ST456: Deep Learning Group Project
Complete the Look: Scene-based Complementary Product Recommendation

Candidate Number

50270 [1] **50691** [1] **37984** [1] **45278** [1]

Department of Statistics, London School of Economics and Political Science

Abstract

This paper replicates a complex contexual fashion recommender system proposed by Kang et al., identifying products that complement a given scene image. We developed the model architecture from the ground up in Keras and trained it on the Complete the Look dataset. Despite computational limitations, we trained, evaluated and interpreted our model using various performance metrics and visualisations, achieving good predictive performance similar to the original paper.

1. Introduction

The fashion industry is rapidly evolving, increasingly integrating machine learning and artificial intelligence to enhance customer-centric fashion analysis. Applications range from virtual ”try-on” functionalities to personalised recommendation of products based on user sentiment. An alternate approach involves contexualising the product recommendation process, by assisting shoppers to assemble their outfit by suggesting items that complement their existing look. To enable this system, we can employ two types of images: scene images (typically sourced from social media) and product images (found on online shopping websites). This raises the following question: *given a scene, can we recommend complementary fashion products for different categories (e.g. clothes, shoes, accessories), to help the shopper complete their look?*

Since this scene-product compatibility system is founded on real-world use cases, it allows for more widespread adoption among shoppers. Rather than searching for official product images on the internet, users can simply upload their own personal images of the outfits they want to style. This makes the system more practical for everyday styling and fashion decisions. To understand and replicate Kang et al., we explore important relevant literature in the field including the Shop-The-Look (STL) paper in Section 2 and in Section 3 we prepare the Complete-The-Look (CTL) data by processing STL data. In Section 4, we establish the model architecture for extracting and learning global embeddings from scene and product images, and local embeddings from the regional segments of the scenes. Furthermore, we illustrate computations for scene-product compatibility (global and local) with category-based attention weights, and the triplet loss functions using anchor, positive and negative images for training the model. Section 5 details the implementation of the methods described above. In Sections 6 and 7 we evaluate and interpret model performance in terms of accuracy, mean reciprocal rank, product-scene compatibility scores for positive and negative products, t-SNE plots and attention maps. Lastly, in Sections 8 and 9 we suggest areas for further research and provide the overall conclusion.

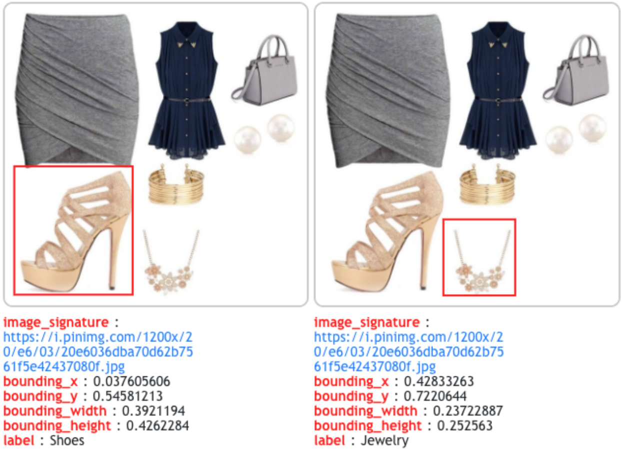
2. Related Work

Since the pioneering work of Chopra et al. on Siamese networks, these architectures have found natural applications in visual similarity and style Bell & Bala and Veit et al.. In summary, these papers explore Siamese network models trained on pairs of similar/complementary and dissimilar/non complementary images and they attempt to learn an embedding space where similar/complementary products are close together. Upon the appearance of a new product, they check which products are similar by applying a nearest-neighbour search on the embedding space. Shiau et al. implements the idea of recommending similar products to the ones that appear on the image at a production level for Pinterest.

Furthermore, Liu et al. proposed a Street2Shop or Shop The Look (STL) – given a scene image and a bounding box surrounding the query product, the authors attempted to extract similar-looking or identical products. They also utilised human-labeled datasets to estimate the product similarities; we adopted their STL datasets and modified it by cropping (see Section 3). Additionally, we adopt their idea to help our model learn a notion of complementarity instead of similarity of products to the scene image. The paper we replicate – Kang et al. – introduces context in the form of full images, whereas all the previous literature performed product to product comparisons without regard to the scene in context. Kang et al. argue that context is important for providing accurate product recommendations and thus embrace a training scheme where the scene/context assigns a distance metric to both a positive (originally in the scene but cropped out) product and a negative (not in the original scene) product. In this manner, they design a practical recommender system based on the images users upload.

3. Data Preparation

To support our experiments on the CTL task, we used the STL-Fashion dataset, a modified version of STL-10 tailored for fashion. This dataset contains labeled images of individual fashion items such as t-shirts, pants, shoes, and coats. Although STL-Fashion doesn't include full-scene outfit images, we recreated a Pinterest-style recommendation environment using structured metadata. Each metadata entry includes a product ID, a scene ID, and a normalized bounding box indicating where the product appears within the scene image. An auxiliary file provides category labels (e.g., "shoes", "coats") that are used during preprocessing to group and organize the data. Using the product and scene IDs, we constructed image URLs for both product and scene images. For each product-scene pair, we extracted the corresponding image URLs, bounding box coordinates, and category labels. Only entries with accessible images, verified through HTTP requests, were retained for further processing.



image_signature :
https://i.pinimg.com/1200x/20/e6/03/20e6036dba70d62b7561f5e42437080f.jpg
bounding_x : 0.037605606
bounding_y : 0.54581213
bounding_width : 0.3921194
bounding_height : 0.4262284
label : Shoes

image_signature :
https://i.pinimg.com/1200x/20/e6/03/20e6036dba70d62b7561f5e42437080f.jpg
bounding_x : 0.42833263
bounding_y : 0.7220644
bounding_width : 0.23722887
bounding_height : 0.252563
label : Jewelry

Figure 1. Product-scene compatibility scores for the scene anchor image against the positive and negative product images

3.1. Triplet Dataset Preparation

To support training under contrastive learning objectives such as triplet loss, we construct a dataset of triplets with an anchor (a scene image that provides contextual style), a positive product appearing within the scene (stylistically compatible) and a negative product from the same category but associated with a different scene (stylistically incompatible). Triplet construction involves grouping products by category (e.g., footwear, topwear) to ensure category-level consistency across triplets. For each valid scene-product pair, an anchor–positive combination is formed, where the anchor is the scene image and the positive is the corresponding product appearing within it. A negative product is then randomly sampled from the same category but linked to a different scene, ensuring it is stylistically incompatible with the context of the anchor. The data statistics after preprocessing are as follows: 29,429 scene images, 38,098 product images and 72,173 pairs, where each pair contains a compatible scene and product.

4. Methodology

Following Kang et al., the aim of our methodology is to learn the scene-product style compatibility, between pairs of a scene image I_s and a product image I_p provided in the dataset. The aim of modelling is to construct global and local embeddings; the former represents the overall style of the scene while the latter captures a more refined, region-specific style of the scene by focusing on different spatial areas within I_s. We adopt a two-pronged approach to assess product-scene compatibility at a global and local level – to evaluate how well the product aligns with the overall scene as well as its most relevant local regions.

4.1. Architectures

We utilise a ResNet-50 architecture pre-trained on ImageNet to conduct feature extraction for the scene and product images I_s and I_p. This base model consists of all the convolutional layers and the weights are frozen during training due to the limited size of the datasets, thus retaining the representations learned from ImageNet. To obtain the global feature vector for the scene $\mathbf{v}_s \in R^{d_1}$, we pass I_s through the ResNet-50 and extract the final layer (`conv5_block3_out`). We perform the same operation to obtain the product feature vector $\mathbf{v}_p \in R^{d_1}$. To capture key context from the local regions of the scene, we extract the feature map $\{\mathbf{m}_i \in R^{d2}\}_{i=1}^{w\times h}$ from intermediate convolutional layers (`conv4_block6_out`).[1]This modelling choice was an attempt to further focus on the most salient features in the scene when considering local, granular regions of I_s.

To create embeddings, we pass the feature vectors and maps into a two-layer feed forward network $g(\Theta;\cdot)$ with the

[1] This resultant layer required reshaping to achieve the final dimension of (7,7) per image, via an additional Global Max Pooling layer. In the original paper they used a ResNet50 version that wasn't available in Keras and so we made this minor modification to the original architecture.

following architecture: *Linear-Batch Normalisation-Relu-Dropout-Linear-L2Norm*. This transformation is detailed in equation 1 and gives rise to the global scene embedding $\mathbf{f}_s$, product embedding $\mathbf{f}_p$ and the local scene embedding $\mathbf{f}_i$.[2]

$$\begin{aligned} \mathbf{f}_s = g(\Theta_g; \mathbf{v}_s), \quad \mathbf{f}_p = g(\Theta_g; \mathbf{v}_p), \\ \mathbf{f}_i = g(\Theta_l; \mathbf{m}_i), \quad \hat{\mathbf{f}}_i = g(\Theta_t; \mathbf{m}_i) \end{aligned} \tag{1}$$

4.2. Compatibility & Loss Function

The objective is to evaluate the hybrid compatibility of scene-product pairs, which includes the global and local compatibility. First, global compatibility is measured by computing the l_2 distance between global scene embedding $\mathbf{f}_s$ and the product embedding $\mathbf{f}_p$; where nearby embeddings indicate high compatibility.

$$d_{\text{global}}(s,p) = \|\mathbf{f}_s - \mathbf{f}_p\|^2, \tag{2}$$

Since the scene image consists of several objects, solely considering the global landscape may lead to overlooking key features for a scene-product match. Thus, we also focus on the local compatibility.

$$\begin{aligned} d_{\text{local}}(s,p) = \sum_{1\le i\le w\times h} a_i \|\mathbf{f}_i - \mathbf{f}_p\|^2, \\ \hat{a}_i = -\|\hat{\mathbf{f}}_i - \hat{\mathbf{e}}_c\|^2, \quad \mathbf{a} = \text{softmax}(\hat{\mathbf{a}}) \end{aligned} \tag{3}$$

After creating the local style embeddings $\mathbf{f}_i$, we use a category-aware attention mechanism to determine which local regions are most relevant for recommending a product of a particular category. This mechanism assigns weights a_i to each region based on its relevance to the product category. The weights are determined based on the proximity of the local region to the product category dependent bias term ($\hat{\mathbf{e}}_c$)[3]. That is, a_i is calculated as the l_2 distance between $\hat{\mathbf{f}}_i$ and $\hat{\mathbf{e}}_c$. We negate this distance such that the higher similarity scores (small Euclidean distance) are rewarded with larger attention weights. These scores are then passed through a softmax layer which ensures that the distances are normalized and thus can be interpreted as an attention measure.

Lastly, local compatibility is computed as the weighted sum of the l_2 distance between each scene region's embedding f_i and product embedding f_p, with attention weights a_i, as shown in equation 3.

The overall hybrid compatibility score is then defined as a combination of both global and local distances, allowing the model to consider both the overall style of the scene and the compatibility of specific regions with the product being recommended.

$$d_*(s,p) = \frac{1}{2}\left[d_{\text{global}}(s,p) + d_{\text{local}}(s,p)\right] \tag{4}$$

For the loss function, we compute a custom hinge loss that measures the triplet loss given the anchor scene image (s), positive image (p^+) and negative image (p^-).

$$\mathcal{L} = \sum_{(s,p^+,p^-)\in\mathcal{T}} \max\left(d_*(s,p^+) - d_*(s,p^-) + \alpha, 0\right) \tag{5}$$

This helps train our models to iteratively learn to decrease the distance between the anchor scene and positive image and the anchor scene and negative image, maintaining α as the margin. In this case, the distance criterion is the hybrid compatibility measure in equation 4. Therefore, the lower this compatibility score for the positive product, the smaller the distance between the scene and the positive product – which is precisely what is required for accurate model predictions.

The margin is present to ensure that the model learns a representation where the positive pairs are significantly closer than the negative; this should help the performance generalise well to previously unseen pairs.

5. Implementation

The architecture – visualised in Appendix A – was implemented using Keras. Due to the complex nature of the architecture, we developed our own custom layers. However, the main difficulty associated with using these custom layers was that the manner in which we programmed them meant that the model could not be trained using GPU[4]. This added a large layer of complexity to the training phase, since the model had 15M trainable parameters.

Following Kang et al., we train this model using Adam and a minibatch with size of 16 over 100 epochs and pick the best model using a validation set evaluated on the triplet loss with a margin of 0.2. The dimension of the embeddings is chosen to be 128 ($\mathbf{f}_i, \mathbf{f}_s, \mathbf{e}_c, \mathbf{f}_p \in \mathbb{R}^{128}$). Due to high computational requirements, we[5] trained our model for 50 epochs using 200 minibatches per epoch that were randomly shuffled for

[2] Note that our notation implies parameter sharing for the global embeddings, similar to Siamese networks. On the other hand, there is no parameter sharing for the local embeddings.

[3] $\hat{a}_i$ is defined like a decoder with the bias term giving it some extra flexibility.

[4] Our best guess is that we used pure python loops inside the layers and the GPU was unable to handle that.

[5] None of the group members had a computer with a powerful enough CPU, but we gained access to an external MacBook Pro Max with a high-performance CPU to train the model.

every epoch. While we did not use a separate validation set, we evaluated the model based on its performance at the final epoch. For all the other hyperparameters, we follow the original paper, including the embedding dimensions.

6. Numerical Results

To evaluate the model we used 10 percent of the sample (7217) as a test set and utilized three main evaluation metrics: training loss, accuracy of binary comparisons and Mean Reciprocal Rank.

Primarily, while training our model for 50 epochs, we monitored the training loss based on the custom triplet loss function seen in equation 5.

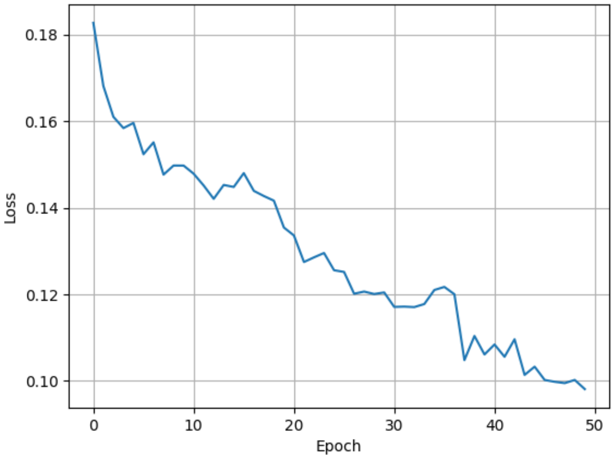


Figure 2. Line chart demonstrating the training loss per epoch

As seen in figure 2, the training loss decreases gradually from 0.18 to 0.09 over the span of 50 epochs, indicating stable and consistent convergence. The contrastive loss-style function encourages the model to learn to discriminate between the positive and negative product embeddings with respect to the scene embeddings. It does so by maximising the compatibility between a scene and positive product (minimising $d_*(s, p^+)$), and minimising the same for the negative product (maximising $d_*(s, p^-)$). Thus, as the loss decreases, the condition $d_*(s, p^+) + \alpha \leq d_*(s, p^-)$ starts to become "binding"[6].

The training plot is informative because we see that in the initial epochs the model does not have a "fashion sense", both distances (positive and negative) are largely equal and the loss during this phase primarily arises from the enforced margin, $\alpha = 0.2$. As the model learns and the margin stays the same, we see the loss decreasing because either the positive distance becomes smaller or the negative one becomes larger.

However, a downside of the training process is observed in figure 2; while there are drastic reductions in the training loss until epoch 40, the model performance is rather stable between epochs 40 to 50. Therefore, an improvement would be to monitor the validation loss[7] to identify which epoch (upper bound) would be sufficient for an early stopping criterion. This would help safeguard the model against the risk of overfitting.

Given our model, we computed the accuracy of the binary comparisons on the test dataset; that is, given a scene, positive and negative products, the model needs to determine which product is more coherent with the scene image. It is defined as follows:

$$\text{Binary accuracy} = \sum_{i=1}^{n_{\text{test}}} \mathbb{1}(d(s,p)_i < d(s,n)_i) \tag{6}$$

Our model achieved a high binary comparison accuracy at 69% in comparison to Kang et al. achieving 70% accuracy on their test set. Hence, for 69% of all the triplets in our test instances, the distance between the scene and positive product is smaller than the distance between the scene and negative product.

To further examine the predictive accuracy of the model, we consider the Mean Reciprocal Rank (MRR) metric; to evaluate how well the model captures the compatibility of scene-product pairs, across different product categories.

$$\text{MRR} = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{\text{rank}_q} \tag{7}$$

We let Q be the set of test instances, where $|Q| = 7217)$. For each test instance, the model is given a scene image and a set of 10 candidate products consisting of 1 positive product, denoted by q, and 9 randomly sampled negative products.[8] We evaluate the compatibility scores of each of these 10 products with the scene image and rank them, with rank_q denoting the rank position of the positive product in the sorted list of candidate products. We then compute the reciprocal rank $\frac{1}{\text{rank}_q}$ of the positive product. In general, we repeat this process for all Q test instances and compute the final average of the reciprocal ranks of the positive images. Overall, the idea is that the MRR measures how highly the positive products are ranked among all the possible negative

[6] It would be binding when loss reaches 0.

[7] In general, we did not have sufficient motivation to use a validation set because we did not conduct any hyperparameter tuning for the model.

[8] The total number of products to rank was chosen guided by computational concerns; it took over 1 hour to get the 72170 distances from the model.

products. For example, in a list of 10 products, the best reciprocal rank (product with highest compatibility score) is 1/1 = 1, whereas the worst reciprocal rank is 1/10 = 0.1.

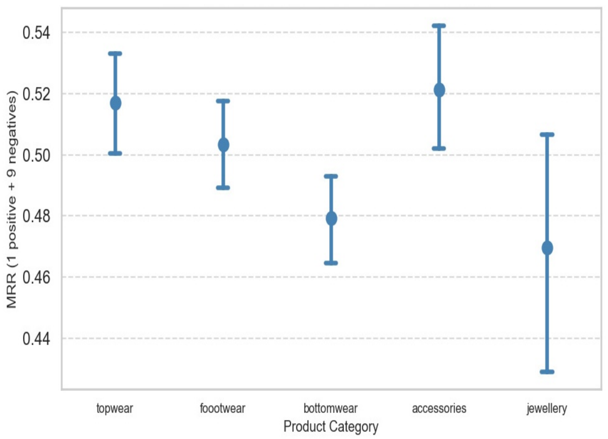


Figure 3. Scatter plot indicating Mean Reciprocal Rank with bootstrap confidence intervals per product category

Figure 3 shows the MRR scores for different product categories; topwear, footwear, bottomwear, accessories and jewellery. Notably, all categories perform well with MRR scores ranging from 0.47 to 0.52. These scores suggest that, on average, when a product is required to be paired with a given scene image, the correct product is within the top two model predictions. This is inferred from inversing the reciprocal ranks - 0.52 approximately has rank 2 $(0.5 = \frac{1}{\text{rank}_q} = 1/2)$. Hence, the model is highly effective in determining compatible products for a given scene image, consistently achieving high compatibility scores and thus ranking the positive product among the top model predictions.

Notably, accessories have the highest MRR at 0.52 indicating that in situations where accessories have to be paired with the scene image, the model enables this match accurately. This could be due to distinctive visual features or a clearer regional context from the scene image that helps identify which accessories are necessary. Figure 3 also demonstrates error bars showing the 95% bootstrap confidence intervals derived from computing the MRR for all 7217 test instances. Although accessories has a high MRR, it also has a wide confidence interval relative to the other categories, implying that this category faces higher variability in model performances across the test samples.

Jewellery and bottomwear exhibit the lowest MRRs, connoting that it is more difficult for the model to assign these products correctly to the scene. This could be attributed to the smaller size of the products in case of jewellery, or partially obscured products in case of bottomwear (e.g. not fully visible in the scene). While bottomwear has the narrowest confidence interval demonstrating stable model performance across samples, jewellery experiences the widest interval across all categories, suggesting its volatile performance. Potential improvements could be to use a more granular filter in the CNN backbone, or a more granular regional grid for the scene image, such that the category-aware attention weights can be increased for smaller regions containing jewellery.

In (Kang et al., 2019) the authors use top K-accuracy to evaluate the model. They define it as “how often the top-K retrieved items contain the ground-truth product”. This implies that for a given scene image we would have to compute distances with all the available products in the database/test set check if the true pairs are in the top-k and repeat this process for all pairs in our test set. This is much more expensive than the traditional approach (Veit et al., 2015),(Bell & Bala, 2015). In said approach the products are in an embedding space and when a new product/query appears, the model places it in the embedding space and looks for the nearest neighbours (in the embedding space) for recommendations. This is likely a motive of why the recommendation system Pinterest (Shiau et al., 2020) implements at scale, does not include contextual information.

7. Interpretation

To understand whether the model results makes sense, we conduct some sanity checks using three methods: displaying examples of test instance triplets to understand whether the model's product recommendation works, t-SNE plots to visualise the proximity of the positive and negative images to the scene image, and attention maps to illustrate whether the attention weights are being calculated correctly.

Firstly, we consider the model's product recommendations shown in figure 4 for three randomly selected triplets from the test set. Following from our previous discussion on a test accuracy score of 69%, this figure shows that in two out of the three instances, the model would recommend the positive product over the negative product, whereas in one instance it would fail to do so. This can be observed in the first triplet – while the model should ideally recommend the blue heels since they are stylistically more appropriate for the outfit in the scene image, it recommends the golden heels. We can confirm this selection because it provides the positive product with a higher compatibility score than the negative product – that is, the positive product has a larger distance (1.301) from the scene image $d_*(s, p^+)$ than the negative product $d_*(s, p^-)$ (1.157).

For the other two examples, the model correctly assigns a lower compatibility score to the positive images – in the second triplet, the black leather bag with score 1.092 is well-

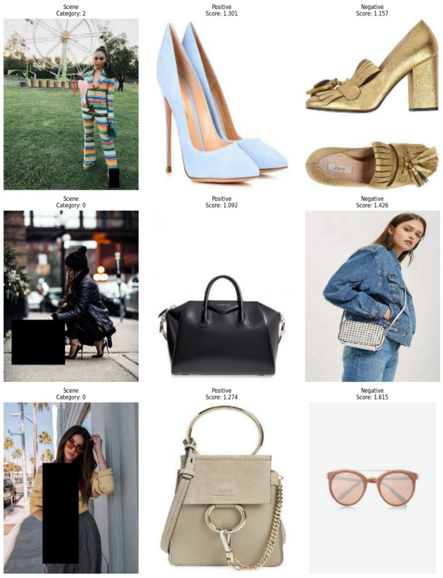


Figure 4. Model's product recommendations along with product-scene compatibility scores for the scene against the positive and negative products

assigned to the all-leather outfit, while the white sling bag is not. Similarly, in the third triplet, the model detects that the scene most likely requires a purse than sunglasses (because they are already contained in the image), and assigns them correct scores of 1.274 and 1.615 respectively. Hence, this validated that our model's product recommendations are appropriate for a given scene.

Following this, we present a t-SNE plot for Category 3 (jewellery) visualising the global embeddings space for 8 triplets[9] in the test set. Similar visualisations for the remaining categories can be found in Appendix B. These embeddings are the output of the two-layer feed forward network $g(\Theta; \cdot)$ following the ResNet-50 backbone and are of size $\mathbb{R}^{128\times 1}$ (final embedding dimension), which we compress in two dimensions using t-SNE to visualise if the scene is closer to the positive product.

In figure 5 we observe that for given scene embeddings in the 2D space, the associated positive product embeddings tend to be closer than the negative product embeddings. This implies that the final compatibility score (equation 4: average of the global and local distances) for the positive product will be lower, indicating that the model has learned to pair the correct product with the scene.

A point to consider is that these global embeddings do not take into account the local embeddings extracted from the intermediate layers of the ResNet-50. However, it can be seen that the good performance of the global embeddings – positive products are closer to scene than negative products – is consistent with the good performance of triplet loss function, due to the declining training loss as seen in figure 2. Hence, an extension would be to visualise the contributions of both global and local embeddings to the functioning of the hybrid compatibility measure, because with figure 5 it appears that the global embeddings are sufficient for the model to correctly distinguish between positive and negative products. Overall, this sanity check validates that the hybrid compatibility measure and triplet loss function are working as expected.

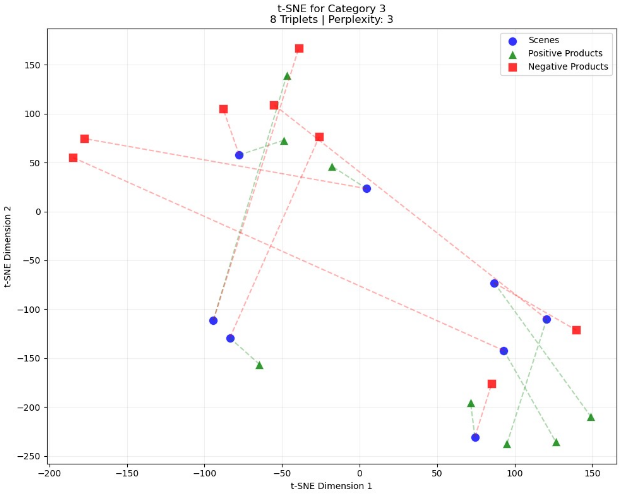


Figure 5. t-SNE plot on sample test data indicating the proximity of the positive and negative product embeddings to the scene embeddings

Lastly for interpretation, we visualise the attention map to intuitively demonstrate which parts of a scene image are important for predicting the complementarity of the positive product with the remaining outfit. Figure 6 highlights the regions of the scene image that receive high attention weights, indicated by bright focal points, while less relevant areas with low attention weights appear darker. We observe that our model tends to focus on meaningful regions, such as the footwear and bottomwear in the first scene, footwear in the second scene and topwear in the third scene. Interestingly, the attention mechanism we employed successfully focuses more on clothing than human faces, suggesting that our model understands that clothing is more relevant for recommending a complementary product than the appearance of the individuals.

However, the limitation is that the attention weights do not fully detect all the relevant parts of the scene image; they do not highlight all the regions in which clothing is present, and while they do avoid majority of the scene's background, some of it is in the spotlight in all of the images. The most likely explanation for why the attention does not seem

[9] We randomly selected 8 triplets to allow the visualisation to be more comprehensible

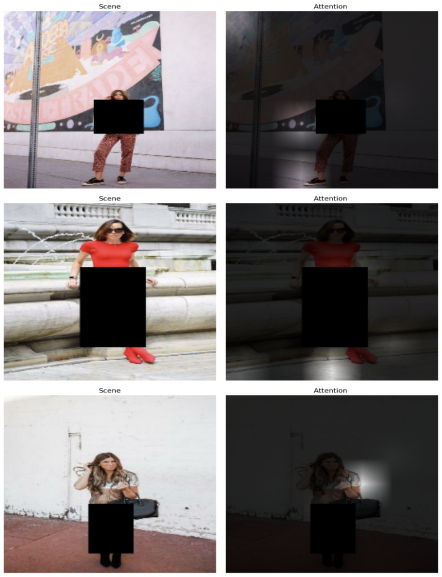


Figure 6. Attention map revealing areas of the scene image deemed important for prediction by attention weights

to be fully working is that we did not train the model on enough data, as in the original paper the authors obtain better attention maps.

8. Limitations and extensions

Even though our baseline performed well, we could not experiment with other types of optimisers, losses, architectures and training regimes. This was due strong computational limitations and the time spent debugging[10].

Few extensions that could help improve the CTL framework involve attention mechanisms and loss functions. Firstly, as observed in figure 6, a more well-suited attention mechanism may be required to fully detect all the relevant parts of the scene image. For example, multi-head self-attention or spatial transformer networks could be employed to better discriminate between and isolate clothing from the background regions. A finer grid may also be considered to allow the attention mechanism to detect finer products such as jewellery – the product category that received the lowest MRR score (Figure 3). Furthermore, explicitly supervising the attention map during training could help the model learn to focus on more semantically relevant areas for product recommendation. For example, segmentation masks – different colours for different categories labelled on the scene image by human annotators – can be used to build a reward/punish system, inspired by reinforcement learning. If training the attention modules encourages the model to focus on more clothing pixels, it is rewarded else penalised.

Exploring alternative loss functions could also enhance the performance of the CTL framework. While we employed the triplet loss function to learn discriminative embeddings, future work could consider losses such as Proxy-NCA, ArcFace, and Contrastive loss. Proxy-NCA uses a representative point (called a proxy) for each class instead of comparing every possible triplet of images. This makes training faster and more efficient, especially when there are many different types of scenes or items. ArcFace enforces angular margins between classes, which produces more compact and well-separated embeddings, which could help the model make finer distinctions between similar-looking clothing items. Contrastive loss, on the other hand, simplifies the learning objective by working on pairs of samples, bringing similar ones closer and pushing dissimilar ones apart, and is often used in Siamese networks. These alternative loss functions could help the model learn in different ways, possibly improving its ability to generalise and understand scenes more effectively.

We also wanted to implement a Siamese network architecture[11], the architecture in itself is relatively simple but the data processing to transform the dataset into product pairs [12] would have been complex. Finally, previous literature (Veit et al., 2015) train their Siamese network on pairs from different categories. It would be interesting to try a similar approach in our case, instead of training on triplets where both products are from the same category.

9. Conclusion

In this paper, we presented our implementation of Complete The Look (CTL), a scene-based complementary product recommendation system, which learns scene-product compatibility through global and local embeddings. CTL can be adopted by e-commerce platforms to provide their users with a virtual fashion advisor, simply enabled by the users providing scene images as input. Thus, the system offers practical support for everyday outfit styling. We constructed a dataset of triplets with an anchor scene image providing the contextual style, positive compatible product, and negative incompatible product. ResNet-50 was adopted as the backbone model for extracting feature vectors, which were then processed by a two-layer feed-forward neural network to generate local and global embeddings for the scene and

[10] Noticing that the issue lay with GPU use and not the architecture in itself took us 3 days.

[11] Kang et al. include a Siamese baseline that performs very well, but they do not provide implementation details.

[12] Full siamese architecture only makes sense if we are embedding pairs of products into a style space; it is unclear how to infuse the scene information in this context and thus we assume that the authors most likely trained their model after transforming the data into product pairs.

products. Category-aware attention mechanism and hybrid (global and local) compatibility metrics were constructed to be used within a triplet loss function, helping the model embed the scene and positive product pair closer than the respective negative pair. The model achieved high accuracy (69%) and competitive MRR scores across all categories (0.47 - 0.52), suggesting that the correct product is typically ranked among the top two recommendations. Model output and interpretability was further verified using test triplets, t-SNE plots, and attention maps. While the model performs well overall, certain limitations — particularly in capturing fine-grained details like jewellery, and capturing the correct regions for attention weights – suggest room for improvement. Future work may focus on enhancing regional granularity, refining attention mechanisms, and trying other model architectures such as Siamese networks. These enhancements could make the system even more precise and practical for real-world fashion applications.

10. Individual Contributions

- 45278: Focused on the evaluation and interpretability aspects of the scene-based recommendation system. Implemented multiple visualization techniques including attention map overlays, t-SNE and UMAP projections of embeddings, and score comparisons across categories to qualitatively and quantitatively assess the model's behavior. Proposed several visualisations for demonstrating the model's product recommendations and attention maps. This enabled a deeper understanding of how the final model responds to different scene inputs and validated the alignment between model attention and recommendation quality.

- 50270: Conducted an in-depth literature review on existing scene-based recommendation techniques, studied compatibility measures, and explored contrastive learning frameworks. Implemented the attention mechanism, local and global compatibility measures and tailored the contrastive loss function to a triplet loss function for this recommendation task, helping guide the model architecture choices. Also developed a draft of the Siamese Network architecture, however due to time constraints, we were unable to run this model on our dataset. Played a key role in evaluating and interpreting model output, recognising areas for technical improvement, and writing and organising the final project report – ensuring clarity and coherence in the presentation of the team's work.

- 50691: Developed the MLP layers that outputted the final embeddings used for the compatibility measures. Connected/reformatted all the custom layers we created, ensuring they were compatible when used together in the final model. Made sure the model could be saved properly; Keras was very specific about saving and reloading custom models. Conducted debugging on the issue we had with the GPU training. Identified the bug but was unable to fix it. Helped with the binary comparison and MRR performance metrics.

- 37984: Handled the coding, model training, and data preparation efforts for the recommendation system. He was responsible for preprocessing raw scene and product data to ensure consistency and model compatibility. He generated ResNet-based embeddings for scene images, which formed the foundational input representation for the recommendation model. His work ensured that the dataset was clean, structured, and ready for training, allowing for smooth integration of downstream components. He contributed to iterative performance improvements through extensive testing and debugging. His work provided a solid foundation for subsequent model iterations and contributed to a clearer understanding of model behavior in the early stages of development.

References

Bell, S. and Bala, K. Learning visual similarity for product design with convolutional neural networks. *ACM Trans. Graph.*, 34(4), July 2015. ISSN 0730-0301. doi: 10.1145/2766959. URL https://doi.org/10.1145/2766959.

Chopra, S., Hadsell, R., and LeCun, Y. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pp. 539–546 vol. 1, 2005. doi: 10.1109/CVPR.2005.202.

Kang, W.-C., Kim, E., Leskovec, J., Rosenberg, C., and McAuley, J. Complete the Look: Scene-Based Complementary Product Recommendation . In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10524–10533, Los Alamitos, CA, USA, June 2019. IEEE Computer Society. doi: 10.1109/CVPR.2019.01078. URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2019.01078.

Liu, S., Song, Z., Liu, G., Xu, C., Lu, H., and Yan, S. Street-to-shop: Cross-scenario clothing retrieval via parts alignment and auxiliary set. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3330–3337. IEEE, 2012.

Shiau, R., Wu, H.-Y., Kim, E., Du, Y. L., Guo, A., Zhang, Z., Li, E., Gu, K., Rosenberg, C., and Zhai, A. Shop the look: Building a large scale visual shopping system at pinterest. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pp. 3203–3212, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3403372. URL https://doi.org/10.1145/3394486.3403372.

Veit, A., Kovacs, B., Bell, S., McAuley, J., Bala, K., and Belongie, S. Learning Visual Clothing Style with Heterogeneous Dyadic Co-Occurrences . In *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 4642–4650, Los Alamitos, CA, USA, December 2015. IEEE Computer Society. doi: 10.1109/ICCV.2015.527. URL https://doi.ieeecomputersociety.org/10.1109/ICCV.2015.527.

A. Model architecture details

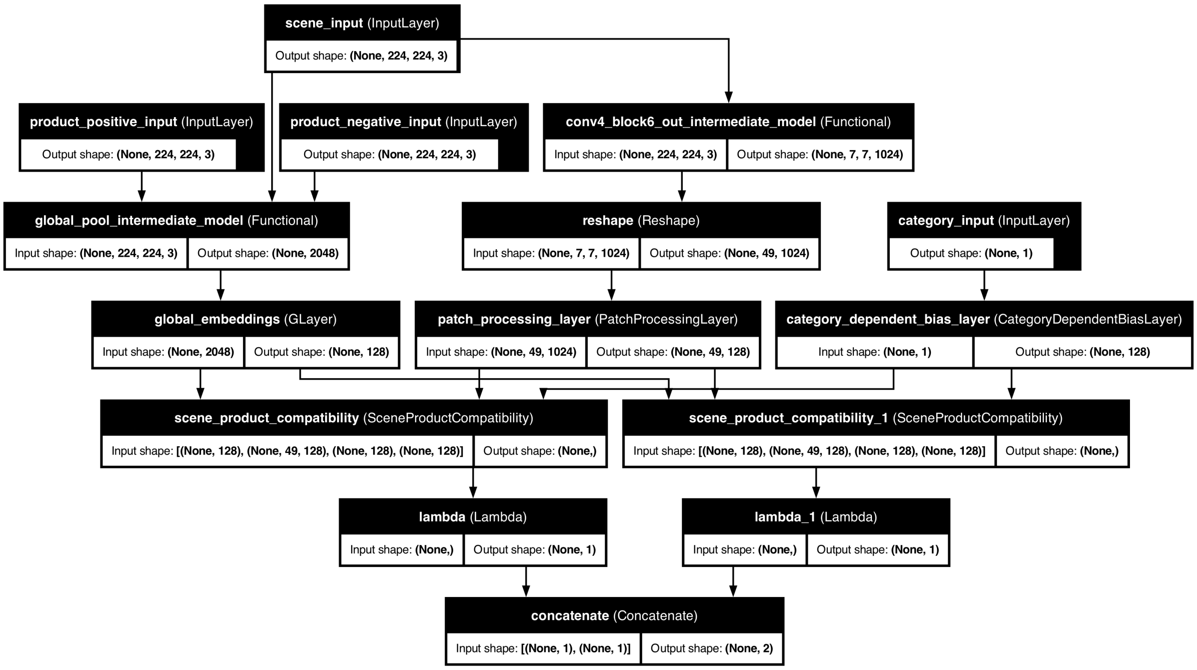

scene_input (InputLayer)
Output shape: (None, 224, 224, 3)
product_positive_input (InputLayer)
Output shape: (None, 224, 224, 3)
product_negative_input (InputLayer)
Output shape: (None, 224, 224, 3)
conv4_block6_out_intermediate_model (Functional)
Input shape: (None, 224, 224, 3)
Output shape: (None, 7, 7, 1024)
global_pool_intermediate_model (Functional)
Input shape: (None, 224, 224, 3)
Output shape: (None, 2048)
reshape (Reshape)
Input shape: (None, 7, 7, 1024)
Output shape: (None, 49, 1024)
category_input (InputLayer)
Output shape: (None, 1)
global_embeddings (GLayer)
Input shape: (None, 2048)
Output shape: (None, 128)
patch_processing_layer (PatchProcessingLayer)
Input shape: (None, 49, 1024)
Output shape: (None, 49, 128)
category_dependent_bias_layer (CategoryDependentBiasLayer)
Input shape: (None, 1)
Output shape: (None, 128)
scene_product_compatibility (SceneProductCompatibility)
Input shape: [(None, 128), (None, 49, 128), (None, 128), (None, 128)]
Output shape: (None,)
scene_product_compatibility_1 (SceneProductCompatibility)
Input shape: [(None, 128), (None, 49, 128), (None, 128), (None, 128)]
Output shape: (None,)
lambda (Lambda)
Input shape: (None,)
Output shape: (None, 1)
lambda_1 (Lambda)
Input shape: (None,)
Output shape: (None, 1)
concatenate (Concatenate)
Input shape: [(None, 1), (None, 1)]
Output shape: (None, 2)

Figure 7. Tree diagram exhibiting each stage of the model architecture from the scene and product inputs to the CNN models, global and local embeddings and the scene-product compatibility measures.

B. t-SNE Plots for all Product Categories

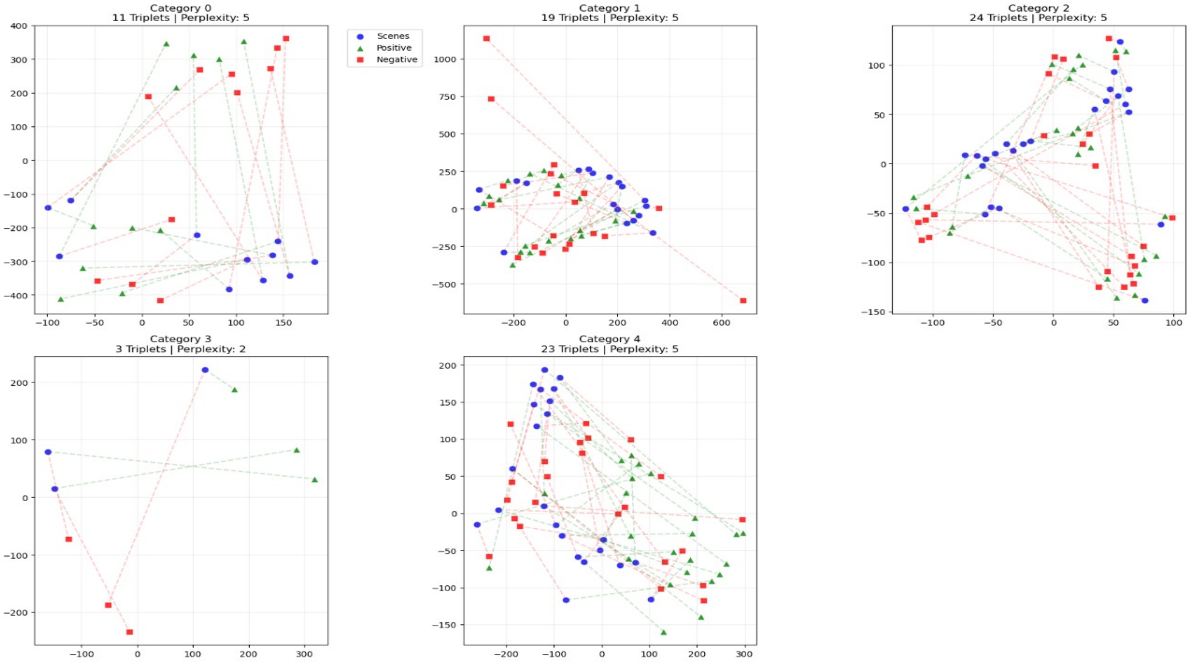


Figure 8. t-SNE Plots the following categories: accessories (0), bottomwear (1), foootwear (2), jewellery (3), topwear (4)