Personalized Item Recommendation for Second-hand Trading Platform

Xuzheng Yu
Shandong University
xuzhengyuuu@gmail.com

Tian Gan*
Shandong University
gantian@sdu.edu.cn

Yinwei Wei
Shandong University
weiyinwei@hotmail.com

Zhiyong Cheng
Shandong Artificial Intelligence Institute, Qilu University of Technology (Shandong Academy of Sciences)
jason.zy.cheng@gmail.com

Liqiang Nie
Shandong University
nieliqiang@gmail.com

ABSTRACT

With rising awareness of environment protection and recycling, second-hand trading platforms have attracted increasing attention in recent years. The interaction data on second-hand trading platforms, consisting of sufficient interactions per user but rare interactions per item, is different from what they are on traditional platforms. Therefore, building successful recommendation systems in the second-hand trading platforms requires balancing modeling items' and users' preference, and mitigating the adverse effects of the sparsity, which makes recommendation especially challenging. Accordingly, we proposed a method to simultaneously learn representations of items and users from coarse-grained and fine-grained features, and a multi-task learning strategy is designed to address the issue of data sparsity. Experiments conducted on a real-world second-hand trading platform dataset demonstrate the effectiveness of our proposed model.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; *Hierarchical data models*; *Personalization*; *Multimedia and multimodal retrieval*.

KEYWORDS

Second-hand Trading Platform; Recommendation; Sparsity

ACM Reference Format:
Xuzheng Yu, Tian Gan, Yinwei Wei, Zhiyong Cheng, and Liqiang Nie. 2020. Personalized Item Recommendation for Second-hand Trading Platform. In *Proceedings of the 28th ACM International Conference on Multimedia (MM '20), October 12–16, 2020, Seattle, WA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3394171.3413640

*Corresponding author.

MM '20, October 12–16, 2020, Seattle, WA, USA

ACM ISBN 978-1-4503-7988-5/20/10…$15.00
https://doi.org/10.1145/3394171.3413640

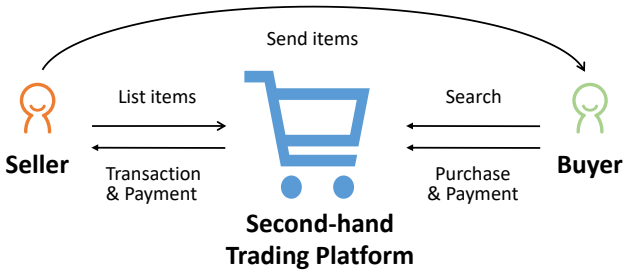

Figure 1: A conceptual diagram of the typical business operation model on the second-hand trading platforms.

1 INTRODUCTION

With rising awareness of environment protection, people are increasingly participating in activities that reduce ecological footprint through recycling. Therefore, the used and second-hand items industries, which involve the transfer of second-hand items to secondary consumers, have attracted increasing attention in recent years [4, 7–10, 20]. For example, a German online shop called Ubup[1], reported that the number of items it had sold has risen by 566% since the beginning of 2014, and this shop had sold 1.5 million used items in 2016. Besides, there are around 99 million users of resale-focused apps as of August 2019, according to the data firm Getui[2]. In addition, China Beijing Environment Exchange[3] estimated that transactions on Alibaba owned re-commerce platform "Idle Fish" has helped to reduce 100,000 tons of carbon emissions between 2014 and 2018.

Compared to traditional trading platforms, second-hand trading platforms are special. Specifically, for products of the same brand, model and style, if they come from different sellers, they will be treated as different items due to reasons like different conditions. Figure 1 is the conceptual figure of the typical business operation model in the second-hand trading platform. Sellers take pictures of items they want to sell and list them on the second-hands trading platform. Meanwhile, buyers will search for items of interest on

[1] www.ubup.com.
[2] www.getui.com.
[3] www.cbeex.com.cn.

the platform. When buyers purchase the desired items, sellers will receive notifications from the platform and send the items to buyers. After buyers receive the items, the platform will pay the sellers to complete the entire transactions. In this case, items in the second-hand trading platform are usually unique, therefore, they bring more challenges in the task of item recommendations: First, the number of the same items is usually only one, and most have been interacted with few users (*e.g.,* in the second-hand trading platform dataset we worked on in this paper, more than ninety-nine percent of items have only been interacted with by less than five people). Therefore, the challenge is how to utilize the sparse interaction data that most of them would be filtered out in the preprocessing stage of traditional methods, which would lead to the removal of most of the data in our settings. Second, different sellers list similar items in the second-hand trading platforms, and these items are usually regarded as the same thing in traditional trading platforms. Without available explicit feedback, we argue that it is useful to apply visual descriptions and hierarchical categories of items, to represent these items which may be the same style but are regarded as different for conditions or other reasons. Hence, it brings another challenge that how to utilize visual information and hierarchical category data reasonably and effectively.

Existing studies investigate the above problems in regular item recommendation scenarios, and the most common way is to filter out low-frequent items or users during preprocessing stage [18, 21, 28, 30]. However, it is not suitable for second-hand platforms because sparse interaction is one of the prominent characteristics of second-hand trading platforms, and therefore most items in second-hand trading platforms will be filtered out under this method. In this sense, the problem we tackled here comes from a typical multimodal scenario; whereas the traditional recommendation methods do not work in this scenario. Meanwhile, several previous studies [17, 21] argued that the category hierarchy information is able to learn more robust visual representations of items. Specifically, these studies focus on the combination of category hierarchy and visual features while they ignore relationships in category hierarchy, which is valuable as well.

In this paper, we propose a novel method to simultaneously learn representations of items and user preferences using visual features and hierarchical categories meanwhile reinforce the relationships among items and users to distinguish items. Additionally, we propose a multi-task learning strategy to address the data sparsity problem.

Our contributions are as follows:

- We propose a novel method that simultaneously learns representations of items and user preferences using visual features and hierarchical categories, which utilizes the hierarchical category information to enhance the relationship among items or users for better latent representation learning.
- We design a multi-task learning strategy of "recommending items to users" and "identifying potential users to items" to further improve the recommendation performance regarding to the problem of data sparsity.
- We quantitatively evaluated our model on a real-world second-hand trading platform dataset. Experimental results demonstrate the effectiveness of our model.
- We additionally applied our proposed multi-task learning strategy to several state-of-the-art methods, and they all achieve great improvement, demonstrating the effectiveness of our proposed multi-task learning strategy.

The rest of this paper is structured as follows. In Section 2, we briefly review the related literature. In Section 3, we detail our proposed model, followed by experimental results and analyses in Section 4. We finally conclude the work in Section 5.

2 RELATED WORK

In this section, we mainly review the studies that are most related to our work, including representation learning for recommendation, and sparsity problems in recommendation systems.

2.1 Representation Learning

Learning representations of items is an important step in recommendation systems [17, 18, 21, 30–32]. He *et al.* [18] proposed a scalable method that incorporates visual features into Matrix Factorization to uncover the "visual dimensions" that can influence people's behavior. He *et al.* [17] also proposed another sparse hierarchical embedding method that simultaneously reveals globally-relevant and subtle visual dimensions efficiently. However, in the second-hand trading platform, the number of interactions related to each item is very small. Under this circumstance, learning latent factors for each item in the above methods will cause overfitting. Meanwhile, a few research work [17, 21] try to utilize additional category information to learn more accurate representations of items. Specifically, Liu *et al.* [21] proposed to obtain style features by subtracting categorical information from visual features. He *et al.* [17] introduced hierarchical category information into visual feature learning. These work inspires our initial idea, though they did not consider the relationships among hierarchical categories as what we do in this paper.

Compared to item representations which are usually refined using various specially designed models, in most previous work [17, 18, 21, 30], user preferences are often learned in a similar way with a trainable embedding matrix. In addition, several researchers further apply attention mechanism to help learn user preferences [12, 13, 34, 35].

Apart from item representations and user preferences, various types of interaction data are also utilized by researchers to train models [11, 13, 28, 35]. Among those attempts to learn from interactions, there is a kind of methods based on graph-based neural networks achieves great success. Wang *et al.* [29] proposed the newly embedding propagation layer to leverage high-order connectivities in the user-item integration graph. He *et al.* [19] argued the unnecessarily complicated design of graph convolutional networks and proposed LightGCN which consists of light graph convolution and layer combination. Wu *et al.* [33] explored the simplest possible formulation of a graph convolutional model. These models learn high-order representations of items and users with interaction data. However, in the second-hand platforms, the average number of occurrences of items in the interaction data is very small, which makes the user-item integration graph too sparse to support operations in GCNs [23].

2.2 Sparsity in Recommendation System

Sparsity problem is one of the major problems encountered by recommendation systems, and the data sparsity has a great impact on the quality of the recommendation [1]. Several attempts have been made to mitigate the negative effects of sparse data [2, 15, 24–26]. Pazzani *et al.* [24] alleviated the problem of sparse user interaction by introducing additional information of users. Rawat *et al.* [25] utilized additional contextual information to address the sparsity problem. And Salakhutdinov *et al.* [26] utilized item based mining retrieval technique to make models perform well on sparse data. Billsus *et al.* [2] applied Singular Value Decomposition to reduce the dimensionality of sparse rating matrices. Guo *et al.* [15] reduced the sparsity of items by pre-training the model with constructed data. In addition, Guo *et al.* [14] presented a Mahalanobis distance and a deep neural network method to effectively model the linear and non-linear correlations between features, which can incorporate side information to overcome the cold-start and data sparsity problems. Despite that there is certain amount of work focusing on solving the sparsity problem, the level of sparsity in their work is different from what we are facing now. They usually first filter out items and users that appear less than five times at preprocessing step, which will lead to the removal of most of the data in our settings.

3 OUR PROPOSED METHOD

3.1 Problem Setting and Model Overview

3.1.1 Problem setting. Before describing our method, we introduce the problem setting first. Formally, let $\boldsymbol{U}$ and $\boldsymbol{I}$ denote the set of users and items respectively. Each user $u \in \boldsymbol{U}$ had interactions (*e.g.,* click, like, or purchase) with a set of items $\boldsymbol{I}_u \subset \boldsymbol{I}$. Each item $i \in \boldsymbol{I}$ is associated with a path $\boldsymbol{C}_i = \{\boldsymbol{C}_i^1, \boldsymbol{C}_i^2, \boldsymbol{C}_i^3, \ldots, \boldsymbol{C}_i^m\}$ on a category hierarchy from the root (*i.e.,* the first level as the highest level) category to a leaf (*i.e.,* the m-th level as the lowest level) category, where m denotes the number of levels in the category hierarchy of all items, and $\boldsymbol{C}_i^k$ is the parent category of $\boldsymbol{C}_i^{k-1}$. In addition, each item i is associated with a visual feature vector $\boldsymbol{f}_i^{\text{vis}} \in \mathbb{R}^{d_{vis}}$ extracted from a pre-trained convolutional neural network, where d_{vis} represents the dimensions of visual features. Our goal is to learn a personalized item recommendation model, which could recommend items to each user $u \in \boldsymbol{U}$ appropriately based on personal preference.

3.1.2 Model overview. Sparsity is one of the prominent characteristics of second-hand platforms. If the recommendation model treat each item as an individual and learn a latent factor for each item, it will severely suffer from the overfitting problem because each item appears rarely in the interaction history. In order to tackle this problem, as shown in Figure 2, we utilized the hierarchical category information to enhance the relationship among items or users for their latent representation learning. We also designed a multi-task learning strategy of "recommending items to users" and "identifying potential users to items" to further improve the recommendation performance.

3.2 Latent Representation Learning

3.2.1 Item latent representation learning. When a user interacts with items on second-hand trading platforms, what s/he actually needs determines the coarse-grained of items (*e.g.,* type, etc.) s/he will interact with. Meanwhile, the fine-grained characteristics (*e.g.,* appearance, condition, etc.) of those items will influence which specific items of in the same type s/he will prefer. Based on this intuition, we propose to learn the latent representations of items by combining the coarse-grained features and the fine-grained features. Specifically, in this paper, we adopt hierarchical category features as the coarse-grained features, and utilize visual comprehensive features that are learned both from visual features and category information as the fine-grained features.

Although the continuous linear transformation and learning an embedding vector for each category have little difference in mathematical principles, we believe it that our proposed method can better learn the characteristics of categories from the whole instead of focusing on each category independently.

To make fully use of the hierarchical relationships among categories, we learn the relationships between adjacent hierarchical categories instead of learning independent features for each category c. Specifically, we learn an embedding vector $\boldsymbol{e}_c^{\text{base}} \in \mathbb{R}^{d_{\text{cate}}}$ for each first-level category $c \in \boldsymbol{C}^1$, where d_{cate} represents the dimension of category feature. We also learn an embedding matrix $\boldsymbol{W}_c^{\text{cat}} \in \mathbb{R}^{d_{\text{cat}} \times d_{\text{cat}}}$ for each rest categories $c \in \boldsymbol{C} \backslash \boldsymbol{C}^1$, which models the relationship between this category c and $\gamma(c)$, where $\gamma(c)$ represents the parent category of the category c. And then we can obtain representations $\boldsymbol{e}_c^{\text{cat}} \in \mathbb{R}^{d_{\text{cat}}}$ for each category $c \in \boldsymbol{C}$ as follows:

$$\boldsymbol{e}_c^{\text{cat}} = \begin{cases} \boldsymbol{e}_c^{\text{base}}, & c \in \boldsymbol{C}^1 \\ \boldsymbol{W}_c^{\text{cat}} \boldsymbol{e}_{\gamma(c)}^{\text{cat}}, & c \notin \boldsymbol{C}^1 \end{cases}, \tag{1}$$

To learn the fine-grained visual comprehensive features, in addition to visual features, we also utilize the lowest-level (the 1^{st}-level category corresponds to the highest-level category) category of items in the hierarchy instead of the whole hierarchical category information which are adopted in previous work [17]. The formulations of visual comprehensive features $\boldsymbol{e}_i^{\text{vis}}$ are defined as follows:

$$\boldsymbol{e}_i^{\text{vis}} = \boldsymbol{W}_{\delta(i)}^{\text{item}} \boldsymbol{f}_i^{\text{vis}} + \boldsymbol{b}_{\delta(i)}^{\text{item}}, \tag{2}$$

where $\delta(i)$ represents the lowest-level category of the item $i \in \boldsymbol{I}$, $\boldsymbol{W}_c^{\text{item}}$ represents the visual-category conjunction matrix for items from category c, $\boldsymbol{f}_i^{\text{vis}}$ represents each item's visual feature vector, and $\boldsymbol{b}_c^{\text{item}}$ represents the visual-category bias vector of items from category c. And the reasons we only apply the lowest-level categories are: 1) the lowest-level category actually implies the information of the upper categories, and 2) the tasks using upper-level categories are actually much more difficult than the tasks using lower-level categories, because using the upper-level categories will lead to the loss of detailed category information.

Finally, we can obtain the latent representations $\boldsymbol{e}_i^{\text{item}}$ of the item $i \in \boldsymbol{I}$ by concatenating the category representations of its hierarchical categories and its visual comprehensive feature. The formulations are defined as follows:

$$\boldsymbol{e}_i^{\text{item}} = \boldsymbol{e}_{\boldsymbol{C}_i^1}^{\text{cat}} \oplus \boldsymbol{e}_{\boldsymbol{C}_i^2}^{\text{cat}} \oplus \ldots \oplus \boldsymbol{e}_{\boldsymbol{C}_i^m}^{\text{cat}} \oplus \boldsymbol{e}_i^{\text{vis}}, \tag{3}$$

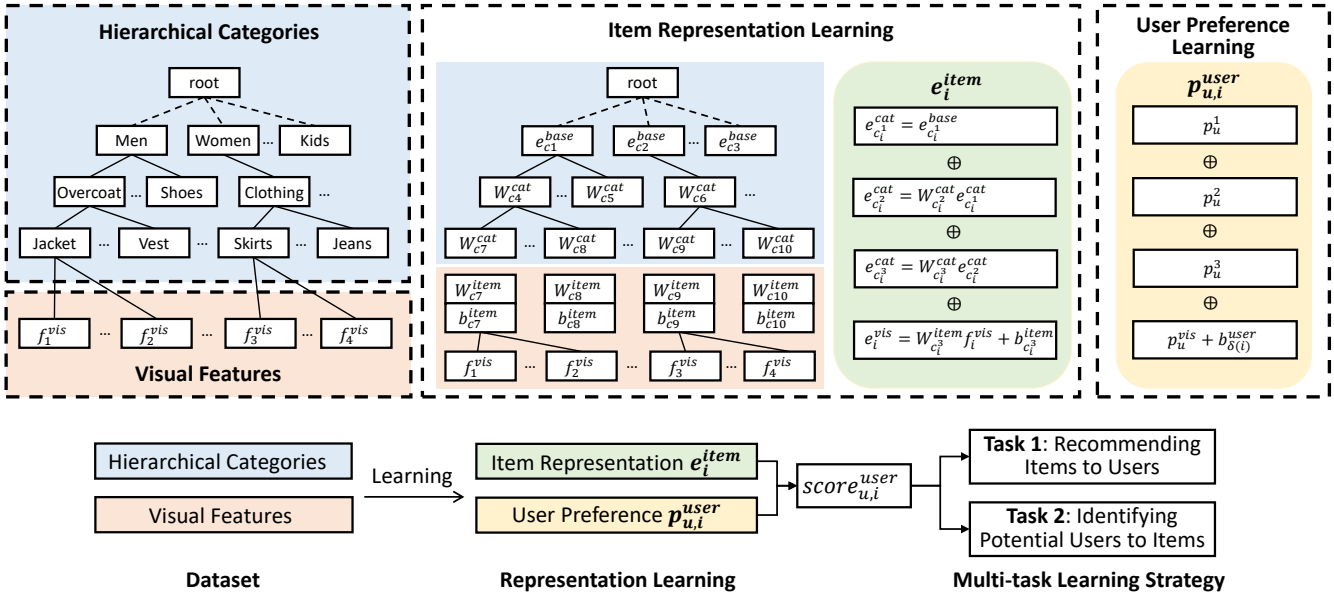

Figure 2: An overview of the proposed framework for personalized item recommendation in secend-hand trading platforms.

where m denotes the number of levels in the category hierarchy of all items, C_i^j represents the j-level hierarchical category of item $i \in \boldsymbol{I}$, and $\oplus$ indicates the operation of concatenating. We adopt this method to fuse the coarse-grained hierarchical category features and the fine-grained visual comprehensive features of items.

3.2.2 User latent representation learning. Similar to the representations of items, for users, we also consider the user preference from the two aspects of what they actually need and which specific items of the same kinds they will prefer. Based on this idea, we learn an embedding vector $\boldsymbol{p}_u^{\mathrm{k}}$ to model each user's preference on each hierarchical category which indicates what the user needs. We also learn an embedding vector $\boldsymbol{p}_u^{\mathrm{vis}}$ to model each user's preference on the fine-grained characteristics of items. In addition, considering that the same person may show different preferences in different categories of items, we apply embedding vector $\boldsymbol{b}_c^{\mathrm{user}}$ to model the bias in users' preferences on specific category c. Then preference on item i for user u can be calculated as follows:

$$\boldsymbol{p}_{u,i}^{\mathrm{user}} = \boldsymbol{p}_u^1 \oplus \boldsymbol{p}_u^2 \oplus \ldots \oplus \boldsymbol{p}_u^{\mathrm{m}} \oplus (\boldsymbol{p}_u^{vis} + \boldsymbol{b}_{\delta(i)}^{user}), \tag{4}$$

where m denotes the number of levels in the category hierarchy of all items, $\boldsymbol{p}_u^1, \boldsymbol{p}_u^2, \ldots, \boldsymbol{p}_u^{\mathrm{m}}$ represent the user's preferences on each hierarchical category, $\boldsymbol{p}_u^{vis}$ represents the user's preference on the fine-grained visual comprehensive features, $\boldsymbol{b}_c^{user}$ represents the bias in users' preferences on specific category c, $\delta(i)$ represents the lowest-level category of the item i, and $\oplus$ indicates the operation of concatenation.

Besides, we also apply additional embedding vectors to model the impacts of users' preference on viewing items, and we denote the impact of the preference of user u as $\boldsymbol{p}_u^{\mathrm{bias}}$. And then we can obtain the personalized item representations of item i for each user $u \in \boldsymbol{U}$ as follows:

$$\widetilde{\boldsymbol{e}}_{i,u}^{\mathrm{item}} = \boldsymbol{e}_i^{\mathrm{item}} + \boldsymbol{p}_u^{\mathrm{bias}}. \tag{5}$$

3.3 Multi-task Learning for Item Recommendation

After the previous steps, we obtain the representations of users and items in the same latent space. we can get user's recommendation score for an item by calculating the inner product between the preference representations of user and item:

$$score(u,i) = \boldsymbol{p}_{u,i}^{\mathrm{user}} \circ \widetilde{\boldsymbol{e}}_{i,u}^{\mathrm{item}}, \tag{6}$$

where $\circ$ represents the inner product operation. Then we consider the problem from the following two perspectives of recommending items to users and identifying potential users to items.

3.3.1 Recommending items to users. In this perspective, we treat our task as a traditional recommended task. In other words, what we need to do is to recommend suitable items to users. We adopt a modified pairwise-based learning method for optimization. In this method, we construct triples $< u,\ i^+,\ i^- >$ for training based on existing data where u and i^+ correspond to the user and one of the interacted items. And for each positive item i^+, we additionally randomly sample several items i^- as negative items which have not been interacted with by user u in the same type of behavior history.

Specifically, for each positive item that has been actually interacted with by a user, we will sample items that have not been interacted with by this user as negative items. The negative items are from different categories while share the same parent category with this positive item, or items in the same lowest-level category of this positive item. In this way, we can balance the differences among different categories and the differences among items in the same categories when training models.

The objective function can be formulated as follows:

$$\mathcal{J}_1 = \sum_{u,i} \ln\left(1 + e^{\text{score(u,i}^-) - \text{score(u,i}^+)}\right) + \lambda_1 \|\theta\|^2, \tag{7}$$

where $score(u, i)$ represents the recommendation score of the user $u \in \boldsymbol{U}$ for the item $i \in \boldsymbol{I}$, θ denotes all the parameters to be estimated in our model, and λ_1 is a hyper-parameter to control the power of regularization.

3.3.2 Identifying potential users to items. In the second-hand platform, most items have only been interacted by very few users. In this case, the recommendation task is difficult, because there is not enough data to learn a suitable representation for all items. However, by analyzing the data, we found out that most items have been only interacted by a very small number of users, but most users have interacted with many items (*e.g.,* in our sampled *Men*'s dataset, which will be introduced in the next section, each item is interacted by only 1.56 people on average, while each user has interacted with 330.69 items on average). In other words, we have enough data for each user to learn his preference's representations, so that the task of identifying potential users to items may be much easier than the original task of recommending items to users. Therefore in this perspective, we treat the interacted users as labels for items, and classify items with multiple labels (*i.e.,* users). The objective function can be formulated as follows:

$$\mathcal{J}_2 = \sum_{u} \sum_{i \in I_u} -\ln\left(\frac{\exp\left(\text{score(u, i)}\right)}{\sum_j \exp\left(\text{score(u, j)}\right)}\right) + \lambda_2 \|\theta\|^2, \tag{8}$$

where $score(u, i)$ represents the recommendation score of the user u for the item i, θ denotes all the parameters to be estimated in our model, and λ_2 is a hyper-parameter to control the power of regularization.

3.3.3 Multi-task learning for item recommendation. Although the objective functions of the above two tasks are different, their purposes are both to make actually interacted user-item pairs obtain higher recommendation scores than the uninteracted ones. Therefore, we apply the idea of multi-task learning to learn the two tasks at the same time so that the representations of users and items can be learnt better and easier than with single task. The final objective function is defined as:

$$\mathcal{J} = \beta_1 \mathcal{J}_1 + \beta_2 \mathcal{J}_2, \tag{9}$$

where β_1 and β_2 are hyper-parameters as coefficients of $\mathcal{J}_1$ and $\mathcal{J}_2$.

3.4 Personalized Item Recommendation

Given users as queries, the model will first calculate general latent representations of users and items, and then calculate the recommendation scores among the queried users and items, finally take out the items with the Top-K scores and recommend them to users.

4 EXPERIMENTS

In this section, we conduct experiments on real-world datasets to evaluate the performance of our proposed methods.

4.1 Datasets

We evaluate our model on one of the largest available second-hand trading platform datasets which called the Mercari dataset. The Mercari dataset contains metadata from Mercari Inc.[4], including 509,838 users, 55,615,152 items, and user behaviors spanning from November 2016 to October 2018. Specifically, it includes user information (ID and status), item metadata (item ID, seller ID, price, brand, category, condition, size, description, and status), item shipping information (methods, from area, duration, and payer), and user behaviors (liked, listed, purchased, and clicked). Meanwhile, there is a three-level category tree associated with Mercari dataset. Figure 2 shows part of the category hierarchy. It has 13 1^{st}-level categories, 149 2^{nd}-level categories, and 1,233 3^{rd}-level categories. There is also a visual feature vector $F_i^{vis} \in \mathbb{R}^{2048}$ extracted from ResNet [16] associated with each item in this dataset.

Table 1: Statistics of the Mercari datasets.

Dataset	#(Users)	#(Items)	#(Interactions)
Men	1,000	212,487	330,690
Women	1,000	267,597	532,246
Kids	1,000	56,628	117,554
Mix	400	200,731	355,264
Artworks	1,000	82,320	133,020

We consider the data of the categories of *Men*, *Women* and *Kids* from the Mercari dataset. The items of these categories mainly consist of clothes and shoes, which are the same with the dataset in previous work [17]. Apart from studying these three categories separately, we also combine them together and denote it as "*Mix*" to explore whether the use of richer hierarchical information will affect the overall recommendation performances. Meanwhile, we consider the category of *Artwork*. Compared with the aforementioned categories, *Artwork* has its own characterstics: the scope of is wider, the content is more abstract, and there are more variances among items. We would like to evaluate whether and how our model can capture user's preference on these items.

To make experiments more feasible, we sample a subset of the dataset while keeping the original data distribution roughly unchanged. Specifically, we first randomly sample 1,000 users for the categories of *Men*, *Women*, *Kids* and *Artwork*. We also randomly sample 400 users for the sub-dataset of *Mix*. And then we select the items that have been interacted by the selected users, and all the interaction data related to the selected users and items. The statistics of the five sampled sub-datasets are shown in Table 1.

4.2 Evaluation Protocol and Parameter Settings

We randomly split each sampled sub-dataset into training, validation, and testing sets with 8:1:1 ratio as the same in previous work [3, 5, 6]. We evaluate the performance of different models using Accuracy [27, 30], NDCG, AUC, and F1 score as the evaluation metrics defined as follows:

$$Accuracy@K = \sum_{u \in U} \frac{\text{Ind}\left[(\text{H}_{u,K} \cap \text{G}_u) \neq \emptyset\right]}{|U|}, \tag{10}$$

$$NDCG@K = \sum_{u \in U} \{ \sum_{t}^{K} \frac{\text{Ind}\left[H_{u,K,t} \in G_u\right]}{|U| \log_2(1+t)} / \sum_{t}^{K} \frac{\text{Ind}\left[t \leq |G_u|\right]}{|U| \log_2(1+t)} \} \tag{11}$$

[4] www.mercari.com

Table 2: Performance comparison among our model and all fully-trained baselines using Accuracy.

Model	*Men*			*Women*			*Kids*			*Mix*			*Artworks*		
	@5	@10	@20	@5	@10	@20	@5	@10	@20	@5	@10	@20	@5	@10	@20
VBPR	0.000	0.000	0.000	0.003	0.003	0.003	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
VBPR-I	0.026	0.049	0.090	0.028	0.041	0.057	0.011	0.027	0.049	0.042	0.063	0.100	0.065	0.085	0.109
VBPR-II	0.064	0.095	0.141	0.028	0.062	0.096	0.059	0.092	0.141	0.084	0.109	0.146	0.093	0.153	0.190
Sherlock	0.013	0.023	0.028	0.010	0.023	0.047	0.011	0.027	0.049	0.054	0.100	0.126	0.004	0.008	0.008
Sherlock-I	0.097	0.146	0.169	0.083	0.119	0.158	0.108	0.168	0.222	0.176	0.226	0.276	0.097	0.153	0.206
Sherlock-II	0.179	0.235	0.302	0.217	0.264	0.328	0.211	0.243	0.292	0.272	0.356	0.385	0.198	0.234	0.290
DeepStyle	0.003	0.008	0.018	0.021	0.047	0.057	0.043	0.054	0.059	0.021	0.033	0.075	0.000	0.000	0.000
Deepstyle-I	0.069	0.097	0.133	0.067	0.080	0.106	0.092	0.151	0.200	0.126	0.172	0.230	0.101	0.141	0.161
Deepstyle-II	0.110	0.161	0.248	0.114	0.150	0.222	0.195	0.232	0.276	0.184	0.264	0.351	0.149	0.198	0.258
OurModel-S	0.118	0.153	0.207	0.098	0.140	0.181	0.146	0.189	0.227	0.180	0.251	0.310	0.125	0.165	0.206
OurModel	**0.210**	**0.266**	**0.309**	**0.243**	**0.295**	**0.336**	**0.222**	**0.281**	**0.335**	**0.293**	**0.360**	**0.393**	**0.210**	**0.262**	**0.339**
%Improv.(vs. Variant-Is)	**116.5%**	**82.2%**	**82.8%**	**192.8%**	**147.9%**	**112.7%**	**105.6%**	**67.3%**	**50.9%**	**66.5%**	**59.3%**	**42.4%**	**107.9%**	**71.2%**	**64.6%**
%Improv.(vs. all models)	**17.3%**	**13.2%**	**2.3%**	**12.0%**	**11.7%**	**2.4%**	**5.2%**	**15.6%**	**14.7%**	**7.7%**	**1.1%**	**2.1%**	**6.1%**	**12.0%**	**16.9%**

Table 3: Performance comparison among our model and all fully-trained baselines using AUC.

Model	*Men*			*Women*			*Kids*			*Mix*			*Artworks*		
	@5	@10	@20	@5	@10	@20	@5	@10	@20	@5	@10	@20	@5	@10	@20
VBPR	0.000	0.000	0.000	0.002	0.002	0.002	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
VBPR-I	0.017	0.027	0.047	0.015	0.025	0.034	0.007	0.013	0.026	0.022	0.030	0.052	0.040	0.053	0.064
VBPR-II	0.044	0.062	0.087	0.024	0.033	0.055	0.036	0.058	0.083	0.046	0.065	0.091	0.060	0.088	0.113
Sherlock	0.008	0.010	0.016	0.004	0.009	0.021	0.011	0.014	0.022	0.029	0.048	0.072	0.001	0.004	0.004
Sherlock-I	0.051	0.081	0.110	0.056	0.072	0.096	0.053	0.093	0.135	0.106	0.137	0.163	0.052	0.082	0.115
Sherlock-II	0.108	0.148	0.190	0.136	0.174	0.216	0.115	0.153	0.194	0.158	0.215	0.255	**0.119**	0.155	0.184
Deepstyle	0.000	0.003	0.009	0.016	0.024	0.035	0.026	0.036	0.047	0.013	0.014	0.032	0.000	0.000	0.000
Deepstyle-I	0.043	0.061	0.081	0.041	0.055	0.063	0.057	0.081	0.120	0.066	0.088	0.130	0.068	0.085	0.104
Deepstyle-II	0.064	0.095	0.131	0.069	0.096	0.132	0.108	0.159	0.195	0.096	0.144	0.199	0.100	0.132	0.166
OurModel-S	0.064	0.091	0.126	0.059	0.082	0.106	0.073	0.114	0.143	0.110	0.155	0.192	0.069	0.097	0.122
OurModel	**0.120**	**0.168**	**0.209**	**0.149**	**0.188**	**0.231**	**0.131**	**0.178**	**0.231**	**0.176**	**0.235**	**0.271**	**0.119**	**0.163**	**0.205**
%Improv.(vs. Variant-Is)	**135.3%**	**107.4%**	**90.0%**	**166.1%**	**161.1%**	**140.6%**	**129.8%**	**91.4%**	**71.1%**	**66.0%**	**71.5%**	**66.3%**	**75.0%**	**91.8%**	**78.3%**
%Improv.(vs. all models)	**11.1%**	**13.5%**	**10.0%**	**9.6%**	**8.0%**	**6.9%**	**13.9%**	**11.9%**	**18.5%**	**11.4%**	**9.3%**	**6.3%**	**0.0%**	**5.2%**	**11.4%**

$$AUC@K = \sum_{u \in U} \frac{\sum_{t1=1}^{K-1} \sum_{t2=t1+1}^{K} \mathrm{Ind}\left[\left(H_{u,K,t1} \in G_u\right) \wedge \left(H_{u,K,t2} \notin G_u\right)\right]}{|U| \times \left|H_{u,K} \cap G_u\right| \times \left(K - \left|H_{u,K} \cap G_u\right|\right)/2} \quad (12)$$

$$F1@K = 2/\left[\frac{|U|}{\sum_{u \in U}\left(\left|H_{u,K} \cap G_u\right|/K\right)} + \frac{|U|}{\sum_{u \in U}\left(\left|H_{u,K} \cap G_u\right|/|G_u|\right)}\right] \quad (13)$$

where $\boldsymbol{U}$ represents the set of users, $H_{u,K}$ represents the set of Top-K items that the model predicts (*i.e.,* recommends) to user $\boldsymbol{u}$, G_u represents the set of items that user $\boldsymbol{u}$ actually interacted with, Ind($[\cdot]$) indicates Indicator function, and the numerator in the Equation 10 represents whether someone actually interacts with items that are recommended by the model.

To train our proposed model, we randomly initialize model parameters with a Gaussian distribution and utilize AdamW [22] algorithm for optimization. We further restrict the length of final representation vector of users or items in each model to be the same for fair comparison. We set the hyper-parameters β_1 and β_2 to 0.75 and 0.25, respectively. We have tried different parameter settings, including the batch size of {64, 128, 256}, the latent feature dimension of {32, 64, 128}, the learning rate of {0.1, 0.3, 0.01, 0.003, 0.001}. As the findings are consistent across the dimensions of latent vectors, if not specified, we only report the results based on dimension of 128, which gives relatively good performance.

4.3 Baselines

To evaluate the effectiveness of our model, we compared our proposed method with several state-of-the-art baselines.

- VBPR [18]: This baseline introduces visual information into the recommendation system for the first time, and incorporates visual features into Matrix Factorization to uncover the "visual dimension" that plays a crucial role in influencing users' behavior.
- Sherlock [17]: It is a sparse hierarchical embedding method to uncover the visual dimensions of users' opinions on top of raw visual features. Sherlock utilizes different layers on the category hierarchy to simultaneously learn both general and subtle information hidden in categories and vision. The similarity scores are obtained by calculating the inner product of users' opinion embedding and obtained item embedding for recommendation.
- DeepStyle [21]: This baseline learns style features by subtracting categorical information from visual features, and make recommendations with the obtained style features.
- Variant-I (*i.e.,* VBPR-I, Sherlock-I, DeepStyle-I): These variants remove the modules for learning each item's latent factors from corresponding original models. To be specific, the item representations learnt in all baseline models (VBPR,

Table 4: Performance comparison among our model and all fully-trained baselines using NDCG.

Model	*Men*			*Women*			*Kids*			*Mix*			*Artworks*		
	@5	@10	@20	@5	@10	@20	@5	@10	@20	@5	@10	@20	@5	@10	@20
VBPR	0.000	0.000	0.000	0.001	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
VBPR-I	0.009	0.009	0.010	0.008	0.007	0.007	0.003	0.003	0.003	0.012	0.012	0.011	0.018	0.016	0.016
VBPR-II	0.024	0.023	0.023	0.017	0.018	0.018	0.017	0.017	0.018	0.025	0.024	0.022	0.043	0.041	0.039
Sherlock	0.003	0.003	0.003	0.003	0.003	0.003	0.004	0.006	0.007	0.014	0.016	0.016	0.001	0.001	0.001
Sherlock-I	0.030	0.029	0.025	0.028	0.026	0.023	0.029	0.027	0.026	0.060	0.056	0.052	0.035	0.037	0.036
Sherlock-II	0.055	0.051	0.048	**0.086**	**0.074**	**0.067**	0.060	0.055	0.054	0.111	0.104	**0.095**	0.071	0.065	0.063
Deepstyle	0.001	0.001	0.001	0.006	0.006	0.005	0.010	0.009	0.008	0.006	0.006	0.007	0.000	0.000	0.000
Deepstyle-I	0.022	0.019	0.018	0.023	0.019	0.017	0.026	0.025	0.025	0.042	0.043	0.041	0.040	0.039	0.034
Deepstyle-II	0.036	0.034	0.036	0.039	0.038	0.037	0.062	0.056	0.053	0.069	0.069	0.069	0.068	0.061	0.059
OurModel-S	0.033	0.032	0.029	0.029	0.027	0.024	0.039	0.037	0.036	0.067	0.062	0.055	0.042	0.040	0.038
OurModel	**0.061**	**0.056**	**0.050**	0.076	0.069	0.061	**0.069**	**0.063**	**0.061**	**0.119**	**0.105**	**0.095**	**0.078**	**0.070**	**0.070**
%Improv.(vs. Variant-Is)	**103.3%**	**93.1%**	**100.0%**	**171.4%**	**165.4%**	**165.2%**	**137.9%**	**133.3%**	**134.6%**	**98.3%**	**87.5%**	**82.7%**	**95.0%**	**79.5%**	**94.4%**
%Improv.(vs. all models)	**10.9%**	**9.8%**	**4.2%**	**-11.6%**	**-6.8%**	**-9.0%**	**11.3%**	**12.5%**	**13.0%**	**7.2%**	**1.0%**	**0.0%**	**9.9%**	**7.7%**	**11.1%**

Table 5: Performance comparison among our model and all fully-trained baselines using F1 score.

Model	*Men*			*Women*			*Kids*			*Mix*			*Artworks*		
	@5	@10	@20	@5	@10	@20	@5	@10	@20	@5	@10	@20	@5	@10	@20
VBPR	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
VBPR-I	0.005	0.005	0.006	0.004	0.004	0.004	0.000	0.001	0.002	0.001	0.002	0.004	0.006	0.007	0.008
VBPR-II	0.015	0.015	0.014	0.010	0.010	0.010	0.012	0.012	0.012	0.011	0.011	0.012	0.020	0.025	0.028
Sherlock	0.000	0.001	0.002	0.000	0.000	0.001	0.002	0.004	0.004	0.002	0.005	0.008	0.000	0.000	0.001
Sherlock-I	0.012	0.015	0.014	0.008	0.010	0.011	0.009	0.013	0.019	0.010	0.015	0.021	0.013	0.019	0.024
Sherlock-II	0.024	0.028	0.031	**0.038**	**0.041**	**0.040**	0.034	0.034	0.035	0.037	0.044	**0.048**	0.036	0.040	0.043
Deepstyle	0.000	0.001	0.001	0.000	0.001	0.002	0.006	0.006	0.004	0.001	0.002	0.002	0.000	0.000	0.000
Deepstyle-I	0.008	0.009	0.011	0.006	0.006	0.007	0.013	0.016	0.017	0.005	0.011	0.015	0.015	0.021	0.021
Deepstyle-II	0.019	0.021	0.026	0.020	0.023	0.024	**0.040**	0.036	0.033	0.027	0.031	0.041	0.038	0.037	0.039
OurModel-S	0.012	0.018	0.020	0.009	0.012	0.014	0.021	0.023	0.024	0.011	0.016	0.021	0.018	0.022	0.026
OurModel	**0.025**	**0.033**	**0.033**	0.033	0.038	0.039	**0.040**	**0.040**	**0.038**	**0.042**	**0.046**	**0.048**	**0.041**	**0.044**	**0.050**
%Improv.(vs. Variant-Is)	**108.3%**	**120.0%**	**135.7%**	**312.5%**	**280.0%**	**254.5%**	**207.7%**	**150.0%**	**100.0%**	**320.0%**	**206.7%**	**128.6%**	**173.3%**	**109.5%**	**108.3%**
%Improv.(vs. all models)	**4.2%**	**17.9%**	**6.5%**	**-13.2%**	**-7.3%**	**-2.5%**	**0.0%**	**11.1%**	**8.6%**	**13.5%**	**4.5%**	**0.0%**	**7.9%**	**10.0%**	**16.3%**

Sherlock and DeepStyle) come from two kinds of information: visually relevant and visually irrelevant ones. Due to the extreme sparsity issue in the second-hand trading platform dataset, learning visually irrelevant information (which is item's latent factors) will usually result in overfitting. Therefore, we remove this information from their corresponding original models for Variant-I.

- Variant-II (*i.e.,* VBPR-II, Sherlock-II, DeepStyle-II): These variants additionally apply proposed multi-task learning strategy on the basis of their corresponding models with Variant-I.
- OurModel-S: This variant removes proposed multi-task learning strategy from the origin proposed model.

4.4 Performances, Quantitative Analysis and Ablation Study

For each dataset, we evaluate all fully-trained models using the metric Accuracy@K, AUC@K, NDCG@K and F1@K where K={5, 10, 15}. We report the results of different methods using different metrics in Table 2, Table 3, Table 4, and Table 5. Since the main task of our work is to recommend items to users, we only report results of this task. We have the following observations with respect to our experimental results.

First, our proposed method achieves the best performance across all the five sub-datasets using Accuracy and AUC, and also achieves the best performance on four sub-datasets except *Women* using NDCG and F1 score, demonstrating the effectiveness of our model. In other words, OurModel does not achieve better performance than Sherlock-II on sub-dataset *Women*. The possible reasons we analyzed are that Sherlock-II pays more attention to the influence of visual features than OurModel, and the coarse-grained features extracted in OurModel are only related to categories, which strengthens the influence of category information, while visual influences may be more dominant than category information for the items in the category *Women*, resulting in the limited performance of OurModel.

Second, all the original models (*i.e.,* VBPR, Sherlock, DeepStyle) perform very poorly on all the five second-hand platform sub-datasets, showing a very obvious phenomenon of overfitting. Besides, after removing the modules for learning each item's latent factors from their corresponding original models, Variant-Is outperform the original models (*e.g.,* Sherlock obtains 0.023 on *Men*'s sub-dataset using Accuracy@10, while Sherlock-I obtains 0.146 on the same dataset), demonstrating the effectiveness of the strategy that we should not learn latent factors for each item separately to avoid overfitting when data is very sparse.

Third, compared with the Variant-Is, OurModel-S (*i.e.,* our proposed model without multi-task learning strategy) still performs well, verifying the effectiveness of our modeling items and users' preferences using visual features and hierarchical categories.

Moreover, compared with the Variant-Is, we notice that the Variant-IIs using multi-task learning strategy all achieve great improvement across all the five sub-datasets (*e.g.,* DeepStyle-I obtains 0.092 on *Kids*'s sub-dataset using Accuracy@5, while DeepStyle-II obtains 0.195 on the same dataset), demonstrating the strategy that identifying potential users to items is effective and make fully use of the special sparsity.

Finally, we find that all models achieve obviously improvements on *Mix*'s sub-dataset (*e.g.,* OurModel obtains 0.243 on *Women*'s sub-dataset using Accuracy@5, while it obtains 0.293 on *Mix*'s sub-dataset). The reason for this phenomenon is that the 1^{st}-level categories of items in this dataset make an effective and positive effect on models, while the 1^{st}-level categories of items in other datasets are all the same (*i.e.,* do not have impacts on modeling).

5 CONCLUSION

With rising awareness of environment protection and recycling, second-hand trading platforms have attracted increasing attention in recent years. Considering the unique features of second-hand platforms, the key to construct a successful recommendation system is to obtain comprehensive representations of item and user preference, and tackle the data sparsity problem. In this paper, we proposed a method to simultaneously learn representations of items and users using visual features and hierarchical categories, and design a multi-task learning strategy for the data sparsity problem. We evaluated our model for personalized item recommendation tasks on a real-world second-hand trading platform dataset. The experiment results show that our proposed model outperforms the state-of-the-art baselines. In addition, we also conducted ablation studies, demonstrating the effectiveness of our proposed representation learning components and the multi-task learning strategy, respectively.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China, No.:61702302, and No.:U1936203; the Shandong Provincial Natural Science Foundation, No.:ZR2019JQ23; the Innovation Teams in Colleges and Universities in Jinan, No.:2018GXRC014. We appreciate the generous support by Mercari Japan Inc.

REFERENCES

[1] G Adomavicius and A Tuzhilin. 2005. Toward the Next Generation of Recommender Systems: A survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering* 17, 6 (2005), 734–749.

[2] Daniel Billsus and Michael J. Pazzani. 1998. Learning Collaborative Information Filters. In *Proceedings of International Conference on Machine Learning, ICML*. 46–54.

[3] Zhiyong Cheng, Xiaojun Chang, Lei Zhu, Rose Catherine Kanjirathinkal, and Mohan S. Kankanhalli. 2019. MMALFM: Explainable Recommendation by Leveraging Reviews and Images. *ACM Transactions on Information Systems* 37, 2 (2019), 16:1–16:28.

[4] Zhiyong Cheng, Ying Ding, Xiangnan He, Lei Zhu, Xuemeng Song, and Mohan S Kankanhalli. 2018. A^3NCF: An Adaptive Aspect Attention Model for Rating Prediction. In *Proceedings of International Joint Conference on Artificial Intelligence, IJCAI*.

[5] Zhiyong Cheng, Ying Ding, Lei Zhu, and Mohan S. Kankanhalli. 2018. Aspect-Aware Latent Factor Model: Rating Prediction with Ratings and Reviews. In *Proceedings of World Wide Web Conference, WWW*. ACM, 639–648.

[6] Zhiyong Cheng, Jialie Shen, and Steven C. H. Hoi. 2016. On Effective Personalized Music Retrieval by Exploring Online User Behaviors. In *Proceedings of International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 125–134.

[7] Tian Gan, Junnan Li, Yongkang Wong, and Mohan S. Kankanhalli. 2019. A Multi-sensor Framework for Personal Presentation Analytics. *Transaction on Multimedia Computing, Communications, and Applications* 15, 2 (2019), 30:1–30:21.

[8] Tian Gan, Shaokun Wang, Meng Liu, Xuemeng Song, Yiyang Yao, and Liqiang Nie. 2019. Seeking Micro-Influencers for Brand Promotion. In *Proceedings of the ACM International Conference on Multimedia*. 1933–1941.

[9] Tian Gan, Yongkang Wong, Bappaditya Mandal, Vijay Chandrasekhar, and Mohan S. Kankanhalli. 2015. Multi-sensor Self-Quantification of Presentations. In *ACMMM*. 601–610.

[10] Tian Gan, Yongkang Wong, Daqing Zhang, and Mohan S. Kankanhalli. 2013. Temporal encoded F-formation system for social interaction detection. In *Proceedings of ACM International Conference on Multimedia*. 937–946.

[11] C. Gao, X. He, D. Gan, X. Chen, F. Feng, Y. Li, T. Chua, L. Yao, Y. Song, and D. Jin. 2019. Learning to Recommend with Multiple Cascading Behaviors. *IEEE Transactions on Knowledge and Data Engineering* (2019), 1–1.

[12] Yuyun Gong and Qi Zhang. 2016. Hashtag Recommendation Using Attention-Based Convolutional Neural Network. In *Proceedings of International Joint Conference on Artificial Intelligence, IJCAI*, Subbarao Kambhampati (Ed.). IJCAI/AAAI Press, 2782–2788.

[13] Yulong Gu, Zhuoye Ding, Shuaiqiang Wang, and Dawei Yin. 2020. Hierarchical User Profiling for E-commerce Recommender Systems. In *Proceedings of ACM International Conference on Web Search and Data Mining, WSDM*. 223–231.

[14] Yangyang Guo, Zhiyong Cheng, Jiazheng Jing, Yanpeng Lin, Liqiang Nie, and Meng Wang. 2020. Enhancing Factorization Machines with Generalized Metric Learning. *arXiv preprint arXiv:2006.11600* (2020).

[15] Yangyang Guo, Zhiyong Cheng, Liqiang Nie, Xin-Shun Xu, and Mohan S. Kankanhalli. 2018. Multi-modal Preference Modeling for Product Search. In *Proceedings of ACM International Conference on Multimedia*. 1865–1873.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. IEEE Computer Society, 770–778.

[17] Ruining He, Chunbin Lin, Jianguo Wang, and Julian J. McAuley. 2016. Sherlock: Sparse Hierarchical Embeddings for Visually-Aware One-Class Collaborative Filtering. In *Proceedings of International Joint Conference on Artificial Intelligence, IJCAI*. 3740–3746.

[18] Ruining He and Julian McAuley. 2016. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of AAAI Conference on Artificial Intelligence*. 144–150.

[19] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 639–648.

[20] Fan Liu, Zhiyong Cheng, Changchang Sun, Yinglong Wang, Liqiang Nie, and Mohan Kankanhalli. 2019. User Diverse Preference Modeling by Multimodal Attentive Metric Learning. In *Proceedings of ACM International Conference on Multimedia*. 1526–1534.

[21] Qiang Liu, Shu Wu, and Liang Wang. 2017. DeepStyle: Learning User Preferences for Visual Recommendation. In *Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 841–844.

[22] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations, ICLR*.

[23] Andreas Loukas. 2020. What graph neural networks cannot learn: depth vs width. In *International Conference on Learning Representations, ICLR*.

[24] Michael J. Pazzani. 1999. A Framework for Collaborative, Content-Based and Demographic Filtering. *Artificial Intelligence Review* 13, 5-6 (1999), 393–408.

[25] Yogesh Singh Rawat and Mohan S. Kankanhalli. 2016. ConTagNet: Exploiting User Context for Image Tag Recommendation. In *Proceedings of ACM International Conference on Multimedia*. ACM, 1102–1106.

[26] R. Salakhutdinov and A. Mnih. 2008. Probabilistic matrix factorization. *Advances in Neural Information Processing Systems* (2008), 1257–1264.

[27] Andreas Veit, Maximilian Nickel, Serge J. Belongie, and Laurens van der Maaten. 2018. Separating Self-Expression and Visual Content in Hashtag Supervision. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*. IEEE Computer Society, 5919–5927.

[28] Mengting Wan and Julian J. McAuley. 2018. Item recommendation on monotonic behavior chains. In *Proceedings of ACM Conference on Recommender Systems, RecSys*. 86–94.

[29] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 165–174.

[30] Yinwei Wei, Zhiyong Cheng, Xuzheng Yu, Zhou Zhao, Lei Zhu, and Liqiang Nie. 2019. Personalized Hashtag Recommendation for Micro-videos. In *Proceedings of ACM International Conference on Multimedia*. ACM, 1446–1454.

[31] Yinwei Wei, Xiang Wang, Weili Guan, Liqiang Nie, Zhouchen Lin, and Baoquan Chen. 2019. Neural multimodal cooperative learning toward micro-video understanding. *IEEE Transactions on Image Processing* 29 (2019), 1–14.

[32] Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, Richang Hong, and Tat-Seng Chua. 2019. MMGCN: Multi-modal graph convolution network for personalized recommendation of micro-video. In *Proceedings of ACM International Conference on Multimedia*. 1437–1445.

[33] Felix Wu, Amauri H. Souza Jr., Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. 2019. Simplifying Graph Convolutional Networks. In *Proceedings of International Conference on Machine Learning, ICML*, Vol. 97. 6861–6871.

[34] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. 2018. ATRank: An Attention-Based User Behavior Modeling Framework for Recommendation. In *Proceedings of AAAI Conference on Artificial Intelligence, Innovative Applications of Artificial Intelligence, and AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 4564–4571.

[35] Meizi Zhou, Zhuoye Ding, Jiliang Tang, and Dawei Yin. 2018. Micro Behaviors: A New Perspective in E-commerce Recommender Systems. In *Proceedings of ACM International Conference on Web Search and Data Mining, WSDM*. 727–735.