OutfitTransformer: Learning Outfit Representations for Fashion Recommendation

Rohan Sarkar[1,2], Navaneeth Bodla[2], Mariya I. Vasileva[2], Yen-Liang Lin[2], Anurag Beniwal[2], Alan Lu[2], and Gerard Medioni[2]

[1]Purdue University, West Lafayette, [2]Amazon

Abstract

Learning an effective outfit-level representation is critical for predicting the compatibility of items in an outfit, and retrieving complementary items for a partial outfit. We present a framework, OutfitTransformer, that uses the proposed task-specific tokens and leverages the self-attention mechanism to learn effective outfit-level representations encoding the compatibility relations between all items in the entire outfit for addressing both compatibility prediction and complementary item retrieval. For compatibility prediction, we design an outfit token to capture a global outfit representation and train the framework using a classification loss. For complementary item retrieval, we design a target item token that additionally takes the target item specification (in the form of a category or text description) into consideration. We train our framework using a proposed set-wise outfit ranking loss to generate a target item embedding given an outfit, and a target item specification as inputs. The generated target item embedding is then used to retrieve compatible items that match the rest of the outfit. Additionally, we adopt a pre-training approach and a curriculum learning strategy to improve retrieval performance. Experiments show that our approach outperforms state-of-the-art methods on compatibility prediction, fill-in-the-blank, and complementary item retrieval tasks.

1. Introduction

Two main tasks for a fashion outfit recommendation system are fashion *compatibility prediction* and large-scale *complementary item retrieval*. For compatibility prediction (CP), the task is to determine whether a set of fashion items in an outfit go well together. For complementary item retrieval (CIR), the task is to complete a partial outfit by finding a compatible item from a large database. Given an outfit, we want to predict how well its constituent items go together. Also, given a partial outfit with different items (such as a bag, shoes, and pants) and a target item description (e.g., "top"), we want to retrieve compatible items to complete the outfit. Figure 1 illustrates our proposed method.

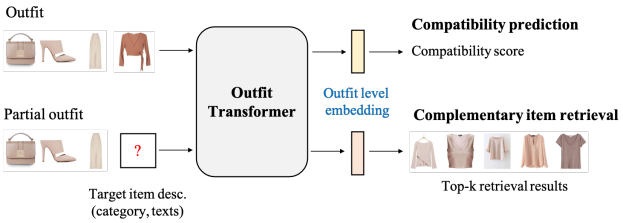


Figure 1. OutfitTransformer learns an outfit-level representation for a set of outfit items to address the CP and CIR tasks. For CIR, it learns a single embedding encoding overall compatibility of the partial outfit, and a target item description that is used to retrieve compatible items cohesively matching the entire outfit using KNN search. For CP, it learns an outfit-level representation capturing overall outfit compatibility to predict a compatibility score.

Prior work such as [28, 27, 25, 22, 34] addresses the pairwise item-level compatibility problem and achieves state-of-the-art results but does not explicitly model outfit-level compatibility. Some methods optimize for compatibility at an outfit-level [9, 6, 11, 12, 5]. However, these approaches are mainly designed for classification tasks: compatibility prediction and fill-in-the-blank (FITB) but they do not address the large-scale CIR task. CSA-Net [16] proposes a method for large-scale CIR, but it does not learn an outfit-level representation that can explicitly capture compatibility of a target item to the outfit as a whole. It searches compatible items for each item in the outfit at a paired-category level (e.g., top to shoe, bottom to shoe) and fuses the ranking scores for the query items to obtain the final rankings.

Instead, our idea is to learn an outfit-level representation for both compatibility prediction and large-scale retrieval of complementary items. Here, we investigate a transformer-

Emails: sarkarr@purdue.edu, navaneeth.bodla@getcruise.com, {vamariy, yenliang, beanurag, alalu, medioni}@amazon.com

based architecture to learn the outfit-level embeddings for the CIR task, as this architecture shows better outfit compatibility prediction performance than the Bi-LSTM [9] and GCN-based [6] architectures (cf. Table 1). Specifically for CIR, the target item should cohesively match all the existing items in an outfit. Using outfit-level representations can more effectively capture complex feature correlations among multiple items in the outfit, as opposed to considering pairs of items at a time. Additionally, users may specify their preference for the target item in the form of a target category (e.g., top) or the text descriptions (e.g., full sleeve shirt with floral design), and our system is able to retrieve the complementary items that match it. We design our framework to learn a target item embedding that operates at the outfit-level and encodes both the overall compatibility of a partial outfit, and the target item specification. We pose this as a set-to-item compatibility learning problem where we model the outfit as a set of items, and extract a single target item embedding to search for complementary items.

The outfit representation learnt for both tasks is invariant to the order of the items, i.e., permutation of the order of the outfit items should generate the same representation. The transformer is a suitable choice for our framework because it captures the higher-order relationships (beyond pairwise) between all constituent items in the outfit and is able to take unordered items as input.

For the task of CP, we train the OutfitTransformer with a classification loss and design an outfit token to capture a global outfit representation that encodes the compatibility relationships among all the items in the outfit. For CIR, we design a target item token that encodes both the compatibility of the partial outfit and a target item description to generate the embedding of the target item. This embedding is used to retrieve compatible items from a database. We train our framework using a proposed set-wise outfit ranking loss, which encourages compatible items to be embedded closer to the overall representation of a set of outfit items. Our design allows extraction of a single target item embedding enabling large-scale indexing and retrieval. [1]

Directly training on the retrieval task leads to poor performance, since the network does not have any prior knowledge regarding compatibility of the partial outfit. To alleviate this problem, we facilitate OutfitTransformer to learn compatibility relationships by pre-training it on the CP task. We find that this improves retrieval performance significantly (cf. Table 4(a)). In addition, we propose a curriculum learning strategy to hierarchically sample more informative negative examples which further boosts retrieval performance (cf. Table 4(b)).

[1]The framework needs to be designed in a way such that it allows individual item embedding extraction (which should not depend on the query image during indexing like SCE-Net [22]) to support large-scale indexing for KNN search. (cf. Sec 3.2.3).

We evaluate our method on the public Polyvore Outfits dataset [25]. Experimental results show that our approach outperforms state-of-the-art techniques in compatibility prediction, fill-in-the-blank (FITB), and complementary item retrieval tasks. In Section 4, we demonstrate that our framework can retrieve complementary items based on the target item category or description.

In summary, our main technical contributions are:

- We propose a new framework, OutfitTransformer, that effectively learns outfit-level representations, which is shown experimentally to outperform state-of-the-art methods on both compatibility prediction (CP) and complementary item retrieval (CIR) tasks.
- We propose task-specific tokens to support both CP and CIR. For CP, the outfit token is designed to capture a global outfit representation. For CIR, the target item token additionally takes the target specification (in the form of category or free-form text) into account.
- Our framework learns a single embedding that enables large-scale indexing and retrieval for complementary items, and has smaller indexing size than previous approaches ([16, 25]) which use subspace embeddings.
- We provide in-depth analysis of different design choices (pre-training and curriculum learning) to improve retrieval performance.

2. Related Work

Outfit Compatibility Prediction. Prior work on fashion outfit compatibility often considers pairwise item comparisons and aggregates item-level scores to predict the final compatibility score [28, 27, 25, 22, 34]. To add global constraints, a number of methods [9, 6, 12, 5] aggregate inputs from all constituent items. Han *et al*. [9] use a BiLSTM to model outfit composition as a sequential process, considering outfits as ordered sequences. However, outfit compatibility should be invariant to the order of items. Some recent approaches [6, 7] use a graph convolutional network (GCN) for CP. Cucurull *et al*. [6] train a GCN that generates embeddings conditioned on the representations of neighboring nodes and predict the outfit compatibility but require large neighbor information for best performance, which is impractical for new items as mentioned in [16]. Cui *et al*. [7] model an outfit as a graph, where each node represents a category and each edge represents interaction between two categories. Chen *et al*. [5] (*hrta.* POG) propose a transformer-based encoder-decoder architecture for generating compatible outfits that are specifically designed for personalization based on historical clicks data. All these approaches [9, 6, 7, 5] are mainly designed for classification tasks (CP and FITB [9]) but do not address large-scale CIR.

Outfit Complementary Item Retrieval. Although the CP score in prior methods can be used for ranking items, it is impractical to do so in a large-scale setting. The framework must support indexing to avoid linearly scanning the entire database. Lin *et al*. [16] addresses large-scale CIR and retrieves items by considering compatibility between the target item and every item in an outfit in a pairwise manner, and then aggregating the scores. However, they do not consider the outfit as a whole and only use attention at a paired-category level. In contrast, we use a transformer model to capture interactions between all the items in an outfit to learn a global outfit representation. Also, our method has a much smaller indexing size than [16] which is important for practical applications (cf. Section 3.2.3). Lorbert *et al*. [17] use a single layer self-attention based framework for outfit generation but do not explicitly model compatibility.

Attention-Based Methods and Vision Transformers (ViT): Transformers have been used in a wide variety of computer vision tasks [3, 30, 18, 32]. ViT [8] and related models [24, 33, 4] decompose each image into an ordered sequence of smaller patches to learn image representation of a single image. On the contrary, we model outfits as an unordered set of different item images and use a transformer-based architecture to learn a global outfit representation that captures overall outfit compatibility. As discussed earlier, attention mechanisms [16, 23, 5, 17] have also been used in fashion recommendation systems. [16, 23] use attention to understand complementary relationships in a pairwise manner. In contrast, we use a transformer model to learn interactions between all the items in an outfit, which attends to higher-order compatibility relationships [13] beyond pairwise [25, 16, 22]. Both POG [5] and [17] use pretrained ImageNet embeddings, while we learn fashion-specific features by training in an end-to-end manner.

Distance Metric Learning: CIR is different from visual similarity search because the complementary item is from a different category and is visually dissimilar from the other items in an outfit. We specifically design a negative sampling strategy for CIR where we sample negatives from the same category as positives, unlike [20, 21]. Also, in contrast to [22, 25, 16], which consider pair-wise compatibility between target items and each individual item in the outfit, we propose a set-wise outfit ranking loss that compares target items with a single embedding for the entire outfit.

3. Proposed Approach

Figure 2 illustrates the overview of our framework. Our framework takes as input each outfit's constituent item images and their text descriptions. For CP, we train the transformer encoder to generate a global outfit representation that can capture higher-order compatibility relationships between all items in the outfit beyond pairwise relationships. This global outfit representation can then be used to predict an outfit compatibility score (details in Section 3.1).

For CIR, given a partial outfit and a target item description (e.g., product category or description), we train the transformer encoder to generate a target item embedding, which can be used to retrieve items that are compatible with the partial outfit and match the target item description. The framework is trained using a proposed ranking loss that enforces the target item embedding to move closer to the positive item and further apart from the negative items. The positive item matches the global style of the overall outfit, whereas the negatives are incompatible with the outfit (details in Section 3.2).

We investigate different training strategies to improve retrieval performance. We employ a pre-training strategy where we first train the model on the CP task as mentioned in Section 3.2.1. We also adopt curriculum learning to select more informative negative samples in different training stages. The details are presented in the Section 3.2.2.

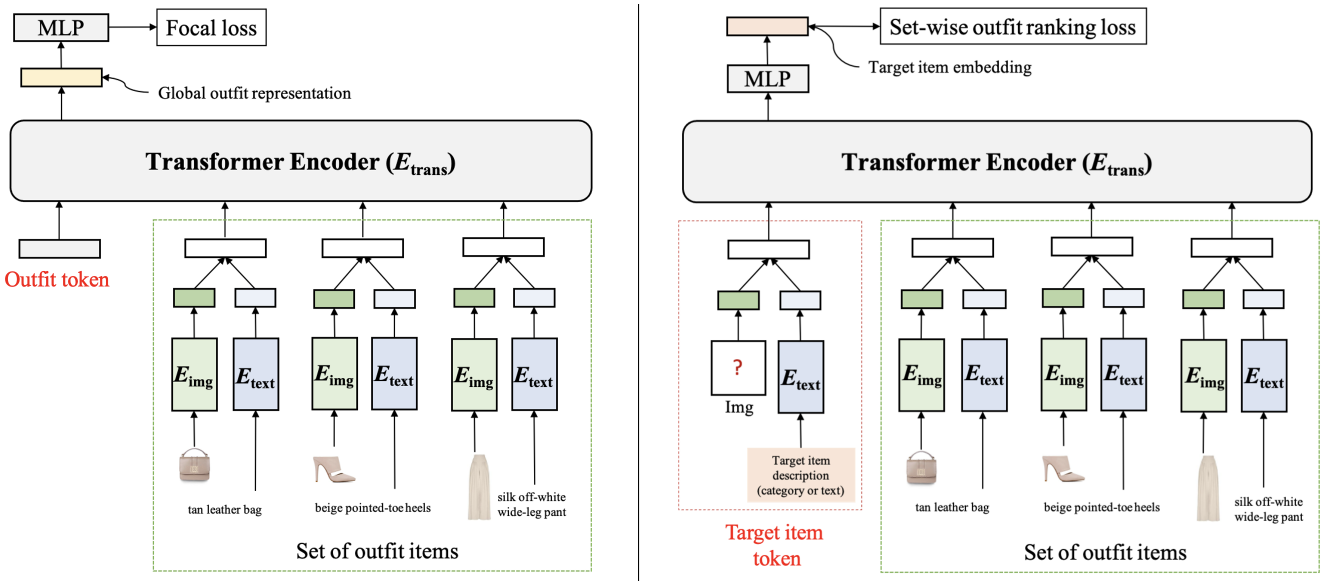
3.1. Fashion Outfit Compatibility Prediction

The compatibility prediction task predicts the compatibility of all the items in an outfit. Given an outfit $O = \{(\mathrm{I}_i, \mathrm{T}_i)\}_{i=1}^{L}$, where I_i is the image, T_i is the corresponding text description for an item i. We learn a non-linear function that predicts a compatibility score in $[0, 1]$, where 1 indicates perfect compatibility.

As shown in Figure 2 (a), the item images and their text descriptions are fed into an image (E_{img}) and text encoder (E_{text}) respectively to extract the image and text feature vectors (see Section 3.3 for details about the image and text encoder architecture). We concatenate the extracted image and text feature vectors to generate an item feature vector $u_i = E_{\mathrm{img}}(\mathrm{I}_i) \parallel E_{\mathrm{text}}(\mathrm{T}_i)$, where $\parallel$ denotes a concatenation operation. The set $F = \{u_i\}_{i=1}^{L}$ represents the feature vectors of all the items in an outfit.

Since the goal of ViT [8] is to produce a classification score for an image, a classifier token is typically used to capture a single image representation from the input image patches. In contrast, we introduce the outfit token whose state at the output of the transformer encoder serves as the global outfit representation. The goal of introducing this token is to learn a global outfit representation that captures compatibility relationships between items in the outfit using the self-attention mechanism. We model outfits as an unordered set of items as the overall outfit compatibility is invariant to the order of the items. Thus, positional encodings used in NLP [26] and ViT [8] are not required for us.

The outfit token ($\mathrm{x}_{\mathrm{Outfit}}$) is a learnable embedding that is prepended to the set of outfit feature vectors F and fed into the transformer encoder $\mathrm{E}_{\mathrm{trans}}$. The state of the outfit token at the output of the transformer encoder serves as the global outfit representation which is subsequently fed into



(a) Compatibility prediction

(b) Complementary item retrieval

Figure 2. System overview of our framework for compatibility prediction and complementary item retrieval. We model outfits as an unordered set of items. We use an image encoder (E_{img}) and a text encoder (E_{text}) to extract the image and text features. (a) For compatibility prediction, we train the transformer encoder using a focal loss [15] and learn a global outfit representation to predict an outfit compatibility score. (b) For complementary item retrieval, given an outfit and a target item description, we train the transformer encoder to learn a target item embedding that can be used for retrieving compatible items to complete an outfit. We train the framework using the proposed set-wise outfit ranking loss in an end-to-end manner. The details of set-wise ranking loss are explained in Section 3.2.2.

the MLP that predicts an overall outfit compatibility score:

$$\mathrm{c} = \mathrm{MLP}\left(E_{\text{trans}}\left(\mathrm{x}_{\text{Outfit}}, F\right)\right) \tag{1}$$

Our framework ($E_{\text{trans}}, E_{\text{img}}, E_{\text{text}}$) is trained end-to-end using focal loss [15].

3.2. Complementary Item Retrieval

The complementary item retrieval task is to retrieve an item that is both compatible with the partial outfit and matches a specified item description to complete the outfit. Specifically, given a set of partial outfit items and a user provided target item specification, the goal is to generate a target item embedding that can be used to retrieve compatible items. Our framework is trained with a proposed set-wise outfit ranking loss (details in Section 3.2.2).

The target item token s (cf. Figure 2 (b)) includes an item description T for the target item that we want to retrieve, and an empty image represented by x_{Img}. The target item token is defined as $\mathrm{s} = \mathrm{x}_{\text{Img}} \parallel E_{\text{text}}\left(\mathrm{T}\right)$.

The intuition behind designing the target item token in this manner is that, during inference, the target image is unknown but users can provide a description for the item they are searching for. We simulate a similar setting when training the framework for the retrieval task. We introduce the target item token whose state at the output of the transformer encoder serves as the target item representation that explicitly takes into consideration both compatibility with the partial outfit, and the target item description. Our framework is generic and the target item description can be provided in different forms such as category, text, tags, etc.

The transformer encoder takes as input the set of feature vectors F of the partial outfit, and the target item specification s, which is subsequently fed into a MLP that generates the target item embedding.

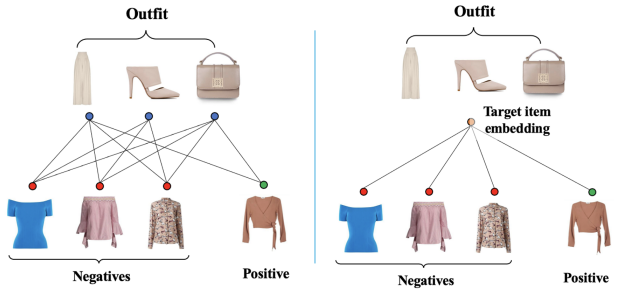
$$\mathrm{t} = \mathrm{MLP}\left(E_{\text{trans}}\left(\mathrm{s}, F\right)\right) \tag{2}$$

To learn this target item embedding, we train our framework with a proposed set-wise outfit ranking loss which is discussed in Section 3.2.2.

3.2.1 Pre-training on Compatibility Prediction

We pre-train the framework on the CP task and use the learned weights to initialize the transformer, image and text encoder for complementary item retrieval. This choice leads to a significant improvement for CIR (cf. Table 4(a)).

We conjecture that the reasons might be: 1) the pre-trained transformer encoder captures compatibility relationships, which is helpful for encoding them into the target item embedding to retrieve compatible items, and 2) the image encoder captures fashion-specific features, which is used to extract better feature vectors for positive and negative samples in the set-wise outfit ranking loss.



(a) Pair-wise outfit ranking loss (b) Set-wise outfit ranking loss

Figure 3. Comparison of pair-wise outfit ranking loss [16] and our set-wise outfit ranking loss. Our framework generates a single target item embedding that captures the compatibility of the entire outfit and does not need pairwise computations with individual items in the outfit as in [16].

3.2.2 Set-wise Outfit Ranking Loss

Previous approaches ([25], [22]) use a triplet loss to learn relationships only between a pair of items but do not consider the relationship between all items in the outfit. To address this, the outfit ranking loss [16] is proposed which considers the pairwise compatibility of target items with all the items in the outfit, as shown in Figure 3 (a). In contrast, our approach generates a single target item embedding t that already captures the compatibility relations for a set of outfit items and hence does not require pairwise comparisons with individual outfit items, as shown in Figure 3 (b).

In practice, only legitimate outfit samples are provided in the dataset, and there are no annotated negative samples. Given an outfit we randomly pick an item as positive and the remaining items as the partial outfit. Here, we investigate a curriculum learning approach to gradually increase the difficulty of negatives for training. Specifically, we train the model in two stages. In the first stage, we sample the negatives from the same high-level category as the positive item. Subsequently in the second stage, we sample harder negatives from more fine-grained categories. Note that since CIR is different from visual similarity search, our negatives are different from the conventional way of constructing triplets (e.g., [20, 21]), where negatives are sampled from other classes.

The set-wise outfit ranking loss is designed to optimize relative distances between samples such that the target item embedding moves closer to the positive embedding and farther apart from the negative embeddings. Note that we use the pre-trained image and text encoders (cf. Sec. 3.2.1) to extract the positive and negative embeddings. Because we have a single target item embedding that encodes the compatibility of the entire outfit, we can directly train our set-wise ranking loss using triplets without requiring pairwise computations as [16]. The set-wise outfit ranking loss is defined as:

$$L\left(\mathrm{t}, p, N\right) = L(\mathrm{t}, p, N)_{All} + L(\mathrm{t}, p, N)_{Hard} \quad (3)$$

$$L(\mathrm{t}, p, N)_{All} = \frac{1}{|N|} \sum_{j=1}^{|N|} \left[d\left(\mathrm{t}, f^{p}\right) - d\left(\mathrm{t}, f_{j}^{N}\right) + m\right]_{+}$$

$$L(\mathrm{t}, p, N)_{Hard} = \left[d\left(\mathrm{t}, f^{p}\right) - \min_{j=1\ldots|N|} d\left(\mathrm{t}, f_{j}^{N}\right) + m\right]_{+}$$

where $[]_{+}$ is the hinge loss, t is the target item embedding, f^{p} is the positive embedding, f_{j}^{N} is the j^{th} negative embedding from the pool of negatives in N, and m is the margin.

The loss has two components as shown in Equation (3). The first component $L(\mathrm{t}, p, N)_{All}$ considers all the sampled negatives for the outfit, while the second component $L(\mathrm{t}, p, N)_{Hard}$ considers the hard negative samples (e.g., [31, 21]). This allows the model to learn discriminatory features to distinguish between items that might have very subtle differences between them. We empirically find that this loss formulation and the hard negative sampling strategy improves complementary item retrieval performance significantly (cf. Tables 4(b), 5). There are other sampling methods that could potentially be used (e.g., [29]), but the investigation thereof is outside the scope of this paper.

3.2.3 Indexing and Retrieval of complementary items

Not all the methods for CP can support indexing for retrieval. For example, SCE-Net [22] requires pairs of images as inputs, which does not allow single item embedding extraction for indexing. We design our framework in a way that allows extraction of individual item feature vectors during indexing and generate a single item embedding during inference. Based on our design, we can use off-the-shelf KNN search tools (e.g., [1, 2]) to perform indexing and retrieval, which makes the search very efficient even for a large database (e.g., with millions of items). Specifically, during indexing, we use the trained image and text encoder to extract the item features. This does not depend on the query images unlike [22]. During inference, given the partial outfit and a target item description, our framework generates a single target item embedding, which is then used to search for compatible items from the database using KNN search.

Our framework offers two advantages as compared to prior works. First, we require smaller indexing size compared to previous approaches that use subspace embeddings. For indexing, Type-aware [25] and CSA-Net [16] generate multiple embeddings of each item for each of the target categories and therefore the indexing size grows linearly with the number of categories. Because we are not learning subspaces, our approach is independent of the number of categories. Second, in [16], for each item in the outfit, a target category-specific embedding is extracted, which is used to retrieve compatible items from the database. This has to be repeated exhaustively for each item

Method	Features	PO-D	PO
BiLSTM + VSE [9]	ResNet-18 + Text	0.62	0.65
GCN (k=0) [14]	ResNet-18	0.67	0.68
SiameseNet [25]	ResNet-18	0.81	0.81
Type-Aware [25]	ResNet-18 + Text	0.84	0.86
SCE-Net [22]	ResNet-18 + Text	-	0.91
CSA-Net [16]	ResNet-18	0.87	0.91
OutfitTransf. (Ours)	ResNet-18	0.87	0.92
OutfitTransf. (Ours)	ResNet-18 + Text	**0.88**	**0.93**

Table 1. Comparison of our model with state-of-the-art methods on the CP task using the AUC metric [9]. The methods are evaluated on Polyvore-Outfits (where -D denotes the disjoint dataset).

in the query outfit. In contrast, our framework can retrieve items in a single step regardless of outfit length.

3.3. Implementation Details

The image encoder uses a ResNet-18 initialized with ImageNet pre-trained weights. The text encoder uses a pre-trained SentenceBERT [19], on top of which we add a fc layer. During training, we finetune the weights of the image encoder and the fc layer of the text encoder. We extract a 64-dimensional image and a 64-dimensional text embedding and concatenate them to generate 128-dimensional item embeddings before feeding them into the transformer encoder. We use a six-layer transformer encoder with 16 heads. For the retrieval task, we set the margin m for the set-wise outfit ranking loss as 2 and sample 10 negatives for each outfit. We use a batch size of 50 and optimize using ADAM with an initial learning rate of $1e^{-5}$ and reducing the learning rate by half in steps of 10.

4. Experiments

We compare our proposed approach with the state-of-the-art baselines such as Bi-LSTM [9], GCN [6], SiameseNet [27], Type-aware [25] , SCE-Net [22] and CSA-Net [16] on the Polyvore Outfits dataset [25]. For evaluation, we compare our method with these state-of-the-art baselines on three different tasks: (1) *Compatibility Prediction (CP) task* that predicts the compatibility of items in an outfit. (2) *Fill in the Blank (FITB) task* that selects the most compatible item for an incomplete outfit given a set of candidate choices (e.g., 4 candidates). (3) *Complementary Item Retrieval (CIR) task* that retrieves complementary items from the database for a target category given an incomplete outfit.

The Polyvore Outfits dataset [25] has two sets, the disjoint and non-disjoint sets. In the disjoint set, the training split items (and outfits) do not overlap with the validation and test splits. In the non-disjoint set, the training split items can overlap with those of validation and test splits, but outfits do not overlap. The non-disjoint set contains 53306 training and 10000 test outfits, while the disjoint set comprises of 16995 training and 15154 test outfits.

For the standard compatibility prediction and FITB tasks, we evaluate our model on the Polyvore Outfits dataset. Since the Polyvore Outfits dataset does not provide the annotations for the complementary item retrieval task, we adopt a modified version of the Polyvore Outfits dataset proposed by CSA-Net [16].

4.1. Outfit Compatibility Prediction (CP)

The goal of this task is to measure the compatibility of an outfit. Our compatibility model in Figure 2 (a) predicts a score that indicates the compatibility of the overall outfit. We compare the performance with the state-of-the-art methods in Table 1 by using the standard metric AUC [9], which measures the area under the receiver operating characteristic curve.[2] While Bi-LSTM models outfits as a sequence of items, SiameseNet, Type-Aware, CSA-Net and SCE-Net learn pairwise compatibility of items and aggregates the pairwise compatibility scores for all possible pairs in an outfit to learn the compatibility. On the contrary, we use self-attention to learn high-order compatibility relationships between outfit items. We observe that using just image features; we outperform other methods that use both image and text features on the compatibility prediction task. Using text features boosts performance further.

The methods [25, 22, 16] employ a pairwise model where they require careful selection of negatives and data augmentation. Our approach uses the outfit compatibility data provided without using any additional strategies and still outperforms the state of the art methods. From Table 1, we observe that transformers can learn better compatibility relationships than other methods [6, 9] that learn compatibility at an outfit-level.

4.2. FITB and Complementary Item Retrieval(CIR)

FITB and CIR tasks deal with completing an outfit. While for FITB, the task is to select the best item among a fixed set of choices that goes well with an outfit, for CIR, the task is to choose the best item from the entire database. For the FITB task we use accuracy and for retrieval we use recall@top-k (abbreviated as R@k) as the metric.

Lobert et al. [17] propose to use pre-trained ImageNet embeddings and category for retrieval using self-attention. For evaluation, we adopt their strategy using our own imple-

[2]The authors do not report the performance of SCE-Net [22] on the disjoint dataset. We report the performance of [6] from the paper [14] as the authors report performance on Maryland Polyvore [9] but not on Polyvore Outfits dataset [25]. [6] requires information from a large number of neighbors in a catalog for best performance, which is impractical for our setting because we do not have prior knowledge about connections between each new item to the existing items, as mentioned in [16]. So, we use k=0 (no neighbors) for a fair comparison.

Method	Polyvore Outfits-D				Polyvore Outfits			
	FITB	R@10	R@30	R@50	FITB	R@10	R@30	R@50
Type-Aware [25]	55.65	3.66	8.26	11.98	57.83	3.50	8.56	12.66
SCE-Net Average [22]	53.67	4.41	9.85	13.87	59.07	5.10	11.20	15.93
CSA-Net [16]	59.26	5.93	**12.31**	**17.85**	63.73	8.27	15.67	20.91
OutfitTransformer (Ours)	**59.48**	**6.53**	12.12	16.64	**67.10**	**9.58**	**17.96**	**21.98**

Table 2. Comparison of our model with state-of-the-art methods on the FITB (using accuracy) and CIR tasks (using recall@top-k).

mentation using a transformer [3] and observe that their FITB accuracy on the Polyvore Outfits dataset is 41.61%. We investigate several strategies such as pre-training on the compatibility prediction task, curriculum learning, a different loss formulation and observe a significant improvement in FITB performance. Our method yields a FITB performance of 58.92% when using images and category and 67.10 % using images and text.

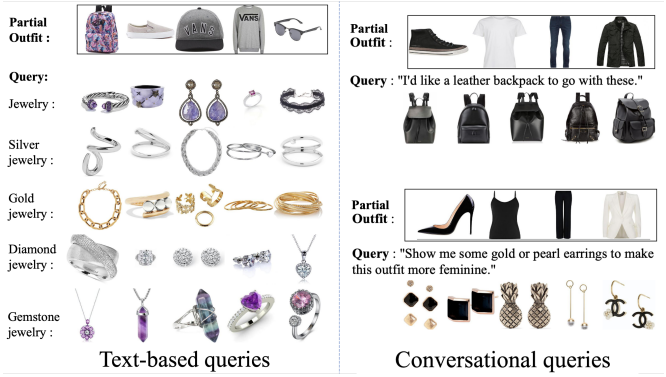
For retrieval, we use the same testing setup as CSA-Net [16], and compare the performance of our method with the state of the art methods CSA-Net [16], Type-aware [25] and SCE-Net average [22][4] For evaluation, we use the category as our target item description for retrieving complementary items and use recall@top-k metric that measures the rank of the ground-truth item similar to [16].

From Table 2, we observe that we outperform all the methods on the non-disjoint dataset. On the disjoint dataset, our performance on recall@top-10 is better than CSA-Net, but is slightly worse on recall@top-30 and recall@top-50. We conjecture that the reason for the performance drop might be because there are fewer outfits on the disjoint set, and transformers typically require large amounts of training data to generalize well. Also, the authors in CSA-Net discuss that the rank of the ground truth is not a perfect measure for evaluating the retrieval performance since the database can contain many complementary items to the query outfits – some of which may be judged by human experts to be equally-good or even better stylistic matches, as can be seen in Figure 4(a). We validated the quality of our results using a user study via Amazon Mturk and observed that users find our retrieved items equally compatible with the outfit as the ground truth. For details of the user study please refer to the supplement.

Example retrieval results using our framework are shown in Figure 4. OutfitTransformer can retrieve compatible items using either target category as shown in Figure 4(a) or text descriptions as shown in Figure 4(b). We show more visualization results in the supplement.

[3] Their code is not publicly available and they did not report numbers on the Polyvore Outfits dataset.

(a) Queries provided as category



(b) Queries provided as free-form text

Figure 4. For the partial outfit and the target category, (a) shows the top-5 retrieved complementary items. The ground truth is indicated by the green bounding box. Similarly, for each partial outfit and a text-based query, (b) shows the top-5 retrieved items that are both compatible with the outfit and matches the text query.

4.3. Ablation studies

Effect of end-to-end training on CP: We compare the effect of end-to-end training and using different input modal-

[4] Type-aware and SCE-Net were adapted for retrieval as reported in [16]. The performance of [10] is not directly comparable to ours, as their method needs to train attributes on another dataset (Shopping100k).

Training strategy for CP task	CP-AUC
ResNet-18 (pre-trained ImageNet)	0.82
ResNet-18 (end-to-end)	0.91
ResNet-18 (end-to-end) + Category	0.92
ResNet-18 (end-to-end) + Text	**0.93**

Table 3. Effect of end-to-end training and different modalities.

Dataset	(a) Pre-training on CP task		(b) Ranking Loss Components	
	without	with	L_{All}	$L_{All} + L_{Hard}$
Polyvore Outfits-D	49.15	**59.48**	55.34	**59.48**
Polyvore Outfits	53.96	**67.10**	64.48	**67.10**

Table 4. Comparison of pre-training and different components of the set-wise outfit ranking loss for the FITB task (using accuracy)

Negative sampling	Polyvore Outfits-D				Polyvore Outfits			
	FITB	R@10	R@30	R@50	FITB	R@10	R@30	R@50
High-level category	55.54	5.14	10.21	14.15	63.33	7.26	13.60	17.78
Fine-grained category	**59.48**	**6.03**	**12.20**	**16.51**	**67.10**	**9.29**	**16.94**	**21.82**

Table 5. Comparison of different negative sampling strategies either from the same high-level or fine-grained category as the positive exemplar for the CIR task.

ities on the CP task in Table 3. We experiment with using pre-computed ResNet-18 image embeddings generated as inputs to our transformer model, and observe that training end-to-end improves performance by 9%. We hypothesize that training end-to-end allows the image encoder to learn better fashion-specific features. This allows the transformer to capture visual relationships between items to learn compatibility better. Using category or text information boosts performance further by 1% and 2%, respectively.

Pre-training: We compare different weight initialization schemes in Table 4(a): 1) We train our framework for the retrieval task where the transformer-encoder is trained from scratch and the image encoder is initialized with pre-trained ImageNet weights. 2) We first pre-train our framework on the compatibility prediction task (CP) and then fine-tune our model on the retrieval task. We see a significant improvement in FITB accuracy with the pre-training.

Set-wise outfit ranking loss components: As mentioned earlier in Section 3.2.2, the outfit complementarity loss has two components. L_{All} optimizes the distances such that the target item embedding is closer to the positive and well separated from the pool of negative samples while L_{Hard} focuses specifically on the hard negatives from the randomly sampled pool. From Table 4(b), we observe that L_{Hard} improves FITB performance by 3-4% on both datasets.

Negative sampling strategies: In Section 3.2.2, we proposed a curriculum learning strategy where we first sample negatives from the same high-level category as the positive and subsequently sample harder and more informative negatives from the same fine-grained category. This strategy leads to stable training and improves both FITB and complementary item retrieval performance substantially as shown in Table 5. Directly training using the hardest negatives from the beginning leads to poor performance.

Comparing different modalities used for retrieval: The OutfitTransformer for retrieval is trained using image and text information. Since the CIR task in [16] is designed for retrieving items given a target category, we feed the category information to our text encoder and use that for our target item query. We experiment with different modalities during inference, such as using an image with either category or text description for the items in the partial outfit. From Table 6, we observe that when using category information, our method outperforms CSA-Net on the non-disjoint dataset, and using text boosts performance further.

5. Conclusion

We present a framework to learn outfit-level representations for compatibility prediction and complementary item retrieval. Experimental results demonstrate that our model outperforms several state-of-the-art approaches on the Polyvore Outfits dataset in three established tasks. We validate that our retrieved results are competitive with the ground truth via a user study, and demonstrate qualitatively that our framework retrieves compatible items using target category or text-based descriptions. In future work, we plan to extend complementary item retrieval to sets of items rather than one-at-a-time.

Method	Input information used		Polyvore Outfits-D			Polyvore Outfits		
	Target Item	Outfit Items	R@10	R@30	R@50	R@10	R@30	R@50
CSA-Net [16]	Category	Image + Catgegory	5.93	**12.31**	**17.85**	8.27	15.67	20.91
OutfitTransformer (Ours)	Category	Image + Category	6.03	12.20	16.51	9.29	16.94	21.82
OutfitTransformer (Ours)	Category	Image + Text	**6.53**	12.12	16.64	**9.58**	**17.96**	**21.98**

Table 6. Comparison of performance using different input information during inference for the CIR task.

References

[1] https://github.com/nmslib/hnswlib.

[2] https://github.com/facebookresearch/faiss.

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *CoRR*, abs/2005.12872, 2020.

[4] Chun-Fu Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. *ArXiv*, abs/2103.14899, 2021.

[5] Wen Feng Chen, Pipei Huang, Jiaming Xu, Xin Guo, Cheng Guo, Fei Sun, Chao Li, Andreas Pfadler, Huan Zhao, and Binqiang Zhao. Pog: Personalized outfit generation for fashion recommendation at alibaba ifashion. *SIGKDD*, 2019.

[6] Guillem Cucurull, Perouz Taslakian, and David Vazquez. Context-aware visual compatibility prediction. In *CVPR*, 2019.

[7] Zeyu Cui, Zekun Li, Shu Wu, Xiaoyu Zhang, and Liang Wang. Dressing as a whole: Outfit compatibility learning based on node-wise graph neural networks. *The World Wide Web Conference*, 2019.

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[9] Xintong Han, Zuxuan Wu, Yu-Gang Jiang, and Larry S. Davis. Learning fashion compatibility with bidirectional lstms. In *ACM MM*, 2017.

[10] Yuxin Hou, Eleonora Vig, Michael Donoser, and Loris Bazzani. Learning attribute-driven disentangled representations for interactive fashion retrieval. In *The International Conference on Computer Vision (ICCV)*, October 2021.

[11] Wei-Lin Hsiao and Kristen Grauman. Learning the latent "look": Unsupervised discovery of a style-coherent embedding from fashion images. In *ICCV*, 2017.

[12] Wei-Lin Hsiao and Kristen Grauman. Creating capsule wardrobes from fashion images. In *CVPR*, 2018.

[13] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer. *ArXiv*, 2018.

[14] Kedan Li, Chen Liu, and David Forsyth. Coherent and controllable outfit generation, 2019.

[15] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *TPAMI*, 2020.

[16] Yen-Liang Lin, S. Tran, and Larry Davis. Fashion outfit complementary item retrieval. In *CVPR*, 2020.

[17] Alexander Lorbert, David Neiman, Arik Poznanski, Eduard Oks, and Larry Davis. Scalable and explainable outfit generation. In *CVPR Workshop*, 2021.

[18] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. *CoRR*, abs/1906.05909, 2019.

[19] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP*, 2019.

[20] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015.

[21] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. *CVPR*, 2016.

[22] Reuben Tan, Mariya I. Vasileva, Kate Saenko, and Bryan A. Plummer. Learning similarity conditions without explicit supervision. In *ICCV*, 2019.

[23] Meet Taraviya, Anurag Beniwal, Yen-Liang Lin, , and Larry Davis. Personalized compatibility metric learning. *KDD Workshop*, 2021.

[24] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers amp; distillation through attention. In *ICML*, 2021.

[25] Mariya I. Vasileva, Bryan A. Plummer, Krishna Dusad, Shreya Rajpal, Ranjitha Kumar, and David Forsyth. Learning type-aware embeddings for fashion compatibility. In *ICCV*, 2018.

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017.

[27] Andreas Veit, Serge Belongie, and Theofanis Karaletsos. Conditional similarity networks. In *CVPR*, 2017.

[28] Andreas Veit, Balazs Kovacs, Sean Bell, Julian McAuley, Kavita Bala, and Serge Belongie. Learning visual clothing style with heterogeneous dyadic co-occurrences. In *ICCV*, 2015.

[29] Chao-Yuan Wu, R. Manmatha, Alexander J. Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *ICCV*, 2017.

[30] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *CoRR*, abs/2105.15203, 2021.

[31] Hong Xuan, Abby Stylianou, Xiaotong Liu, and Robert Pless. Hard negative examples are hard, but useful. In *ECCV*, 2020.

[32] Fuzhi Yang, Huan Yang, Jianlong Fu, Hongtao Lu, and Baining Guo. Learning texture transformer network for image super-resolution. *CoRR*, abs/2006.04139, 2020.

[33] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal self-attention for local-global interactions in vision transformers. *CoRR*, abs/2107.00641, 2021.

[34] X. Yang, Yunshan Ma, Lizi Liao, M. Wang, and Tat-Seng Chua. Transnfcm: Translation-based neural fashion compatibility modeling. *ArXiv*, 2019.