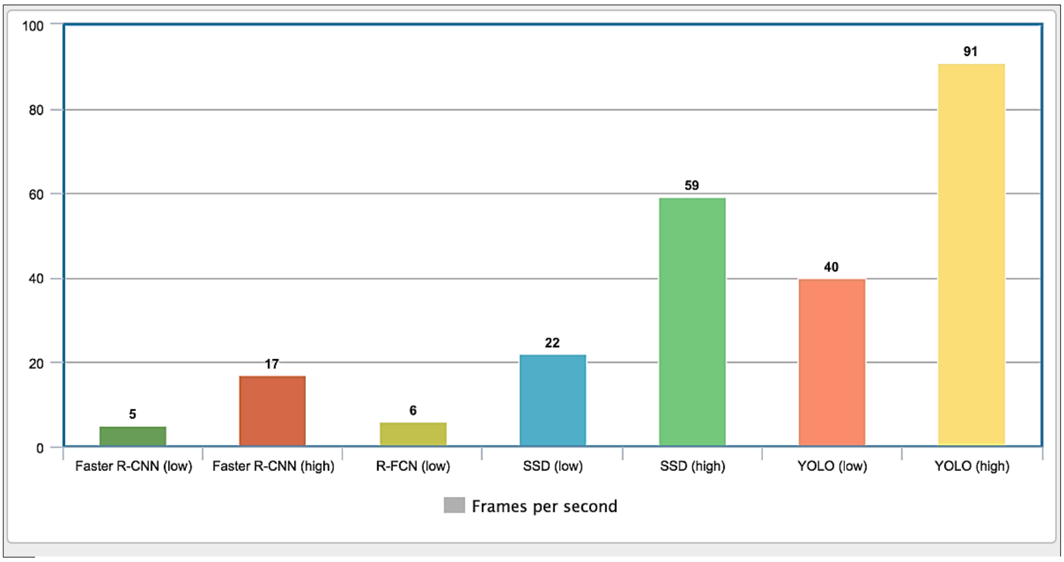


Figure 2-11 Highest and lowest Frames Per Second (FPS) values

It is clear that YOLO object detection is faster by a huge margin compared to other object detection methods. Now let's look at mAP scores for various methods against the MS COCO Dataset.

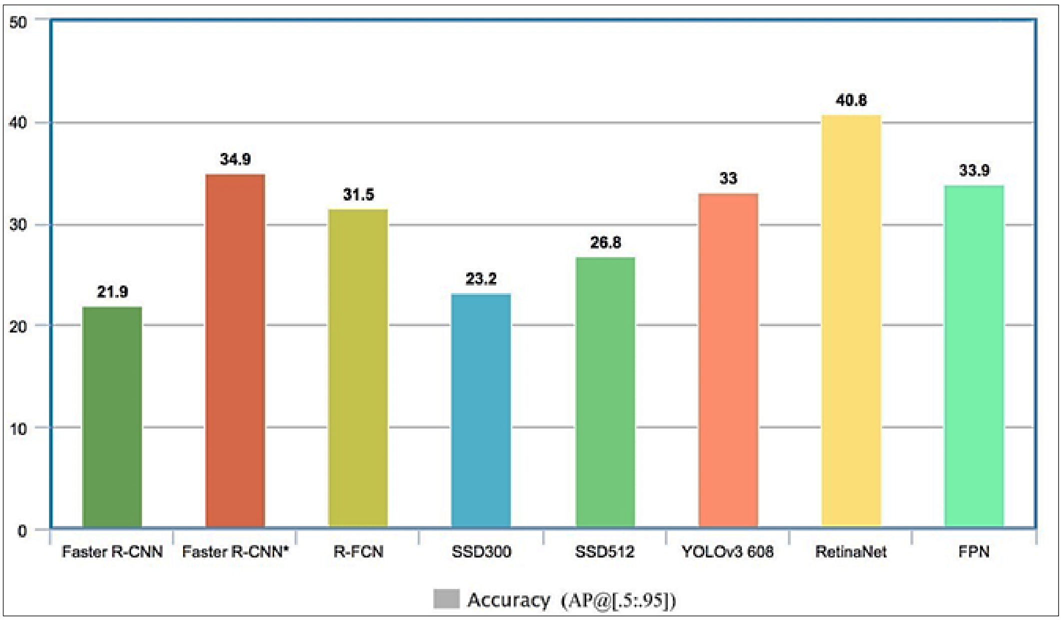


Figure 2-12 mAP Scores for various methods against the MS COCO Dataset

From the plot 1-12, we can see that RetinaNET has the highest mAP score of 40.8 followed by Faster-RCNN, FPN and YOLO. We can safely assume that YOLO will provide us with the right balance of Accuracy and Speed for our task of Fashion product detection.

Hence, we'll use YOLO object detection (You Only Look Once: Unified, Real-Time Object Detection, n.d.) for detecting fashion products in three categories - top wear, bottom wear, and footwear. YOLO algorithms divide a given input images into the SxS grid system. Each grid is responsible for object detection. Now the Grid cells predict the boundary boxes for the detected object. For every box, we have five main attributes: x and y for coordinates, w and h for width and height of the object, and a confidence score for that the box containing the object. We have followed (How to Train YOLOv5 On a Custom Dataset, n.d.) for training an object detector using your custom dataset.

For training the custom Object Detection model we'll use Transfer Learning. We'll load YOLOv5s (s stands for small variant of YOLOv5) (ultralytics/yolov5, n.d.) weights trained on the COCO Dataset and use transfer learning to fine-tune the model for our use case.

We have used (Roboflow Dataset, n.d.), of 180 images (135 train and 45 validation) with annotated top wear, bottom wear and footwear. We were able to achieve a mAP score of 0.99 @ 0.5 IoU threshold and a a mAP score of 0.69 @ 0.95 IoU threshold on a test set of 45 images.

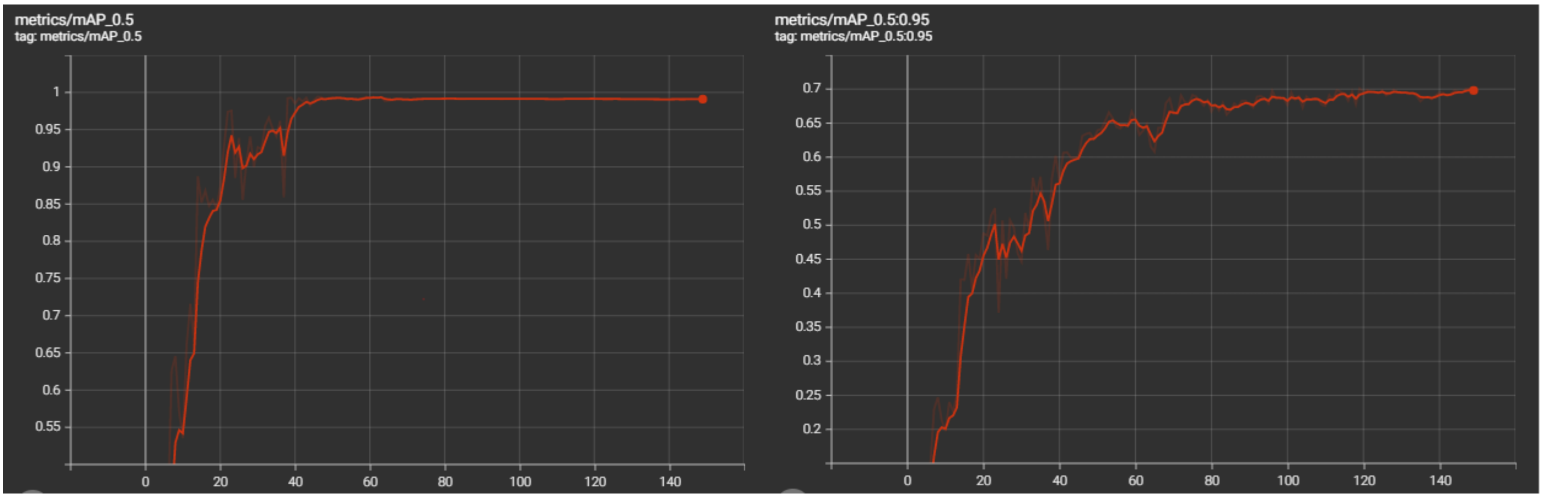


Figure 2-13 Training Results

Below is a more detailed description of the metrics and losses of our model. It is worth mentioning that both Precision, which measures how many of the bounding boxes are correct and Recall, which measures how much of the true bounding boxes were correctly predicted are at high levels, which means that the majority of the images are labelled correctly

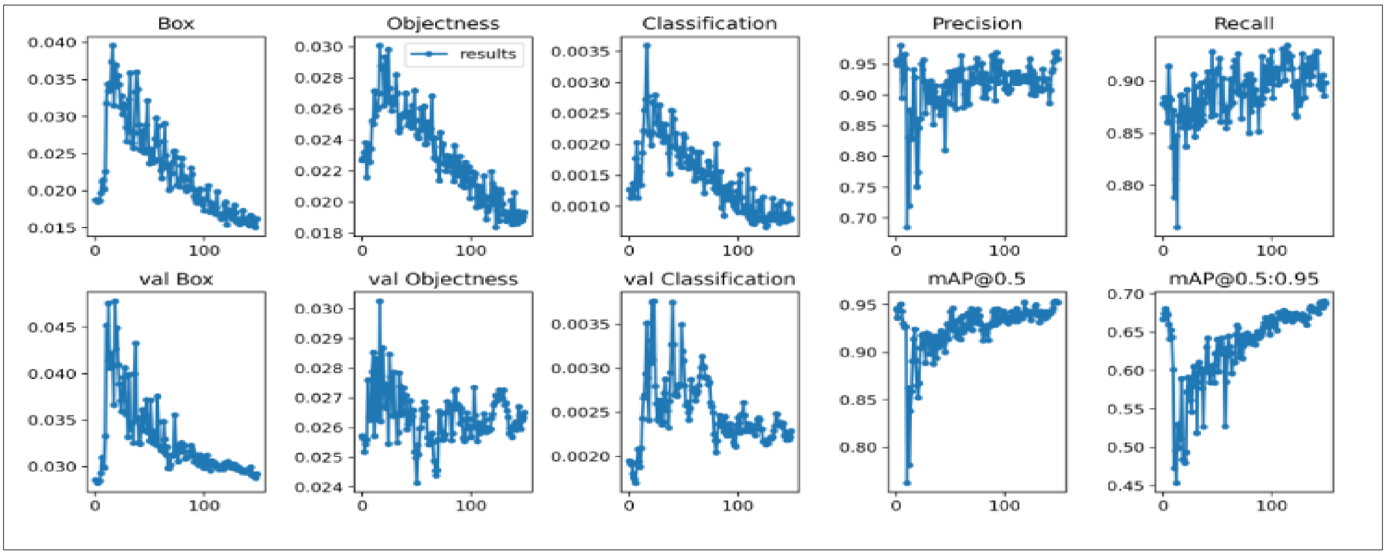


Figure 2-14 Training Results

Here are some object detection results on top wear, bottom wear, and footwear detection.

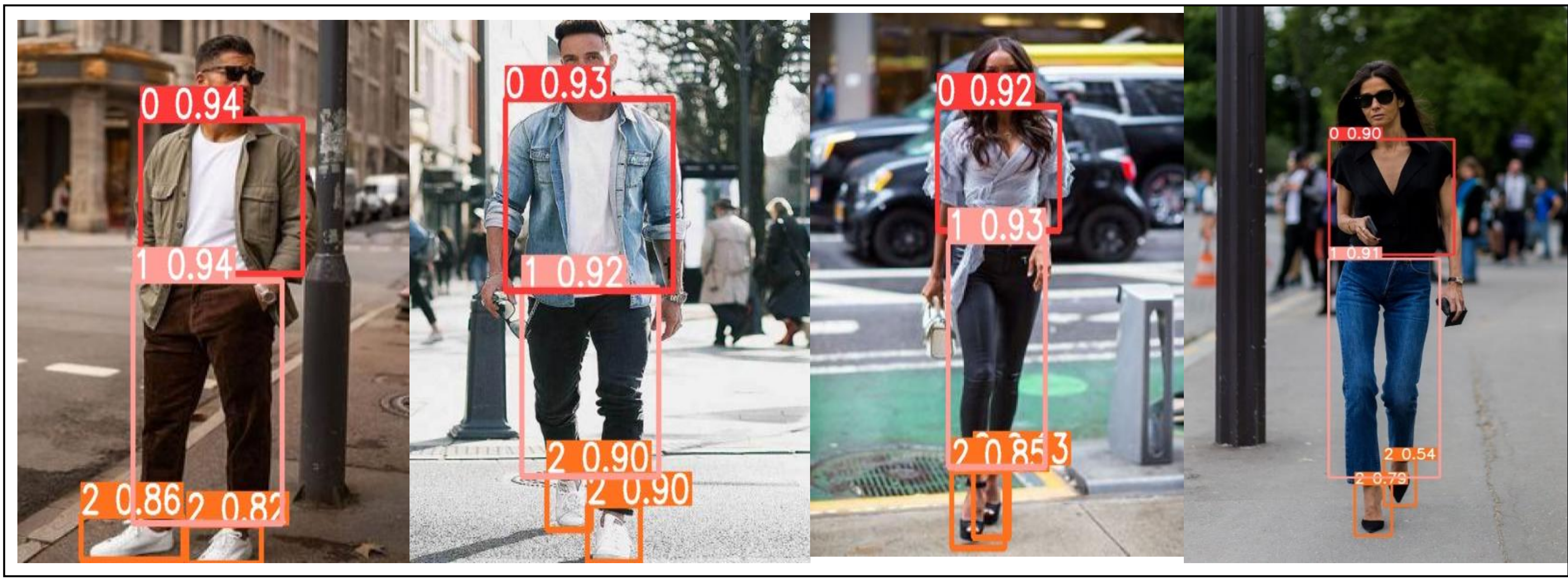


Figure 2-15 Object Detection Results

As a final step we have created a function to crop the query image based on the output of the object detection model:

Figure 2-16 Output from Object Detection with Query Image

2.6 Embedding Generation

We use embedding generation to represent images/products such that similar ones are grouped together whereas dissimilar ones are moved away so that in order to retrieve products that are similar to the products present in the query image. There are various ways to calculate similarity between images after they are converted into a vector with n-dimensions. Cosine similarity and Euclidean distance are two examples. We can define Cosine similarity and Euclidean distance as

$$\text{Cosine Similarity}(x_q, x_i) = \frac{x_q^T x_i}{||x_q|| \cdot ||x_i||}$$

$$\text{Squared Euclidean Distance} = (x_q - x_i)^T (x_q - x_i)$$

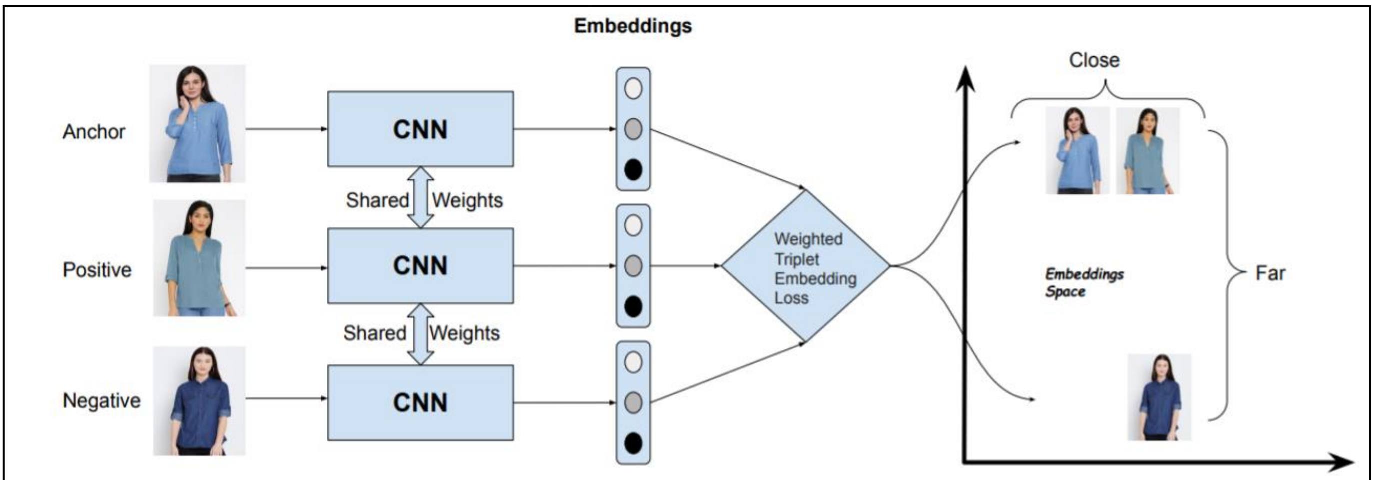
$x_q = \text{query image}$

$x_i = i^{th}$ image from the catalog

$x_q, x_i \in \mathbb{R}^n$

Figure 2-17 Cosine Similarity and Euclidean Definition

In order to convert images to n-dimensional vector representations of the images, we'll use a special type of Neural Network called Siamese Network. This network which takes in three inputs - anchor, positive and negative, such that anchor, and positive inputs are similar whereas anchor and negative inputs are dissimilar. The output is three n-dimensional vectors representing anchor, positive and negative inputs generated by a CNN backbone. This network requires a special kind of loss function that minimizes the distance metric between anchor and positive while maximizing the distance metric between anchor and negative hence we'll need an intermediate layer that computes the distance between anchor and positive outputs and anchor and negative outputs. One such loss is the triplet loss which in our case we've defined as –



Where D is the distance function used in the previous layer

$$\mathrm{L}(\mathrm{a}, \mathrm{p}, \mathrm{n}) = max(D(a,p) - D(a,n) + margin, 0)$$

Figure 2-18 Triple Loss Definition

In order to train the network above we will use "Shop The Look" dataset, which is taken from (Wang-Cheng Kang, Eric Kim, Jure Leskovec, Charles Rosenberg, Julian McAuley (2019). Complete the Look: Scene-based Complementary Product Recommendation, n.d.) It provides a dataset for fashion with outdoor scene and product pairs. Product is an image of a product in professional setting whereas a scene is the image of the same product but in casual or non-professional setting. Each dataset contains the scene-product pairs in the following format, where scene and products are encoded with a signature that can be converted to an URL by using a function provided in the official GitHub repo (Shop the Look Dataset, n.d.).

```
Example (fashion.json):
{
    "product": "0027e30879ce3d87f82f699f148bff7e",
    "scene": "cdab9160072dd1800038227960ff6467",
    "bbox": [
        0.434097,
        0.859363,
        0.560254,
        1.0
    ]
}
```

Figure 2-19 Conversion of Signature to Url with a Function

We can easily create anchor and positive pairs from this dataset and for generating negative samples we can randomly select scenes from the dataset which is not same as the positive sample. Let’s see an example of Anchor, Positive, and Negative pairs for Topwear, Bottomwear, and Footwear.

- ✓ **Top wear**

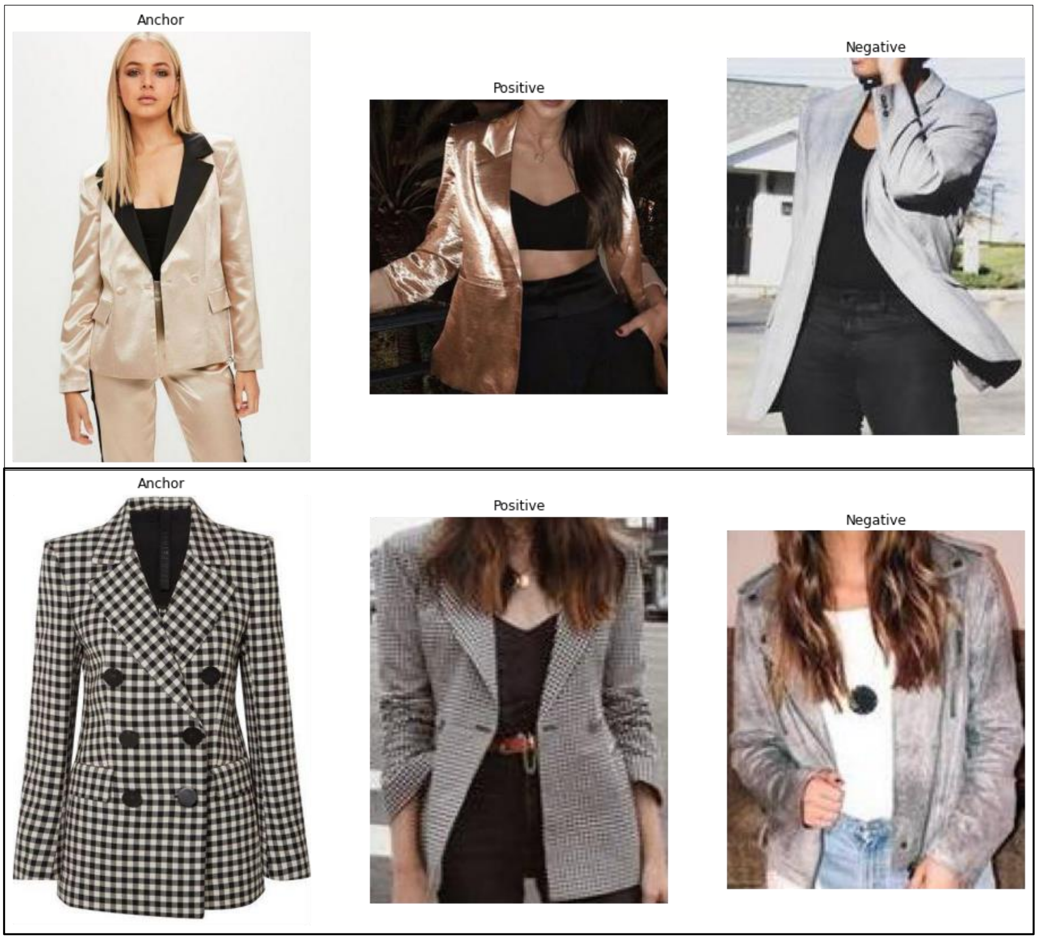


Figure 2-20 Anchor, Positive, and Negative pairs for Topwear

✓ Bottom wear

Figure 2-21 Anchor, Positive, and Negative pairs for Bottomwear

✓ Footwear

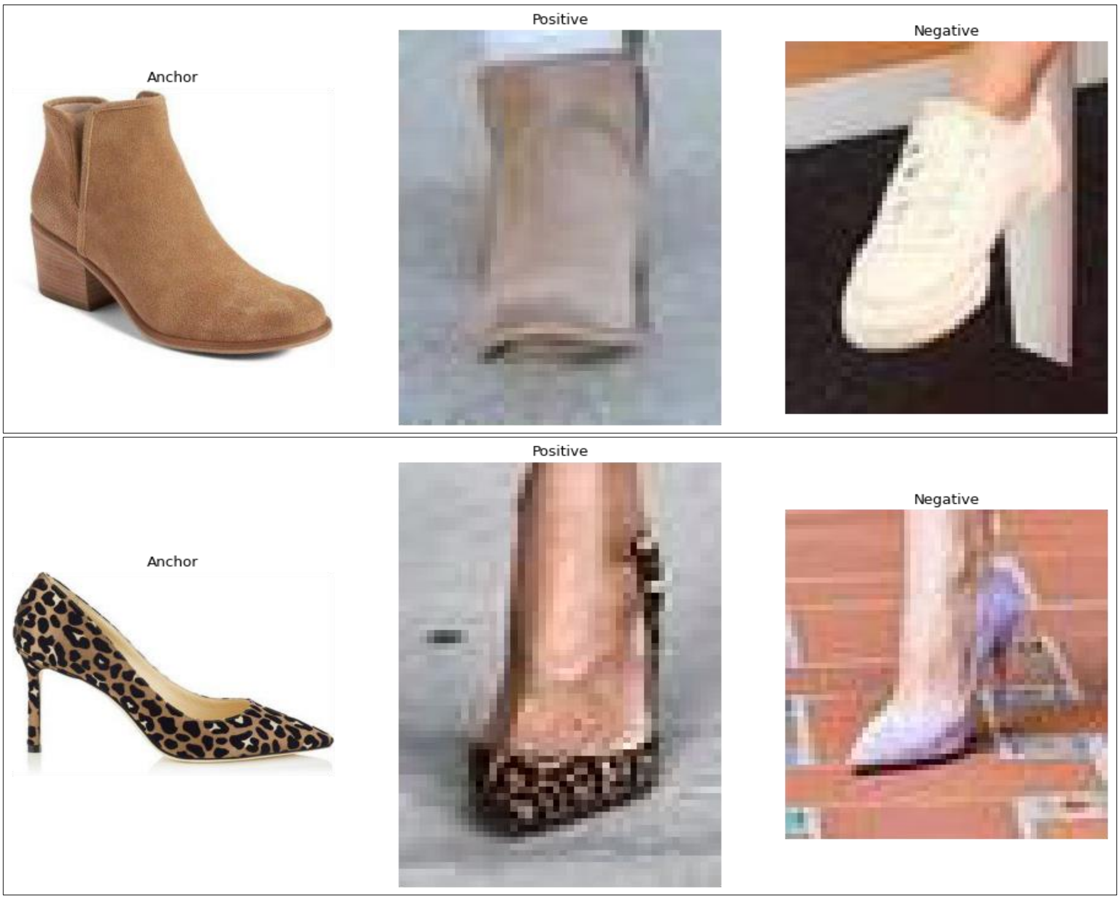


Figure 2-22 Anchor, Positive, and Negative pairs for Footwear

The triplets above will be given as input to the Siamese network. The triplet loss function tries to maximize the distance between anchor image and negative image while minimizing the distance between anchor image and positive image thereby learning to differentiate similar images to non-similar images.

We now move onto to define the Siamese network using TensorFlow and Keras. For a detailed explanation on the definition of the network, we have followed (Image similarity estimation using a Siamese Network with a triplet loss, n.d.). We have used a pre-trained ResNet50 as part of the subnetwork that generates the feature embeddings. By using transfer learning, we have reduced the training time and size of the dataset. We have fine-tuned the weights of the final layers of ResNet50 and we have kept the rest untouched. As a final step we have created the Distance Layer, which is responsible for computing the distance between the anchor embedding and the positive/negative embedding. The goal of our network is to differentiate a set of images. Triplet loss is evaluating how good a job the Siamese network does in separating the embeddings depending on whether they belong to similar images. Below we can see the losses of the 3 networks.

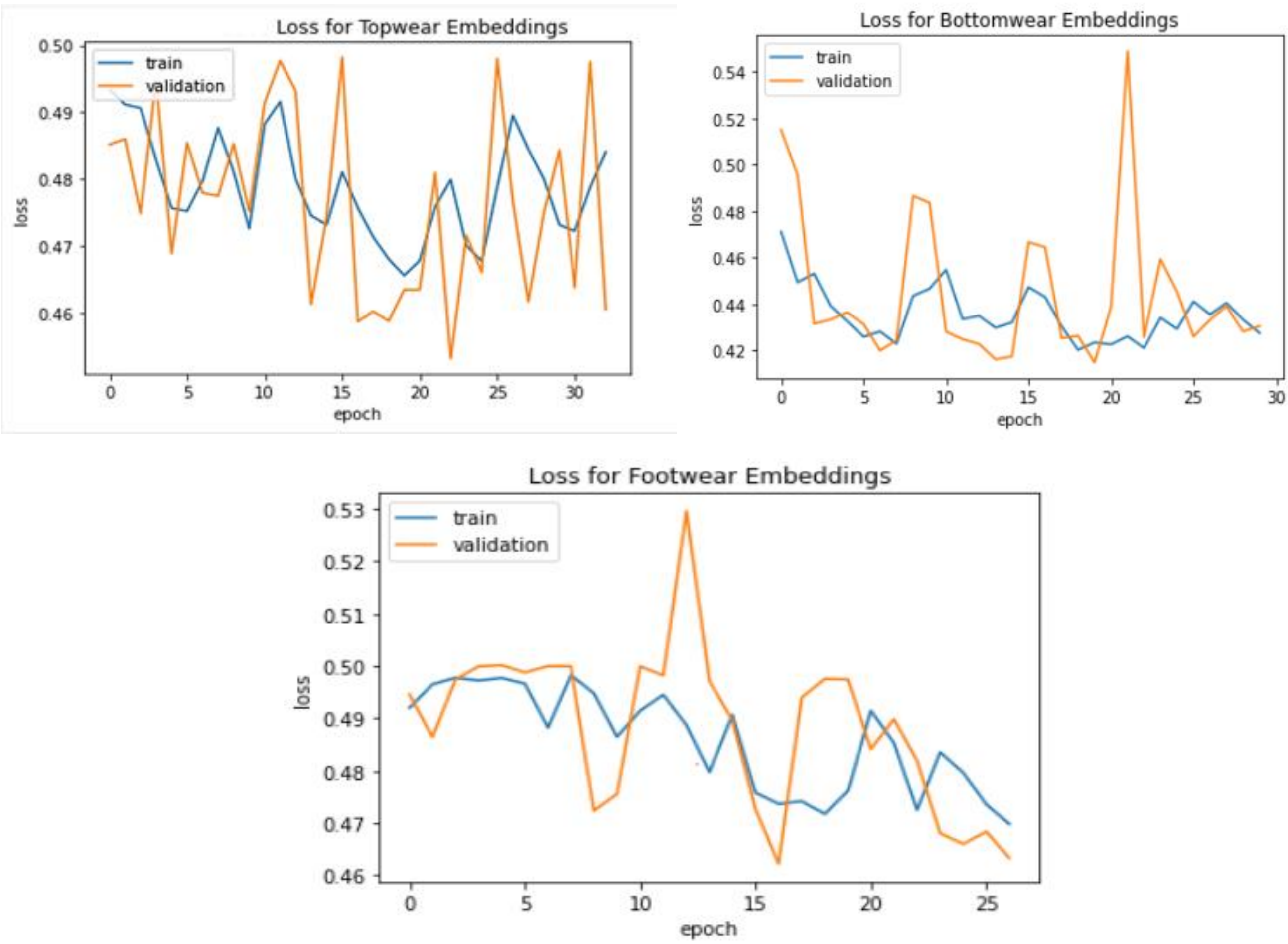


Figure 2-23 Loss Metric for the Siamese Networks

At this point, we can check how the network learned to separate the embeddings depending on whether they belong to similar images. We can use cosine similarity to measure the similarity between embeddings. Let's pick a sample from the dataset to check the similarity between the embeddings generated for each image.

- **Top wear**

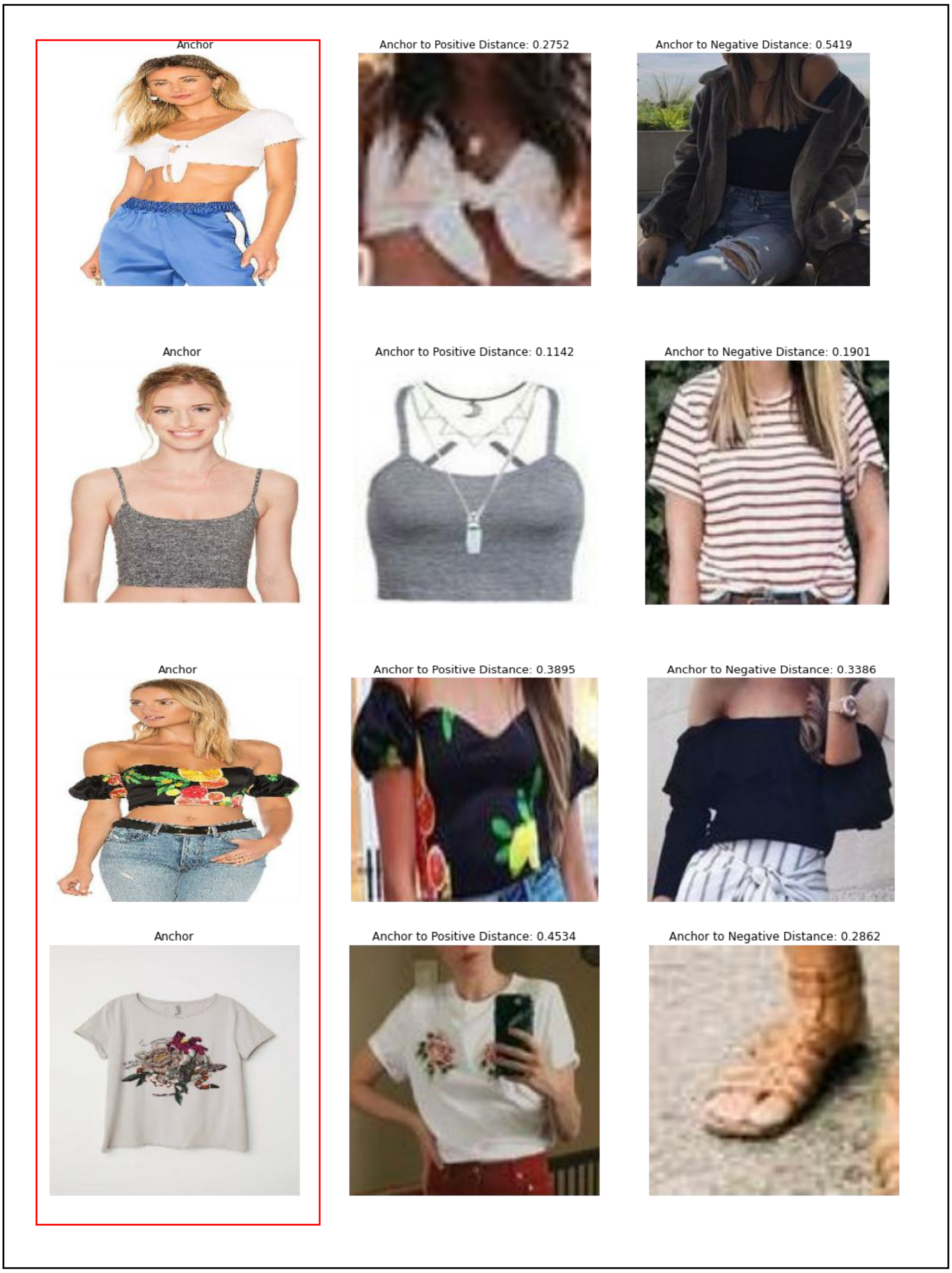


Figure 2-24 Similarity example between the embeddings generated for each image

➢ Bottom wear

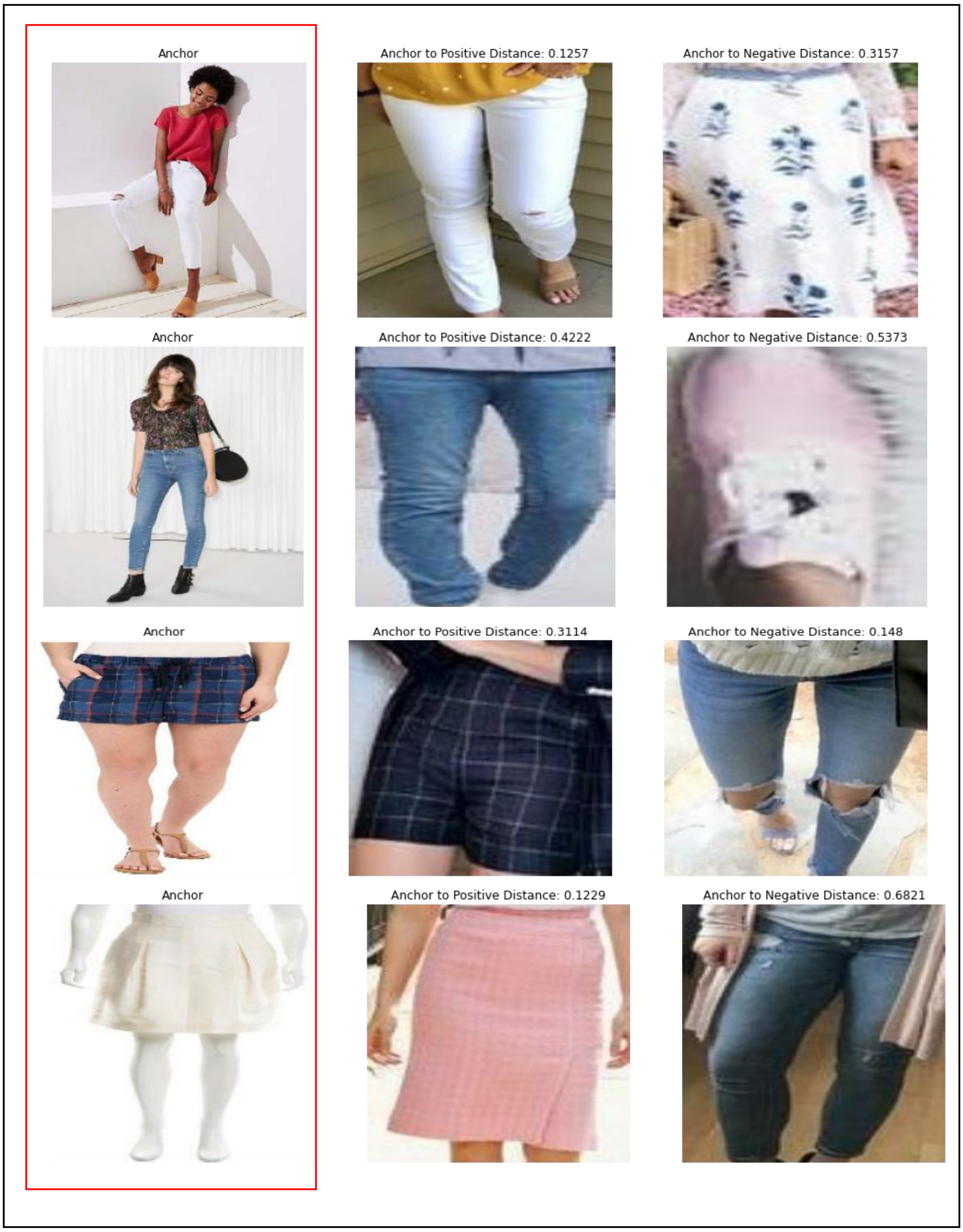


Figure 2-25 Similarity example between the embeddings generated for each image

- **Footwear**

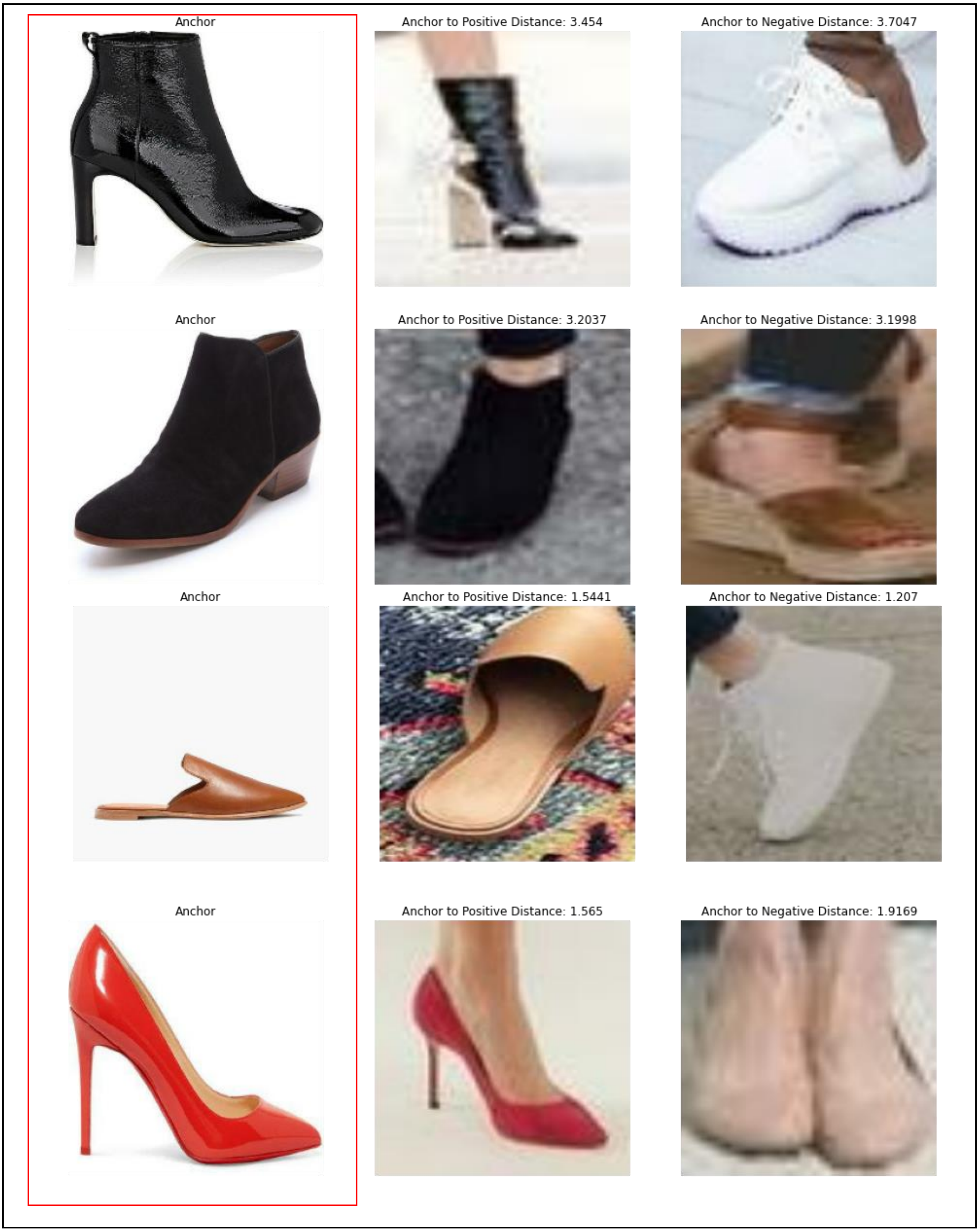


Figure 2-26 Similarity example between the embeddings generated for each image

Having trained the embedding generation model, we have to generate and save embeddings for the catalogue of fashion products. Let's try now to check the similarity between the embeddings generated for a catalogue image, which is not included in the training dataset.

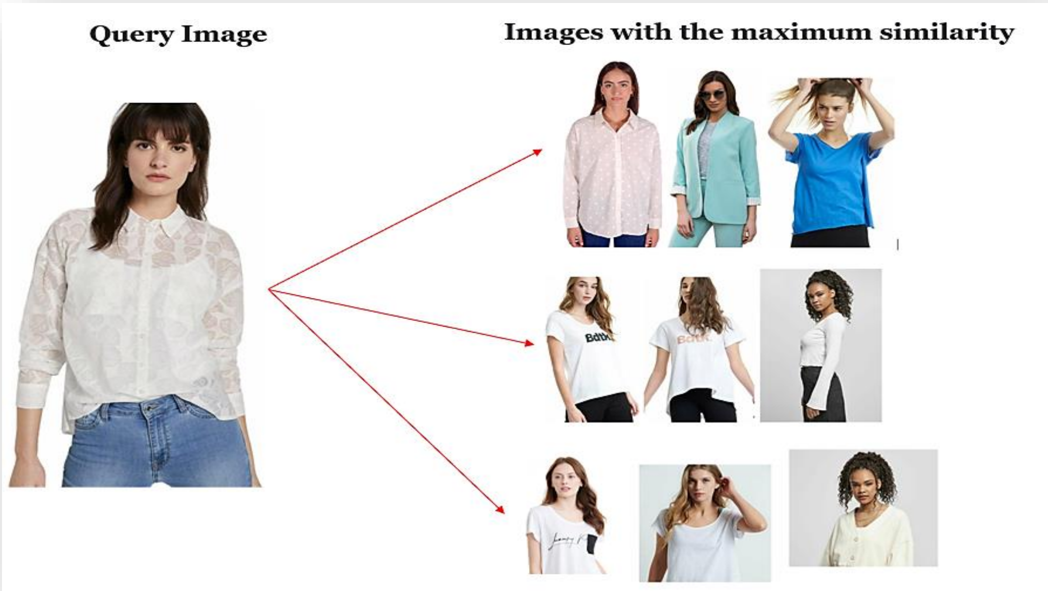


Figure 2-27 Similarity Results between the embeddings generated for a catalogue image

3.1 Final Pipeline

Having generated and stored the embeddings for the catalogue of fashion products, the final step is to combine all the modules into a pipeline that takes in a query image and outputs the relevant recommendations. Here are some example recommendations from the final pipeline. The first image is the query image followed by the recommendations.

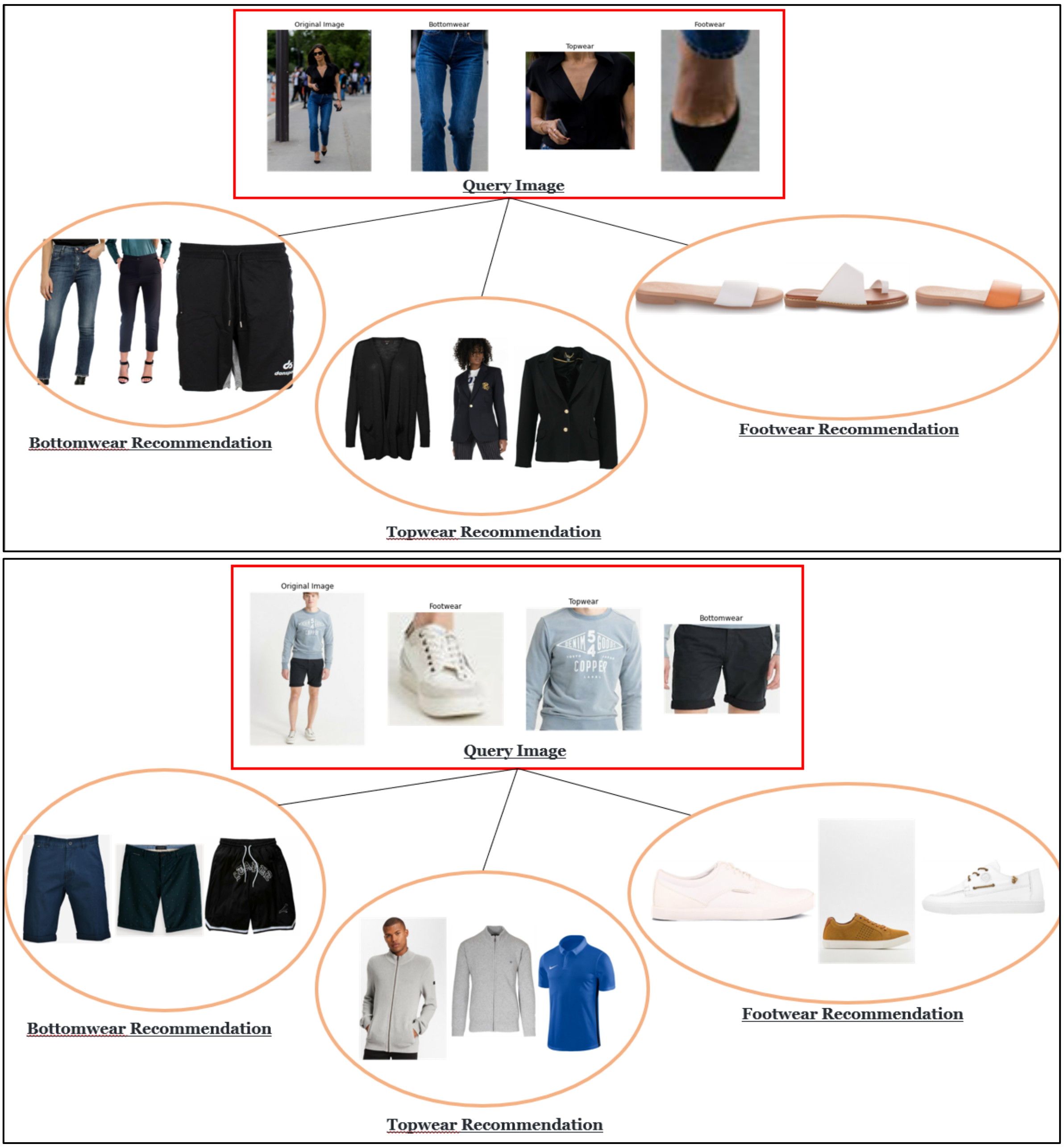


Figure 3-1 Recommendation Examples from Final Pipeline

Figure 3-2 Recommendation Examples from Final Pipeline

3.2 Further improvements

This system can be considered as a POC (Proof of Concept) for an image similarity-based item recommendation engine, further improvements can be made by changing the existing modules with better performing alternatives as time progresses.

For example, we can change the object detection pipeline to any SOTA algorithm of time for better detection. Similarly, we can retrain the embedding generation model regularly from time to time as new products are being added to the catalogue. The current object detection only detects top wear, bottom wear, and footwear. We can improve it by adding specific classes instead of in broad terms, like shirts, t-shirts, suits, pants, trousers, jeans, and shoes, and also support for other fashion accessories like handbags, sunglasses, watches etc.

The STL-Dataset included a lot of pairs where the product and the scene did not match. We can clean the dataset of such pairs and retrain the embedding generation model for a higher number of epochs.

One more thing that will improve our solution's output is to enrich our database of images with fashion items of more categories. For example we can include in our catalogue of images a greater variety of women’s t shirts and men's jackets. By doing this we will have the opportunity to output a more precise recommendation to the users.

Lastly, we need to deploy this system for users to try. Before we start predicting from our saved TensorFlow models we can further optimize them by using a technique called post training quantization. Post-training quantization is a conversion technique that can reduce model size while also improving CPU and hardware accelerator latency, with little degradation in model accuracy. One can quantize an already-trained float TensorFlow model when it is converted to TensorFlow Lite format using the TensorFlow Lite Converter. We can virtually reduce the model size by half by converting all the float32 weights to float16.

4. Members, Roles & Timeplan

A small team of two people in a complex and demanding project was a big challenge. Designing and implementing this project required full dedication from both team members and continuous evolution and collaboration in all project phases. Therefore, this project was a team effort end to end.

Spyros acted as the Business Analyst of the project. He captured the project objectives and ensured a thorough understanding of the problem that we were facing. He was responsible for requirement analysis and creating documentation. Having a thorough knowledge and understanding of the ecommerce fashion domain helped to assess the impact of the solution and decide the levels of metrics that would be acceptable beyond which the deep learning models can be declared successful. In addition, Spyros contributed to the development of the code throughout all phases of the project, providing valuable feedback and different perspectives on challenging technical problems. Thanasis, having experience as a Data Scientist, leveraged his coding skills to collect the data and build the machine learning and deep learning models. He analyzed the data to identify patterns and trends, interpreted the data to discover solutions and opportunities and finally communicated the findings using visualization techniques.

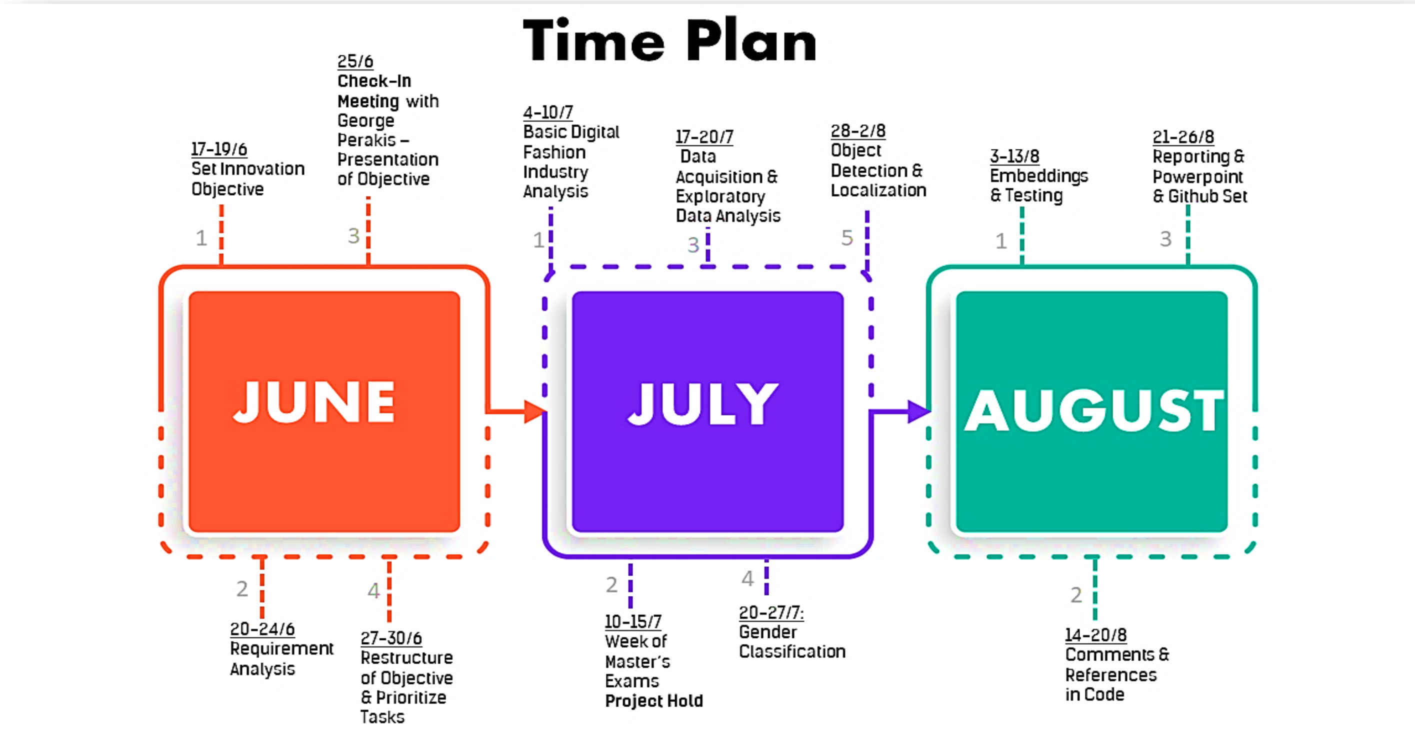


Figure 4-1 Timeplan for the Shop The Look Project

5. Conclusions

Conducting the above machine learning project with different tests and plots we ended up with final models that have given the best results. This project was a great opportunity to experiment with a variety of different machine learning tools. From the idea exploration to the implementation and promotion we saw end-to-end the phases of delivering a machine learning project. As we mentioned in “Feature Improvements” section, our solution for sure in not unmatched. We are continuously discovering new tools and methodologies that will improve our recommendation system to the users. Therefore, the main goal from this project was to get a better understanding of the problems and the difficulties that a machine learning projects has. Last but not least, we believe that “Shop The Look” idea, has a great potential and requires a deeper dive to each module separately

6. Bibliography

1. *Breaking Down Mean Average Precision (mAP)*. (n.d.). Retrieved from https://towardsdatascience.com/breaking-down-mean-average-precision-map-ae462f623a52

2. *Buy Me That Look: An Approach for Recommending Similar Fashion Products*. (n.d.). Retrieved from https://arxiv.org/pdf/2008.11638.pdf

3. *Complete the Look: Scene-based Complementary Product Recommendation*. (n.d.). Retrieved from https://arxiv.org/pdf/1812.01748.pdf

4. *How to Train YOLOv5 On a Custom Dataset*. (n.d.). Retrieved from https://blog.roboflow.com/how-to-train-yolov5-on-a-custom-dataset/

5. *Image similarity estimation using a Siamese Network with a triplet loss*. (n.d.). Retrieved from https://keras.io/examples/vision/siamese_network/

6. *Men / Women Classification Kaggle Dataset* . (n.d.). Retrieved from https://www.kaggle.com/datasets/playlist/men-women-classification

7. *Object detection: speed and accuracy comparison*. (n.d.). Retrieved from https://jonathan-hui.medium.com/object-detection-speed-and-accuracy-comparison-faster-r-cnn-r-fcn-ssd-and-yolo-5425656ae359

8. *Roboflow Dataset*. (n.d.). Retrieved from https://universe.roboflow.com/new-workspace-w4orl/fashion_obj_detection/browse?queryText=&pageSize=50&startingIndex=0&browseQuery=true

9. *Shop the Look Dataset*. (n.d.). Retrieved from https://github.com/kang205/STL-Dataset

10. *ultralytics/yolov5*. (n.d.). Retrieved from https://github.com/ultralytics/yolov5

11. *You Only Look Once: Unified, Real-Time Object Detection*. (n.d.). Retrieved from https://arxiv.org/pdf/1506.02640v1.pdf