Fashion.ai

Contents

1. KPI identification

1. The first KPI is to evaluate the compatibility score based on the suggested item generated by our model and it should be more than 0.5.
2. The second KPI is to evaluate the similarity score based on the suggested item generated by our model and it should be more than 0.5.
3. The third KPI is to evaluate the accuracy of the system's compatibility predictions. We will calculate the average score of 7 suggested outfit sets from the compatibility model for a given dataset of fashion combinations labeled from "least compatible" to "most compatible" with 1 to 5 scale from a sample group of customers (50 peoples) and fashion professionals (5 peoples). The average score of 7 suggested outfit sets that are filled in by both 50 sample groups and 5 fashion professionals should be more than 4.

The steps to measure this KPI includes the following steps.

1. Launch a survey in 50 sample groups and 5 fashion professionals, distributing from gen Z until gen Y.
2. There will be 7 sets of input items. In each input set, we will show the recommended photo results from the compatibility model.
3. In a scale of 1 to 5 (1 is least compatible and 5 is most compatible), the sample group will rate the score based on how compatible they are between the input image and the suggested images from our model. In each set, there will be 3 questions based on 3 different criterias including color, style, balance. Each criteria will have different weight : color 30% , style 30% , and balance 40%. We weight balance to be 40% because overall appearance would be the most important when we evaluate outfit score as a whole. For another two factors, it would be the same level of significance but less important compared to balance so we give only 30% for each of them.

2. Literature survey and review

2.1 How Good Is the Aesthetic Ability of a Fashion Model?

In this study, A100 (Aesthetic 100) is introduced to evaluate the aesthetic capability of the models for fashion compatibility. A100 has a number of positive traits: 1.Completeness, through two assessments, the LAT (Liberalism Aesthetic Test) and AAT (Academicism Aesthetic Test), it covers all categories of standards in the fashion aesthetic system.2. Reliability, it is diagnostic training data that is in line with key markers. It offers a just and impartial assessment for model comparison. 3. Explainability. Better than all prior indications, the A100 further pinpoints key

aesthetic elements in fashion, demonstrating the model's effectiveness on more minutely detailed dimensions like Color, Balance, Material, etc.

Analysis of Completeness

It covers all types of standards in the fashion aesthetic system through two tests

1. LAT (Liberalism Aesthetic Test- multiple choice questions).The LAT represents the aesthetic standard of Bottom-up.
    a. The source images are collected from mainstream outfit datasets. The missing product needed to be evaluated by experts will then be automatically generated following the proposed Outfit Generation Principle
    b. Then manually verified by experts with a fashion background. For each question, there has already formed a collective consensus with only one correct answer
    c. Finally, they built a questionnaire website and released the LAT to the fashion community to obtain the ground truth of this test

As the answers of each participant are not the same. Thus, the LAT has two scores: *1. LATs (LAT score)* The hard score follows the majority, the most selected choice of each question will be scoring 1 and others are 0. *2. mLATs (mean LATs)* : The score of each choice equals the probability of it being selected.

2. AAT (Academicism Aesthetic Test - multiple choice questions) AAT represents the aesthetic standard of Top-down. The creation of the AAT has significant highly professional requirements.
    a. According to a referred fashion professional in The Hong Kong Polytechnic University How Good Is Aesthetic Ability of a Fashion Model research.There are six dimensions that should be examined when judging the aesthetic ability of the model, including Color, Style, Occasion, Season, Material, and Balance.
    b. Then, the questions and choices in the AAT are rigorously designed following these dimensions and their sub-dimensions.
    c. Each question is limited to focusing on testing the model’s performance from only one dimension. This strategy enables our test to uniquely show the characteristic performance of the model on a fine-grained level in addition to the overall performance.
    d. The accuracy is denoted as AATs (AAT score), while the accuracy of each dimension set is called detailed index, e.g., Color Index, Style Index, Occasion Index, etc.

Analysis of Reliability

They start by looking at A100's precision in assessing model performance. The performance of four popular fashion compatibility techniques, including Bi-LSTMs, FHN, SCE-Net, and CSN, is specifically compared by the author. Three commonly used FITB test sets—the Maryland FITB test set, the Polyvore-630 FITB test set, and the Type-aware FITB test set—have quantifiable results that are given. It is important to note that when the performance of the models indicated by these three test sets has disagreements, the voting process will be used for judgment. The outcomes of this perception experiment don't quite serve as a standard. They underline that all models employ the default options and settings for two reasons: first, fairness (considering them to be pre-built models); and second, because no two models have the same input data needs and training conditions.

Analysis of Explainability

They also present the results of detailed indices of Bi-LSTMs, FHN, SCE-Net, and CSN to show the Explainability of the A100. The AAT's design process enables the A100 to demonstrate a model's aesthetic performance on fine-grained characteristics. The 100 questions were specifically broken up into six categories: color, style, occasion, season, material, and balance.The model can then be developed by concentrating on the areas that are lacking. For instance, as shown by A100, current models perform relatively poorly on the dimension of Balance. To sample more data relevant to the Balance is a straightforward notion for model improvement.

They present the A100 for evaluating fashion compatibility models. It stipulates that fine-grained indexes can further expose the models' deficiencies. The performance analysis can yield improved insights for model improvement by including these evaluations. The thorough investigation shows how effective A100 is. The author is hoping that the new indications of aesthetic perception will aid in the identification of the contemporary fashion intelligence system and stimulate real-world applications of fashion AI.

2.2 Fashion Recommendation and Compatibility Prediction Using Relational Network

In recent years, personalized fashion advice has drawn more and more attention. The majority of fashion recommendation systems base their recommendations on similarities and user evaluations and co-clicked and co-purchased histories. However, the idea of compatibility is essential to create a stylish outfit. Developing an artificial intelligence that can learn whether

objects are compatible with one another is challenging, because compatibility determination is a very subjective and complicated procedure.

The three primary drawbacks of the existing algorithms for compatibility-based fashion suggestions, which mostly emphasize visual similarity, are:

- Can only determine pairwise compatibility
- A category label and numerous attributes are required.
- Require a fixed order and number of items

Moosaei et al. developed two neural networks, FashionRN and FashionRN-VSE, based on the idea of a relational network to overcome this constraint. By inferring the link between things in an outfit based on visual or textual information about those objects, the goal is to model fashion compatibility.

In this study, outfits were created using data from Polyvore (a positive sample). Popular fashion website Polyvore.com allows users to generate and upload outfit information. The user-uploaded outfits are regarded as being compatible. The writers explored incorporating data from other sources but chose Polyvore as it had a clear context and description.The negative examples are made by swapping out a piece of clothing from one outfit for a piece from another. 14,922 sets were used for testing, 14,922 sets for validation, and 69,636 sets were used for training (both positive and negative).

The two components of the proposed model are compatibility scoring and relationship creation.

Each pair of components in an outfit has a non-linear relationship that is learned via the relation creation algorithm. The elements in the outfit are treated as objects and the ensemble is treated as a scene. First, vector representations (embeddings) of the images are produced using pre-trained CNN (DenseNet). The Fully Connected (FC) layer is then used to minimize the number of dimensions by passing the embeddings across it. The relation embedding is created by concatenating the lower-dimensional visual features of each item pair and running them through an FC layer.

For FashionRN-VSE, which combines the concept of Visual Semantic Embedding (VSE), the encoding of item descriptions (brands, texture, material, price point, etc.) is created using one-hot encoding. The item's image embedding and the text embeddings are then joined together.

All pairwise relations can be added together to determine the overall outfit's compatibility score, which is learned through cross-entropy loss.

The models are assessed using two tests, the compatibility prediction test and the fill-in-the-blank (FITB) test, using the Polyvore dataset.

- Compatibility prediction test: binary classification task; True if the outfit is compatible, and vice versa. The Area Under Curve (AUC) score is used to compare each model.
- Fill-in-the-blank (FITB) test: Choose the item that best complements the other items in the outfit from a set of candidate items and an outfit. Each model is compared using accuracy.

According to the study's findings, the models were able to solve some of the shortcomings of earlier models and showed state-of-the-art performance in both the fill-in-the-blank test and the compatibility prediction job. The necessity to provide item category labels as input to the network is removed when creating a compatibility framework based on RNs because the network is able to implicitly learn such information. Additionally, this method does not take into account the amount of things in an outfit or the order in which they appear.

2.3 Theme-matters: fashion compatibility learning via theme attention

Fashion compatibility is a crucial component of the fashion business that has attracted more attention in recent years as a result of the rising need for customized fashion advice.

Lai et al. divide the fashion compatibility learning techniques into two categories: outfit-wise compatibility learning, which models the process of putting together an outfit as sequence learning, and pair-wise compatibility learning, which formulates it as a pair-wise learning task and develops measurement methods (such as metric learning) for pair-wise compatibility (i.e., LSTM). The semantic linkages between various fashion items have not been taken into account by existing fashion compatibility analysis, which mostly focus on visual similarities. The authors created a theme attention mechanism that combines fashion themes into the examination of fashion compatibility in order to overcome this constraint.

The authors created a new fashion dataset, called Fashion32, which includes detailed annotations of outfit themes and fine-grained fashion categories because there was no dataset that could predict theme-aware fashion compatibility. Additionally, the authors divide the current fashion datasets into two groups: social media datasets and online shopping datasets. These two different datasets can be incredibly noisy and random. The data was collected via crawling into the online retailer JD.com, and annotated by fashion stylists from brand vendors, making it more realistic and convincing. Fashion32 has 152 fine-grained categories for more than 40K outfit pieces and 32 theme tags for more than 13K outfits.

Two obstacles have to be overcome in order to develop theme-aware fashion compatibility models from this dataset: how to compute pairwise compatibility and how to link theme to pairwise compatibility to compute outfit-wise compatibility.

The theme-attention model put out by the authors is based on the category-specific embedding space. The ImageNet pre-trained ResNet-50 model is used to first project paired items into the category-specific subspace, and the Triplet Network is then used to project compatible items closer to one another and separate incompatible items further. Negative samples for triplet networks are created by exchanging one piece from a compatible outfit with a piece chosen at random from the other outfits. The authors then create a theme-attention model to link the themes to pairwise compatibility and determine outfit-wise compatibility. The Attention network was utilized in the paper to learn theme compatibility, and it was treated as a classification task (cross-entropy loss). Area under the curve (AUC) and fill-in-the-blank (FITB) accuracy were utilized for objective evaluation. The authors also used the Subjective Experiment on Online Fashion Shop to evaluate the model's performance by calculating the click rates of customers' browsing histories (outfits with a higher click rate have a higher compatibility score).

The authors claimed that because of the particular item order and number, existing work such as the Bi-LSTM framework is less adaptable. However, their work performance relied on the quality of the image captions. Additionally, the authors developed a new dataset with fine-grained categories, which have a strong connection to fashion themes because of their characteristics, as opposed to using coarse categories (such as top, bottom, shoe, etc.). T-shirts, for instance, are more casual, shirts, more formal, and Polo shirts, in between. The resulting model performed better for both AUC and FITB tests than a number of state-of-the-art techniques.

2.4 GP-BPR: Personalized Compatibility Modeling for Clothing Matching

The paper presents a personalized compatibility modeling approach for clothing matching, named GP-BPR, which aims to provide automatic clothing matching solutions that cater to individual preferences. GP-BPR employs a deep neural network architecture that takes as input the user's historical interactions with fashion items and outputs a personalized preference vector for each user.

The general compatibility modeling component works on learning the latent compatibility space shared by complementary items to characterize the item-item interactions towards clothing matching. The personal preference modeling component leverages multi-modal data of fashion

items, including visual images, categories, attributes, and item descriptions, to capture user preferences from both the visual and textual perspectives.

During training, GP-BPR learns two sets of embeddings: one for fashion items and one for users. The learned embeddings are then used to compute compatibility scores between pairs of fashion items based on their latent representations in the learned compatibility space. The embeddings for fashion items and users are generated using a deep neural network architecture. Specifically, the authors use a multi-modal embedding network that takes input as the visual images, categories, attributes, and item descriptions of fashion items. The multi-modal embedding network consists of three sub-networks: a visual sub-network that processes the visual images of fashion items using a convolutional neural network (ResNet50), a textual sub-network that processes the categories, attributes, and item descriptions of fashion items using an embedding layer and a fully connected layer, and a user sub-network that processes the user's historical interactions with fashion items using an embedding layer. The outputs of these three sub-networks are concatenated to form a joint representation for each fashion item. The joint representation is then passed through several fully connected layers to generate the final embeddings for each item. During training, the embeddings are learned by minimizing the Bayesian personalized ranking (BPR) loss function, a pairwise ranking loss function.

In this paper, the authors constructed a large-scale dataset from the online fashion community IQON, which comprises 308,747 outfits created by 3,568 users with 672,335 fashion items. The dataset contains multi-modal data of fashion items, including visual images, categories, attributes, and item descriptions. Each outfit in the dataset consists of a set of fashion items that are worn together by a user. The outfit data is used to construct positive item pairs (i.e., pairs of items that are known to be compatible) for training and evaluation purposes. To construct negative item pairs (i.e., pairs of items that are known not to be compatible), the authors used a negative sampling strategy that randomly selects an item from the pool of all possible items that are not present in the positive pair. The dataset is split into training and testing sets based on user-outfit interactions. Specifically, outfits created by each user are randomly split into training and testing sets with a ratio of 4:1. The training set is used to learn the model parameters, while the testing set is used to evaluate the model performance.

The model is evaluated using the area under the receiver operating characteristic curve (AUC) metric. AUC measures the ability of the model to distinguish between positive and negative item pairs based on their predicted compatibility scores. To evaluate the model, a leave-one-out strategy is adopted, where one top-bottom pair for each user is randomly sampled and retained as the testing sample. The remaining data is used to generate a quadruple set consisting of training, validation, and testing sets according to a specific equation. For each positive top-bottom pair of

a user, a negative bottom item is randomly sampled from the whole bottom dataset to form a quadruplet. The model is trained on the training set and evaluated on the testing set using AUC. The AUC score ranges from 0.5 (random guessing) to 1 (perfect prediction). Higher AUC scores indicate better performance in distinguishing between positive and negative item pairs.

2.5 Bootstrapping Complete The Look at Pinterest

The research paper describes the development of the Complete The Look (CTL) system, which is designed to recommend complementary fashion items to users on Pinterest. The system uses a dataset of "polyvore"-style images, which are outfit collages that follow a specific visual style.

To create this dataset, the researchers trained an image style classifier to identify polyvore-style images on Pinterest. They then ran an object detection model to gather bounding boxes and category labels for items on these images. The resulting dataset was cleaned up using post-processing criteria for higher quality images.

The researchers then ran comprehensive sets of experiments comparing multiple methods for recommending complementary fashion items. They used offline metrics such as R@K to evaluate the performance of these methods.
The Complete The Look (CTL) system has three main components: the visual featurizer, category predictor, and style extractor.

The visual featurizer is a convolutional neural network (CNN) that extracts visual features from input images. It takes in an image and outputs a feature vector that represents the visual content of the image. This feature vector is then passed on to the category predictor and style extractor components.

The category predictor is implemented as two fully connected layers. It takes in the feature vector from the visual featurizer and outputs a probability distribution over all possible product categories. The ground truth labels for this classifier come from a detection model that identifies bounding boxes and category labels for items on polyvore-style images.

The style extractor is also implemented as two fully connected layers. It takes in the feature vector from the visual featurizer and outputs a 128-dimensional style embedding that represents the visual style of an image. This embedding is used to find complementary fashion items that match the user's current outfit.

During training, all three components are trained jointly using triplet loss, which compares the similarity between three images: an anchor image, a positive image (with similar style), and a

negative image (with dissimilar style). This training method helps each component learn to distinguish between different styles and product categories, generating embeddings that capture these differences.

During inference, given an input image of an outfit, CTL first uses its category predictor to identify the product category of each item in the outfit. Then it uses its style extractor to generate a 128-dimensional embedding representing the overall visual style of the outfit. Finally, it searches through its database of fashion items to find complementary products that match both the product categories and overall visual style of the user's current outfit.

The evaluation of the Complete The Look (CTL) system is described in the research paper. The authors used multiple methods to evaluate the performance of their model, including offline metrics and user studies.

For offline metrics, the authors used a test set of 25,000 outfits (109,847 items) to measure retrieval recall and Fill-in-the-Blank (FITB) accuracy. Retrieval recall measures how many relevant items are retrieved in the top K results, while FITB accuracy measures how well the model can predict missing items in an outfit. The authors also performed end-to-end evaluation on Pinterest's real product corpus, leveraging their in-house fashion specialists for labeling.

In addition to offline metrics, the authors launched CTL internally and conducted user studies to gain valuable insights about how real users think when they use the product. These user studies helped them understand how users interact with CTL and what factors influence their decision-making process when selecting complementary fashion items.

3. Theory Backup

3.1 Image embedding

Images are typically represented as a vast array of integers, such as a 3D array for RGB images. In order to analyze high-resolution images, we have to keep track of millions of numbers. It requires a high computational cost to scale any modeling using arrays of integers. Therefore, image embedding is a necessary technique used in computer vision and machine learning to represent an image.

Image embedding is a process of converting an image into a representation in the form of a numerical matrix. Image embedding aims to efficiently and compactly capture the features of an image, making it possible to perform mathematical operations and comparisons on images. The process of image embedding involves transforming the image into a set of features and mapping the features into a high-dimensional space. The mapping is performed so that the distance between the embeddings of similar images is small. On the other hand, the distance between the embeddings of dissimilar images is large.

There are several approaches to image embedding, such as deep learning and transfer learning. Deep learning approaches, such as Convolutional Neural Networks (CNNs), learn the image embeddings automatically from large amounts of training data. Transfer learning is a technique where a pre-trained model is fine-tuned for a specific task using a smaller amount of training data. In this paper, we propose two commonly used deep learning models in image embedding, including Convolutional Neural Networks (CNNs) and transfer learning embedding.

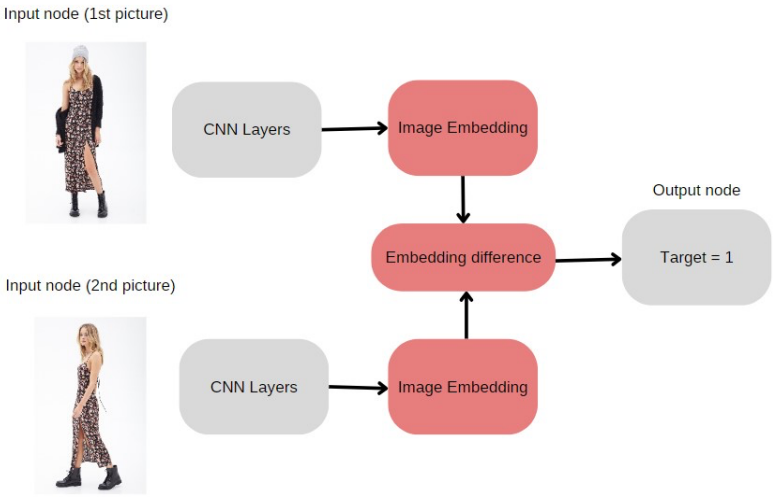


Figure 1 : *Image Embedding from input node to output node*

3.2 Convolutional Neural Networks (CNNs)

Implementing CNNs for image embedding involves the following steps:

1. Pre-processing: The input image is typically pre-processed to resize it to a standard size and to normalize the pixel values to a standard range.
2. Convolutional layers: The initial stage of a CNN consists of several convolutional layers, where filters are applied to the input image to extract local features, such as edges, corners, and textures. The filters slide over the input image, and the output of each filter is a feature map representing a different aspect of the image.
3. Pooling layers: After the convolutional layers, pooling layers are used to minimize the feature maps' spatial dimensions and increase the features' translation invariance. Pooling layers typically compute the output using max pooling or average pooling operations.
4. Fully connected layers: The final stage of a CNN consists of one or more fully connected layers, where the features from the pooling layer are fed into a dense neural network to perform classification or regression tasks.
5. Embedding layer: In order to use a CNN for image embedding, the final fully connected layer is replaced with an embedding layer, which outputs a fixed-length vector that represents the image. The embedding layer is trained to preserve the semantic relationships between images in the high-dimensional space, resulting in a small distance between the embeddings of similar images and a large distance between different images.

For various computer vision tasks, such as image retrieval, classification, and similarity search, it is possible to automatically train a compact and discriminative representation of pictures by utilizing CNNs for image embedding.

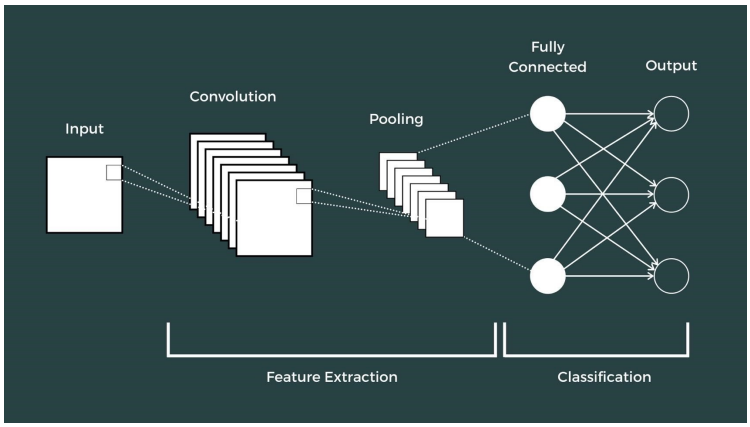


Figure 2 : *Implementing Convolutional Neural Networks for image embedding*

3.3 Siamese Network

A Siamese network is a type of neural network architecture that consists of two or more identical subnetworks that share the same weights and architectures. The subnetworks process two different inputs (for example, two images), and their outputs are compared to determine how similar the inputs are.

Siamese networks are typically used for tasks that involve comparing or matching pairs of inputs, such as image recognition, face verification, text similarity, or signature verification. They are particularly useful for tasks where labeled training data is scarce or where the similarity between pairs of inputs needs to be learned from scratch.

The basic idea behind a Siamese network is to extract a feature representation for each input using the shared subnetworks, and then compare the feature representations using a distance metric (such as Euclidean distance or cosine similarity). The distance metric output can then be thresholded to determine if the inputs are a match or not.

3.4 Triplet Loss

Triplet loss is a widespread loss function for supervised similarity or metric learning. In machine learning algorithms, the triplet loss function compares a baseline input to a positive and negative input. The triplet loss is used to train a model to distinguish between similar and dissimilar items by minimizing the difference between the distances between the similar and dissimilar pairs in the learned feature space. The triplet loss function is formulated as following:
$L(A, P, N) = max(d(A, P) - d(A, N) + margin, 0)$

where is the anchor image, is the positive image, is the negative image, is the distance$A$ between the representations $P$of the anchor and$N$positive image, d(is$A$the, P)distance between the representations of the anchor and negative image. Margin is$d$(aAhyperparameter, N) that sets the minimum difference required between the distances. The anchor image and the positive image belong to the same class, while the anchor image and the negative image belong to different classes. The model is trained to produce representations of the images such that the

distance between the anchor and positive image is smaller than the distance between the anchor and negative image by a margin. The use of triplet loss is effective in visual search models, as it helps the model learn discriminative representations that can distinguish between similar and dissimilar examples. Using triplet loss can improve performance on tasks such as image retrieval.

There are three types of triplets depending on how the loss is defined:

- **Easy triplets**: triplets which have a loss of 0, because d(a,p) + margin < d(a,n)
- **Hard triplets**: triplets where the negative is closer to the anchor than the positive, d(a,n) < d(a,p)
- **Semi-hard triplets**: triplets with positive loss even when the negative is not nearer the anchor than the positive : d(a,p) < d(a,n) < d(a,p) + margin

Each of these definitions depends on how far away from the anchor and positive the negative is. So, we may also divide the negatives into three categories: **easy negatives**, **semi-hard negatives**, and **hard negatives.**

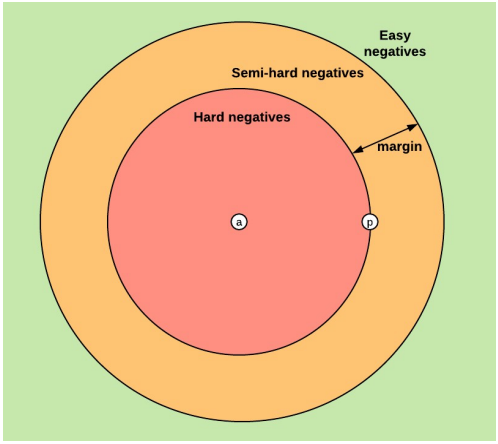


Figure 3 *: The three types of triplets includes* easy negatives, semi-hard negatives, and hard negatives

Easy negatives	Semi-hard negatives	Hard negatives

are examples that the network classifies correctly with high confidence. These examples are typically the least informative since they don't pose much of a challenge to the network.	are examples that the network classifies correctly but with a relatively low confidence score. These examples are somewhat challenging, but not as much as hard negatives.	are examples that the network classifies incorrectly with high confidence. These examples are typically the most challenging and important to learn from since they are the ones that the network is currently struggling with

Table 1 *: Comparison between three types of triplet loss*

Choosing semi-hard negatives for training a neural network can be useful because they can provide a good balance between hard negatives and easy negatives.

Using only hard negatives can make the training process very challenging and can slow down the rate of learning since the network is constantly being presented with examples that it struggles to classify correctly. On the other hand, using only easy negatives can lead to overfitting, where the network becomes too specialized on the training data and does not generalize well to new examples.

Semi-hard negatives offer a compromise by providing examples that are challenging but not too difficult for the network to learn from. By focusing on these examples, the network can improve its performance without becoming overwhelmed by examples that are too difficult to classify correctly. Furthermore, since the network is already classifying semi-hard negatives correctly, training on them can lead to fine-tuning the decision boundary, which can improve the network's ability to generalize to new examples.

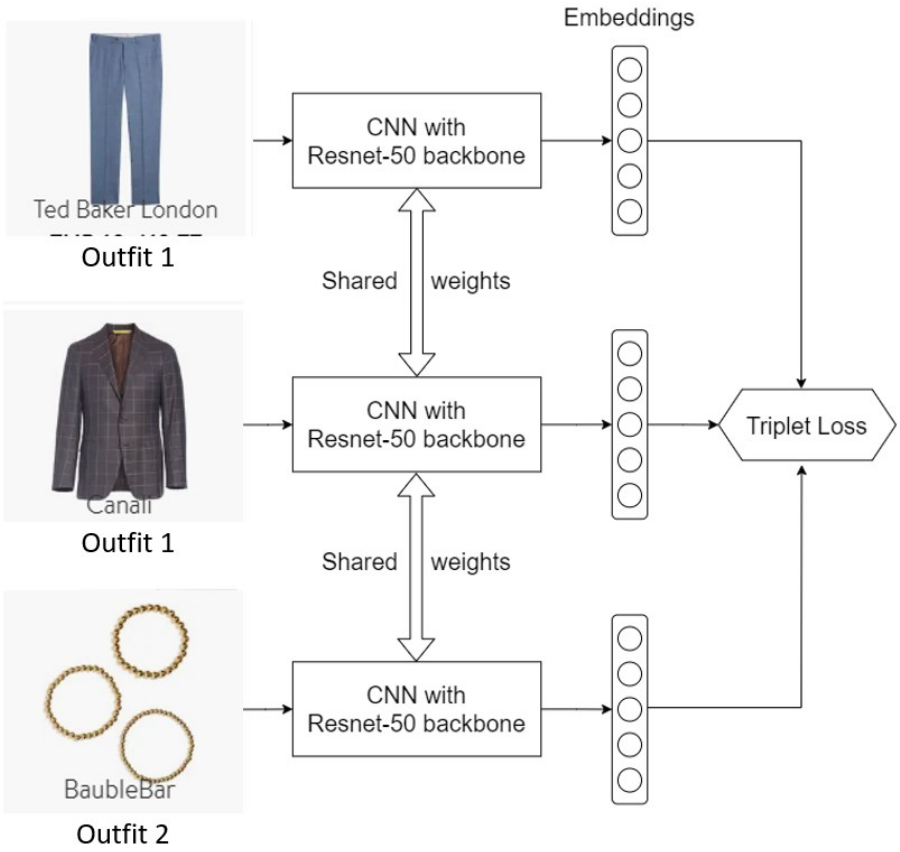


Figure 4 *: Train siamese network using triplet loss*

3.5 Object detection

Object detection is a critical process in computer vision that involves identifying objects' labels and estimating their locations by drawing bounding boxes around them. It serves as the foundation for various visual recognition activities. Single-class object detection focuses on detecting a specific object class, while multi-class object detection aims to detect and classify multiple objects present in an image.

Convolutional Neural Networks (CNNs) have emerged as a powerful approach for object detection. CNNs are feed-forward neural networks that utilize weight sharing to efficiently process image data. CNNs employ convolution, which involves overlapping functions, to extract meaningful features. The input image is convolved with an activation function to generate feature maps. Pooling layers are then applied to the feature maps to reduce spatial complexity and obtain abstracted feature representations. This process is repeated for the desired number of

filters, producing a set of feature maps. Finally, fully connected layers process these feature maps to generate output for object recognition, including confidence scores for predicted class labels.

CNNs employ different types of pooling layers to reduce network complexity and parameter count. Pooling layers, are translation-invariant and operate on patches within the selected feature maps. Activation maps serve as input to pooling layers, which process them to generate downsampled feature representations.

Given the limited availability of large-scale annotated fashion datasets, transfer learning is crucial in object detection for fashion images. Pretrained models, trained on large generic datasets like ImageNet, provide a starting point for training fashion-specific object detection models. Fine-tuning these models on fashion-specific datasets significantly enhances performance and convergence.

3.6 YOLOv4

YOLOv4 is the fourth iteration of the popular real-time object detection system used for image detection, which involves identifying and localizing objects of interest within an image.

YOLOv4 can be used for fashion image detection, which involves identifying and localizing different types of clothing and accessories in images. This can be useful in a variety of applications, such as e-commerce, social media, and fashion industry analysis. With YOLOv4, a neural network model can be trained on a large dataset of fashion images to detect various clothing items and accessories, such as shirts, pants, dresses, hats, shoes, and bags. The model can then be used to automatically analyze new images and identify the clothing items and accessories in them.

YOLOv4 uses a neural network architecture consisting of a backbone network, neck network, and head network to extract features from the input image and predict bounding boxes and class probabilities for objects in real-time. The backbone network is based on the CSPDarknet-53 architecture, while the neck network is a feature fusion module that combines features from different layers. The head network uses anchor boxes with different aspect ratios and the Mish activation function to improve the model's accuracy in detecting objects of different shapes and sizes.

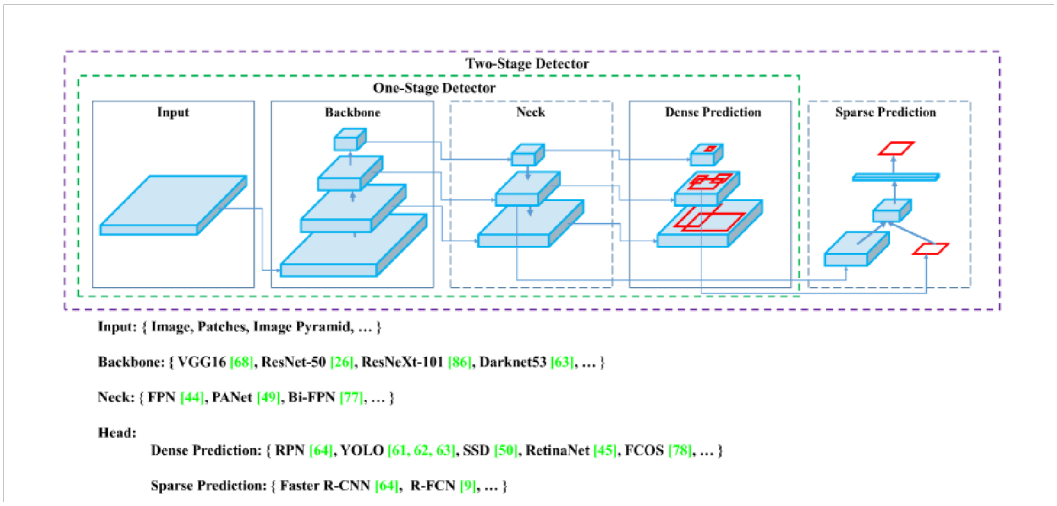


Figure 5 : *YoloV4 Architecture*

One potential use case for YOLOv4 in fashion image detection is in e-commerce, where it can be used to automatically tag and categorize products, making it easier for users to search for and find the items they are looking for. It can also be used for trend analysis and forecasting in the fashion industry, by analyzing large amounts of fashion images to identify popular styles, colors, and brands.

3.7 Darknet

Darknet is a widely-used open-source neural network framework written in C and CUDA, which can be used to train and deploy neural networks such as YOLOv4. Darknet plays a crucial role in fashion image detection, and can support YOLOv4 in several ways, including:

1. Training the YOLOv4 model: Darknet is utilized to train the YOLOv4 model on a large dataset of fashion images. During the training process, Darknet adjusts the weights of the neural network to improve its accuracy in detecting and classifying fashion items in images, resulting in a highly effective model for fashion image detection.
2. Data augmentation: Darknet can be employed to augment the fashion image dataset, producing new images with variations in lighting, rotation, and scale. This enhances the diversity of the training dataset, leading to improved robustness of the YOLOv4 model.
3. Fine-tuning the YOLOv4 model: Darknet can be utilized to fine-tune the pre-trained YOLOv4 model on a smaller dataset of fashion images. This enables the model to learn specific features of the fashion items in the images, resulting in even higher accuracy and performance.

4. Inference for new images: After the YOLOv4 model is trained, Darknet can be used to perform inference on new fashion images, detecting and localizing the items in the images. This is a critical step in fashion image detection, as it enables automated analysis of large volumes of images, leading to improved efficiency and productivity in the fashion industry.

3.8 Flask

The Flask web application serves as the interface between the users and the underlying models. It receives user requests and provides the corresponding responses based on the selected options. The Python application acts as the backbone of the web application, handling the core functionalities and coordinating the interactions between the different components. It manages the image uploads, processes user inputs, and triggers the appropriate operations based on the selected options.

The similarity search model was responsible for finding similar clothing items based on the uploaded image or the selected clothing part. It utilized a visual embedding model that was trained in semester 1 and comparison algorithms to identify garments with similar attributes. The YOLO object detection model played a role in the automated cropping process. It analyzed the uploaded image and detects clothing objects within it. The detected objects were then used to automatically crop the relevant clothing part before proceeding with the similarity search. The Flask web application ensured a seamless workflow by routing the user requests to the corresponding components. It received the uploaded image and passed it to the Python application for further processing. Depending on the selected options, the Python application interacted with the similarity search model or the YOLO object detection model to retrieve the desired results.

3.9 Front-End Development (HTML, CSS, JavaScript)

HTML (Hypertext Markup Language): HTML is the standard markup language used to structure the content of a web page. It defines the elements and their semantic meaning, such as headings, paragraphs, links, images, tables, forms, etc. HTML provides the basic structure and layout of a webpage.

CSS (Cascading Style Sheets): CSS is a style sheet language used for describing the presentation and visual appearance of a web page written in HTML. It is used to define colors, fonts, layouts, and other visual aspects of the content. With CSS, we would separate the presentation from the structure and enhance the design and user experience of a webpage.

JavaScript: JavaScript is a programming language that enables interactivity and dynamic behavior on a webpage. It can be used to manipulate and modify HTML and CSS, handle events, link the system to the photos browsing, interact with APIs (Application Programming Interfaces), and much more. JavaScript is commonly used to add functionality and enhance user interactions in web applications.

Together, HTML, CSS, and JavaScript form the foundation of the modern web. HTML provides the structure, CSS adds styling and layout, and JavaScript adds interactivity and dynamic functionality. They work together to create rich and interactive web pages and web applications.

4. Methods

4.1 Web Application Implementation

4.1.1 Backend development process

We created a Flask web application that combines our Python application, the similarity search and compatibility search model, and the YOLO object recognition model to allow people to use our models.

Users and the underlying models are connected through the Flask web application. It accepts user queries and produces suitable answers based on the selected options. The online program's core is the Python application, which manages crucial tasks and streamlines communication between various parts. According to the chosen parameters, it controls picture uploads, analyzes user inputs, and starts the necessary procedures.

The similarity search model was responsible for finding similar clothing items based on the uploaded image. It utilized a visual embedding model that was trained in semester 1 and comparison algorithms to identify garments with similar attributes. The fashion compatibility model was responsible for finding compatible clothing items based on the uploaded image. The

YOLO object detection model played a role in the automated cropping process. It analyzes the uploaded image and detects clothing objects within it. The detected objects were then used to automatically crop the relevant clothing part before proceeding with either the similarity search or compatibility search based on user's selection. The Flask web application ensured a seamless workflow by routing the user requests to the corresponding components. It received the uploaded image and passed it to the Python application for further processing. Depending on the selected options, the Python application interacted with the similarity search model or the YOLO object detection model to retrieve the desired results.

The workflow of the user's input of the Flask web application could be visualized in the image below.

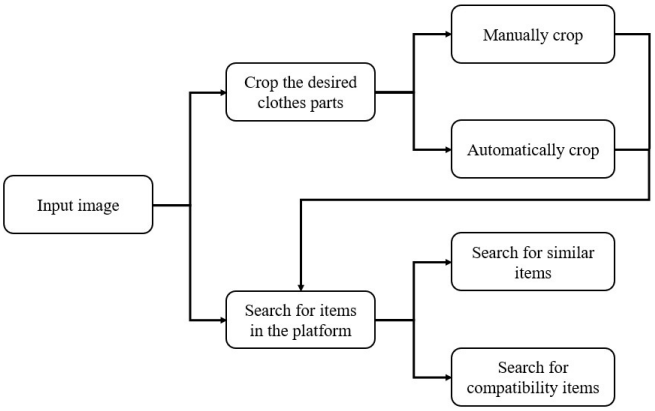


Figure 6 : *The workflow of the user's input in the Web application*

When users enter the web application, firstly, they choose an image to upload to the system. Then, the application provides users with two options: finding items from our platform or cropping the clothing part before finding similar items. If the users select to crop the clothing parts, there are two options: manually cropping or using the YOLO model to automatically crop the clothing part. The application provides two options for finding items from our platform, finding for similar items or compatible items.

4.1.2 Frontend development process

By combining the power of web technologies including HTML5, CSS3, and JavaScript, we crafted a visually captivating and user-friendly fashion similarity and compatibility item search platform. Our frontend development efforts resulted in a platform that seamlessly blends functionality, aesthetics, and usability to deliver an enjoyable and immersive fashion experience.

HTML (Hypertext Markup Language): HTML provides the structure and organization of the platform's content. It allows for the creation of different sections, such as product listings, descriptions, header and user interface components. HTML ensures the proper presentation and accessibility of the content.

CSS (Cascading Style Sheets): CSS is used to enhance the visual appearance and layout of the platform. It enables designers to apply styles, such as colors, fonts, and spacing, to HTML elements. CSS helps create an aesthetically pleasing and consistent user interface, making the platform visually appealing to users.

JavaScript: JavaScript plays a crucial role in creating dynamic and interactive features within the fashion product recommendations platform. It allows for real-time updates, user interactions, and personalized experiences. JavaScript can enable users to help connect the users to browse for desired products from the computer to show on the system platform.

Together, HTML, CSS, and JavaScript enable the creation of a visually appealing, user-friendly, and interactive fashion product recommendations platform. HTML provides the structure, CSS enhances the visual presentation, and JavaScript adds functionality and interactivity. These technologies work together to deliver personalized recommendations, improve user engagement, and create a seamless shopping experience for users.

4.1.3 User Journey on our web application

We have developed the prototype of our web application as follows:

On the home page, The primary focus is to showcase a curated selection of fashion products for users to explore and purchase. Users can also use the search bar to find out more products that they want in our store. Moreover, a search icon serves as a gateway to our platform's main features: product visual search and product compatibility search. By clicking on

the search icon, users can access these powerful tools that further enhance their fashion exploration.

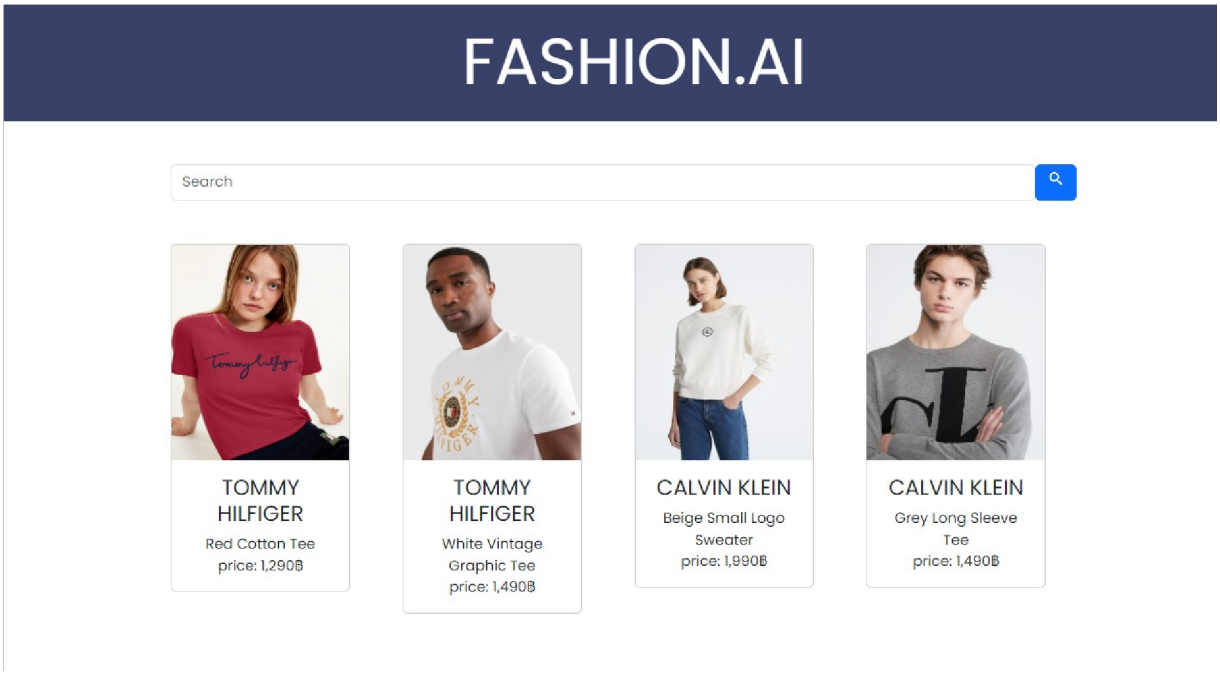


Figure 7 *: The homepage interface that shows product lists in the store*

After the users click the search icon, they can choose to upload input images that they want to find similar or compatible items on our platform.

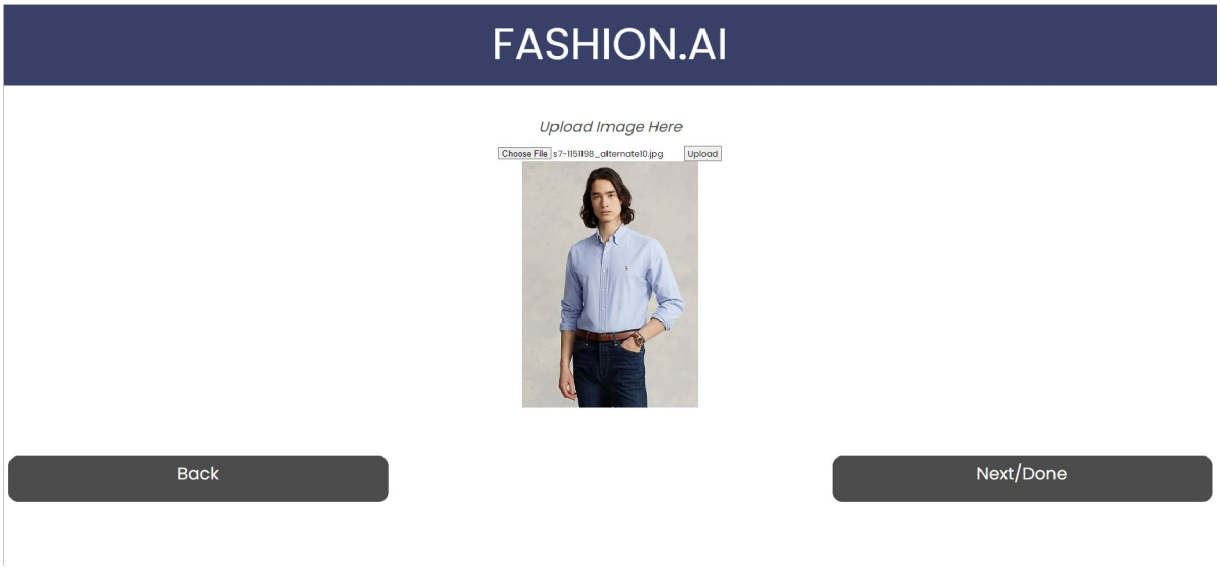


Figure 8 *: Users can choose and upload the input image into the platform*

Then, users can decide whether to crop or not crop the input image before proceeding to the platform. Between the two options, users can manually crop images by themselves if they choose a custom crop. On the other hand, if users select auto crop, the system will auto-detect the most obvious object to crop and proceed to the next step.

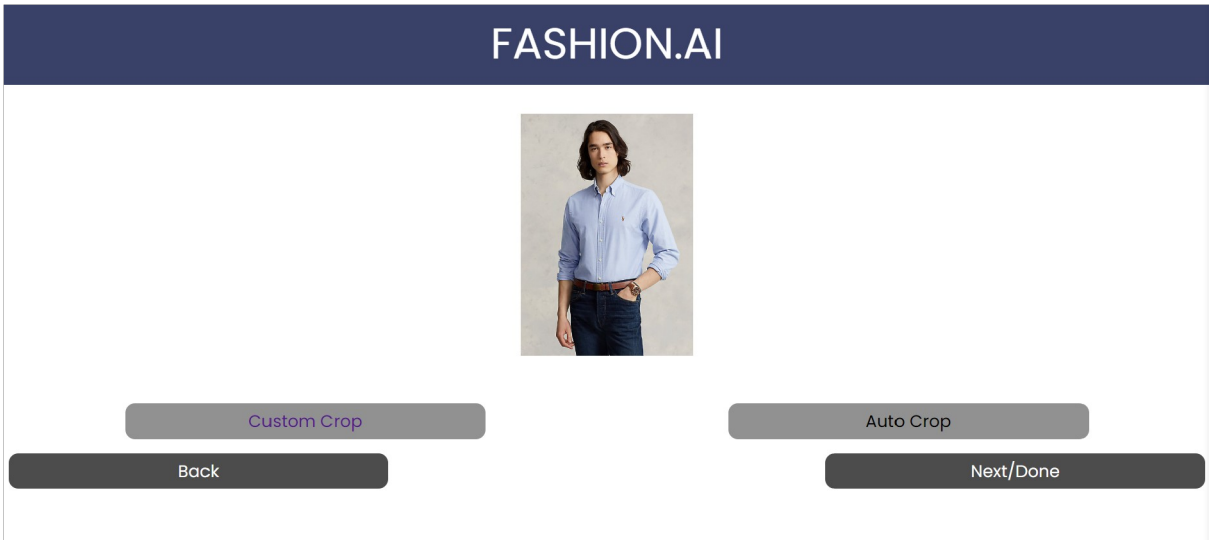


Figure 9 *: Users can choose to crop or not crop the image before proceed to the next step*

If users select to crop the input image by themselves (manual crop), they have to select the area they want to crop. In the example below, they want to focus only on the upper part body which is the shirt of the input image. Then, they can select a type of product such as top in this case to be more specific.

Figure 10 *: Users can select to crop the area that they focus before using our 2 features*

After users already crop the image, they can choose between 2 options to proceed. First, the product similarity search feature allows users to find similar products, while the product compatibility search helps users discover fashion items that go well together, providing valuable style recommendations.

Figure 11 *: Users can select to search similar products or compatible products from the input image*

If the user selects the product similarity search, the result will pop up a similar item from the input image. In this example, the result shows a similar blue shirt compared to the input image.

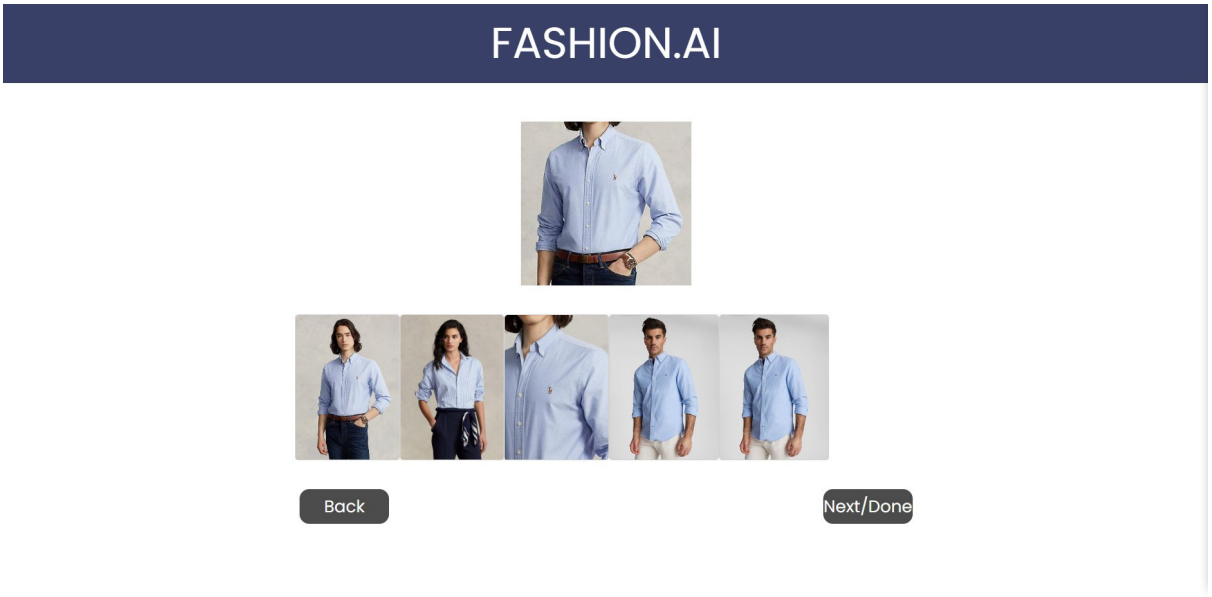


Figure 12 *: The result of product similarity search*

If the user selects the product compatibility search, the result will pop up the matching item from the input image. In this example, the result shows several items (eg. white sneakers, green shorts, beige linen trousers, navy loafer) that match the input image.

Figure 13 *: The result of product compatibility search*

4.2 Training

We developed the fashion compatibility model by incorporating an additional training phase utilizing the IQON dataset and employing triplet loss as an optimization objective to the search model, which was initially developed in the prior semester. This enhancement helps the model in generating image embeddings within the compatibility space, as opposed to the similarity space, which was the initial focus. For this training phase, we have utilized a dataset comprising 154,285 outfits. Additionally, a separate validation set encompassing 38,572 outfits, which represents 20% of the size of the training dataset.

The methodology deployed during this training phase, as well as the fine-tuning procedures, are similar to the previous semester.

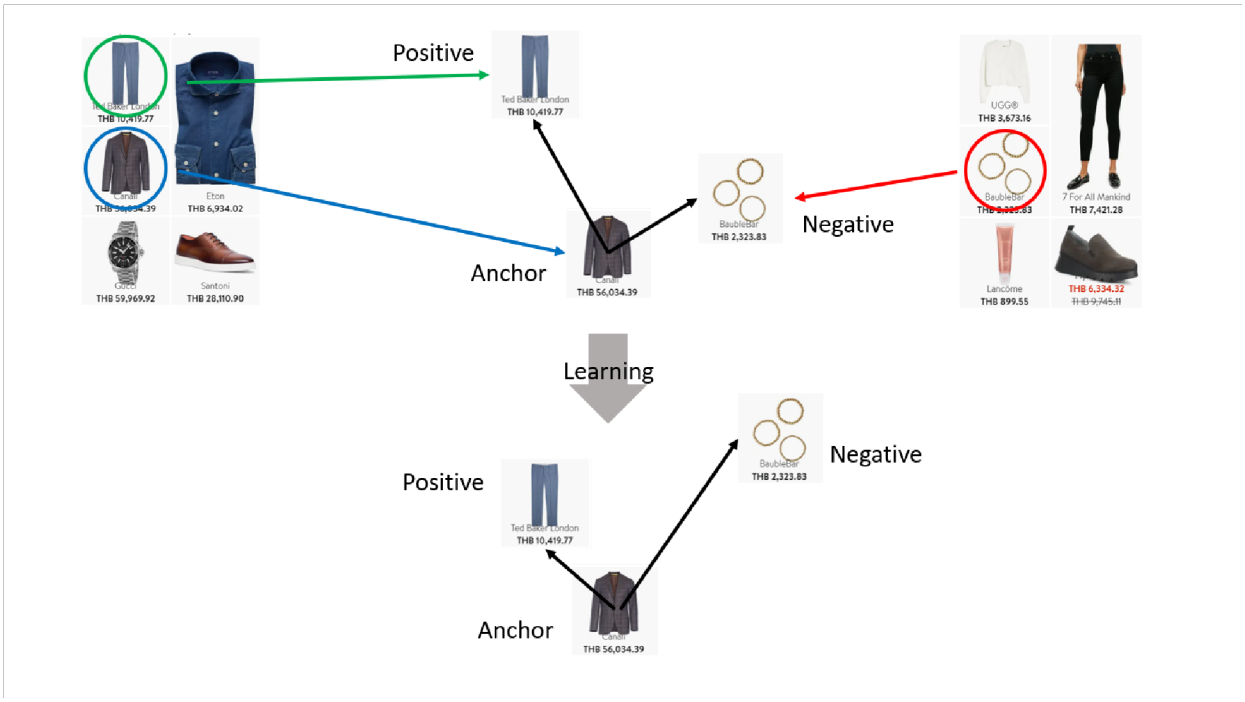


Figure 14 **:** *Triplet loss minimizes the distance between the anchor and a positive, both of which are from the same outfit, and minimizes the distance between the anchor and a negative from a different outfit.*

The backbone of the model is the similar image retrieval model from the previous semester. During the training stage, the pipeline includes following steps:

1. The base model is the similar image retrieval model from the previous semester. The model is then followed by a global average pooling layer, fully connected layer of dimension d, and l2 normalization layer.

2. We randomly sample K images to constitute a training batch. In this paper, we set K = 256.

3. We resize each image into n x n pixels. Each image is represented by a 3D numpy array. We use image size of 224 x 224 pixels

4. Then we convert the input images from RGB to BGR, then will zero-center each color channel with respect to the ImageNet dataset, without scaling.

5. The model outputs an image embedding with d dimensions. We used 64 dimensions

7. The image embedding is used to calculate triplet loss The margin m of triplet loss is set to be 0.2.

8. Adam method is adopted to optimize the model. The initial learning rate is set to be 1e^-4 and is decreased by 0.1 when validation loss has stopped improving. In Total there are 30 training epochs.

Similarity Search

Querying similar items given the images: We use **Cosine Similarity**.Cosine similarity is a commonly used technique in recommendation systems, including fashion recommendations. It measures the similarity between two vectors by computing the cosine of the angle between them. In the context of fashion recommendation, vectors could represent items such as clothes, accessories, or shoes.

The formula for cosine similarity is as follows:

cosine_similarity(A, B) = dot_product(A, B) / (norm(A) * norm(B))

where: A and B are two vectors to be compared, dot_product(A, B) is the dot product of A and B, norm(A) and norm(B) are the Euclidean norms of A and B, respectively.

Overall, cosine similarity can be a useful tool for fashion recommendation systems to suggest items that are likely to be of interest to the user based on their past behavior or preferences.

To perform image searches using embeddings, we used a library called Faiss, which can speed up the search processes.

FAISS (Facebook AI Similarity Search) is a widely used library for efficient similarity search and clustering of large datasets. One of the most common use cases of FAISS is to find the k nearest neighbors of a given query vector based on cosine similarity. The k-nearest neighbors (KNN) search algorithm in FAISS is implemented using an inverted index structure that stores the feature vectors and their corresponding metadata. To find the KNN of a query vector, FAISS first normalizes the query vector and performs a cosine similarity search against the stored vectors. The top-k vectors with the highest cosine similarity scores are returned as the KNN. This process is computationally efficient and scalable, making it suitable for high-dimensional datasets.

5. Experimentation

5.1 Dataset

The dataset for training are considered in the below table.

Dataset	**Polyvore**	**Fashion32**	**Complete-the-look-dataset**	**IQON3000**
Source	Polyvore.com	JD.com	pinterest.com	iqon.jp
Year created	2017	2019	2020	2019
Scale	60K outfits	13K outfits 40K items	100K outfits 453K items	308K outfits 672K items
Diversity	Unknown	32 theme, 6 coarse-grained category, 152 fine-grained category	Unknown	Unknown
Realistic	The outfit are created by the users	The outfit fashion stylists from the brand vendors	The outfit are created by the users	The outfit are created by the users
Quality	Each item contains rich multi-modal information such as product images, text descriptions, associated tags,	Each outfit has rich meta information and various labels, as well as real model pictures. The items are annotated with	Each item includes the image, and the category of the item.	Each item includes the visual image, categories, attributes, item description, price and number of

	popularity score, and type information.	design/style, tags of texture/fabric, tags of color, and a paragraph for product descriptions.		likes.

Table 2 : *Comparison between each available datasets to be considered for training the model*

5.1.1 Training dataset (IQON3000)

From the four considered datasets: Polyvore, Fashion32, Complete-the-look-dataset, and IQON3000, we decided to proceed with IQON3000 because of its large dataset (308K outfits, 672K items) and detailed annotation (categories, attributes, item description, price and a number of likes) for each data item. IQON is a Japanese platform that allows users to freely combine images of fashion items sold on over 200 e-commerce sites to create matching outfits.

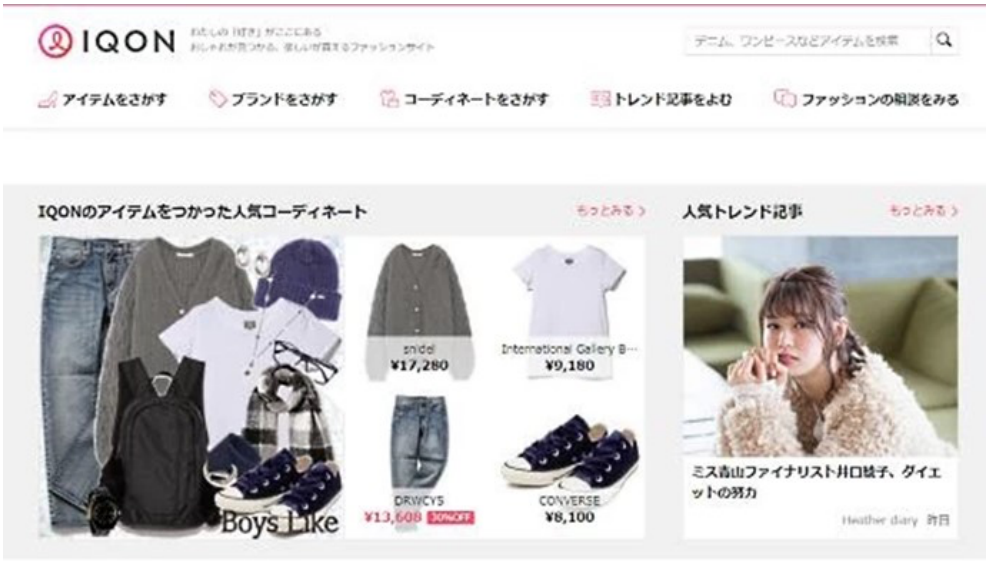


Figure 15 : *The interface of IQON, a Japanese platform that allows users to freely combine images of fashion items sold on over 200 e-commerce sites*

One important thing to take into consideration is the annotation from the IQON's dataset provided is in the Japanese language.

```
{'setId': 3319306,
 'setUrl': 'https://www.iqon.jp/sets/3319306',
 'likeCount': 14,
 'user': 2340358,
 'items': [{'imgUrl': 'https://img.iqon.jp/items/9718751/9718751_m.jpg',
   'price': '26,352',
   'category x color': 'コート × 黒 (ブラック) ',
   'itemId': 9718751,
   'itemName': '[L]キュプラ混トレンチコート',
   'itemUrl': 'https://item.iqon.jp/9718751/',
   'breadcrumb': [['IQONトップ',
     'ブランドをさがす',
     'インディヴィ バイ［INDIVI V.A.I.] ',
     'コート',
     'トレンチコート',
     '[L]キュプラ混トレンチコート'],
    ['IQONトップ', 'カテゴリでさがす', 'コート', 'トレンチコート', '[L]キュプラ混トレンチコート']],
   'brands': ['ブランド：インディヴィ バイ［INDIVI V.A.I.] '],
   'categorys': ['カテゴリ：インディヴィ バイ［INDIVI V.A.I.] × コート'],
   'options': ['素材: キュプラキュプラ × コートキュプラ × トレンチコート'],
   'colors': ['色: 黒 (ブラック) × コート黒 (ブラック) × トレンチコート'],
   'expressions': ['アイテム説明 [INDIVI / V.A.I] ・シックな光沢感が魅力的なキュプラ素材を使用したトレンチコートです。落ち感
のあるなめらかな質感が大人の女性らしい洗練された印象を漂わせています。すとんと落ちるすっきりしたシルエットも魅力的。付属のベル
トでかちっとしたウエストシェイプスタイルも楽しめます・サイズ違いの商品もご用意しております。・R:127-98306 S:127-98604\u3000
・店舗へのお問い合わせ商品番号: 127-98706・主に春に着用いただける商品です。【送料】全国一律送料496円(税込)。 2,000円 (税込)
以上購入で送料無料。\r\n【配送期間】在庫がある場合は注文登録完了後、ご注文当日～翌々日に発送。\r\n【ポイント】「楽天スーパーポ
イント」利用が可能。1ポイント1円として購入時に利用可能。\r\n※記載事項に関しまして、購入サイトにて変更が生じる場合がございま
す。さらにくわしい情報をみる']},
```


Figure 16 : *Example of the annotation data*

Within the dataset, some outfits are not complete. For example, there is only one item like one accessory in some of the outfits. To filter out those outfits, we first focused on the categories of the items. There are a total of 62 categories presented in this dataset, we first selected the category that fitted into the tops and bottoms categories. We identified tops and bottoms according to the description below.

tops = ['ブラウス','チュニック','トップス','T シャツ','タンクトップ','カーディガン','ニット']
bottoms = ['ショートパンツ','ロングパンツ','スカート','ロングスカート','レッグウェア']

Then we kept only outfits with at least one top and one bottom, resulting in 192,857 outfits. Each outfit contains from 2 to 20 items, and we could also further filter the outfit based on the number of items if needed.

Figure 17 : *Example of incomplete outfit (no bottom part)*

Figure 18 : *Example of a complete outfit (both top and bottom parts, with additional accessories)*

5.1.2 Testing dataset (scraped dataset 100 outfits)

We evaluated the accuracy of the model's predictions in determining the compatibility perception of fashion items input by our target customers who preferred classic styles of clothing and were in the age range of 18-35 and fashion professionals.

We started to collect data to test the model from the real clothes-selling website, including Nordstrom, Connor, nohow-style, and many other shop-by-look websites. There are 100 sets of matching outfits in our dataset, 60 sets are men's outfits and another 40 sets are for women. Each outfit set consists of the mandatory parts which are top, bottom, and shoes, and some outfit sets also include matching jackets, bags, and other accessories.

After that, we also put the product’s description in the Google sheet which had already been divided into 100 outfit sets with 6 categories including top, bottom, shoes, bag, and accessories. The description of each product in Google sheet will be correlating with the testing outfit dataset.

	Outfit Set	Top	Bottom	Jacket	Shoe	Bag	Acc
2	1	Raglan Sleeve Cashmere Sweater	Wool & Mohair Straight Leg Pants		Marcie Loafer (Women)		Round Brilliant Lab Created Diamond Pendant Necklace
3	2	Floral Peplum Blouse	Preen Wide Leg Jeans		Maxwell Mary Jane Flat (Women)		Dot Hammered Buddha Belly Earrings
4	3	Cross Back Sleeveless Linen Top	Pleated High Waist Wide Leg Pants		Kaitlynn Lizard Embossed Pointed Toe Pump (Women)	Mini Leather Bucket Bag	Maggie Large Hoop Earrings
5	4	High/Low Crepe Blouse	Tachini2 Stretch Cotton Ankle Pants		Milah Penny Loafer (Women)	Small Grand Ambition Bucket Bag	Logo Drop Back Earrings
6	5	Delphine Print Silk Blend Button-Up Shirt	Raw Hem Straight Leg Jeans		Saimy Block Heel Sandal (Women)	Small Marcie Leather Crossbody Bag	Pavé Cubic Zirconia Cross Stud Earrings
7	6	Oversize Short Sleeve Tunic	Double Face Mélange Pants	Double Breasted Stretch Wool Blazer	Parker Slide Sandal (Women)	Mini Puzzle Colorblock Leather Bag	
8	7	Keyhole Front Rib Sweater	Flare Trousers	Double Breasted Stretch Wool Blazer	Ronnie Platform Loafer (Women)	Crossgrain Leather Convertible Shoulder Bag	
9	8	Aida Wrap Blouse	Santiago Straight Leg Trousers		Eleanor Pointed Toe Pump (Women)	Mini Ann Leather Tote	Multicut Cubic Zirconia Hoop Earrings
10	9	Ellis Organic Cotton Shirt	Lark Ankle Bootcut Jeans		dragonfly espadrille (Women)		Mini Bubble Initial Gold Stud Earring
11	10	Brightside '90s Tank	90s High Waist Loose Stretch Denim Jeans		Finaa Sandal (Women)	Amini Betty Quilted Satin Top Handle Bag	Chunky Oversize Hoop Earrings
12	11	Sleeveless Turtleneck Matte Crepe Top	Preen Wide Leg Jeans	Dorset Crop Wool Blend Blazer	Becarda Pointed Toe Loafer (Women)	Small Cabata Calfskin Leather Tote	
13	12	Liv Slit Balloon Sleeve Stretch Cotton Top	Saige Raw Hem High Waist Ankle Wide Leg Jeans		Blondie Slingback Sandal (Women)		Lion Head Double-G Clip-On Earrings
14	13	High/Low Crepe Blouse	Kelsey Knit Trousers		Kiara Pump (Women)	Shopping Leather Tote	Pear Diamond Pendant Necklace
15	14	Tie Neck Sleeveless Crêpe de Chine Blouse	Drapey Pleated Cuff Trousers		Rhyme Time Slingback Flat (Women)	Large Le Pliage Recycled Canvas Shoulder Tote	54mm Gradient Round Sunglasses
16	15	Gauze Button-Up Shirt	Farrah High Waist Skinny Jeans	Pleated Sleeve Blazer	Hi-Line Chelsea Boot (Women)		Maillon Chain Belt
17	16	Carey Pleat Poplin Top	Perinua Wide Leg Trousers	Angelina Tweed Blazer	Kendria Ankle Strap Sandal (Women)		
18	17	Reannia Polo Sweater	Preen Wide Leg Jeans		Marcie Loafer (Women)		21-Inch Fine Beaded Chain
19	18	Gauze Button-Up Shirt	Stretch Knit Trousers	Grace Open Front Knit Jacket	Gommettine Buckle Pointed Toe Flat (Women)		14K Gold Diamond Marquise Stud Earrings
20	19	Split Neck Kind/Wool Sweater	Faux Feather Trim Swirl Print Satin Crop Pants		Zayne Mary Jane Platform Pump (Women)	Extra Small Le Pliage Leather Crossbody Bag	Men's Point Stud Earrings
21	20	EVERLY CAMI	WIDE LEG JEANS	MIA BLAZER	CIABATTE HEELS		DIEGO 10 NECKLACE
22	21	AMELIA HALTER ONE PIECE	TWILL TROUSERS	SHIRT JACKET	ROSIE 12 HEELS	GIA SHOULDER BAG	
23	22	THE BOYFRIEND SHIRT	DAHL MINI SKIRT	DOUBLE BREASTED SUIT JACKET	COMBAT BOOTS	MINI LOVE BAG	
24	23	SIGNATURE SHIRT	MADELYNN TROUSERS		THE LINE STRAPS	MINI SOHO FLAT GRAIN BAG	BLOCK SEMI PATENT BELT
25	24	VEILED TOP	WIDE LEG JEANS		LYDIA SANDALS	MINI LISBON CLUTCH	
26	25	SHAPED RIB TOP	ANKLED D'ARCY JEANS	ROMY CROPPED JACKET	GLITTER PUMPS	CHAMPAGNE BOW BAG	
27	26	REESE TOP	NEW DIDI PANTS	KATHERINE VEST	GROMMET KURT HIGH TOP SNEAKERS	MINI SOHO FLAT GRAIN BAG	BLOCK SEMI PATENT BELT
28	27	WAITER BLAZER	POLEN PANTS		WANDA ANKLE BOOTS		DIEGO 2 NECKLACE
29	28	ORGANIC COTTON POPLIN OVERSIZED SHIRT	L148 DENIM MOTT FLARED ANKLE JEAN		NAPPA LEATHER & SUEDE PACKABLE LOAFER		LUNE BRASS HOOP EARRINGS

Figure 19 : *The products description of each outfit set, correlating with the datasets derived from shop-by-look, clothes-selling websites*

5.1.3 Platform's item dataset

In the first semester, we collect both tops and bottoms product categories to test on the platform and it consists of 270 images from 10 hi-street brand products from which represent products in our platform. This dataset will be used to evaluate our model. In this semester, we have collected more product categories including jacket, shoe, bag, and acc to help model achievable predict more wide range of product.

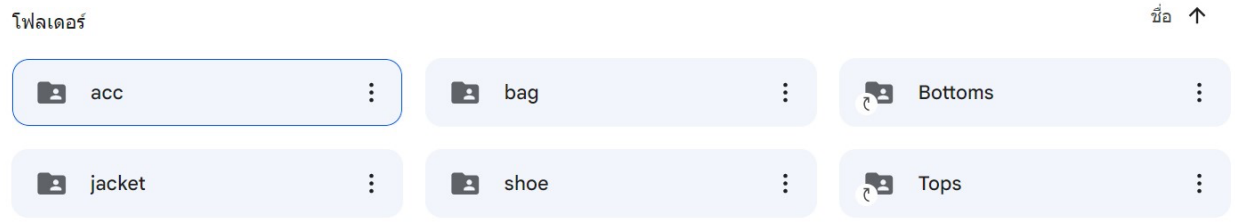


Figure 20 : *Platform items dataset in different categories*

5.2 Evaluation Metric

5.2.1 Evaluation of suggested sets with compatibility-tagged items in the dataset accuracy metric

- Compatibility score

The compatibility score is a metric used to evaluate the performance of a model during training. It quantifies the number of clothing items generated by the model that appear in a set of possible compatible outfits. The compatibility score serves as an indicator of how well the model is performing in generating clothing outfits that align with the concept of compatibility within a given set of outfits.

Example of the calculation of compatibility score in evaluating our model is shown in the image below.

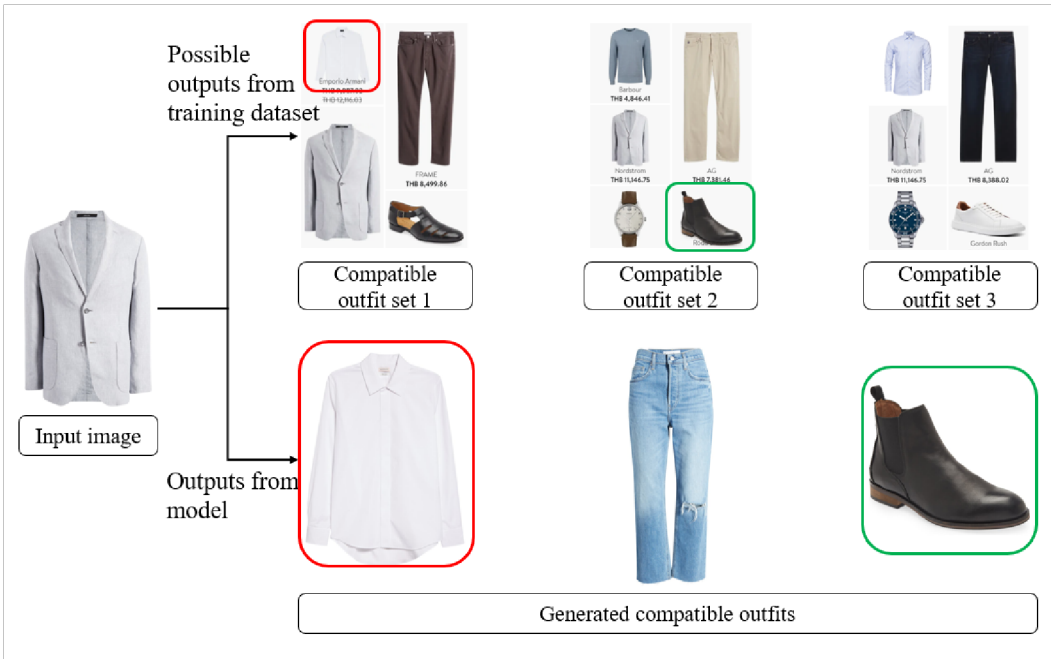


Figure 21 : *Example of the calculation of compatibility score*

According to the figure, given the input image, there are 3 compatible outfits which include this input clothing item in the training dataset. The model generated different clothing items for each category (top, bottom, and shoe). Analyzing each category, the generated top matches with the compatible outfit set 1, resulting in a compatibility score of 1. However, the generated bottom does not appear in any of the compatible outfits, yielding a compatibility score of 0. On the other hand, the generated shoe corresponds to the shoe in the compatible outfit set 3, giving a compatibility score of 1. Therefore, the compatibility score will be 13+03+13 = 0.66.

- Outfit similarity score

The outfit similarity score is used to evaluate our model using an evaluation dataset representing our platform's items. This score assesses the similarity between the generated outfits from our platform's items and the outfits in the training dataset. To calculate the similarity score, we begin by finding the most similar clothing item in the training dataset based on the input image. From this similar item, we identify the compatible outfits it belongs to in the training dataset. Next, we generate full outfits using our platform's items for each category. For the bottom, shoe, and accessory categories, we measure their similarity with the corresponding items in the compatible outfit sets using cosine similarity. Finally, we average these similarity scores for each generated item. The Outfit Similarity Score provides a comprehensive evaluation of how closely our model's generated outfits align with the outfits present in the training dataset. Example of the calculation workflow of outfit similarity score in evaluating our model is shown in the image below.

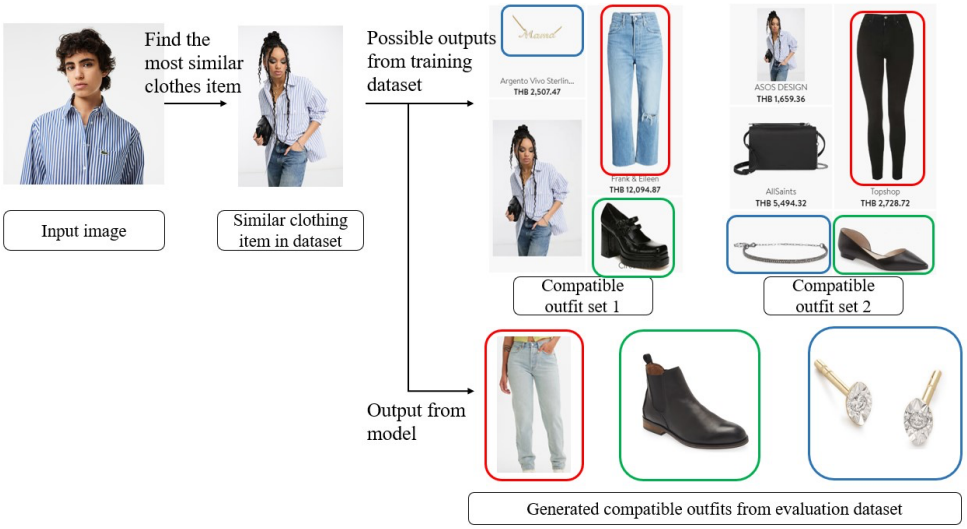


Figure 22 : *Example of the calculation workflow of outfit similarity score*

- Human compatibility perception Accuracy Metric (Qualitative)

We defined a metric to measure the accuracy of the system's compatibility predictions. We calculated the percentage of correct predictions for a given dataset of fashion combinations labeled from "incompatible" to "compatible" with 0 to 5 scale from a sample group of customers (50 peoples) and fashion professionals (10 peoples).

6. Results

The model is trained several times to perform hyperparameter tuning in order to find the optimal model.

6.1 Influence of batch size

From previous semester experimentation and literature review, we found out that increasing batch size for models trained with triplet loss affects the score tremendously. Therefore, we start by finding the largest batch size that our computational power allows. Image size and final embedding dimension, triplet loss margin are fixed at 224x224, 64, and 0.1 respectively.

Batch size	64	96	128	256
Compatibility score	0.2298	0.2574	0.2850	0.3061
Outfit similarity score	0.6202	0.6413	0.6752	0.6984

Table 3 : *Compatibility score and Outfit similarity score from fine-tuning the batch size.*

The red highlighted cells show the optimal batch size that gives the best evaluation score.

6.2 Influence of input image size

For input image size, we use the same image size from the previous semester which is 224x224.

6.3 Influence of image embeddings dimension

Using a batch size of 256, we test different final embedding dimensions to find the largest dimension that our computational power allows. As a result we use the final dimension of 64.

6.4 Influence of triplet loss margin

Three different triplet loss margins are tested to find the optimal triplet loss margin. Batch size, image size, and embedding dimension are fixed at 256, 224 x 224, and 64 respectively.

Table showing the effect of triplet loss margins

Margin	0.5	0.2	0.1

Compatibility score	0.2426	0.3257	0.3118
Outfit similarity score	0.6490	0.7056	0.6987

Table 4 : *Compatibility score and Outfit similarity score from fine-tuning the triplet loss margin.*

Different combinations of hyperparameters are tested to get the model that gives the best results within our computation power limit.

The finalized model hyperparameter is as follows:

- Batch Size: 256
- Image Size: 224 x 224
- Embedding dimension: 64
- Margin: 0.2

6.5 Result from testing the model with training dataset

The evaluation results for the compatibility model after training with IQON dataset and employing triplet loss as an optimization objective are shown in Table 5 and Table 6.

Compatibility score	**Outfit similarity score**
0.3257	0.7056

Table 5 : *Overall accuracy of the trained compatibility model*

Query image	Predicted compatible outfits
	
	

Table 6 : *Examples of predicted compatible outfits*

To summarize, the performance of our model resulted in a low compatibility score and a high outfit similarity score. This suggested that the model has difficulty accurately predicting the same things as those in the training set. However, the model recommended clothing that looked similar to the training outfit. Upon manual analysis, we noticed that the majority of the predicted items had colors that were similar to the input query image, rather than proposing appropriate items with different colors. As a result of this scenario, the compatibility score was low and the outfit similarity score was high. Examples of this scenario can be seen in Table 7.

Query image	Training compatible outfits
	
	Predicted compatible outfits
	

Table 7 : *Examples of training compatible outfits compared with predicted compatible outfits which resulted in a low compatibility score and a high outfit similarity score*

In this example, considering the bottom item, the training dataset suggested that the shirt is compatible with the pink skirt. However, when predicting outfits based on an image, the model mainly focuses on matching colors rather than different colors. As a result, it suggests a gray skirt as a compatible option, which is not the same as the training item, leading to a compatibility score of 0. On the other hand, these 2 skirts are similar in the embedding space, resulting in a high outfit similarity score.

6.6 The result comparison between each model (Pre-trained Resnet 50, Similar Image retrieval model, Compatibility model) on our platform items

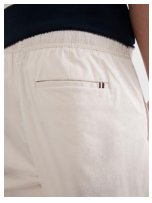
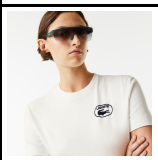
Query image	model	Result				
	Pre-trained Resnet 50					
	Similar Image retrieval model					
	Compatibility model					

Table 8 : *Examples of recommendation for a white pants from different models*

Query image	model	Result

	Pre-trained Resnet 50					
	Similar Image retrieval model					
	Compatibility model					

Table 9 : *Examples of recommendation for a black sweater from different models*

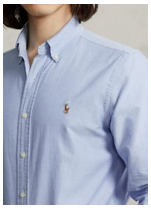
Query image	model	Result				
	Pre-trained Resnet 50					
	Similar Image retrieval model					
	Compatibility model					

Table 10 : *Examples of recommendation for a blue shirt from different models*

6.7 The result from testing the model with testing dataset

The evaluation results for the compatibility model using the testing dataset are shown in Table 11.

Compatibility score	**Outfit similarity score**
0.2238	0.7922

Table 11 : *Overall accuracy of the trained compatibility model with testing dataset*

6.8 The result from testing the model with our platform's item dataset

Since our platform's dataset contains only clothing items not the entire outfit, we evaluated the model with only outfit similarity score. The evaluation results for the compatibility model using the testing dataset are shown in Table 12.

Outfit similarity score
0.6834

Table 12 : *Overall accuracy of the trained compatibility model with our platform's item dataset*

7. End Results

First KPI : The first KPI is to evaluate the compatibility score based on the suggested item generated by our model and it should be more than 0.5.

Evaluated dataset	**Compatibility score**
testing dataset	0.2238

Table 13 : *Compatibility score evaluated from testing dataset*

Second KPI : The second KPI is to evaluate the similarity score based on the suggested item generated by our model and it should be more than 0.5.

Evaluated dataset	**Outfit similarity score**
testing dataset	0.7922
our platform's item dataset	0.6834

Table 14 : *Outfit similarity score evaluated from testing dataset and our platform's item dataset*

Third KPI : The third KPI is to evaluate the accuracy of the system's compatibility predictions. We calculated the average score of 7 suggested outfit sets from the compatibility model for a given dataset of fashion combinations labeled from "least compatible" to "most compatible" with 1 to 5 scale from a sample group of customers (50 peoples) and fashion professionals (5 peoples). The average score of 7 suggested outfit sets that are filled in by both 50 sample groups and 5 fashion professionals should be more than 4.

The results from platform users and fashion professional model compatibility surveys are shown in Table 15.

- Women's Outfit Score by sample group of customers

SET	**Color (30% weighted)**	**Style (30% weighted)**	**Balance (40% weighted)**	**Total after weight**
1	3.92	3.84	3.8	3.848
2	3.92	4.08	4.04	4.016
3	4.36	4.28	4.2	4.272
4	4.16	4.08	4.08	4.104
5	4.12	3.88	4	4
6	4.68	4.48	4.72	4.636
7	4.56	4.56	4.36	4.48

Table 15 : *The average score between 7 sets of women's outfit*

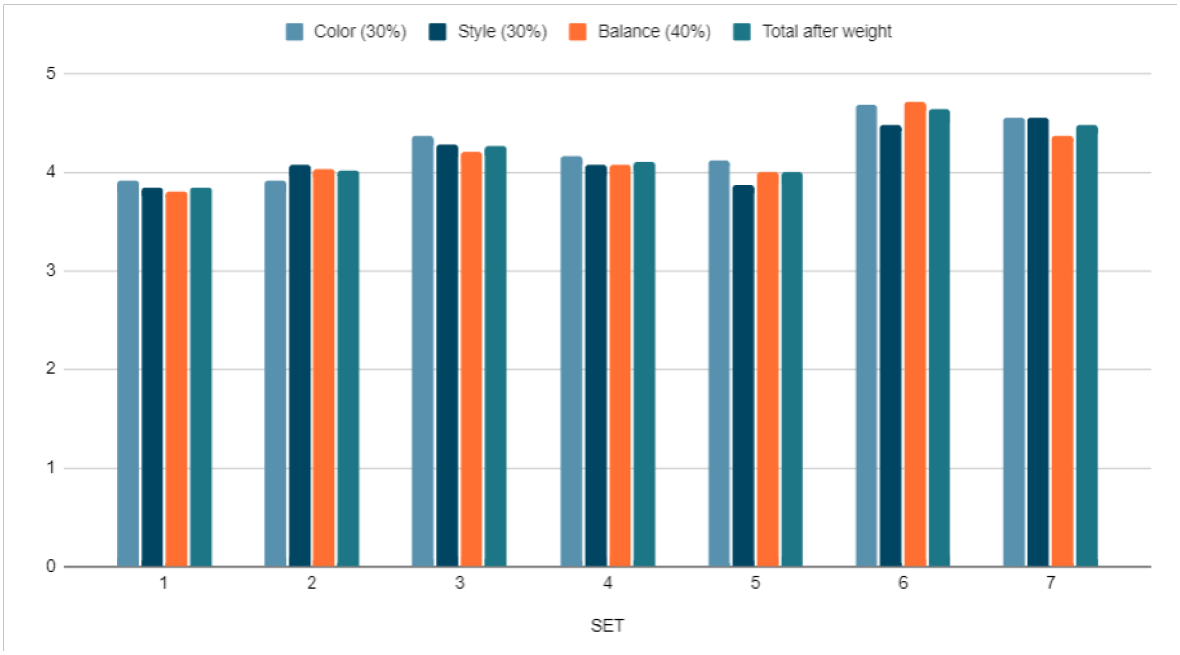


Figure 23 : *Bar graph shows the average score between 7 sets of women's outfit*

- Men's Outfit Score by Sample group of customers

SET	**Color (30%)**	**Style (30%)**	**Balance (40%)**	**Total after weight**
1	3.68	3.56	3.8	3.692
2	4.4	3.64	3.72	3.9
3	4.04	4	3.8	3.932
4	4.08	4.2	4.24	4.18
5	4.44	4	4.28	4.244
6	4.16	3.88	3.96	3.996
7	4.56	4.28	4.28	4.364

Table 16 : *The average score between 7 sets of men's outfit*

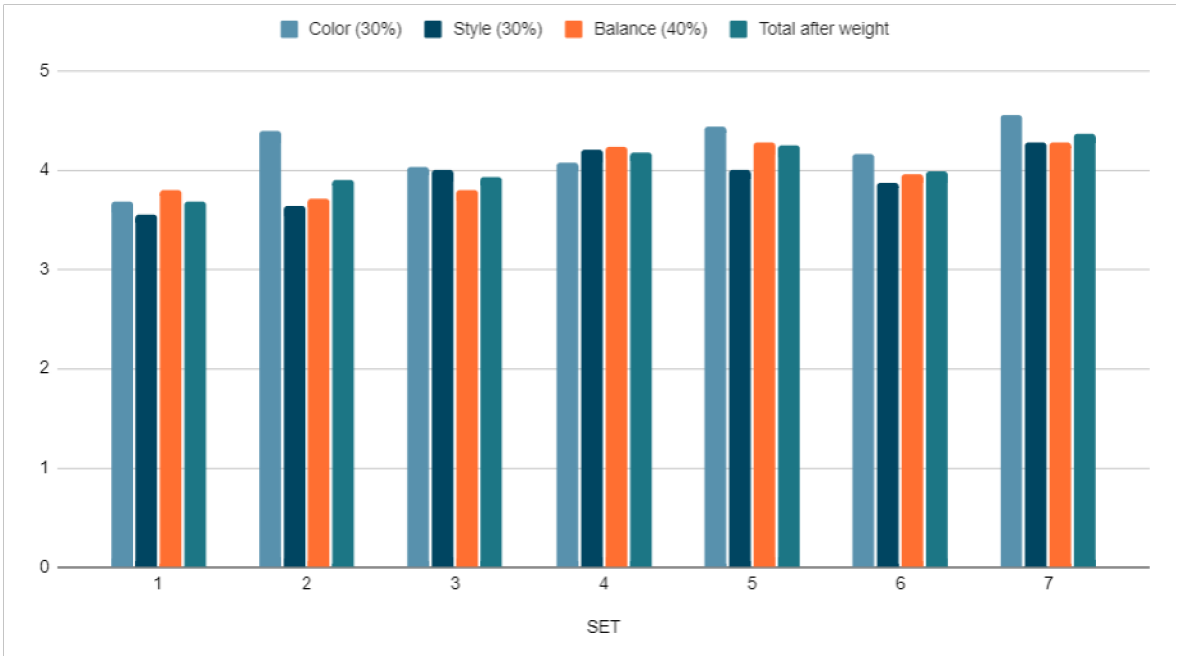


Figure 24 : *Bar graph shows the average score between 7 sets of men's outfit*

- Women's Outfit Score by Fashion professionals

SET	**Color (30%)**	**Style (30%)**	**Balance (40%)**	**Total after weight**
1	4.333333333	4	4	4.1
2	4.333333333	4	4	4.1
3	4	4.333333333	4.666666667	4.366666667
4	4.333333333	4.333333333	4.333333333	4.333333333
5	4.333333333	3.666666667	3.666666667	3.866666667
6	5	4.666666667	5	4.9
7	4.666666667	4.333333333	4.333333333	4.433333333

Table 17 : *The average score between 7 sets of women's outfit*

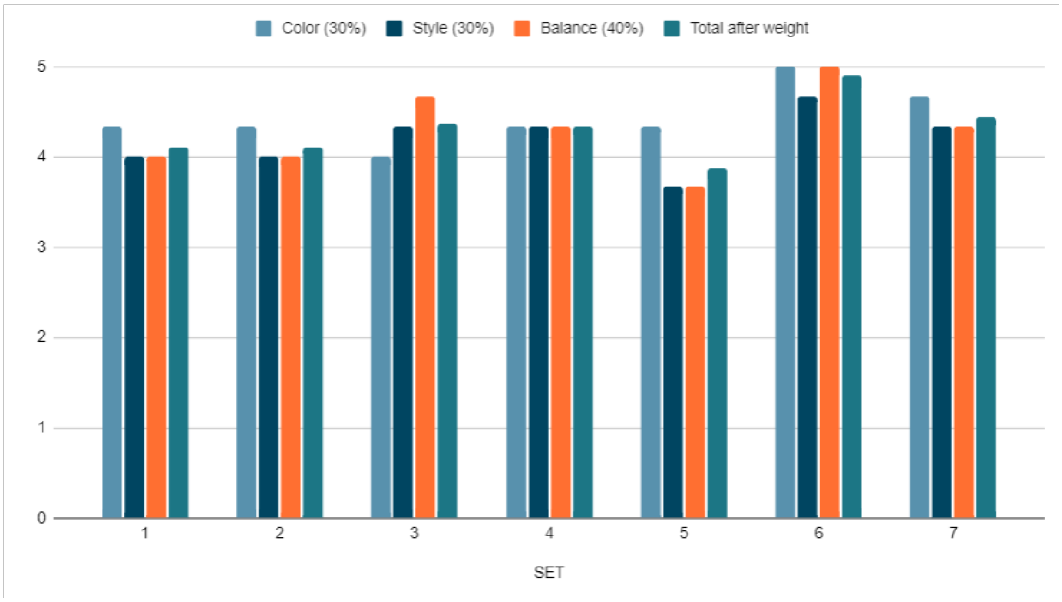


Figure 25 : *Bar graph shows the average score between 7 sets of women's outfit*

Average score of 7 suggested sets from compatibility model (full scale of 5)		
Sample group of customers		Fashion professionals
Women	Men	
4.193714286	4.044	4.3
4.118857143		**4.3**

Table 18 : *Comparison between survey result from platform sample group of customers and fashion professionals*

From the survey, we can see that the average score of the 7 sets of compatibility tests in suggested items after querying the desired image both from a sample group of customers and fashion professionals are 4.12 and 4.3 respectively. These scores reached our KPI of 4 points in full scale out of 5 successfully.

Consequently, we've successfully achieved our second and third Key Performance Indicators (KPIs) outlined in our objectives. Despite not fully meeting our first KPI with regards to accuracy, the predictive capacity of our platform remains commendable. While there's room

for improvement, the overall predictive results still align acceptably with our anticipated outcomes. Therefore, we consider this to be a positive step in our journey towards optimal performance.

References

- Tf.keras.callbacks.reducelronplateau : tensorflow V2.12.0. TensorFlow. (n.d.). https://www.tensorflow.org/api_docs/python/tf/keras/callbacks/ReduceLROnPlateau - Moindrot, O. (2018, March 19). *Triplet loss and online triplet mining in tensorflow*. Olivier Moindrot blog. https://omoindrot.github.io/triplet-loss
- Tensorflow. (n.d.). *Tensorflow/tensorboard: Tensorflow's visualization toolkit*. GitHub. https://github.com/tensorflow/tensorboard
- Igareta, A. (2021, July 28). *Taking the tensorboard embedding projector to the next level*. Medium. https://towardsdatascience.com/taking-the-tensorboard-embedding-projector-to-the-next-l evel-bde53deb6bb7
- Rink, K. (2023, April 4). *Mean average precision at K (map@k) clearly explained*. Medium. https://towardsdatascience.com/mean-average-precision-at-k-map-k-clearly-explained-53 8d8e032d2 https://towardsdatascience.com/mean-average-precision-at-k-map-k-clearly-explained-53 8d8e032d2
- Nagda, R. (2019a, November 8). *Evaluating models using the top N accuracy metrics*. Medium. https://medium.com/nanonets/evaluating-models-using-the-top-n-accuracy-metrics-c0355 b36f91b
- Khare, T. (2020, June 10). *Custom object detection using darknet*. Medium. https://towardsdatascience.com/custom-object-detection-using-darknet-9779170faca2 - Yolov4: Optimal Speed and accuracy of object detection - arxiv.org. (n.d.). https://arxiv.org/pdf/2004.10934.pdf
- Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016, May 9). *You only look once: Unified, real-time object detection*. arXiv.org. https://arxiv.org/abs/1506.02640
- Pramoditha, R. (2021, December 6). *How RGB and grayscale images are represented in numpy arrays?*. Medium. https://towardsdatascience.com/exploring-the-mnist-digits-dataset-7ff62631766a
- Medium. (n.d.). https://towardsdatascience.com/neural-network-embeddings-explained-4d028e6f0526 Understanding-transfer-learning-for-deep-learning
- *Siamese neural network*. Siamese Neural Network - an overview | ScienceDirect Topics. (n.d.). https://www.sciencedirect.com/topics/computer-science/siamese-neural-network

- Brownlee, J. (2021, January 26). *A gentle introduction to object recognition with deep learning*. MachineLearningMastery.com. https://machinelearningmastery.com/object-recognition-with-deep-learning/
- A semi-supervised learning approach for automatic detection and fashion ... (n.d.-a). https://www.researchgate.net/publication/362701696_A_Semi-Supervised_Learning_Approach_for_Automatic_Detection_and_Fashion_Product_Category_Prediction_with_Small_Training_Dataset_Using_FC-YOLOv4/fulltext/62fc6f43aa4b1206fab8b848/A-SemiSupervised-Learning-Approach-for-Automatic-Detection-and-Fashion-Product-Category-Prediction-with-Small-Training-Dataset-Using-FC-YOLOv4.pdf
- Javed, M. (2020, November 4). *Using cosine similarity to build a movie recommendation system*. Medium. https://towardsdatascience.com/using-cosine-similarity-to-build-a-movie-recommendation-system-ae7f20842599
- Hervé Jegou, M. D. (2018, June 28). *Faiss: A library for efficient similarity search*. Engineering at Meta. https://engineering.fb.com/2017/03/29/data-infrastructure/faiss-a-library-for-efficient-similarity-search/
- Koehrsen, W. "Neural network embeddings explained" *Towards Data Science,* 2 October 2018, https://towardsdatascience.com/neural-network-embeddings-explained-4d028e6f052
- M. Hadi Kiapour, Xufeng Han, Svetlana Lazebnik, Alexander C. Berg, Tamara L. Berg "Where to Buy It: Matching Street Clothing Photos in Online Shops" *CVF Open Access*, 18 Feb 2016, https://openaccess.thecvf.com/content_iccv_2015/papers/Kiapour_Where_to_Buy_ICCV_2015_paper.pdf
- Ming Du, Arnau Ramisa, Amit Kumar K C, Sampath Chanda, Mengjiao Wang, Neelakandan Rajesh, Shasha Li, Yingchuan Hu, Tao Zhou, Nagashri Lakshminarayana, Son Tran,Doug Gray "Amazon Shop the Look: A Visual Search System for Fashion and Home." *ACM DL Digital Library*, 14 August 2022, https://assets.amazon.science/2a/43/43c7dd6b4bffbfe72de627456a76/amazon-shop-the-lo ok-a-visual-search-system-for-fashion-and-home.pdf
- Sanghyuk Park, Minchul Shin, Sungho Ham, Seungkwon Choe, Yoohoon Kang. "Study on Fashion Image Retrieval Methods for Efficient Fashion Visual Search." *CVF Open Access*,

https://openaccess.thecvf.com/content_CVPRW_2019/papers/FFSS-USAD/Park_Study_on_Fashion_Image_Retrieval_Methods_for_Efficient_Fashion_Visual_CVPRW_2019_paper.pdf#page9

- Usbser. (n.d.). *USBSER/yolov3_darknet: Convolutional Neural Networks*. GitHub. https://github.com/usbser/yoloV3_darknet
- Sarıgöz, Y. “Triplet loss - advanced Intro” *Towards Data Science*, 25 March 2022, https://towardsdatascience.com/triplet-loss-advanced-intro-49a07b7d8905
- Ziwei Liu, Ping Luo,Shi Qiu, Xiaogang Wang, Xiaoou Tang. “DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations.” *CVF Open Access*, https://openaccess.thecvf.com/content_cvpr_2016/papers/Liu_DeepFashion_Powering_Robust_CVPR_2016_paper.pdf#page6