Hierarchical Fashion Graph Network for Personalized Outfit Recommendation

Xingchen Li
Zhejiang University
xingchenl@zju.edu.cn

Xiang Wang
National University of Singapore
xiangwang@u.nus.edu

Xiangnan He
University of Science and Technology of China
xiangnanhe@gmail.com

Long Chen
Zhejiang University
longc@zju.edu.cn

Jun Xiao*
Zhejiang University
junx@zju.edu.cn

Tat-Seng Chua
National University of Singapore
dcscts@nus.edu.sg

ABSTRACT

Fashion outfit recommendation has attracted increasing attentions from online shopping services and fashion communities. Distinct from other scenarios (*e.g.,* social networking or content sharing) which recommend a single item (*e.g.,* a friend or picture) to a user, outfit recommendation predicts user preference on a set of well-matched fashion items. Hence, performing high-quality personalized outfit recommendation should satisfy two requirements — 1) the nice compatibility of fashion items and 2) the consistence with user preference. However, present works focus mainly on one of the requirements and only consider either user-outfit or outfit-item relationships, thereby easily leading to suboptimal representations and limiting the performance.

In this work, we unify two tasks, fashion compatibility modeling and personalized outfit recommendation. Towards this end, we develop a new framework, *Hierarchical Fashion Graph Network* (HFGN), to model relationships among users, items, and outfits simultaneously. In particular, we construct a hierarchical structure upon user-outfit interactions and outfit-item mappings. We then get inspirations from recent graph neural networks, and employ the embedding propagation on such hierarchical graph, so as to aggregate item information into an outfit representation, and then refine a user's representation via his/her historical outfits. Furthermore, we jointly train these two tasks to optimize these representations. To demonstrate the effectiveness of HFGN, we conduct extensive experiments on a benchmark dataset, and HFGN achieves significant improvements over the state-of-the-art compatibility matching models like NGNN [6] and outfit recommenders like FHN [29]. Our code has been released[1].

*Jun Xiao is the corresponding author.
[1]https://github.com/xcppy/hierarchical_fashion_graph_network

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SIGIR '20, July 25–30, 2020, Virtual Event, China
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401080

CCS CONCEPTS

• **Information systems** → **Personalization**; **Recommender systems**.

KEYWORDS

Personalized Fashion Recommendation, Fashion Compatibility, Graph Neural Networks

ACM Reference Format:
Xingchen Li, Xiang Wang, Xiangnan He, Long Chen, Jun Xiao, and Tat-Seng Chua. 2020. Hierarchical Fashion Graph Network for Personalized Outfit Recommendation . In *43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3397271.3401080

1 INTRODUCTION

Fashion recommendation plays an activate role in daily life, with the rapid growth of online shopping platforms (*e.g.,* Amazon and Taobao) and fashion related social networks (*e.g.,* Instagram). It can improve user experience and bring great profit to the shopping platforms. Personalized outfit recommendation is an emerging service, which targets at selecting a set of visually-compatible items as an outfit for a target user. Distinct from traditional fashion item recommendation which only routes a single product to users, it should satisfy two requirements — 1) *compatibility of fashion items*, meaning that the items within the same outfit should be visually compatible with each other, (*e.g.,* the long sleeve wear matches the high-heel shoes well in the outfit o_4 as Figure 1 shows); and 2)*consistence with personal taste*, meaning that the whole outfit should holistically match user preference, *i.e.,* each user might have individual dressing style (*e.g.,* as Figure 1 shows, the outfit o_1 fits the casual style of user u_1, while user u_2 has special interests on the outfit o_4 due to its long sleeve style).

Nevertheless, most present works focus only on one of the requirements — that is, either compatibility matching [6] or outfit recommendation [29] — while seldom modeling such two tasks simultaneously. In particular, a research line on compatibility matching solely exploits the mapping relationships between outfits and single items to estimate whether multiple fashion items form a good match (*i.e.,* outfit). For example, NGNN [6] builds a fashion graph upon a taxonomy of fashion categories, representing an outfit as a graph instance involving compatible items as its nodes. However, these works leave the personal interests of

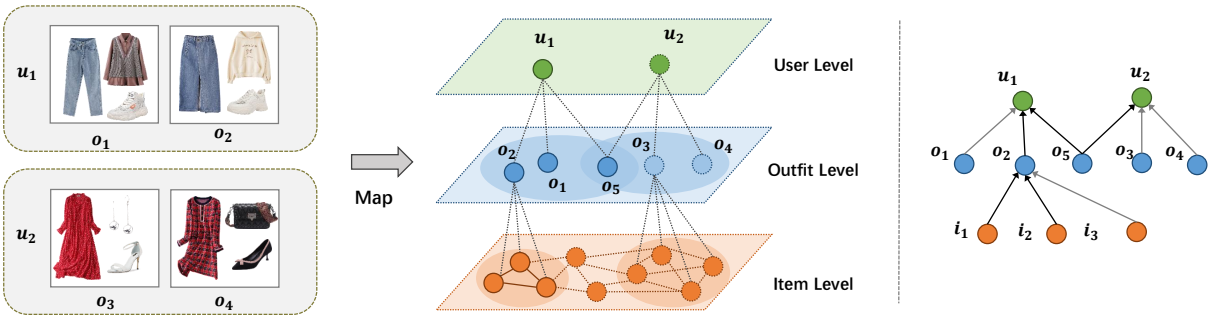


Figure 1: The illustration of our hierarchical fashion graph network, HFGN. HFGN consists of three levels (*i.e.,* user, outfit and item level). The message can propagate from lower level to higher level.

users untouched, hence typically serving as the tool to generate outfit features and failing to meet the second requirement of personalized recommendation. Another straightforward solution to outfit recommendation is to subsume such task under the framework of traditional recommendation, relying largely on the user-outfit interactions. Specially, the fashion items are simply seen as the pre-existing features of the outfit; as such, traditional recommenders can be adopted in such scenarios. For example, VBPR [13] and ACF [2] can be employed on users' historical interactions with outfits to perform future recommendation. However, these methods are not tailored to outfit recommendation, forgoing the compatibility matching between fashion items.

Studies on jointly conducting compatibility modeling and outfit recommendation is under explored until recent FPITF [20] and FHN [29]. To be more specific, FPITF [20] aggregates user preference on each item within an outfit, and integrates them with pairwise compatibility scores between items as a user's holistic preference on the outfit. FHN [29] resorts to the similar paradigm. Examining such paradigm, we find that the outfit-item mappings are treated as isolated data instance towards compatibility matching, while forgoing the relationships among data instances (*e.g.,* outfits with co-occurred items); analogously, the user-outfit interactions are fed individually into the recommender, while ignoring their relationships (*e.g.,* co-purchased outfits or behaviorally similar users). This paradigm obscures the complex relationships among users, outfits, and items, easily leading to suboptimal representations and further limiting the recommendation performance.

In this work, to solve the foregoing limitations, we explicitly present the complex relationships among users, outfits, and items as a hierarchical fashion graph. To be more specific, it consists of three levels — user, outfit, and item levels — where each level contains of the corresponding type of nodes. Distinct from traditional graphs, such a hierarchical graph highlights the connections cross levels. For example, if outfit o contains item i, the node o in the outfit level will be connected with the node i in the item level; analogously, if user u has purchased outfit o, there is an edge between nodes u and o across the user and outfit levels.

Thereafter, we build a new framework, termed **Hierarchical Fashion Graph Network (HFGN)**, upon the hierarchical fashion graph. In particular, HFGN employs the information propagation mechanism from graph neural networks (GNNs) to distill useful signals from the bottom to the top, inject the relationships into representations and facilitate the compatibility matching and outfit recommendation. More specifically, we assign each user/outfit with an ID embedding while representing each item with its visual features. The information propagation rule aggregates useful signals from fashion items to update outfit representations, and further refine user representations by integrating messages passing from his/her historical outfits. Furthermore, we propose a joint learning scheme to conduct the compatibility matching and outfit recommendation simultaneously. To demonstrate effectiveness of HFGN, we conduct extensive experiments, and the experimental results show that our model outperforms the state-of-the-art models *w.r.t.* both two tasks.

To summarize, we make the following main contributions in this paper:

- We propose a Hierarchical Fashion Graph Neural Network (HFGN) to obtain more expressive representations for users and outfits. Benefiting from the message propagation rules, the representations can be updated by the neighbour embeddings iteratively.
- Different from the existing methods which only consider item-level semantics for outfits, we incorporate outfit-level semantics into the representations for outfits.
- Compared to separately considering compatibility matching and personalized recommendation, we regard the compatibility information as a passing message in the graph and encode this information into item and outfit representations. Experiments show the rationality and effectiveness of this modeling.

2 HFGN FRAMEWORK

We now present our HFGN framework which is equipped with three key components: 1) embedding initialization, which initializes embeddings for user, outfit, and item nodes; 2) hierarchical graph convolution, which refines the node embeddings by propagating information from lower levels to higher levels — that is, gather information from item nodes to update outfit representations, and then augment user representations via the historical outfits; and 3) model prediction, which outputs the prediction score for personalized recommendation and compatibility prediction.

2.1 Embedding Initialization

As Figure 1 shows, we organize users, outfits, and items in the form of a hierarchical fashion graph, where these three types of nodes are at the top, internal, and bottom levels, respectively. To characterize the latent features, we represent each user/outfit/item ID with a vectorized representation (*aka.* embedding). More formally, we denote IDs of user u, outfit o, and item i separately as $\mathbf{u} \in \mathbb{R}^d$, $\mathbf{o} \in \mathbb{R}^d$, and $\mathbf{i} \in \mathbb{R}^d$, where d is the embedding size.

As a result, we can obtain an embedding table for all the nodes as follows:

$$\mathbf{E} = [\ \underbrace{\cdots, \mathbf{i}, \cdots}_{\text{item embeddings}}\ ,\ \underbrace{\cdots, \mathbf{o}, \cdots}_{\text{outfit embeddings}}\ ,\ \underbrace{\cdots, \mathbf{u}, \cdots}_{\text{user embeddings}}\ ], \tag{1}$$

where $\mathbf{E} \in \mathbb{R}^{(N_U+N_O+N_I)\times d}$ concatenates embeddings of users, outfits, and items; N_U, N_O, and N_I are the number of users, outfits, and fashion items, respectively.

As only ID information is available for users and outfits, we get inspirations from the mainstream CF models [17, 32] and project each user/outfit ID into an embedding. Such trainable embeddings are used to memorize the latent features of users and outfits. As for each fashion item i, we have its visual feature $\mathbf{x}_i$. Moreover, since items are associated with different fashion categories (*e.g.,* jeans, T-shirt), we use category-aware encoders to extract useful information from the visual features as item embeddings. More formally, the initial embeddings of items are encoded as:

$$\mathbf{e}_{(i)} = f_c(\mathbf{x}_i), \tag{2}$$

where $f_c(\cdot)$ is the encoder for category c, which is implemented by a two-layer MLP. As such, the item features are projected into the same latent space to that of users and outfits, facilitating the further modeling of their complex relationships.

2.2 Hierarchical Graph Convolution

By organizing users, outfits, and items in a hierarchical graph, we can leverage their connectivities to help exhibit their underlying relationships. For example, As the right in Figure 1 shows, the path $i_1 \rightarrow o_2$ states the fact that outfit o_2 includes item i_1, while $o_2 \rightarrow u_1$ presents the behavior that user u_1 purchases outfit o_2; meanwhile, the longer path $i_1 \rightarrow o_2 \rightarrow u_1$ might reflect user u_1's interest on item i_1, while $\{i_1, i_2\} \rightarrow o_2$ and $o_5 \rightarrow \{u_1, u_2\}$ separately indicate the compatibility of items and behavioral similarity of users. Hence, exploiting such connectivities is of crucial importance to explore relationships among users, outfits, and items, and is a promising solution to unify compatibility modeling and outfit recommendation.

Recent studies on graph neural networks [11, 22, 37] show that the information propagation over graph structure is able to effectively distill useful information from multi-hop neighbors and encode higher-order connectivity into the representations. Inspired by this, we devise a new hierarchical graph convolution (HGC) to perform the embedding propagation mechanism over our fashion graph, so as to refine their embeddings. In particular, there are three embedding-propagation steps — 1) information propagation across items, which refines item embeddings by incorporating the compatibility modeling; 2) information propagation from items to outfits, which aggregates item semantics into outfit embeddings; and 3) information propagation from outfits to users, which integrates historical outfits as user representations. In what follows, we elaborate these three ingredients.

2.2.1 ***Information Propagation Across Items.*** Items are at the lowest level of HFGN, providing visual features of individual items, as well as their compatible relationships. For example, the connectivities $\{i_1, i_2\} \rightarrow o_2$ not only describe that items i_1 and i_2 belong to the same outfit o_2, but also reflect that i_1 and i_2 are compatible. Hence, such compatibility information suggests that compatible items should have more information interchange than that in different outfits. Towards presenting compatibility among items in an explicit fashion, we construct an item graph for each outfit first.

Item Graph Construction. Before constructing item graphs for individual outfits, we first build a uniform category graph [6] for all outfits, where the category information of items serve as the prior knowledge of items. In particular, each item is assigned with only one specific category, such as *shirts*, *sandals*, and *jeans*. Different category pairs are associated with varying co-occurrence frequencies, reflecting the coarse-grained compatibility of items at a granular level of category. For instance, *necklaces* co-occurs more frequently with *coats* in outfits, than *sandals*. Hence, we build a weighted category graph $\mathcal{G}_c = \{(c, c', w(c, c'))|c, c' \in C\}$, where C is the set consisting of 60 categories in total (*cf.* data statistics in Table 2). Wherein, each category pair (c, c') is assigned with a weight as follows:

$$w(c, c') = \frac{g(c, c')/g(c')}{\sum_{c''\in C} g(c, c'')/g(c'')}, \tag{3}$$

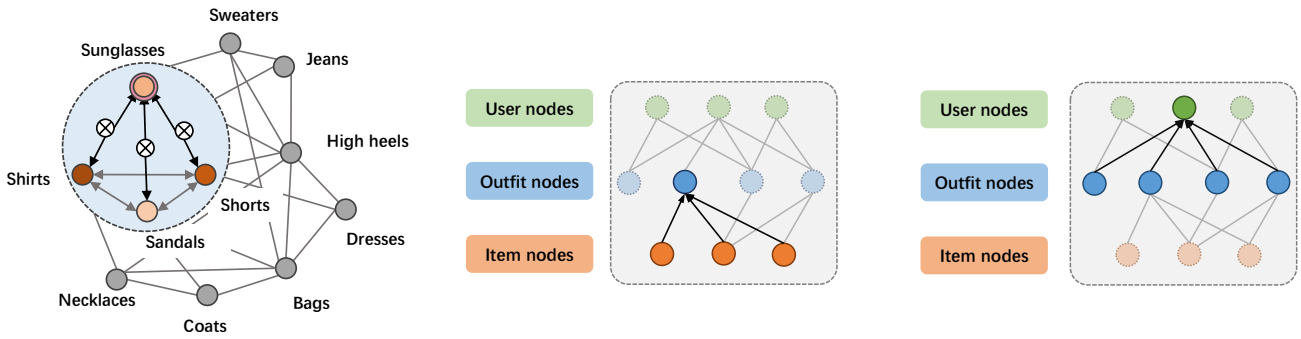
where $g(c, c')$ denotes the co-occurrence frequency of categories c and c' appearing in the same outfits, while $g(c)$ counts the frequency of c in the outfit-item and item-category mappings.

Having established $\mathcal{G}_c$ for all outfits, we now construct an item graph tailored $\mathcal{G}_o$ for a single outfit o. In particular, we activate the category nodes which appear in outfit o (*e.g.,* the orange nodes shown in Figure 2(a)), while removing the others. More formally, $\mathcal{G}_o$ is defined as $\{(c, c', w(c, c'))|c, c' \in \mathcal{N}_o\}$, where $\mathcal{N}_o$ is the item set of outfit o. Clearly, the weights of $\mathcal{G}_o$ directly inherit from the original category graph, and only parts of nodes with their connections are activated as the blue circle in Figure 2(a) shows.

Item-Wise Information Construction. Presenting the coarser-grained compatibility in the form of item graph, we now focus on one specific item and distill useful signals from its neighbors, where the compatibility *w.r.t.* categories is encoded. In particular, the information being propagated from the neighboring item i' to the ego item i is formalized as:

$$\mathbf{m}_{i'\rightarrow i} = w(i, i')\sigma(\mathbf{W}_1(\mathbf{i} \odot \mathbf{i}')), \tag{4}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d\times d}$ is the trainable weight matrix to perform transformation; $\sigma(\cdot)$ is a nonlinear activation function, which is set as LeakyReLU [30] here; $\odot$ is the element-wise product. Hence, the signal $\mathbf{i} \odot \mathbf{i}'$ accounts for the visual compatibility between items i and i', encouraging compatible items to contribute more. Furthermore, the weight $w(i, i')$ takes the categorical compatibility into consideration to control how much signals are being passed



(a) Information propagation across items. (b) Information propagation from item to outfit level. (c) Information propagation from outfit to user level.

Figure 2: Information propagation across three levels in HFGN. (a) presents information propagation across items; (b) presents information propagation from item to outfit level; (c) presents information propagation from outfit to user level.

across categories. Such weights also can be seen as the discounting factors adopt in GNN models [22, 37].

Cross-Item Information Aggregation. For each item node, we can leverage the signals, which are pertinent to its affinity with co-occurred items (*i.e.,* neighbors), to update its embedding. Here we employ the sum aggregator on item i's neighbors as follows:

$$\mathbf{i}^* = \mathbf{i} + \sum_{i' \in \mathcal{N}_i} \mathbf{m}_{i' \to i}, \tag{5}$$

where $\mathbf{i}^*$ is the updated embedding of item i. Here only the sum aggregator is applied, leaving the exploration of other aggregators, such as attention networks [4, 37], in future work. As a result, the compatibility information carried in the first-order connectivity are encoded into the item embeddings. We can stack more layers to synthesize richer semantics in higher-order connectivity, leaving this exploration in future work.

2.2.2 ***Information Propagation from Item to Outfit Level.*** Intuitively, an outfit can be described by its involved items. Taking Figure 1 as an example, outfit o_1 consists of a sweater, jeans, and running shoes, while outfit o_2 is composed of a pullover, jeans, and sneakers. Rich item features help reveal underlying relationships between outfits *w.r.t.* visual similarity and categorical compatibility. For example, outfits o_1 can serve as a substitute for o_2, due to the closely compatibility style; meanwhile, the style of outfit o_4 is different from that of o_1 and o_2, since they have no overlapping categories or items. We hence augment ID embeddings of outfits with the representations of the involved items, to improve the quality of embedding. In particular, we build a heterogeneous graph involving item and outfit nodes, where the edges are the item-outfit links.

Item-Wise Information Construction. Focusing on an outfit, we refine the information that are influential to it from the involved items. Formally, the messages being passed from the neighboring item i to the ego outfit o is:

$$\mathbf{m}_{i \to o} = \frac{1}{|\mathcal{N}_o|} \sigma(\mathbf{W}_2 \mathbf{i}^*), \tag{6}$$

where $\mathbf{W}_2 \in \mathbb{R}^{d \times d}$ is a trainable matrix to perform transformation; $\frac{1}{|\mathcal{N}_o|}$ is the normalization term to handle the varying number of involved items and stabilize the training.

Outfit-Wise Information Aggregation. Analogous to the cross-item information aggregation, we combine the information from all involved items together as the final representation of an outfit, as follows:

$$\mathbf{o}^* = \mathbf{o} + \sum_{i \in \mathcal{N}_o} \mathbf{m}_{i \to o}. \tag{7}$$

Obviously, the refined representation $\mathbf{o}^*$ is composed of the ID embeddings and item-aware features. Different from prior studies [20, 29] that build outfit representations upon the visual features of items solely, this information aggregation additionally considers the compatibility scores.

2.2.3 ***Information Propagation from Outfit to User Level.*** Present studies [16, 21, 23] have shown that personal history directly profiles a user's preference. For instance, analyzing the historical outfits (*i.e.,* o_1 and o_2) of user u_1, we might summarize her dressing style; moreover, $o_5 \to \{u_1, u_2\}$ indicates the behavioral similarity between user u_1 and u_2. Furthermore, collected personal histories reflect CF signals [40, 43], referring that behaviorally similar users would have similar preference on outfits. We hence enrich ID embeddings of users by incorporating the representations of historical outfits. Towards that, we organize the user-outfit interactions in the form of heterogeneous graph.

Outfit-Wise Information Construction. For a target user u, we focus on the items he/she adopted before $\mathcal{N}_u$, and extract useful information from each outfit o as follows:

$$\mathbf{m}_{o \to u} = \frac{1}{|\mathcal{N}_u|} \sigma(\mathbf{W}_3 \mathbf{o}^*), \tag{8}$$

where $\mathbf{W}_3 \in \mathbb{R}^{d \times d}$ is the transformation matrix; here we assume that different outfits might contribute equally to profile a user, hence using $\frac{1}{|\mathcal{N}_o|}$ as priors and leaving the exploration of attentive weights in future.

User-Wise Information Aggregation Thereafter, we employ the sum aggregator on the historical items, updating the user's representation as follows:

$$\mathbf{u}^* = \mathbf{u} + \sum_{o \in \mathcal{N}_u} \mathbf{m}_{o \to u}, \tag{9}$$

where the final representation $\mathbf{u}^*$ consists of two components — the ID embedding, which characterizes the intrinsic features of u, and the outfit-aware features, which present her dressing style explicitly. Distinct from prior works [20, 29] which use ID embeddings for users only, our HFGN results in better representation ability.

After propagating information within the hierarchical fashion graph, we allow the information flow from the bottom to the top levels, exploiting the complex relationships among items, outfits, and users to guide the representation learning.

2.3 Model Prediction

Thereafter, we propose a joint learning scheme to conduct the compatibility matching and outfit recommendation simultaneously.

2.3.1 ***Personalized Outfit Recommendation.*** To predict how likely user u would purchase outfit o, we employ the inner product on their representations as:

$$\hat{y}_{uo} = \mathbf{u}^{*\top} \mathbf{o}^*, \tag{10}$$

which casts the predictive task as the similarity estimation between u and o in the same latent space. As the main focus of this work is representation learning, we leave the exploration of interaction modeling in future work.

2.3.2 ***Compatibility Matching.*** To estimate whether multiple fashion items form a good outfit, we utilize the item representations (*cf.* Equation 5) to calculate the matching score. Distinguished from present works [20, 29] that simply aggregate the pairwise compatibility scores together, we argue that items have varying importance for the outfit. For example, in outfit o_3, as the long dress determines the holistic style, it is more important than the accessories. We hence differentiate importance of items in one outfit via a self-attention mechanism, which generates a R-view attention map and a R-view score map. Formally, the attention map is calculated as:

$$\mathbf{A} = \rho\Big(\mathbf{W}_4 \sigma(\mathbf{W}_5 \mathbf{I}^\top)\Big), \tag{11}$$

where $\mathbf{I} \in \mathbb{R}^{n \times d}$ is embedding matrix of one outfit and n is the length of the outfit; $\mathbf{W}_4 \in \mathbb{R}^{R \times v}$ and $\mathbf{W}_5 \in \mathbb{R}^{v \times d}$ are two trainable weight matrices. After transformation by $\mathbf{W}_4$ and $\mathbf{W}_5$, we can get an R views attention map $\mathbf{A} \in \mathbb{R}^{r \times n}$. $\rho(\cdot)$ is set as the softmax function to normalize the attention scores over items.

Thereafter, we establish the R-view score map for each outfit, considering each view as a latent factor which is influential for compatibility, as follows:

$$\mathbf{C} = \sigma\Big(\mathbf{W}_6 \sigma(\mathbf{W}_7 \mathbf{I}^\top)\Big), \tag{12}$$

where $\mathbf{W}_6 \in \mathbb{R}^{R \times v}$ and $\mathbf{W}_7 \in \mathbb{R}^{v \times d}$ are weight matrices. As such, we project the original item representations into a latent space, describing each item from multiple factors.

Based on such attention and score maps, we get the weighted compatibility score of the outfit o as follows:

$$\hat{s}_o = \sum_{r=1}^{R} \mathbf{a}_r^\top \mathbf{c}_r, \tag{13}$$

where $\mathbf{a}_r \in \mathbb{R}^n$ and $\mathbf{c}_r \in \mathbb{R}^n$ is the r-th rows of $\mathbf{A}$ and $\mathbf{C}$, respectively.

2.4 Optimization

In the following, we introduce the objective function for our model and training strategy.

2.4.1 ***Objective function.*** We adopt Bayesian Personalized Ranking (BPR) algorithm [32] for both tasks. BPR assumes the observed interaction has higher prediction scores than unobserved ones. The objective functions for two tasks are,

$$\mathcal{L}_{mf} = \min_{\Theta} \sum_{(u,o,o') \in \mathcal{H}} -\ln \mu(\hat{y}_{uo} - \hat{y}_{uo'}), \tag{14}$$

$$\mathcal{L}_{com} = \min_{\Theta} \sum_{(o,o') \in \mathcal{H}'} -\ln \mu(\hat{s}_o - \hat{s}_{o'}), \tag{15}$$

where $\mathcal{H} = \{(u, o, o')\}$ is the training set for outfit recommendation, where each triple (u, o, o') denotes user u's historical interaction with outfit o and a unobserved interaction with outfit o'; $\mathcal{H}' = \{(o, o')\}$ is the training set for compatibility modeling, where each pair (o, o') denotes the observed outfit o (*i.e.,* positive samples) and a unobserved outfit o' (*i.e.,* negative samples by randomly generated); $\mu(\cdot)$ is the sigmoid function; and Θ is the set of model parameters, on which L_2 normalization is conducted to avoid overfitting.

2.4.2 ***Training Strategy.*** Due to the imbalance of the training data (*i.e.,* recommendation task has 1.6M positive samples while compatibility task has 0.027M positive samples in our dataset shown in table 1 and table 2.) and different model convergence speeds, we set different learning rates for two tasks. And we jointly optimize $\mathcal{L}_{mf}$ and $\mathcal{L}_{com}$ in one epoch in our training process.

3 EXPERIMENT

We perform our experiments on a benchmark dataset, POG [5], and we aim to answer the following questions:

- **RQ1**: How does HFGN perform as compared with the state-of-the-art methods on personalized outfit recommendation task?
- **RQ2**: How does the message propagation on each level affect HFGN?
- **RQ3**: How does HFGN perform as compared with the state-of-the-art methods on compatibility matching task?

3.1 Experiment settings

3.1.1 ***Dataset.*** As personalized outfit recommendation task is a fresh task in fashion community, the avaliable dataset is limited. POG [5] is the only available large-scale dataset that meets our requirements. The dataset collects click actions from 3.57 million users, including 1.01 milion outfits and 583 thousand individual items with context information.

POG dataset provides three files: 1) user data, which records the user clicks on both outfits and items, 2) outfit data, which lists the items composed of the outfit, and 3) item data, which provides the

Table 1: Statistics of the dataset for personalized outfit recommendation.

#Users	#Outfits	#Interactions	#Items
53,897	27,694	1,679,708	42,563

Table 2: Statistics of the dataset for compatibility matching.

Dataset	#Outfits	#items	#categories
Training set	27,694	42,563	60
Test set	6,924	15,984	60

context details of items including categories, text descriptions and image download links. In our experiment, we utilize the user clicks on outfits to construct our dataset. We extract visual features for items from a pretrained Resnet-152.

Dataset for recommendation. To ensure the quality of dataset, we only retain the users with at least 20 interactions and the outfits with at least 10 interactions. The number of item categories is 60. The statistics of recommendation dataset is shown in Table 1. To evaluate the recommendation performance, we randomly select 80% of click histories of each user to build the training set, and take the remaining histories as the test set. We also split 10% of training set as validation set to tune the hyper-parameters.

Dataset for compatibility matching. We take the positive outfits in recommendation dataset as training set for compatibility matching, and additionally select 6,924 (10% of training dataset) outfits that has never appeared in training set to construct test dataset. The statistics are shown in Table 2. To evaluate the performance of HFGN on compatibility matching, we adopt a widely used task, Fill-in-the-Blank [12].

Fill-in-the-Blank (FLTB). Fill-in-the-Blank (FLTB) [12] is a task to select the most compatible item from the candidates to fill in the blank in one outfit, as Figure 4 shows. For each outfit in test dataset, we randomly mask one item as a blank, and then randomly select 3 items from other outfits to form a candidate set with the masked item. We set the masked item as the ground truth, assuming that the masked item is more compatible than other candidates.

3.1.2 ***Parameter Settings.*** We perform grid search to tune the hyper-parameters for our model and baselines. We search the batch size in $\{256, 512, 1024\}$, and we tune regulation rate and learning rate in $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$ and $\{0.0001, 0.0005, 0.001, 0.005, 0.01\}$, respectively. Moreover, we optimize all the models with Adam optimizer.

3.2 Personalized Outfit Recommendation

In this section, we will discuss the performance of our model on personalized outfit recommendation task compared to the state-of-the-art methods.

3.2.1 ***Evaluation Metric.*** To assess the performance of Top-K recommendation, we adopt four widely-used metrics [17, 40]: HR@K, NDCG@K, Recall@K and Precision@K.

Table 3: Overall performance comparison on personalized outfit recommendation.$*$ denotes the statistical significance for $p < 0.001$.

	HR@10	NDCG@10	Recall@10	Precision@10
FPITF	0.1006	0.0420	0.0183	0.0112
FHN	0.1109	0.0490	0.0208	0.0119
MF	0.2121	0.0872	0.0434	0.0239
VBPR	0.2201	0.0949	0.0449	0.0248
NGCF	**0.2619**	**0.1143**	**0.0554**	**0.0310**
HFGN	**0.2833***	**0.1241***	**0.0605***	**0.0339***
%Improv.	8.17%	8.57%	9.20%	9.35%

- HR@K measures weather any positive samples present in the top K position. It is 1 if yes, otherwise 0.
- Recall@K is the proportion of positive samples that have been successfully recommended to the user.
- Precision@K is the proportion of the recommended items that are relevant to the user.
- NDCG@K is a widely used measure to evaluate the quality of the ranked list, considering the graded relevance among positive ad negative samples within the top K ranking list.

We set K as 10 in our experiment. For all the metrics, we report the mean of all users as the final score.

While the validation set is used to tune the hyper-parameters, we report the performance on the test set.

3.2.2 ***Baselines.*** As the personalized outfit recommendation task is fresh, and only a few works [20, 29] has researched on it. To demonstrate the effectiveness of our proposed model, we select some traditional recommendation models (*e.g.,* MF, VBPR) as our baselines. Besides, we also select a graph-based recommendation model to compare since we exploit graph structure in our method.

- **FPITF** [20]: FPITF represents users and items with ID embeddings and visual features respectively. The prediction score is composed of user's preference score for each item within one outfit and the compatibility score for each item pair in the outfit.
- **FHN** [29]: FHN encodes the visual features with category encoders and then learn binary code for both user and item embeddings. The final score is also composed of two parts, preference score for user-item pairs and compatibility score for item-item pairs. For each part, FHN additionally introduces a diagonal weighting matrix to better model the interactions.
- **MF** [32]: Matrix Factorization (MF) model is one of the most popular techniques in personalized recommendation task. In this model, users and items are both projected into a same latent space and represented by the vectors in this space and their interaction score is calculated by inner product.
- **VBPR** [13]: Compared to MF model, VBPR additionally considers the user preference on visual factors. Here, we represent the visual feature of one outfit by averaging the visual features of the items within the outfit.
- **NGCF** [40]: NGCF utilizes graph structure to model the high-order interaction between users and items. Node embeddings are refined by stacking multiple propagation layers in

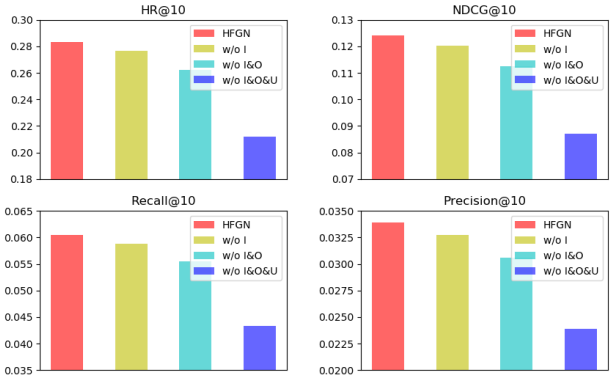


Figure 3: Performance comparison of ablation study on HFGN.

interaction graph. Here, we conduct a two-layer propagation and the nodes in interaction graph are users, outfits and items.

3.2.3 ***Performance Comparsion (RQ1).*** Table 3 reports the performance results on personalized outfit recommendation. From the table, we have the following observations:

- FPITF and FHN perform much worse than other baselines. In both FPITF and FHN, they evaluate the user preference on one outfit only by averaging the preference score on each item. However, outfits and items have different semantics, it's insufficient to only consider item-level preference. In other models, they employ ID embeddings of outfits to capture this information, enhancing the performance greatly in personalized recommendation task.
- Compared to BPR, VBPR achieves better performance. Such improvement indicates the importance of incorporating visual signals into prediction formulations.
- From the Table 3, we can observe that NGCF performs much better than MF and VBPR. Benefiting from multiple propagation layers in interaction graph, NGCF is capable of modeling high-order connectivities among users, outfits and items.
- HFGN yields the best performance on all the metrics. In particular, HFGN achieves the improvement over the strongest baseline (*i.e.,* NGCF) of 9.35% *w.r.t.* Precision@10. Benefiting from message propagation mechanism, HFGN is capable of distilling useful signals from the bottom to the up and modeling complex relationships including user-outfit interactions and outfit-item mappings.

3.2.4 ***Study of HFGN (RQ2).*** To demonstrate the effectiveness of message propagation on each level, we conduct an ablation study. We disable the message propagation for each level and compare the performance, the results are shown in Figure 3.

- First, we disable the message propagation on item level, termed $\text{HFGN}_{\text{w/o I}}$. The message discarded in this operation is the compatibility of pair of items in one outfit, which is an important factor that influences user's interest. From Figure 3, we can see that $\text{HFGN}_{\text{w/o I}}$ slightly underperforms HFGN. It seems that discarding this part of information doesn't hurt the performance much in our experiment. That's because the negative samples in our test dataset is the remaining outfits that users haven't clicked. These outfits are collected directly from the website and have a good compatibility. Therefore, the compatibility influences slightly on evaluation results. In fact, our model has a good capability to analysis compatibility, which will be discussed in Section 3.3. As we mainly research on the architecture of personalized outfit recommendation, we leave the expansion of the test dataset in future work.
- Then, we disable the message propagation from item to outfit level, termed $\text{HFGN}_{\text{w/o I\&O}}$. It means the representation of an outfit only has its ID embedding without item information. From Figure 3, we can see that $\text{HFGN}_{\text{w/o I\&O}}$ performs worse than both HFGN and $\text{HFGN}_{\text{w/o I}}$. It makes sense since that item information plays a significant role in modeling outfit representation. It hence illustrates the rationality and effectiveness of our message propagation rule from item to outfit level.
- Finally, we disable the message propagation from outfit level to user level, termed $\text{HFGN}_{\text{w/o I\&O\&U}}$. It means that we only utilize ID embeddings for both users and outfits, which equals to MF model. From Figure 3, we can observe that the performance decreases greatly compared to the models mentioned above. Compared to $\text{HFGN}_{\text{w/o I\&O}}$, the message discarded in this operation is the historical interactions of users while modeling user profiles, which demonstrates that the historical interaction plays a significant role in modeling user preference.

Table 4: Overall performance comparison on compatibility matching.$*$ denotes the statistical significance for $p < 0.001$.

	FLTB
Random	0.2425
SiameseNet	0.5039
Bi-LSTM	0.6384
FHN	0.7422
NGNN	**0.8422**
HFGN	**0.8797***

3.3 Compatibility Matching

3.3.1 ***Evaluation Protocols.*** For each outfit in test dataset, we randomly select an item as the blank, and set three negative candidates. The target is to select the correct answer from four candidates to fill in the blank masked in the outfit. We report the accuracy to assess the performance.

3.3.2 ***Baselines.*** We compare our model with the following baselines:

- **Random**: The result for FLTB is randomly selected from 4 candiates.

Figure 4: Real examples of HFGN and two strong baselines (*i.e.,* NGNN and FHN) on Fill-in-the-Blank task. The green box circled the correct answer.

- **SiameseNet** [36]: SiameseNet sends the a pair of items into the a siamese network to project them into a style space and compare the distance between them. The compatibility score of the whole outfit is calculated by averaging the pairwise similarities in our experiment.
- **Bi-LSTM** [12]: Bi-LSTM takes an outfit as a sequence. It applies a bidirectional LSTM to learn the compatibility and predict the masked item.
- **FHN** [29]: FHN encodes the visual features with category encoders and then learn binary codes for item embeddings. FHN only classifies the fashion items into 3 categories (*i.e.,* up, bottom and shoes). The outfit score is the mean of pairwise compatibility score of items within the outfit.
- **NGNN** [6]: NGNN exploits the category information to represent items as the nodes in the category graph. Thereafter, the node embeddings are updated through a one-layer graph convolution and a GRU cell. Finally, NGNN calculates the scores from each node embedding and introduces a self-attention mechanism [35] to add them together as the output.

3.3.3 ***Performance Comparision.*** We report the accuracy of test results in Table 4. From Table 4, we have the following observations:

- Compared to other methods, SiameseNet achieves poor performance, indicating that only averaging the compatibility scores of item pairs is insufficient to learn high-order compatibility knowledge due to the overlook of the integrity of outfit, further limiting the performance.
- We can see that Bi-LSTM performs better than SiameseNet. The reason might be that introducing Bi-LSTM can better learn the potential knowledge on compatibility. Compared to SiameseNet which directly averages the pairwise scores of items, Bi-LSTM takes the whole outfit as a sequence and learns high-order relations beyond item-level comparison.
- FHN performs much better than SiameseNet, although it just averages the pairwise compatibility as well. Such improvement might be attributed to the introduced category knowledge. It verifies the importance of category information in modeling compatibility. Nevertheless, it only considers 3 category labels, leading to miss compatibility relations on fine-grained categories.
- The two graph based methods, NGNN and HFGN, achieves better performance compared to other methods, indicating that the graph structure can better model complex interactions among items, further effectively inferring compatibility information. The improvement over Bi-LSTM and FHN method verifies the graph representation can better model the item interactions than sequence representation and pairwise representation.
- Our model, HFGN, achieves the best performance. Benefiting from the graph structure and message propagation across items, HFGN is capable of modeling complex interaction among items within the same outfit. Compared to NGNN, we enhance item embedding with the compatibility information with its neighbours, the other items within the same outfit. Besides, we estimate the compatibility of outfits by introducing a R-view attention map, which can better capture the potential compatibility knowledge, further enhancing the model performance.

3.3.4 ***Case Study.*** In figure 4, we visualize several test examples on Fill-in-the-Blank task to compare our model with the strong baselines (*i.e.,* NGNN and FHN). From example 1, we can see that all the three models infer that the outfit lacks of a pair of shoes and correctly select the answer. From example 2, we can see that although FHN correctly selects complementary category (*i.e.,* shoes), it misses the correct answer as it just considers 3 coarse-grained category information and is incapable of exploiting the fine-grained category information (*e.g.,* the compatibility distinction between high-heels and sports shoes). From example 3, we can see that both NGNN and HFGN have inferred that the query outfit lack of a bag. Nevertheless, HFGN chooses a more compatible bag than NGNN. That might be attributed to the more expressive propagation rules and compatibility formulation in HFGN. These examples demonstrate the rationality and effectiveness of our model on compatibility matching task.

4 RELATED WORK

In this section, we first review works on graph neural networks, and then we introduce the two related tasks in our work: personalized recommendation and compatibility matching.

4.1 Graph Neural Network

Graph neural networks (GNNs) are a kind of structure to model a set of elements and their relations. Due to its great expressive

power, GNNs have been widely used in many tasks involving rich relations including molecular analysis [7, 9], image retrieval [18, 19] and visual comprehension [3].

Gori *et al.* are the first to propose graph neural networks (GNNs) [10] which are capable of directly processing graphs to retain topological information. Although GNNs can be applied on most types of graphs, *e.g.,* acyclic, cyclic, directed and undirected, the primitive GNNs have difficult to train for a fixed point. Thereafter, GCN [22] was proposed to generalize convolutions to graph domain, which has achieved great success. GCN can perform a convolution on the graph and aggregate the information derived from all the neighbours to update the node embeddings. While distinct from GCN, GraphSAGE [11] updates a node embedding by uniformly sampling and aggregating features from its local neighbours. Although GNNs can update a node embedding by propagating information from neighbours with arbitrary depth, a long-term message propagation might cause some problems. To remedy this, recent advanced works [26] attempt to introduce Gate Recurrent Units (GRU) in the propagation process.

Most of the methods mentioned above have been proposed to solve node-focused problems. However, the edges in the graph may also contain rich information. In order to model relations between nodes, TransE [1] has been proposed to embed a graph into a continuous space, where each entity is represented as a point and each relation is modeled as a translating operation. However, TranE has flaws in dealing with complex relations, *e.g.,* 1-to-N, N-to-1 and N-to-N. To overcome this flaw, Wang *et al.* proposed TransH [42] to model a relation as a translating operation on a hyperplane.

Because of the powerful modeling capabilities for complex relationships, GNNs have been widely used in personalized recommendation and fashion analysis, which will be introduced in the following.

4.2 Personalized Recommendation

Recommender system has been widely deployed to capture user preference in online platforms. There have been a series of works committed to model user behavior effectively [17, 24, 25, 40].

MF [24, 32] is one of the most popular techniques, which maps each user and item as a vector with ID information and models the user-item interaction with inner product.

In order to enhance the model performance, some works attempted to incorporate side information into the prediction model. For instance, VBPR [13] incorporated visual features of products to enhance item representations. NSCR [39] utilized social relations to help model user behaviors. Besides, a recent work [41] introduced knowledge graph to provide complementary information while making recommendations. Apart from the works mentioned above, some effort devoted to model user-item interaction by exploiting deep learning techniques. As inner product only linearly models the interaction behavior between users and items, they are insufficient to explore nonlinear and complicated relationships. To remedy this, He *et al.* proposed NCF [17] to capture nonlinear relationships by leveraging a multi-layer perceptron.

Due to the great expressive power of graphs on modeling complex relations, some graph based recommender systems have been proposed recently. For instance, an advanced model, HOP-Rec [43], enriched the training data by performing random walks on the interaction graph. NGCF [40] stacked multiple propagation layers to aggregate high-order information into node representations. KGAT [38] integrates user-item interaction graph and knowledge graph by linking items with their attributes. In such a hybrid architecture, node embeddings can be recursively refined by message propagation. In addition, one recent research, LightGCN [15], has found that two common designs in GCNs—feature transformation and nonlinear activation— might degrade the recommendation performance and discarding these operations would benefit the model effectiveness.

4.3 Compatibility Matching

Fashion analysis [6, 8, 27, 28] has attracted increasing attention in recent years. In this paper, we focus on one fashion related task, compatibility matching.

Some works focus on estimating pairwise compatibility between a pair of items. McAuley *et al.* proposed to map the items into a style space where compatible items will fall close to each other [31]. Following that, Veit *et al.* fed a pair of item images into an end-to-end siamese network and measured the distance between them. He *et al.* proposed Monomer [14] to model heterogeneous relationships of items beyond mere visual similarity. Except for visual features, recent works highlight the importance of exploiting multi-modality features in fashion related task. For instance, Song *et al.* utilized multiple modalities (*e.g.,* visual and contextual modalities) to learn a latent compatibility space [34] via a dual autoencoder network. Afterwards, they proposed a knowledge distillation scheme [33] to learn from both data samples and domain knowledge so that the general knowledge rules can play a role as a teacher to guide the training process.

Recently, outfit related tasks have aroused interest in fashion community and there has been much effort devoted to model the compatibility of the whole outfit. The key of modeling outfit compatibility is to represent an outfit properly. Han *et al.* represented an outfit as a sequence and exploited bidirectional LSTMs to predict the compatibility score of one outfit [12]. Benefiting from the BiLSTM architecture, the model can also predict a compatible item to fill the outfit with a blank. There are also some works represent an outfit as a graph [6, 44]. In NGNN [6], Cui *et al.* represented an outfit as a graph to model complex relations among items, which has been demonstrated to be more effective than sequence and pairwise representation.

5 CONCLUSION

In this paper, we have proposed a new framework, Hierarchical Fashion Neural Network (HFGN), to solve the task of personalized outfit recommendation, which requires the recommended targets not only have a nice compatibility but also meet user's personal taste. We build a hierarchical graph structure upon user-outfit interactions and outfit-item mappings. The graph consists of three levels (*i.e.,* user, outfit and item level), where each level contains the corresponding type of nodes. Through the message propagation on such hierarchical graph, the representations of nodes on each level can be refined by aggregating the interaction

information derived from their neighbours. By introducing ID embedding for the outfit, we incorporate outfit-level semantic into the outfit representation, which has been overlooked by the previous works [20, 29]. And distinct from these works which separately considers compatibility matching and personalized recommendation, we regard the compatibility information as a passing message in the graph and encode it into the node representation. Extensive experiments have demonstrated the rationality and effectiveness of HFGN.

This work explores the potential of graph neural network in personalized outfit recommendation task. Benefiting from such hierarchical graph and message propagation, we can extend our work by incorporating multiple features (*e.g.,* textual feature) into the graph network to refine the node embeddings and model higher-order relations among them. In future work, we will focus on exploring the relationship between the two tasks (*i.e.,* outfit recommendation and compatibility modeling) and devising a more effective formulation to exploit this knowledge.

ACKNOWLEDGMENTS

This work was supported by the National Key Research and Development Project of China (No. 2018AAA0101900), the National Natural Science Foundation of China (U19B2043, 61976185, U19A2079), Zhejiang Natural Science Foundation (LR19F020002, LZ17F020001), the Fundamental Research Funds for the Central Universities, and Chinese Knowledge Center for Engineering Sciences and Technology. Long Chen was supported by 2018 Zhejiang University Academic Award for Outstanding Doctoral Candidates. This research is also part of NExT++ project, supported by National Research Foundation, Prime Minister's Office, Singapore under its IRC@Singapore Funding Initiative.

REFERENCES

[1] Antoine Bordes, Nicolas Usunier, and Alberto Garcia-Duran. 2013. Translating Embeddings for Modeling Multi-relational Data. In *NIPS*. 2787–2795.
[2] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive Collaborative Filtering: Multimedia Recommendation with Item- and Component-Level Attention. In *SIGIR*. 335–344.
[3] Long Chen, Hanwang Zhang, Jun Xiao, Xiangnan He, Shiliang Pu, and Shih-Fu Chang. 2019. Counterfactual Critic Multi-Agent Training for Scene Graph Generation. In *ICCV*. 4613–4623.
[4] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. 2017. SCA-CNN: Spatial and Channel-Wise Attention in Convolutional Networks for Image Captioning. In *CVPR*. 5659–5667.
[5] 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. In *KDD*. 2662–2670.
[6] 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 *WWW*. 307–317.
[7] Alex Fout, Jonathon Byrd, Basir Shariat, and Asa Ben-Hur. 2017. Protein Interface Prediction using Graph Convolutional Networks. In *NIPS*. 6530–6539.
[8] Yuying Ge, Ruimao Zhang, Xiaogang Wang, Xiaoou Tang, and Ping Luo. 2019. DeepFashion2: A Versatile Benchmark for Detection, Pose Estimation, Segmentation and Re-Identification of Clothing Images. In *CVPR*. 5337–5345.
[9] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural message passing for Quantum chemistry. In *ICML*. 1263–1272.
[10] Marco Gori, Gabriele Monfardini, and Franco Scarselli. 2005. A New Model for Learning in Graph Domains. *IJCNN* 2 (2005), 729–734.
[11] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *NIPS*. 1024–1034.
[12] Xintong Han, Zuxuan Wu, Yu-Gang Jiang, and Larry S. Davis. 2017. Learning Fashion Compatibility with Bidirectional LSTMs. In *MM*. 1078–1086.
[13] Ruining He and Julian McAuley. 2016. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. In *AAAI*.
[14] Ruining He, Chales Packer, and Julian McAuley. 2016. Learning Compatibility Across Categories for Heterogeneous Item Recommendation. In *ICDM*. 937–942.
[15] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *SIGIR*.
[16] Xiangnan He, Zhankui He, Jingkuan Song, Zhenguang Liu, Yu-Gang Jiang, and Tat-Seng Chua. 2018. NAIS: Neural Attentive Item Similarity Model for Recommendation. *TKDE* 30 (2018), 2354–2366.
[17] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW*. 173–182.
[18] Richang Hong, Lei Li, Junjie Cai, Dapeng Tao, Meng Wang, and Qi Tian. 2017. Coherent Semantic-Visual Indexing for Large-Scale Image Retrieval in the Cloud. *IEEE Transactions on Image Processing* 26 (2017), 4128–4138. Issue 9.
[19] Richang Hong, Yang Yang, Meng Wang, and Xian-Sheng Hua. 2015. Learning Visual Semantic Relationships for Efficient Visual Retrieval. *IEEE Transactions on Big Data* 1 (2015), 152–161. Issue 4.
[20] Yang Hu, Xi Yi, and Larry S. Davis. 2015. Collaborative Fashion Recommendation: A Functional Tensor Factorization Approach. In *MM*. 129–138.
[21] Santosh Kabbur, Xia NIng, and George Karypis. 2013. FISM: Factored Item Similarity Models for Top-N Recommender Systems. In *KDD*. 659–667.
[22] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.
[23] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD*. 426–434.
[24] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42 (2009), 30–37. Issue 8.
[25] Wenqiang Lei, Xiangnan He, Yisong Miao, Qingyun Wu, Richang Hong, Min-Yen Kan, and Tat-Seng Chua. 2020. Estimation-Action-Reflection: Towards Deep Interaction Between Conversational and Recommender Systems. In *WSDM*. 304–312.
[26] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. 2016. Gated Graph Sequence Neural Networks. In *ICLR*.
[27] Lizi Liao, Xiangnan He, Bo Zhao, Chong-Wah Ngo, and Tat-Seng Chua. 2018. Interpretable Multimodal Retrieval for Fashion Products. In *MM*.
[28] Kuan-Hsien Liu, Ting-Yen Chen, and Chu-Song Chen. 2016. MVC: A Dataset for View-Invariant Clothing Retrieval and Attribute Prediction. In *MM*.
[29] Zhi Lu, Yang Hu, Yunchao Jiang, Yan Chen, and Bing Zeng. 2019. Learning Binary Code for Personalized Fashion Recommendation. In *CVPR*. 10562–10570.
[30] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. 2013. Rectifier Nonlinearities Improve Neural Network Acoustic Models. *ICML* 30 (2013), 3. Issue 1.
[31] Julian McAuley, Christopher, Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *SIGIR*. 43–52.
[32] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*. 452–461.
[33] Xuemeng Song, Fuli Feng, Xianjing Han, Xin Yang, Wei Liu, and Liqiang Nie. 2018. Neural Compatibility Modeling with Attentive Knowledge Distillation. In *SIGIR*. 5–14.
[34] Xuemeng Song, Fuli Feng, Jinhuan Liu, Zekun Li, Liqiang Nie, and Jun Ma. 2017. NeuroStylist: Neural Compatibility Modeling for Clothing Matching. In *MM*. 753–761.
[35] Ashish Vaswani, Noam Shazee, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*. 6000–6010.
[36] Andreas Veit, Balazs Kovacs, Sean Bell, Julian McAuley, Kavita Bala, and Serge Belongie. 2015. Learning Visual Clothing Style with Heterogeneous Dyadic Co-occurrences. In *ICCV*.
[37] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. 950–958.
[38] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. KGAT: Knowledge Graph Attention Network for Recommendation. 950–958.
[39] Xiang Wang, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2017. Item Silk Road: Recommending Items from Information Domains to Social Users. 185–194.
[40] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *SIGIR*. 165–174.
[41] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. 2019. Explainable Reasoning over Knowledge Graphs for Recommendation. 5329–5336.
[42] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. In *AAAI*. 1112–1119.
[43] Jheng-Hong Yang, Chih-Ming Chen, Chuan-Ju Wang, and Ming-Feng Tsai. 2018. HOP-rec: high-order proximity for implicit recommendation. 140–144.
[44] Xun Yang, Yunshan Ma, Lizi Liao, Meng Wang, and Tat-Seng Chua. 2019. TransNFCM: Translation-Based Neural Fashion Compatibility Modeling. In *AAAI*, Vol. 33. 430–410.