OutfitNet: Fashion Outfit Recommendation with Attention-Based Multiple Instance Learning

Yusan Lin
yusalin@visa.com
Visa Research
Palo Alto, California

Maryam Moosaei
maryamoosaei@gmail.com
Visa Research
Palo Alto, California

Hao Yang
haoyang@visa.com
Visa Research
Palo Alto, California

ABSTRACT

Recommending fashion outfits to users presents several challenges. First of all, an outfit consists of multiple fashion items, and each user emphasizes different parts of an outfit when considering whether they like it or not. Secondly, a user's liking for a fashion outfit considers not only the aesthetics of each item but also the compatibility among them. Lastly, fashion outfit data is often sparse in terms of the relationship between users and fashion outfits. Not to mention, we can only obtain what the users like, but not what they dislike.

To address the above challenges, in this paper, we formulate the fashion outfit recommendation problem as a multiple-instance-learning (MIL) problem. We propose OutfitNet, a fashion outfit recommendation framework that includes two stages. The first stage is a Fashion Item Relevancy network (FIR), which learns the compatibility between fashion items and further generates relevancy embedding of fashion items. In the second stage, an Outfit Preference network (OP) learns the users' tastes for fashion outfits using visual information. OutfitNet takes in multiple fashion items in a fashion outfit as input, learns the compatibility among fashion items, the users' tastes toward each item, as well as the users' attention on different items in the outfit with the attention mechanism.

Quantitatively, our experiments show that OutfitNet outperforms state-of-the-art models in two tasks: fill-in-the-blank (FITB) and personalized outfit recommendation. Qualitatively, we demonstrate that the learned personalized item scores and attention scores capture well the users' fashion tastes, and the learned fashion item embeddings capture well the compatibility relationships among fashion items. We also leverage the learned fashion item embedding and propose a simple fashion outfit generation framework, which is shown to produce high-quality fashion outfit combinations.

CCS CONCEPTS

• **Information systems** → *Collaborative and social computing systems and tools*; • **Computing methodologies** → *Artificial intelligence*; • **Applied computing** → *Arts and humanities.*

KEYWORDS

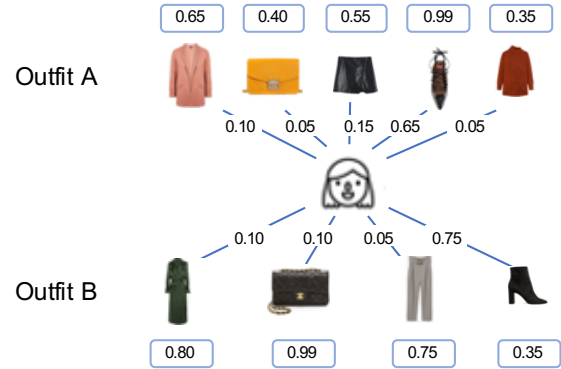fashion outfit recommendation, fashion item relevancy, fashion outfit generation

Figure 1: High-level idea of Outfit Preference network (OP). Values in the boxes indicate the personalized item scores, and values on the lines indicate the personalized attentions.

1 INTRODUCTION

Fashion is an immense industry. Today, the average person purchases 60%, or more, clothing items overall compared to fifteen years ago.[1] By 2020, the global revenue of fashion is expected to rise to $721.9 billion USD, compared to $481.2 billion USD in 2018. Leveraging machine learning and artificial intelligence have helped increase revenue growth in the industry.[2] In particular, *personalized recommendations* has influenced 43% of online fashion purchases, and 95% of online fashion retail companies view personalized recommendation as an essential and urgent business strategy.[3]

Recommending products to consumers has been a widely and thoroughly studied research problem, and, in particular, recommendations based on consumers' personal tastes. When making clothing recommendations, many approaches have been proposed. Among them, the most successful methodology so far is shown to leverage the visuals of clothing items [15, 18, 34], which is also the current standard when learning about personal aesthetics [5, 8, 24, 31]. Previous studies have focused on recommending single fashion

---

[1]https://www.greenpeace.org/international/publication/6969/fashion-at-the-crossroads/
[2]https://www.mckinsey.com/industries/retail/our-insights/the-state-of-fashion
[3]https://www.shopify.com/enterprise/ecommerce-fashion-industry

items to users. However, besides viewing individuals' preferences towards each fashion clothing item, one important aspect of the nature of fashion products is their *combination*. When shopping for fashion clothing items, consumers are usually more drawn to purchase items that they know how to put together into a fashion outfit. A line of research has addressed this problem by learning the *compatibility* of fashion clothing items [11, 12, 16, 27, 28]. However, learning individuals' tastes towards fashion outfits, which are combinations of several individual fashion clothing items, still requires further investigation.

In this paper, we address the problem of recommending fashion outfits that consist of a set of clothing items to individuals based on their fashion tastes. In particular, our goal is to investigate problems similar to the one depicted in the toy example of Figure 1. As shown, the user is presented with two fashion outfits, and each consists of a set of fashion clothing items. When assessing the fashion outfits, the user's thought process is two-fold. Firstly, the user has her preference towards each fashion item, as denoted in the boxes. Secondly, the user puts different weights on fashion items in outfits, as indicated by the number on the lines. In this example, a majority of the items in outfit B receive higher scores from the user than items in outfit A. However, since the user places more emphasis on shoes when assessing a fashion outfit, outfit A has a higher *personalized fashion outfit score* than outfit B in this particular case.

There are several challenges involved when attempting to create a fashion outfit recommendation system that handles the scenario presented in Figure 1. Firstly, creating appropriate representations for the fashion outfits before inputting them to learn users' outfit preferences is important, and the most crucial characteristic to embed is their compatibility. Many works have proposed different ways to learn the compatibility in fashion outfits, while most of them rely heavily on rich meta information, complicated neural network models, or simplified assumptions on outfit compositions [11, 28, 29, 33]. Secondly, most of the data regarding the user's preference for fashion outfits are only labeled at the outfit level, not at the item level. Finding a way to infer a user's preference for fashion items included in fashion outfits is critical. This distinction can also help make individual fashion recommendations to users. Finally, the data on the relationships between users and fashion outfits are often sparse. In our dataset, over 68% of fashion outfits are liked by one user only. Also, we only have data regarding what the users explicitly liked, but not what they disliked.

In this paper, we address the above challenges and formulate the outfit recommendation task into a multiple instance learning (MIL) problem. We then develop a personalized fashion outfit recommendation system, OutfitNet. OutfitNet includes two stages: a Fashion Item Relevancy network (FIR) and an Outfit Preference network (OP). FIR learns the relevancy of a fashion item given a subset of fashion items, a positive item, and a negative sample. OP takes in fashion outfits with a various number of clothing items as inputs and a *personalized attention layer* is designed to aggregate the multiple clothing items in fashion outfits, as well as learning individuals' emphasis on the clothing items when judging a fashion outfit. Through both quantitative and qualitative evaluations, we show the effectiveness of OutfitNet when recommending fashion outfits. More specifically, we evaluate OutfitNet with two recommendation tasks: fill-in-the-blank (FITB) and personalized outfit

recommendation. In both tasks, we show that OutfitNet outperforms other state-of-the-art methods. We also show the fashion item embedding learned by FIR and the personalized attention learned by OP through a case study.

Quantitatively, compared with other state-of-the-art baseline methods, OutfitNet performs better in both the FITB and personalized fashion outfit recommendation tasks. Specifically, in the FITB task, fashion item embeddings learned by FIR can achieve 88.63% accuracy. In the personalized outfit recommendation task, OutfitNet has an 84.07% Area Under Curve (AUC) in recommending all fashion outfits, and 79.70% in recommending outfits that are only liked by one user in our dataset.

Qualitatively, with the learned FIR, we can obtain embeddings of fashion items that reflect the underlying compatibility. When showing the learned fashion item embeddings on a tSNE-transformed two-dimensional space, compatible fashion items are closer to each other than incompatible fashion items, compared to embeddings extracted from pre-trained DenseNet. We also further leverage the learned fashion item embedding and propose a simple yet effective fashion outfit generation framework, OutfitExpansion. We show that the quality of the generated fashion outfits are high. Additionally, when considering a given user and fashion outfit, one can obtain a user's personalized item score, personalized outfit score, and personalized attention scores for the items in the outfits. Through a case study, we show that the learned item scores capture the users' tastes toward fashion items, and the learned attention scores capture the users' emphasis on fashion items among the whole outfit.

The contributions of this work are as follows.

(1) The proposed Fashion Item Relevancy network (FIR) is a lightweight yet effective approach to learn the compatibility between fashion items. Compared to existing state-of-the-art models, FIR has a simple architecture and does not require additional information on fashion items other than the visuals.

(2) We formulate the fashion outfit recommendation as a multiple instance learning (MIL) problem. This approach has the following two benefits: (1) allows us to handle multiple clothing items as inputs for each fashion outfit, and (2) enables us to learn not only the users' preferences towards fashion outfits (labeled in the data) but also their preferences towards fashion clothing items (not labeled in the data).

(3) Unlike the embeddings learned by the state-of-the-art CNN models, which mainly capture the visual similarity such as colors and silhouettes, the fashion item embedding learned by OutfitNet can capture the underlying styles of personal tastes regarding fashion items. We further leverage the learned embedding and propose a simple fashion outfit generation framework, OutfitExpansion, which is shown to produce high-quality fashion outfits.

(4) We design a *personalized attention layer* in OutfitNet, which captures the users' personal emphasis on each fashion item in fashion outfits when assessing their preference for the whole fashion outfit ensemble.

We organize the rest of this paper as follows. In section 2, we review the literature related to this paper. We propose our design

of OutfitNet in section 3, and evaluate OutfitNet by comparing it with other baseline models in section 4. We then further investigate the learned fashion item embedding and personalized attention and also propose a simple fashion outfit generation framework in section 5. Finally, we conclude this work in section 6.

2 RELATED WORK

In this section, we provide a literature review in the fields related to this work, which are fashion recommendation, multiple instance learning and attention mechanism.

2.1 Fashion Recommendation

Starting in 2013, the field of quantitative fashion research started to receive more attention. Most fashion research has been leveraging data obtained on online platforms for analysis and evaluation, such as Instagram [7], and other social media channels, as well as fashion-specific social networks, including Polyvore and Lookbook.nu [19]. In the early stages, more focus was placed on the high-end fashion industry [7, 20, 21, 30]. Recently, the focus has shifted towards understanding the consumers' tastes and recommending fashion products accordingly.

Research has been conducted on recommending fashion products to consumers. Some research utilize product characteristics to make personalized fashion product recommendations [4], or use reviews on retail websites [18], while others use visual information to make recommendations [34]. Kang et al. further design a recommendation model that recommends fashion clothing items based on visual information and generates personalized fashion clothing images [15].

In addition to considering fashion items separately, one important quality is the combination of fashion items in creating fashion outfit ensembles. There are several works exploring the compatibility in fashion outfits: Han et al. views fashion outfits as sequences of fashion items and encodes the outfits using bi-directional LSTM [11]; Song et al. encodes fashion domain knowledge to neural networks with attentive knowledge distillation scheme [27]; Kang et al. extracts products from scene-based images and infers the compatibility among fashion products [16]; Vasileva et al. leverages the categories of fashion items to learn the compatibility embedding [29]; and Cui et al. models the relationships of fashion items into a graph to learn the compatibility embedding [9]. The closest work to our approach of learning compatibility is by Yin et al. [33], in which they leverage the triplet (positive, anchor, negative) and Bayesian Personalized Ranking (BPR) to push the learned embedding of anchor to be closer to positive samples than negative samples. The difference between our FIR and this approach is that we enable the anchor to be considered as sets, which we will discuss further in Section 3.

Although plenty of works have been published on recommending individual fashion items to users, a limited amount of work has been conducted on recommending fashion outfits as a whole. The primary efforts include Lin et al. proposing a neural network that takes inputs as pairs of tops and bottoms [18], and Chen et al. proposing a neural network that takes in five items as an outfit input [6]. These works, however, fall short on considering the varying number of items included in different fashion outfits, the inference

of personal preferences from outfit ensembles to individual items, and the attentive nature of personal preferences on fashion outfits, which we address in our proposed OutfitNet.

2.2 Multiple Instance Learning and Attention

A critical aspect of learning an individual's taste for fashion outfits pertains to the various number of fashion clothing items involved in each fashion outfit ensemble. The aggregation among fashion items in outfits should handle a different number of fashion items and view the fashion items as order-invariant. One related attempt is by Han et al., where they propose using bidirectional LSTM to encode fashion outfits [11]. However, they assume there is a fixed number of items in an outfit, and a fixed order of items in terms of their categories. Instead of using a sequence of fixed length to encode fashion outfits, we formulate our fashion outfit recommendation problem as a multiple instance learning (MIL) problem. A MIL problem assumes that the labels for a set of items are known, while the labels for the individual items in the set are unknown.

Maron et al. first raised the problem of MIL back in 1998 [23]. In 2017, Zaheer et al. explored MIL in neural networks, [35]; however, the pooling operators considered at the time only included operators, such as mean, max, convex maximum (i.e., log-sum-exp), noisy-or and noisy-and. Subsequently, Ilse et al. then leveraged *attention* to improve MIL [14]. Attention, which was first introduced by Bahdanau et al. in 2015, was used to improve machine translation in encoder-decoder RNN models [2]. Xu et al. then used the concept of attention to improve the performance of image captioning by focusing on specific parts of the images when generating words in captions [32]. Based on the idea of attention, Ilse et al. proposed a neural network model that incorporates an attention layer to pool from the multiple inputs. The model is shown to predict multiple-instance-learning tasks better, and the learned attention is shown to capture the emphasis put across multiple inputs when making predictions, which achieves key instance detection.

In this paper, we incorporate the idea of using attention in multiple instance learning to make personalized fashion outfit recommendations.

3 OUTFITNET

In this section, we present the design of our proposed two-fold system, OutfitNet. First, we introduce the intuition and problem formulation. Next, we detail the model architectures of both stages in OutfitNet and discuss how the parameters in OutfitNet are learned. For clarity, we summarize the symbols used in this paper in Table 1.

3.1 Intuition and Problem Formulation

Our goal is to first learn the fashion item embeddings that capture the item compatibility and relevancy, then further recommend fashion outfits to users based on their fashion tastes. Our proposed system, OutfitNet, includes two stages: the *Fashion Item Relevancy network (FIR)* and the *Outfit Preference network (OP)*. FIR learns the relevancy and compatibility among fashion items and generates fashion item embeddings that capture such characteristics. Next, OP takes the embeddings learned by FIR and further learns user preferences towards fashion outfits.
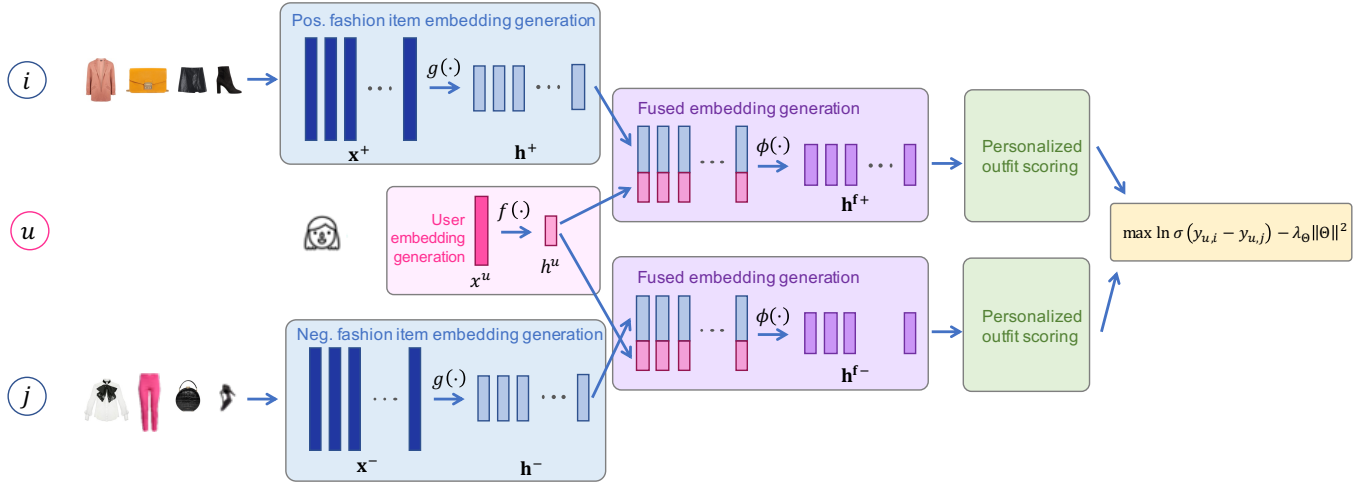
Figure 2: Outfit preference network (OP)

Table 1: Symbol definition for OutfitNet.

Symbol	Definition
$\mathcal{D}$	Dataset
$\mathcal{S}$	Outfit set
$\mathcal{I}$	Item set
$\mathcal{U}$	User set
x	Encoded inputs of user/clothing items
$y_{u,i}$	Score of user u liking the ith outfit
$\pi_{u,i,k}$	Score of user u liking the kth item in the ith outfit
ζ, ψ	Fully-connected layers in FIR
f, g, ϕ, ρ, η	Fully-connected layers in OP
h	Embeddings generated in the hidden layers
$e_{u,k}$	User u's attention for the kth item
$\alpha_{u,k}$	Normalized $e_{u,k}$
λ	Triplet loss margin

In most real-world data, we obtain knowledge of what the users *like*, but not what they *dislike*. Similarly, we only collect information on what fashion items are compatible with each other, but not incompatible with each other. Therefore, rather than determining negative samples and further design the models into classification models, we leverage the idea of triplet loss in the Siamese network [3, 26] and Bayesian Personalized Ranking [25]. Intuitively, in the learned embedding space, the compatible fashion items should be closer to each other compared to incompatible fashion items. Also, the user should be closer to outfits she likes, and further from the outfits she does not like. Therefore, the two problems formulated for OutfitNet are as follows.

Fashion item relevancy network (FIR): Given a set of fashion items $\widetilde{S}$, a positive item x_i that has co-occurred with all of the items in $\widetilde{S}$ in an outfit, and a negative item x_j that has never co-occurred with $\widetilde{S}$ in an outfit before, FIR learns the score of $\widetilde{S}$ co-occurring with x_i, $s(x_i|\widetilde{S})$, and $\widetilde{S}$ co-occurring with x_j, $s(x_j|\widetilde{S})$, where $s(x_i|\widetilde{S}) > s(x_j|\widetilde{S})$.
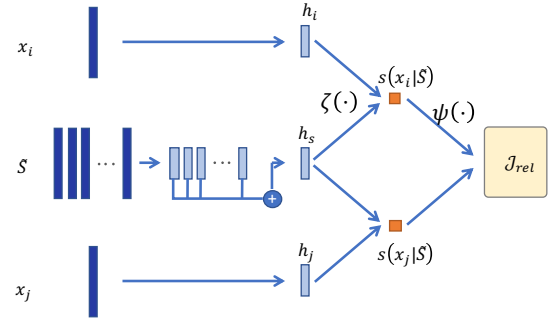

Figure 3: Fashion item relevancy network (FIR)

Outfit preference network (OP): Given an user u, items in a positive outfit i that u likes in the dataset $\mathcal{D}_u$, and items in a negative outfit j that u has not liked in $\mathcal{D}_u$, OP learns the score of u liking i, $y_{u,i}$, and u liking j, $y_{u,j}$, where $y_{u,i} > y_{u,j}$.

Both of the problem settings above fall into the formulation of multiple instance learning. For FIR, the labels for $(\widetilde{S}, x_i)$ and $(\widetilde{S}, x_j)$ are compatible and incompatible, respectively. However, which item in $\widetilde{S}$ is compatible/incompatible with x_i/x_j is unknown. For OP, the labels for (u, i) and (u, j) are like and dislike, respectively. Similar to FIR, which item in outfits i and j the user u likes/dislikes is unknown. Through learning from the two models, we aim to infer the labels of each item.

3.2 Fashion Item Relevancy Network

One crucial part of fashion outfit recommendation is to first learn the relevancy among fashion items in fashion outfits. For this purpose, we develop a *fashion item relevancy network (FIR)* to capture the relevancy among fashion items. The idea is that for fashion items that are more compatible/relevant to each other, they should be closer to each other in the embedding space. FIR can be viewed as an extension of the type-aware network proposed in [29] and the Siamese-network-based model in [33], where we consider beyond

the pair and triplet of individual items' compatibility. The detailed design of FIR is as follows.

The input of FIR is a triplet $(\widetilde{S}, x_i, x_j)$, where $\widetilde{S}$ is a partial set of items in a fashion outfit, x_i is an item that has appeared with $\widetilde{S}$ in an outfit, and x_j is an item that has not appeared with $\widetilde{S}$ in an outfit. In other words, $\widetilde{S} \cup \{x_i\} \subseteq S \exists S \in \mathcal{S}$ and $\widetilde{S} \cup \{x_j\} \not\subseteq S \forall S \in \mathcal{S}$. $(\widetilde{S}, x_i, x_j)$ are inputs to FIR, and all items x are passed through a fully-connected layer ζ. Items in $\widetilde{S}$ are pooled to a single embedding h_s, and x_i and x_j become embeddings h_i and h_j, respectively.

After the embeddings are generated, h_i and h_s together generate a relevancy score $s(x_i|\widetilde{S})$, and h_j and h_s together generate a relevancy score $s(x_j|\widetilde{S})$. The relevancy scores are calculated as follows.

$$s(x_i|\widetilde{S}) = \psi(h_i + h_s) \tag{1}$$

where the embeddings h_i and h_s are element-wise added together, then passed through a fully-connected layer $\psi(\cdot)$. The same is applied to getting relevancy score $s(x_j|\widetilde{S})$. Finally, the scores are passed to the final layer, and the difference between $s(x_i|\widetilde{S})$ and $s(x_j|\widetilde{S})$ are calculated.

3.3 Outfit Preference Network

Figure 2 shows the design of outfit preference network (OP), which consists of two stages: the *fused embedding generation* stage and the *personalized scoring* stage. The input for the model are triplets $(u, i, j) \in \mathcal{D}$, where

$$\mathcal{D} = \{(u, i, j) | u \in \mathcal{U} \wedge i \in \mathcal{S}_u^+ \wedge j \in \mathcal{S} \setminus \mathcal{S}_u^+\}.$$

$i \in \mathcal{S}_u^+$ is an outfit liked by user u, and $j \in \mathcal{S} \setminus \mathcal{S}_u^+$ is an outfit not liked by user u yet. Each triplet (u, i, j) is pre-processed before sending into OutfitNet. User u is encoded as a one-hot-encoding x^u, with only the value of the corresponding user being 1, and others being 0s. For outfits i and j, each clothing item's inputs are represented by the embedding generated from a trained FIR's layer h. We denote the embeddings of outfit i as $\mathbf{x}^+ = \{x_l^+ | l = 0...L\}$ and the embeddings of the outfit j as $\mathbf{x}^- = \{x_m^- | m = 0...M\}$, where L and M are the number of clothing items in outfits i and j, respectively. Note that both $\mathbf{x}^+$ and $\mathbf{x}^-$ are sets and the order of clothing items does not matter. Below we describe the two stages of OutfitNet after the pre-processed inputs are sent in.

Stage 1: Fused Embedding Generation. Given a triplet (u, i, j), their pre-processed x^u, $\mathbf{x}^+$ and $\mathbf{x}^-$ are passed into the network. User's one-hot-encoding x^u is passed through a fully-connected layer (FC layer) $f(\cdot)$ to reduce to a lower-dimensional *user embedding*, h^u. For both outfits i and j, their pre-processed sets of items' FIR embeddings $\mathbf{x}^+$ and $\mathbf{x}^-$ are passed through a FC layer $g(\cdot)$ to reduce to lower-dimensional sets of *fashion item embeddings*, $\mathbf{h}^+$ and $\mathbf{h}^-$. Each of the item embedding h in the sets of fashion item embeddings $\mathbf{h}^{\cdot}$ are concatenated with the user embedding h^u, then passed through a FC layer $\phi(\cdot)$ to generate a set of lower-dimensional *fused embeddings*, $\mathbf{h}^{f+}$ and $\mathbf{h}^{f-}$. This process is the fused embedding generation stage, which is summarized as follows.

$$\mathbf{h}^{f\cdot} = \phi\left(\left[\left(\underbrace{f(x^u)\, f(x^u)\, \cdots f(x^u)}_{k \text{ elements}}\right) \| g(\mathbf{x}^{\cdot})\right]\right) \tag{2}$$
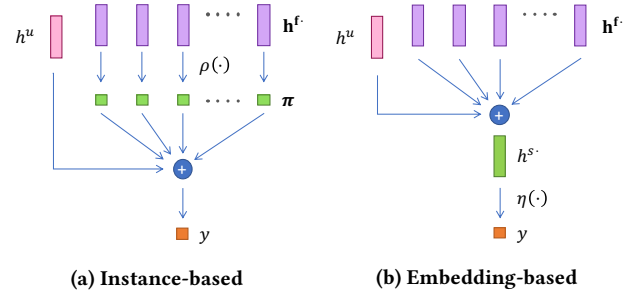

(a) Instance-based   (b) Embedding-based

Figure 4: Approaches for personalized outfit scoring stage.

where the superscripts $\cdot$ are $+$ or $-$, indicating the positive or negative outfits. k is the number of clothing items in the outfit, and $\|$ is the concatenation between embeddings.

Stage 2: Personalized Outfit Scoring. After obtaining the fused embedding, the model leverages them to generate personalized outfit scores $y_{u,i}$ and $y_{u,j}$. For this stage, OP has two alternatives for generating scores: the instance-level approach and the embedding-level approach.

Instance-level approach: Each of the fused embeddings $h^{f\cdot}$ is passed through a FC layer $\rho(\cdot)$, and output as *personalized item scores* $\boldsymbol{\pi}$. The score $\pi_{u,i,k}$ indicates the score of u liking the kth item in the positive outfit i. The scores are passed through a pooling layer to generate a *personalized outfit score* $y_{u\cdot}$. $y_{u,i}$ represents u's score toward the positive outfit i. Here we design the pooling layer as an attention layer. For clarity, we use the positive outfit to explain, and the negative outfit works in the same way.

$$y_{u,i} = \sum_{k=1}^{L} \alpha_{u,k} \pi_{u,k} \tag{3}$$

where α is u's attention for the kth item in the outfit, and L is the number of clothing items in outfit i. $\alpha_{u,k}$ is calculated as follows.

$$\alpha_{u,k} = \frac{\exp(e_{u,k})}{\sum_{l=1}^{L} \exp(e_{u,l})} \tag{4}$$

$e_{u,k}$ is an output of a feed-forward neural network, with inputs as both the user embedding and item's fused embedding:

$$e_{u,k} = a(h^u, h_k^{f+}) \tag{5}$$

where a is a non-linear FC layer, h^u is user u's embedding, and $h_k^{f\cdot}$ is the kth item's fused embedding. By incorporating both inputs, the learned attention is able to be *personalized*. The approach is depicted in Figure 4a.

Embedding-level approach: Unlike the instance-level approach, the embedding-level approach consists of a pooling layer, which is applied to the fused embeddings $h^{f\cdot}$. This pooling layer is an attention layer that outputs a *contextual embedding* $h^{s\cdot}$. For clarity, we use the positive outfits to explain.

$$h^{s+} = \sum_{j=1}^{k} \alpha_{uj} h_j^{f+} \tag{6}$$

h^{s+} is generated similarly as $y_{u,i}$ in Eq. (3), while it is a weighted sum over embeddings rather than a scalar value. h^{s+} is then passed through a FC layer $\eta(\cdot)$, which generates a score $y_{u,i}$, indicating u's preference towards the whole outfit. The approach is depicted in Figure 4b.

For comparison, we summarize how $y_{u,i}$ is generated in each approach.

$$y_{u,i} = \begin{cases} \sum_{k=1}^{L} \alpha_{u,k} \rho(h_k^{f+}) & \text{, instance approach} \\ \eta\left(\sum_{k=1}^{L} \alpha_{uk} h_k^{f+}\right) & \text{, embedding approach} \end{cases}. \quad (7)$$

$y_{u,j}$ is generated in the same way by replacing $+$ with $-$, and L with M.

In both approaches, we use the attention layer as the pooling layer. There are several points to pay attention to regarding our design of the attention layer. Firstly, we do not consider the *order* of the items. Hence the inputs of the attention layer are independent of each other. Secondly, the attention weights α are *personalized* because when generating e in Eq. (5), the user embedding h^u is also passed through the feed-forward network a as one of the two inputs.

3.4 Parameter Learning

The objective of FIR is to maximize the difference between the two relevancy scores, so that embeddings of compatible fashion items are closer to each other, compared to embeddings of incompatible fashion items.

$$\mathcal{J}_{rel}(\Theta_{rel}) = \max \sigma\left(s(x_i|\widetilde{S}) - s(x_j|\widetilde{S})\right) - \lambda_{\Theta_{rel}} ||\Theta_{rel}||^2 \quad (8)$$

where $\Theta_{rel}\{\zeta, \psi\}$ are the parameters to learn from FIR, and $s(x_i|\widetilde{S})$ and $s(x_j|\widetilde{S})$ are the relevancy scores obtained by Eq. (1), and λ is the margin.

Similarly, the objective of OP is to learn the embeddings of users and outfits, so that users' embeddings are closer to the embeddings of outfits they like, compared to outfits they do not like. The objective function of OutfitNet is as follows.

$$\mathcal{J}_{pre}(\Theta_{pre}) = \max \ln \sigma(y_{u,i} - y_{u,j}) - \lambda_{\Theta_{pre}} ||\Theta_{pre}||^2 \quad (9)$$

where $\Theta = \{f, g, \phi, \eta, \rho\}$ are the parameters to learn from OutfitNet, $y_{u,i}$ and $y_{u,j}$ are the scores of a user liking the positive and negative outfits, respectively (obtained by Eq. (7)), and λ is the margin.

4 EVALUATION

To evaluate the effectiveness of OutfitNet, in this section, we show our experiment results. First, we introduce the dataset collected and used for our experiments. Next, we present the experiment setting and OutfitNet's performance in two tasks: fill-in-the-blank (FITB) and fashion outfit recommendation, compared with other current state-of-the-art models.

4.1 Datasets

In this paper, we leverage two datasets for experiment and benchmarking purposes: Polyvore and iFashion.

4.1.1 Polyvore. We created a dataset by crawling Polyvore[4], a fashion social network allowing users to create fashion outfits from

different items, or like and save outfits created by others. Launched in 2007, Polyvore has attracted 20 million users, with 87,000 fashion outfits created on average every day by users.[5] We crawled profiles of 150 randomly selected users $\mathcal{U}$ to generate a dataset of 66,000 fashion outfits S and 158,503 fashion items I.

4.1.2 iFashion. Alibaba research put together the dataset collected from iFashion, a Taobao fashion service, and provided the dataset for research purposes [6]. iFashion works similarly as Polyvore, where users create fashion outfits using existing fashion item data. The dataset consists of 3.5 million users $\mathcal{U}$, 127,000 fashion outfits S and 4.4 million fashion items I.

Table 2: Dataset statistics.

Dataset	Users	Fashion outfits	Fashion items
Polyvore	150	66,000	158,503
iFashion	3,569,112	127,169	4,463,302

For both datasets, we randomly sample 70% as training $\mathcal{P}$, 10% as validation $\mathcal{V}$, and the rest of the 20% as testing $\mathcal{T}$.

4.2 Experiment Setting

The embeddings of fashion clothing items' images are generated with a DenseNet pre-trained on ImageNet, resulting with features in 50176 dimension [10, 13]. OutfitNet (including both FIR and OP) is implemented using Tensorflow [1]. The dimension of the fully-connected (FC) layers are shown in Table 3, where all are single layers except g, which consists of two layers.

Table 3: Dimension of each layer.

Layer	ζ	ψ	f	g	ϕ
Dimension	50176	256	16	6272, 256	16

We initialize all weights with Xavier initialization and use `leakyRelu` as the activation function. We train the model with batch size 64 and use AdaDelta as the optimizer. The training stops until the objective functions in Eq. (1) and (9) converge, or until the maximum number of epochs (100) is reached.

4.3 Fill-in-the-Blank Prediction

Our first evaluation task is fill-in-the-blank (FITB), a widely conducted and standard test across fashion compatibility research. We design the FITB task as follows. We use the data in $\mathcal{P}$ to train the fashion item relevancy network (FIR) and the comparing baselines. Then, given an outfit in the test data $\mathcal{T}$, we randomly masked out one of the fashion items. We then randomly select three fashion items and predict which one is most likely to be in the given partial outfit, i.e., which fashion item is the most compatible with the rest of the items. The performance of this task is measured by accuracy, and compared with the following baseline methods.

(1) **Random:** A model based on random guesses.

Table 4: FITB results of different models.

	(1) Random	(2) PopRank	(3) Triplet	(4) SetNN	(5) F-LSTM	(6) Bi-LSTM	(7) FIR	(7) v.s. best
Polyvore	24.97%	19.03%	84.70%	48.42%	46.19%	50.19%	**88.63%**	+4.63%
ifashion	25.00%	17.95%	85.44%	50.16%	59.14%	60.22%	**86.06%**	+0.73%

Table 5: Fashion outfit recommendation results of different models.

		(1) Random	(2) PopRank	(3) MF	(4) BPR	(5) DVBPR	(6) OP-Ins	(7) OP-Emb	(6) v.s. best	(7) v.s. best
Polyvore	All outfits	50.00%	49.56%	49.73%	52.25%	54.08%	78.11%	**84.07%**	44.43%	55.45%
	Cold outfits	50.00%	46.91%	48.98%	51.11%	49.93%	**79.70%**	73.64%	55.93%	44.08%
iFashion	All outfits	50.00%	48.24%	49.05%	53.76%	57.83%	79.29%	**86.42%**	37.11%	49.44%
	Cold outfits	50.00%	44.03%	45.26%	52.39%	55.34%	**81.65%**	80.91%	47.54%	46.21%

(2) **PopRank:** A model that ranks the items based on its occurrences in outfits, which infers the popularity of the item.

(3) **Triplet:** A Siamese-network-based model proposed in [33], where each element in the triplet is single fashion item.

(4) **SetNN:** An end-to-end model proposed in [17], where fashion items in outfits are also pooled with the multiple-instance approach, but explicit negative (i.e., incompatible) samples are required.

(5) **F-LSTM:** A model that, given a sequence of items embedded with a forward LSTM, predicts the next item [11].

(6) **Bi-LSTM:** A model that, given a sequence of items embedded with a bi-directional LSTM, the model predicts the next item [11].

(7) **FIR (ours):** Our model that learns the compatibility among items by considering the relationship between a subset of items and a single item.

Note that for (5) and (6), the prediction is made directly using the trained model (i.e., predicting the next item given a sequence). For (3), (4), and (7), we first trained the models, then use the trained models to generate embeddings for items in the test data. The prediction is finally made using the distances between items' embeddings. We measure the performance by the percentage of times the model correctly chooses the ground truth items over the other randomly selected items. We show the results of the FITB task using different models in Table 4.

As shown in Table 4, our proposed light-weight FIR outperforms the baseline models. The most comparable is Triplet, which is a simpler version of FIR, where S only consists of single items. Also, its performance is very close to FIR when evaluated on the iFashion dataset. The reason is that outfits in the iFashion dataset, in general, have fewer items than outfits in Polyvore. Therefore, the sampled S often only include single items, which is equivalent to Triplet.

4.4 Personalized Outfit Prediction

We compare the performance of OutfitNet in recommending fashion outfits to users with the following baselines.

(1) **Random:** A model based on random guesses.

(2) **PopRank:** A model that ranks the outfits a given user may like based on the popularity of the outfits. In our dataset, we use the number of likes an outfit receives as the popularity measure.

(3) **MF:** State-of-the-art recommendation algorithm, Matrix Factorization. For this method, we construct a user-outfit matrix as inputs for training.

(4) **BPR:** State-of-the-art recommendation algorithm, Bayesian Personalized Ranking, proposed in [25]. Similar to MF, we construct a user-outfit matrix as inputs for training.

(5) **DVBPR:** The fashion item recommendation model, Deep Visually-Aware Bayesian Personalized Ranking, proposed in [15]. DVBPR was originally designed to recommend fashion items to individuals, rather than fashion outfits as a whole. To compare with OutfitNet fairly in testing, for each fashion outfit a user likes, we assume the user likes all of the items included in the outfit, and vice versa. We then train DVBPR based on user-item relationships. After training, we use the trained DVBPR to generate users' preference scores for fashion items included in the fashion outfit and taking an average over the scores to serve as a personalized fashion outfit score.

(6) **OP-Ins:** The instance-based approach of the proposed OP in this paper.

(7) **OP-Emb:** The embedding-based approach of the proposed OP in this paper.

To evaluate the Outfit Preference network (OP)'s recommendation quality against the above methods, we compute the positive and negative outfits' scores, $y_{u,i}$ and $y_{u,j}$, by using Area Under Curve (AUC) as our performance metrics. The AUC is calculated as follows.

$$AUC = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{1}{|\mathcal{D}_u|} \sum_{(i,j) \in \mathcal{D}_u} \mathbb{I}(y_{u,i} > y_{u,j}) \quad (10)$$

where $\mathcal{D}_u = \{(i,j)|(u,i) \in \mathcal{T}_u \wedge (u,j) \notin (\mathcal{P}_u \cup \mathcal{V}_u \cup \mathcal{T}_u)\}$, and $\mathbb{I}(\cdot)$ is an identity function counting the number of times $y_{u,i} > y_{u,j}$ is true.

Figure 5: Visualization of learned FIR item features with t-SNE transformation.
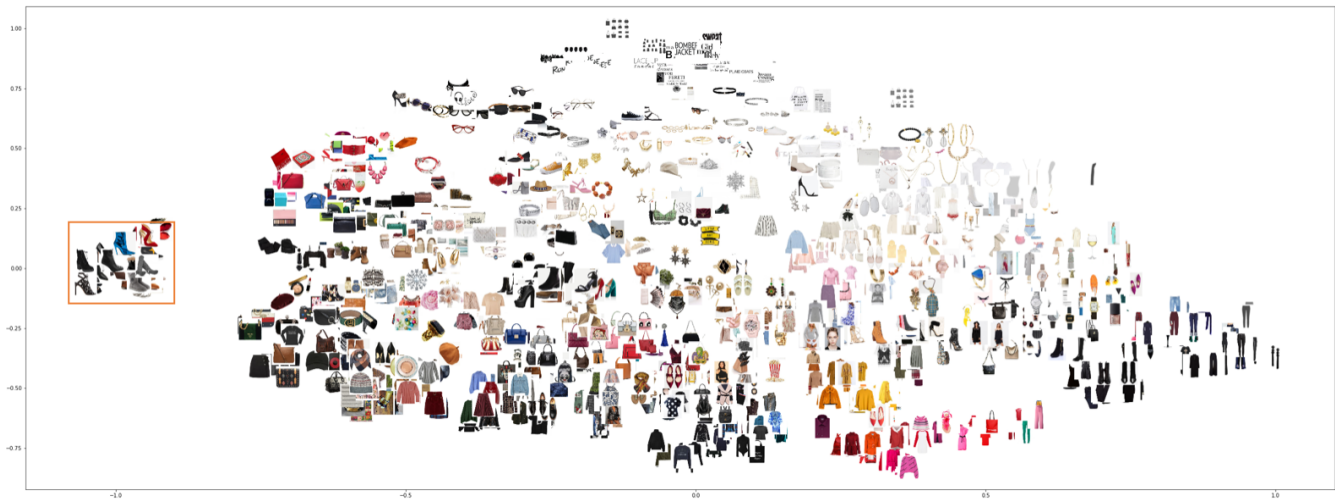

Figure 6: Visualization of DenseNet item features with t-SNE transformation.

The results of the recommendations are shown in Table 5. As shown, the performance of the two OP approaches, both instance approach and embedding approach, exceed other state-of-the-art recommendation models. Besides considering recommending all outfits in the testing set, we also test on recommending cold outfits (with only one user liking it), which tests the ability to handle sparse datasets. Both of our proposed approaches, OutfitNet-Ins and OutfitNet-Emb, still outperform the other comparing methods. Interestingly, OP-Ins performs better than OP-Emb on cold outfits.

5 OUTFITNET APPLICATIONS

After OutfitNet is trained, in addition to using it to make fashion outfit recommendations, the parameters learned in the model can also be used in other applications. Here we present three applications

that are enabled with a learned OutfitNet: fashion item embedding, outfit generation, and personalized fashion outfit attention.

5.1 Fashion Item Embedding

As we train the fashion item relevancy network (FIR), the network pushes fashion items that are more compatible/relevant with each other to be closer in the embedding space. Figure 5 shows the visualization of learned item embedding in a two-dimensional space (transformed using t-SNE [22]), compared to embeddings directly obtained from DenseNet pre-trained on ImageNet [13]. As one can see, when taking a closer look, instead of having visually similar items close to each other in the embedding space, FIR produces embeddings that have compatible fashion items closer to each other. More specifically, one can see there exists a cluster of shoes (in the orange rectangle) in Figure 6, which are all visually similar items

Algorithm 1 OutfitExpansion algorithm. Procedures starting with uppercase letters are what we define; procedures starting with lowercase letters are commonly used general functions.

```
1:  procedure OUTFITEXPANSION(S, k)
2:      H_items ← []
3:      for x ∈ S do
4:          H_items ← append(H_items, FIR(x))
5:      h_outfit ← averagePool(H_items)
6:      I_tmp ← I − S
7:      while |S| < k do
8:          T ← kdTree(I_tmp)
9:          x_candidate ← queryNearest(T, h_outfit)
10:         if CategoryValid(x_candidate, S) then
11:             S ← S ∪ {x_candidate}
12:             I_tmp ← I_tmp − {x_candidate}
13:             H_items ← append(H_items, FIR(x_candidate))
14:             h_outfit ← averagePool(H_items)
15:     return S
16: procedure CATEGORYVALID(x_candidate, S)
17:     c_candidate ← GetFirstLevelCategory(x_candidate)
18:     C_items ← {}
19:     for x ∈ S do
20:         C_items ← C_items ∪ { GetFirstLevelCategory(x) }
21:     if C_items ∩ {c_candidate} = ϕ then
22:         return True
23:     return False
24: procedure GETFIRSTLEVELCATEGORY(x)
25:     c ← GetCategory(x)
26:     while parent(c) ≠ root do
27:         c ← parent(c)
28:     return c
```


Figure 7: Examples of OUTFITEXPANSION's outputs, where $k = 5$.

(including their product category and silhouette). However, such a cluster cannot be observed in Figure 5. Instead, an example is the orange box in Figure 5, where the items in the box show a feasible outfit composition. Not only the repeating categories of items are not close to each other in the embedding space (which do not make sense to be put together in the same outfit), but also items that have similar styles are close together. These kinds of fashion item embeddings are suitable to help find compatible fashion items that form fashion outfits. We demonstrate its ability to generate fashion outfits in a later section.

5.2 Fashion Outfit Expansion

Based on the learned fashion item embedding, given a set of fashion items, we can find the additional fashion items that are closest to the given set to create a complete fashion outfit. This approach is a step further than the fill-in-the-blank (FITB) task, as discussed and evaluated in Section 4. We view this as a set expansion problem, which we propose an OUTFITEXPANSION algorithm to achieve it, as summarized in Algorithm 1.

Given a set of fashion items S, called *seed set*, and the desired size of expanded fashion outfit k, OUTFITEXPANSION algorithm iteratively grows S into a fashion outfit by adding fashion items that are closest to S in the learned embedding space, while not adding fashion items with duplicate categories.
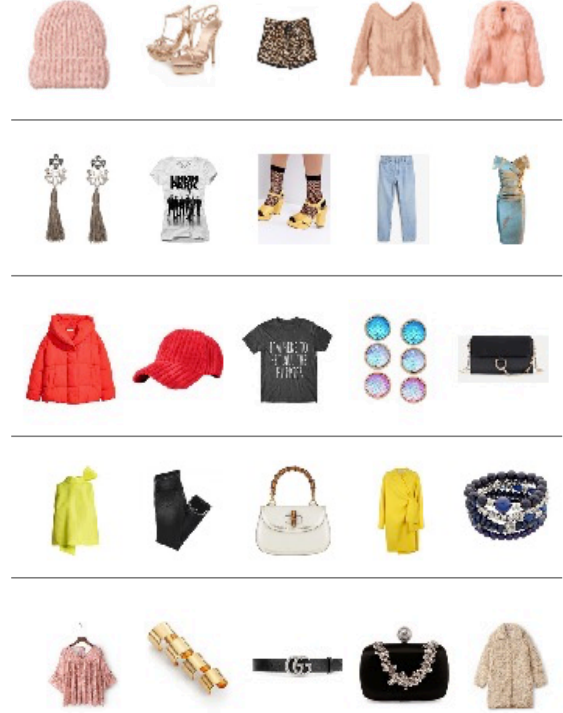
How OUTFITEXPANSION works in detail is as follows. First, all the items in S are passed through a trained FIR to obtain fashion item embeddings. Then these embeddings are pooled into a single embedding h_{outfit} to represent the current partial outfit's embedding. As long as the size of S has not achieved the desired size k, in each iteration, a spatially-indexed tree (e.g., KD tree), T, based on fashion item embeddings of I_{tmp} is built, where I_{tmp} is all of the items in I except the ones in S. We then query an item, $x_{candidate}$, that is nearest to h_{outfit} based on T.[6] If $x_{candidate}$ is in a valid category according to CATEGORYVALID, we add it to S, otherwise, we continue to search for the next one.

The purpose of CATEGORYVALID is to ensure that we don't add fashion items with categories that already exist in S. There are many ways to infer an item's fashion category, such as through image classification and textual description. However, since the fashion categories are hierarchical, considering whether there is a repeat at the most fine-grained level does not make sense. Instead, we only care about whether the items repeat at the highest level, which we call the first-level categories, such as outer, top, bottom, shoes, handbag, hat, and accessory. We construct a fashion category

---

[6]We build T using I_{tmp} instead of I because we want to prevent nearest items queried being the ones already in S.

Table 6: Example of learned personalized attention scores for three outfits.

	Outfit 1			Outfit 2			Outfit 3		
$\pi_{u,k}$	0.372	0.388	0.391	0.211	0.162	0.179	0.218	0.367	0.401
$\alpha_{u,k}$	0.319	0.339	0.342	0.367	0.306	0.327	0.214	0.369	0.417

hierarchy and determine the first-level categories of fashion items accordingly.[7]

Figure 7 shows five examples of outfits created using the OUTFIT-EXPANSION algorithm where $k = 5$ and S includes one fashion item. As one can see, by leveraging the learned fashion item embedding, one can generate outfits that include fashion items that are highly compatible with each other. We believe this is an effective approach to generate compatible outfits since it does not require complicated filtering and only requires categories of the items as additional metadata. One also does not need to specify detailed rules of an outfit, such as the items' orders and the categories required in an outfit.

5.3 Personalized Fashion Outfit Attention

One of the novel designs of OutfitNet's Outfit Preference network (OP) is its ability to incorporate users' personalized attention when learning their tastes toward fashion outfits. For the ease of explanation, we leverage one approach of OP, OP-Ins, in this section. Take OP-Ins as an example: after OP-Ins is trained, when passing a triplet of (u, i, i) into the trained OP-Ins, not only do we get the personalized fashion outfit score $y_{u,i}$ for outfit i, personalized fashion item scores $\pi_{u,i}$ for each item in outfit i, we also get α_u, which is u's personalized attention for the fashion items.

To demonstrate the learned personalized attention, we use an example shown in Table 6. As shown, we present the learned personalized item scores and personalized attention scores to three different outfits, based on the same user: user A. Among these three outfits, user A only liked outfit 1 in our dataset, and not the other two outfits. Visually, one can see that the main difference between outfit 1 and the other two outfits is the color theme. Outfit 1 has a somewhat neutral color theme, while outfits 2 and 3 each have brighter color combinations. A closer look at the learned personalized item scores shows that user A has an overall high (above 0.3) scores for all items in outfit 1, and an overall low (below 0.3) scores for all items in outfit 2. As for outfit 3, although user A has not liked it in our dataset, OP-Ins predicts a high personalized item score on the sweater, which has a similar color theme to what we infer user A may like from outfit 1.

Another aspect to see is the learned personalized attention. For both outfits 1 and 2, user A shows equally distributed attention scores across items in the outfits. This may be because the visual

styles across these items are relatively close to each other within an outfit. However, user A shows a specific higher attention on the sweater in outfit 3. This may be because the sweater is closer to user A's personal taste, and the visual style is significantly different from the other two items (shorts and hat) in outfit 3.

The above case study demonstrates OutfitNet's capability to learn users' fashion tastes toward individual clothing items in fashion outfits with only the labels of fashion outfits available. And by learning their personalized attention towards fashion clothing items in fashion outfits, we can better recommend fashion outfits to users.

6 CONCLUSION

In this paper, we proposed OutfitNet, a neural network model that learns the fashion item compatibility and recommends fashion outfits to users based on their fashion tastes. We propose a lightweight, yet effective design of a neural network to learn the fashion item compatibility, Fashion Item Relevancy network (FIR). We then formulate the fashion outfit recommendation problem into a multiple-instance-learning problem, and propose the Outfit Preference network (OP). Unlike the previously proposed methods, OutfitNet considers the multiple numbers of fashion clothing items involved in fashion outfits without assuming an underlying order of items. It also incorporates a personalized attention layer to learn the individuals' emphasis put on fashion items when assessing a fashion outfit. When learning, OutfitNet leverages Bayesian Personalized Ranking to learn the users' fashion tastes, which eliminates the difficulty in finding negative samples.

OutfitNet's effectiveness is demonstrated both quantitatively and qualitatively. Quantitatively, we show that our proposed FIR outperforms other state-of-the-art models in predicting fashion compatibility task, i.e., fill-in-the-blank (FITB) task. We also show that the proposed OP outperforms other state-of-the-art models in recommendation fashion outfits to users. Qualitatively, through visualization, we show that the fashion item embeddings learned by FIR capture the underlying fashion item compatibility, beyond just the color and silhouette of fashion items. We also present a framework for generating fashion outfits from a subset of fashion items based on the learned fashion item embeddings. Finally, we demonstrate that the learned personalized fashion item scores and personalized attention scores capture well the users' tastes toward fashion items, as well as their emphasis put on different fashion items when judging an outfit.

---

[7]Due to space limitations, we do not attach the fashion category hierarchy in this paper, which is rather space-consuming. However, we will release the digital version upon paper publication.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek Gordon Murray, Benoit Steiner, Paul A. Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: A System for Large-Scale Machine Learning. In *12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016, Savannah, GA, USA, November 2-4, 2016.* 265–283. https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* abs/1409.0473 (2014). arXiv:1409.0473 http://arxiv.org/abs/1409.0473

[3] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. Signature verification using a" siamese" time delay neural network. In *Advances in neural information processing systems.* 737–744.

[4] Ângelo Cardoso, Fabio Daolio, and Saúl Vargas. 2018. Product Characterisation towards Personalisation: Learning Attributes from Unstructured Data to Recommend Fashion Products. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018.* 80–89. https://doi.org/10.1145/3219819.3219888

[5] Kuang-Yu Chang, Kung-Hung Lu, and Chu-Song Chen. 2017. Aesthetic Critiques Generation for Photos. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017.* 3534–3543. https://doi.org/10.1109/ICCV.2017.380

[6] Wen Chen, Pipei Huang, Jiaming Xu, Xin Guo, Cheng Guo, Fei Sun, Chao Li, Andreas Pfadler, Huan Zhao, and Binqiang Zhao. 2019. POG: Personalized Outfit Generation for Fashion Recommendation at Alibaba iFashion. *arXiv preprint arXiv:1905.01866* (2019).

[7] Jorge Alé Chilet, Cuicui Chen, and Yusan Lin. 2016. Analyzing social media marketing in the high-end fashion industry using Named Entity Recognition. In *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining.* IEEE Press, 621–622.

[8] John Collomosse, Tu Bui, Michael J. Wilber, Chen Fang, and Hailin Jin. 2017. Sketching with Style: Visual Search with Sketches and Aesthetic Context. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017.* 2679–2687. https://doi.org/10.1109/ICCV.2017.290

[9] Zeyu Cui, Zekun Li, Shu Wu, Xiaoyu Zhang, and Liang Wang. 2019. Dressing as a Whole: Outfit Compatibility Learning Based on Node-wise Graph Neural Networks. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019.* 307–317. https://doi.org/10.1145/3308558.3313444

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA.* 248–255. https://doi.org/10.1109/CVPRW.2009.5206848

[11] Xintong Han, Zuxuan Wu, Yu-Gang Jiang, and Larry S. Davis. 2017. Learning Fashion Compatibility with Bidirectional LSTMs. In *Proceedings of the 2017 ACM on Multimedia Conference, MM 2017, Mountain View, CA, USA, October 23-27, 2017.* 1078–1086. https://doi.org/10.1145/3123266.3123394

[12] Wei-Lin Hsiao and Kristen Grauman. 2017. Learning the Latent "Look": Unsupervised Discovery of a Style-Coherent Embedding from Fashion Images. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017.* 4213–4222. https://doi.org/10.1109/ICCV.2017.451

[13] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2017. Densely Connected Convolutional Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017.* 2261–2269. https://doi.org/10.1109/CVPR.2017.243

[14] Maximilian Ilse, Jakub M. Tomczak, and Max Welling. 2018. Attention-based Deep Multiple Instance Learning. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018.* 2132–2141. http://proceedings.mlr.press/v80/ilse18a.html

[15] Wang-Cheng Kang, Chen Fang, Zhaowen Wang, and Julian McAuley. 2017. Visually-Aware Fashion Recommendation and Design with Generative Image Models. In *2017 IEEE International Conference on Data Mining, ICDM 2017, New Orleans, LA, USA, November 18-21, 2017.* 207–216. https://doi.org/10.1109/ICDM.2017.30

[16] Wang-Cheng Kang, Eric Kim, Jure Leskovec, Charles Rosenberg, and Julian J. McAuley. 2018. Complete the Look: Scene-based Complementary Product Recommendation. *CoRR* abs/1812.01748 (2018). arXiv:1812.01748 http://arxiv.org/abs/1812.01748

[17] Yuncheng Li, Liangliang Cao, Jiang Zhu, and Jiebo Luo. 2017. Mining Fashion Outfit Composition Using an End-to-End Deep Learning Approach on Set Data. *IEEE Trans. Multimedia* 19, 8 (2017), 1946–1955. https://doi.org/10.1109/TMM.2017.2690144

[18] Yujie Lin, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Jun Ma, and Maarten de Rijke. 2018. Explainable Fashion Recommendation with Joint Outfit Matching and Comment Generation. *CoRR* abs/1806.08977 (2018). arXiv:1806.08977 http://arxiv.org/abs/1806.08977

[19] Yusan Lin, Heng Xu, Yilu Zhou, and Wang-Chien Lee. 2015. Styles in the fashion social network: an analysis on Lookbook.nu. In *International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction.* Springer, 356–361.

[20] Yusan Lin, Yilu Zhou, and Heng Xu. 2014. The hidden influence network in the fashion industry. In *The 24th Annual Workshop on Information Technologies and Systems (WITS),* Vol. 1.

[21] Yusan Lin, Yilu Zhou, and Heng Xu. 2015. Text-Generated Fashion Influence Model: An Empirical Study on Style.com. In *48th Hawaii International Conference on System Sciences, HICSS 2015, Kauai, Hawaii, USA, January 5-8, 2015.* 3642–3650. https://doi.org/10.1109/HICSS.2015.438

[22] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[23] Oded Maron and Tomás Lozano-Pérez. 1997. A Framework for Multiple-Instance Learning. In *Advances in Neural Information Processing Systems 10, [NIPS Conference, Denver, Colorado, USA, 1997].* 570–576. http://papers.nips.cc/paper/1346-a-framework-for-multiple-instance-learning

[24] Jian Ren, Xiaohui Shen, Zhe L. Lin, Radomír Mech, and David J. Foran. 2017. Personalized Image Aesthetics. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017.* 638–647. https://doi.org/10.1109/ICCV.2017.76

[25] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009.* 452–461. https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=1630&proceeding_id=25

[26] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 815–823.

[27] Xuemeng Song, Fuli Feng, Xianjing Han, Xin Yang, Wei Liu, and Liqiang Nie. 2018. Neural Compatibility Modeling with Attentive Knowledge Distillation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018.* 5–14. https://doi.org/10.1145/3209978.3209996

[28] Xuemeng Song, Fuli Feng, Jinhuan Liu, Zekun Li, Liqiang Nie, and Jun Ma. 2017. NeuroStylist: Neural Compatibility Modeling for Clothing Matching. In *Proceedings of the 2017 ACM on Multimedia Conference, MM 2017, Mountain View, CA, USA, October 23-27, 2017.* 753–761. https://doi.org/10.1145/3123266.3123314

[29] Mariya I. Vasileva, Bryan A. Plummer, Krishna Dusad, Shreya Rajpal, Ranjitha Kumar, and David A. Forsyth. 2018. Learning Type-Aware Embeddings for Fashion Compatibility. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XVI.* 405–421. https://doi.org/10.1007/978-3-030-01270-0_24

[30] Sirion Vittayakorn, Kota Yamaguchi, Alexander C Berg, and Tamara L Berg. 2015. Runway to realway: Visual analysis of fashion. In *2015 IEEE Winter Conference on Applications of Computer Vision.* IEEE, 951–958.

[31] Wenguan Wang and Jianbing Shen. 2017. Deep Cropping via Attention Box Prediction and Aesthetics Assessment. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017.* 2205–2213. https://doi.org/10.1109/ICCV.2017.240

[32] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. 2015. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015.* 2048–2057. http://jmlr.org/proceedings/papers/v37/xuc15.html

[33] Ruiping Yin, Kan Li, Jie Lu, and Guangquan Zhang. 2019. Enhancing Fashion Recommendation with Visual Compatibility Relationship. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019.* 3434–3440. https://doi.org/10.1145/3308558.3313739

[34] Wenhui Yu, Huidi Zhang, Xiangnan He, Xu Chen, Li Xiong, and Zheng Qin. 2018. Aesthetic-based Clothing Recommendation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018.* 649–658. https://doi.org/10.1145/3178876.3186146

[35] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabás Póczos, Ruslan R. Salakhutdinov, and Alexander J. Smola. 2017. Deep Sets. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA.* 3394–3404. http://papers.nips.cc/paper/6931-deep-sets