Modeling Product's Visual and Functional Characteristics for Recommender Systems

Bin Wu, Xiangnan He, Yun Chen, Liqiang Nie, Kai Zheng, and Yangdong Ye

Abstract—An effective recommender system can significantly help customers to find desired products and assist business owners to earn more income. Nevertheless, the decision-making process of users is highly complex, not only dependent on the personality and preference of a user, but also complicated by the characteristics of a specific product. For example, for products of different domains (*e.g.,* clothing vs. office products), the product aspects that affect a user's decision are very different. As such, traditional collaborative filtering methods that model only user-item interaction data would deliver unsatisfactory recommendation results.

In this work, we focus on fine-grained modeling of product characteristics to improve recommendation quality. Specifically, we first divide a product's characteristics into visual and functional aspects—*i.e.,* the *visual appearance* and *functionality* of the product. One insight is that, the visual characteristic is very important for products of visually-aware domain (*e.g.,* clothing), while the functional characteristic plays a more crucial role for visually non-aware domain (*e.g.,* office products). We then contribute a novel probabilistic model, named *Visual and Functional Probabilistic Matrix Factorization* (VFPMF), to unify the two factors to estimate user preferences on products. Nevertheless, such an expressive model poses efficiency challenge in parameter learning from implicit feedback. To address the technical challenge, we devise a computationally efficient learning algorithm based on alternating least squares. Furthermore, we provide an online updating procedure of the algorithm, shedding some light on how to adapt our method to real-world recommendation scenario where data continuously streams in. Extensive experiments on four real-word datasets demonstrate the effectiveness of our method with both offline and online protocols.

Index Terms—Matrix factorization, Implicit feedback, Product recommendation, Online learning.

1 Introduction

Recent years have witnessed continued efforts in improving the effectiveness and efficiency of recommender systems [1], [2], [3], [4]. Such systems have become an indispensable component in helping users to find their desired items and assisting business owners to earn more income, ranging from movies [5], news [6], to point-of-interests (POIs) [7]. Distinct from these applications, user consumption behaviors in E-commerce are even more complex, which are usually influenced by many internal and external factors [8]. This makes it difficult to accurately model a user's true preference. Consider a real-life online shopping scenario where a user is holding a basket of products. A high-level question naturally arises: what affects the user's decision making? Namely, which aspects of products drive the user to purchase the products?

Essentially, a product's intrinsic (*i.e.,* functionality) and extrinsic (*i.e.,* visual appearance) characteristics have a profound yet different effect on consumer consumption behaviors. In particular, in visually non-aware product domain (*e.g.,* office products), users are likely to pay more attention on product's functional aspects [9]. While in visually-aware product domain (*e.g.,* fashion products), the purchasing philosophy is totally different — one would not purchase a product without seeing it, no matter what ratings or reviews the product had. As an example, Figure 1 presents the historical shopping baskets of two users in different product domains in Amazon [1]. In fashion product domain, Helen has purchased products, including $dressskirt, handbag, necklace, highheel$. In this case, the probability of a specific product being bought by her largely depends on whether she is satisfied with its visual appearance (*e.g.,* a red hat is compatible with products in her shopping basket). In contrast, in office product domain, Helen has bought $\{pen, notebook, ink, pencilcase\}$. At the next time, she may purchase products (*e.g.,* $ruler$) that are functionally complementary with other products in her shopping basket, rather than substitutable products (*e.g.,* two pens of different brands).

With the above considerations, we believe that when developing a recommender system for E-commerce, the extrinsic and intrinsic characteristics of products should be accounted for. Furthermore, it motivate us to investigate the varying importance of product aspects for products of different domains. Although several recent works [10], [11], [12] have successfully taken the visual appearance into account for visually-aware product recommendation, these works only consider extrinsic characteristic as side information and forgo modeling the functional factors. As such, they may result in suboptimal recommendation accuracy,

- *Corresponding author: Yangdong Ye*
- *Bin Wu, Yun Chen, and Yangdong Ye are with the School of Information Engineering, Zhengzhou University, Henan 450001, China. E-Mail: wubin@gs.zzu.edu.cn, ieyunchen@gs.zzu.edu.cn, and yeyd@zzu.edu.cn.*
- *Xiangnan He is with the School of Computing, National University of Singapore, Singapore, 117417. E-Mail: xiangnanhe@gmail.com.*
- *Liqiang Nie is with the School of Computer Science and Technology, Shandong University, Shandong 266237, China. E-Mail: nieliqiang@gmail.com.*
- *Kai Zheng is with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Sichuan 611731, China. E-Mail: zhengkai@uestc.edu.cn.*

1. https://www.amazon.com/

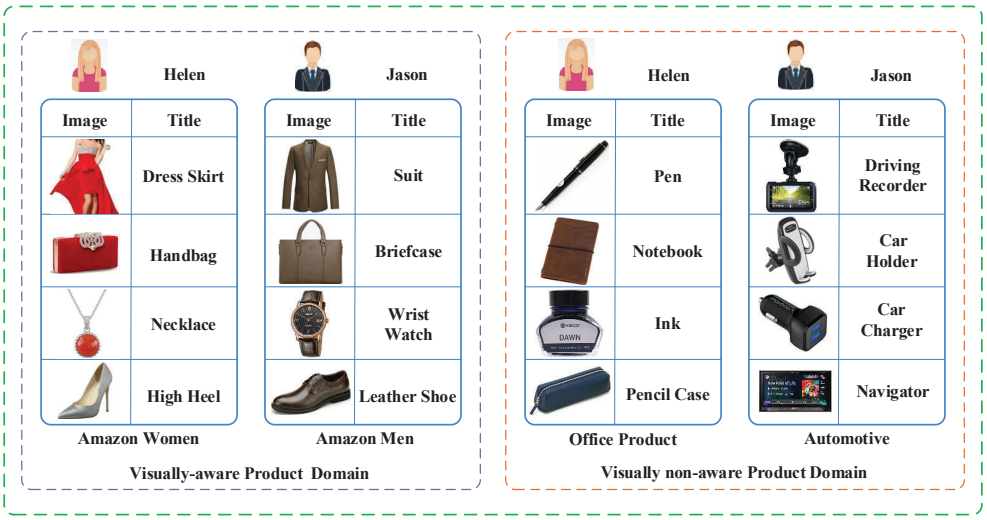


Fig. 1: Illustration of user preference on products of different domains. Best viewed in color.

especially for products of visually non-aware domain.

Moreover, from the perspective of practical recommender systems, it is important for a recommendation algorithm to be scalable in training, due to the sheer number of user-item interactions and their fast generation speed. Currently, most methods utilize Alternating Least Squares (ALS) [13], [14] or Stochastic Gradient Descent (SGD) [15], [16] to optimize a recommender model. However, the standard ALS [13], [14] updates a set of parameters as a whole, making it prohibitive to do fast training on large-scale data [3] (since the time complexity is cubic *w.r.t.* the number of latent factors). While SGD is a more generic solver for a wide range of models and is a suitable choice for online learning, it requires more iterations to converge [17], and more importantly, its accuracy is highly sensitive to the choice of learning rate and negative examples [18] for top-N recommendation.

Driven by the motivation that a user's basket will be affected by varying aspects for products of different domains, we propose a new recommendation solution to integrate the visual and functional characteristics of products to improve the recommendation accuracy. However, a downside is that such an expressive model poses efficiency challenges in learning model parameters with traditional optimization schemes. To make our method suitable for practical use, we develop a fast optimization algorithm based on the idea of ALS but using it on each element level (also referred to as *element-wise ALS* [3]); furthermore, we prove its convergence in theory and extend it for online learning setting. Extensive results demonstrate both the effectiveness and efficiency of our method for product recommendation. We summarize the key contributions of this paper as follows:

- We propose a *Visual and Functional Probabilistic Matrix Factorization* (VFPMF) model to capture product characteristics in a fine-grained manner to model user preference.
- We develop a *Fast Alternating Least Square* (FALS) algorithm to efficiently learn VFPMF parameters for product recommendation with implicit feedback.
- We provide a rigorous proof on the convergence of FALS and design an online updating strategy to adapt it to real-time streaming data.
- We conduct empirical studies on four real-world datasets, showing that our method significantly outperforms several state-of-the-art competitors.

The remainder of this paper is organized as follows. We first review related work in Section 2. We then formulate the problem solved by VFPMF in Section 3, and afterwards elaborate our method in Section 4. We conduct experiments in Section 5, before concluding the paper in Section 6.

2 Related Work

To begin with, we discuss some related works on Visually non-aware recommender systems, which are closely related with Collaborative Filtering (CF). Next, we discuss the literature on visually-aware recommendation that exploits visual content. Finally, we give a brief review on online learning techniques for recommendation.

Visually non-aware Recommender Systems. Generally speaking, for the task of item ranking, CF-based recommendation can be categorized into two branches: pointwise [3], [13], [19] and pairwise methods [18], [20], [21]. The former methods aim to approximate the absolute score of interactions by minimizing a pointwise loss, such as L_2 loss and log loss. Specifically, Hu *et al.* [13] introduced a Weighted Regularized Matrix Factorization (WRMF) method, which optimized L_2 loss with varying weights on observed feedback and missing data to alleviate the class imbalance problem. While a uniform weight was applied to all missing data in WRMF, He *et al.* [3] proposed a frequency-aware varied-weighting strategy on missing data, leading to improved modeling fidelity and recommendation performance. Recently, He *et al.* [19] presented a Neural Collaborative Filtering (NCF) framework which optimized log loss; the best performed model Neural Matrix Factorization (NeuMF) unified the strengths of linearity of MF and non-linearity of multi-layer perception in learning user preference. Distinct from point-wise methods, pairwise methods aim to obtain relative ranking by maximizing the likelihood of pairwise preference over positive feedback and negative samples. For instance, Rendle *et al.* [20] proposed a Bayesian Personalized Ranking (BPR) framework to optimize the relative ranking between positive purchase activities and unobserved ones.

Recently, Ding *et al.* [18] improved the negative sampler in BPR for purchase-oriented product recommendation by leveraging view data. The work that is relevant to our work is [22], which learns user's intrinsic (*i.e.,* personal taste) and extrinsic (*i.e.,* geographical distance) interests for POI recommendation. From the perspective of probabilistic modeling, a typical method of this type is Matrix Co-Factorization for social recommendation, which factorize the user-item rating matrix and the social relation matrix by sharing a common latent user feature matrix [23], [24], [25], [26]. However, due to the lack of geographical information and social relations in our scenario, these methods can not be applied for product recommendation.

Visually-aware Recommender Systems. Just as the famous saying goes, "A picture is worth a thousand words", product images contain rich information and have the potential to facilitate top-N recommendation. In recent years, the importance of visual content has been investigated in recommendation [27], [28], [29], [30]. For instance, He *et al.* [10] recently proposed Visual Bayesian Personalized Ranking (VBPR), which extended the BPR method [20] by incorporating visual content into the predictive model. Liu *et al.* [31] proposed DeepStyle to learn style features of items and sensing preference of users. More recently, Yu *et al.* [32] integrated deep aesthetic features into the MF model for clothing recommendation. In another scenario of POI recommendation, Wang *et al.* [33] exploited visual signal of POIs and demonstrated improved performance.

Online Learning. To provide real-time personalization, an important aspect for commercial recommender systems is how to efficiently refresh model parameters when new data comes in. Online updating strategies have been extensively studied for both tasks of rating prediction and top-N recommendation in literature [3], [34], [35], [36]. In an early work [34], an online updating algorithm is developed for regularized kernel MF that allows to solve the new-user/new-item problem. In [35], an incremental learning algorithm is devised for kernel-based attribute-aware MF model, which addresses the item cold-start problem by integrating item attribute information. In [36], an online learning framework is employed for probabilistic MF and top-one probability based ranking MF, which scales linearly with the number of observed ratings. In terms of online learning, the work that is most relevant with ours is [3], which develops an incremental learning strategy for the element-wise ALS algorithm. However, their algorithm can only be applied to the simple MF model, being non-trivial to extend to other models. Due to the dedicated modeling of both extrinsic and intrinsic characteristics of products, our VFPMF method is more expressive and complicated than MF. We adopt the similar strategy of element-wise ALS to achieve fast optimization for VFPMF and devise online learning strategies correspondingly. Moreover, we theoretically prove the convergence of the algorithm, which is not studied in the previous work [3].

3 Problem Formulation

Notations. In this paper, all column vectors are represented by bold lowercase letters (*e.g.,* $\boldsymbol{h}$), all matrices are denoted by bold uppercase letters (*e.g.,* $\boldsymbol{H}$), and a numeric value is represented by lowercase letters (*e.g.,* r_{uj}). The j^{th} row of a matrix $\boldsymbol{H}$ is represented by $\boldsymbol{h}_j$. Sets are denoted as calligraphic letters (*e.g.,* $\mathcal{R}$). Euclidean and Frobenius norms are represented by $\|\cdot\|_2$ and $\|\cdot\|_F$, respectively. An estimated value is denoted by having a $\hat{}$ (hat) over it (*e.g.,* $\hat{r}_{uj}$). For the sake of clarity, we list some key notations and their explanations in Table 1.

TABLE 1: Notations

Notation	Explanation
k	the dimension of latent features
c	the dimension of Deep CNN features
$\boldsymbol{R}$	consumption matrix ($m \times n$)
$\boldsymbol{w}_u$	user-specific latent feature vector
$\boldsymbol{h}_j$	product-specific latent feature vector
$\boldsymbol{F}$	product extrinsic characteristic matrix($n \times c$)
$\boldsymbol{E}$	embedding kernel matrix($c \times k$)
$\boldsymbol{S}$	product intrinsic characteristic matrix ($n \times n$)
$\boldsymbol{f}_j$	extrinsic characteristics of product j
$\mathcal{R}$	the set of observed consumptions
$\mathcal{R}_u$	the set of products that have been interacted by user u
$\mathcal{R}_j$	the set of users that have purchased product j
$\mathcal{S}$	the set of non-zero entries of the matrix $\boldsymbol{S}$
$\mathcal{F}$	the set of non-zero entries of the matrix $\boldsymbol{F}$

In E-commerce sites, user consumption behaviors are influenced by many internal and external factors, such as functional complementarity and visual compatibility (as illustrated in Figure 1). The complexity in human consumption leads to the inability of existing methods for modeling a user's true preferences on products and interpreting her purchase decision-making process. Inspired by the studies in psychology [8], [37] and economics [38], we consider modeling a product from two aspects: 1) visual characteristic, which affects user purchasing decision with the product's visual appearance, and 2) functional characteristic, which affects user purchasing decision with the product's functionality. Formally, we define them as follows:

Definition (Visual Characteristic) *It is an external aspect of property and determined by the product's manufacturer.*

For instance, a user may purchase products that are visually compatible with other products in her current basket. In the example of Figure 1, since Helen already has $\{dressskirt, handbag, necklace, highheel\}$ in her current basket, it is more appropriate to recommend her *red hat* or other visually compatible clothing products. It is known that high-level CNN features are more discriminative and expressive than low-level manually crafted features (*e.g.,* color histogram and SIFT) in describing an image [39], [40]. As such, we choose high-level CNN features to represent extrinsic characteristic. Same as [10], [41], the visual features are extracted by a pre-trained deep CNN on 1.2 million ImageNet images (ILSVRC2010). Specifically, we used the Caffe reference model [42], which has 5 convolution, 3 max pooling, 3 fully-connected and 1 softmax layers, to extract $c = 4,096$ dimensional visual features $\boldsymbol{f}_j \in \mathbb{R}^c$ from the FC7 layer.

Definition (Functional Characteristic) *It is an internal aspect of property and determined by the product's inherent functionality.*

For instance, a user may purchase products that are functionally complementary with other products in her current basket. In the example of Figure 1, since Helen already has $\{pen, notebook, ink, pencilcase\}$ in her current basket, it is more appropriate to recommend her *ruler*, rather

than other products of the same functionality. To capture this, we use the "*frequently-bought-together*" information to encode the intrinsic characteristic. The intuition is that E-commerce products that are frequently bought together by users should be functionally complementarity [9]. Specifically, let $\boldsymbol{S}$ denote the $n \times n$ intrinsic characteristic matrix. For two products j and i, $s_{ji} = 1$ denotes that they are functionally complementary, and $s_{ji} = 0$ otherwise[2].

With the aforementioned notations and definitions, we define the problem to solve as follows:

Problem Definition

Given: *The user-product consumption matrix $\boldsymbol{R}$, visual characteristic matrix $\boldsymbol{F}$, and functional characteristic matrix $\boldsymbol{S}$.*

Goal: *Recommend user u with top-N products that the user has never purchased before but may purchase in future.*

4 Proposed Method

We first describe how to model the visual and functional characteristics of products respectively. We then demonstrate how two aspects are seamlessly combined using the graphical model and devise a computationally efficient algorithm to solve the unified model. Finally, we provide theoretical analysis for the learning algorithm and design an online updating strategy to adapt streaming data.

4.1 Visual Characteristic Modeling

MF for implicit feedback is operated on a user-product consumption matrix $\boldsymbol{R} = [r_{uj}]_{m \times n}$ with m users and n products. r_{uj} is a binary variable, where $r_{uj} = 1$ means that user u has bought product j; otherwise $r_{uj} = 0$. Let $\boldsymbol{W} \in \mathbb{R}^{m \times k}$ and $\boldsymbol{H} \in \mathbb{R}^{n \times k}$ be user-specific and product-specific latent feature matrices, with column vectors $\boldsymbol{w}_u \in \mathbb{R}^k$ and $\boldsymbol{h}_j \in \mathbb{R}^k$ denoting the k-dimensional latent feature vector for user u and product j, respectively. We model user preference with product's extrinsic characteristic as a probabilistic generative model. Inspired by the success of Probabilistic Matrix Factorization (PMF) [43], the preference r_{uj} of the matrix $\boldsymbol{R}$ is independently generated by Gaussian distributions:

$$P(\boldsymbol{R} \mid \boldsymbol{W}, \boldsymbol{H}, \boldsymbol{E}) = \prod_{u=1}^{m} \prod_{j=1}^{n} \mathcal{N}(r_{uj} \mid \boldsymbol{w}_u^\top \boldsymbol{h}_j, (C_{uj}^r)^{-1}), \tag{1}$$

where $\mathcal{N}(x \mid \mu, \sigma^{-1})$ is the Gaussian distribution with mean μ and precision σ. Here, C_{uj}^r serves as a confidence hyper-parameter for the preference r_{uj} estimated from $\boldsymbol{w}_u^\top \boldsymbol{h}_i$. To alleviate the class imbalance problem, we introduce different confidence hyper-parameters C_{uj}^r for different preferences r_{uj} similar to that for WRMF [13]. In particular, we set C_{uj}^r a higher value when $r_{uj} = 1$ than that of $r_{uj} = 0$ (*i.e.*, $C_{uj}^r = \gamma$ if $r_{uj} = 1$ where $\gamma > 1$, and $C_{uj}^r = 1$ otherwise). In order to reduce the dimension of product's visual characteristic, we learn an embedding kernel matrix $\boldsymbol{E} \in \mathbb{R}^{c \times k}$, which linearly transforms high-dimensional visual features $\boldsymbol{f}_j \in \mathbb{R}^c$ into a much lower-dimensional visual style space by $\boldsymbol{E}^\top \boldsymbol{f}_j$. For each product j, we draw product latent offset $\boldsymbol{\epsilon}_j \sim \mathcal{N}(0, \beta^{-1}\boldsymbol{I})$ and set the product latent vector as $\boldsymbol{h}_j = \boldsymbol{\epsilon}_j + \boldsymbol{E}^\top \boldsymbol{f}_j$. Note that $\boldsymbol{h}_j = \boldsymbol{\epsilon}_j + \boldsymbol{E}^\top \boldsymbol{f}_j$, where $\boldsymbol{\epsilon}_j \sim \mathcal{N}(0, \beta^{-1}\boldsymbol{I})$, is equivalent to $\boldsymbol{h}_j \sim \mathcal{N}(\boldsymbol{E}^\top \boldsymbol{f}_j, \beta^{-1}\boldsymbol{I})$. Same as PMF, the prior distributions over parameters are assumed to be Gaussian:

$$\begin{aligned} P(\boldsymbol{W}) &= \prod_{u=1}^{m} \mathcal{N}(\boldsymbol{w}_u \mid 0, \lambda_w^{-1}\boldsymbol{I}), \\ P(\boldsymbol{E}) &= \prod_{t=1}^{c} \prod_{d=1}^{k} \mathcal{N}(e_{td} \mid 0, \lambda_e^{-1}), \\ P(\boldsymbol{H} \mid \boldsymbol{E}) &= \prod_{j=1}^{n} \mathcal{N}(\boldsymbol{h}_j \mid \boldsymbol{E}^\top \boldsymbol{f}_j, \beta^{-1}\boldsymbol{I}), \end{aligned} \tag{2}$$

where λ_w, λ_e, and β control the precisions of the three Gaussian distributions and $\boldsymbol{I}$ is the identity matrix. Here, we define the joint probability as $P(\boldsymbol{W}, \boldsymbol{H}, \boldsymbol{E}, \boldsymbol{R})$. $P(\boldsymbol{R})$ is the marginal probability (*a.k.a.* the evidence) that an interaction r_{uj} is seen, regardless of whether it is a positive or negative example. Thereby, combining the prior distributions and what the data reveals using Bayes' rule[3], the posterior distribution of $\boldsymbol{W}, \boldsymbol{H}$ and $\boldsymbol{E}$ is given by:

$$\begin{aligned} &P(\boldsymbol{W}, \boldsymbol{H}, \boldsymbol{E} \mid \boldsymbol{R}) \\ &= P(\boldsymbol{W}, \boldsymbol{H}, \boldsymbol{E}, \boldsymbol{R}) / P(\boldsymbol{R}) \\ &\propto P(\boldsymbol{W}, \boldsymbol{H}, \boldsymbol{E}, \boldsymbol{R}) \\ &= P(\boldsymbol{R} \mid \boldsymbol{W}, \boldsymbol{H}, \boldsymbol{E}) P(\boldsymbol{W}) P(\boldsymbol{H} \mid \boldsymbol{E}) P(\boldsymbol{E}). \end{aligned} \tag{3}$$

4.2 Functional Characteristic Modeling

Let $\boldsymbol{H} \in \mathbb{R}^{n \times k}$ and $\boldsymbol{Z} \in \mathbb{R}^{n \times k}$ be product-specific latent feature matrices, with column vectors $\boldsymbol{h}_j$ and $\boldsymbol{z}_i$ denoting product-specific latent feature vectors, respectively. To reflect the asymmetric nature of the functionally complementary relations (*e.g.*, after a user purchases a phone, it is more appropriate to recommend screensavers, not vice versa.), we model the functional complementarity between product j and i by $\boldsymbol{h}_j^\top \boldsymbol{z}_i$ rather than $\boldsymbol{h}_j^\top \boldsymbol{h}_i$. $\boldsymbol{z}_i$ characterizes the behaviors of "to be functionally complemented by others" of product i. Similar to Eq. (1), we define the conditional distribution over $\boldsymbol{S}$ as:

$$P(\boldsymbol{S} \mid \boldsymbol{H}, \boldsymbol{Z}) = \prod_{j=1}^{n} \prod_{i=1}^{n} \mathcal{N}(s_{ji} \mid \boldsymbol{h}_j^\top \boldsymbol{z}_i, (C_{ji}^s \alpha)^{-1}), \tag{4}$$

where α is a balancing hyper-parameter that regulates the importance of the functional characteristic in modeling user preference. Same as C_{uj}^r, we also introduce different confidence hyper-parameters for different scores s_{ji} (*i.e.*, $C_{ji}^s = \gamma$ if $s_{ji} = 1$ and $C_{ji}^s = 1$ otherwise). Similar to previous definitions on $\boldsymbol{W}$ and $\boldsymbol{E}$, the priors of $\boldsymbol{H}$ and $\boldsymbol{Z}$ are modeled as zero-mean spherical Gaussian distributions:

$$\begin{aligned} P(\boldsymbol{H}) &= \prod_{j=1}^{n} \mathcal{N}(\boldsymbol{h}_j \mid 0, \lambda_h^{-1}\boldsymbol{I}), \\ P(\boldsymbol{Z}) &= \prod_{i=1}^{n} \mathcal{N}(\boldsymbol{z}_i \mid 0, \lambda_z^{-1}\boldsymbol{I}), \end{aligned} \tag{5}$$

where λ_h and λ_z are the precisions of the two Gaussian distributions. Hence, similar to Eq. (3), through a Bayesian inference, we can compute the posterior distribution over the latent features of $\boldsymbol{H}$ and $\boldsymbol{Z}$ as:

$$\begin{aligned} &P(\boldsymbol{H}, \boldsymbol{Z} \mid \boldsymbol{S}) \\ &= P(\boldsymbol{H}, \boldsymbol{Z}, \boldsymbol{S}) / P(\boldsymbol{S}) \\ &\propto P(\boldsymbol{H}, \boldsymbol{Z}, \boldsymbol{S}) \\ &= P(\boldsymbol{S} \mid \boldsymbol{H}, \boldsymbol{Z}) P(\boldsymbol{H}) P(\boldsymbol{Z}). \end{aligned} \tag{6}$$

2. Note that the aim of this work is not to discover functionally complementary or supplementary products, which is another line of research [9]. As such, we only use the "*frequently-bought-together*" statistics provided in the data to build the $\boldsymbol{S}$ matrix.

3. posterior=prior $\times$ likelihood / evidence

4.3 Unified Model

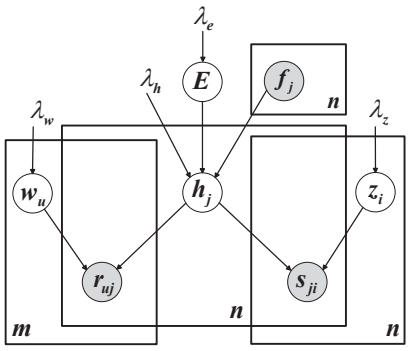


Fig. 2: Graphical model representation of VFPMF. Solid lines denote generative process. Gray and white nodes denote observed variables and latent variables, respectively.

By far, we have shown how to model the visual and functional characteristics of products, respectively. As described in an earlier example (see Figure 1), user's attentions would vary in different product domains. In this subsection, we present a unified probabilistic model to capture the varying importance of product aspects for users in different product domains, named as VFPMF, by simultaneously considering product's visual and functional characteristics. Given an observed interaction of user u purchases product j and a functional complementarity between product j and product i, Figure 2 illustrates how we integrate the visual and functional characteristics of products into a unified model by sharing the product-specific latent feature matrix $\boldsymbol{H}$.

Based on Figure 2, the posterior distribution over various features is given by:

$$\begin{aligned} & P(\boldsymbol{W}, \boldsymbol{H}, \boldsymbol{Z}, \boldsymbol{E} \mid \boldsymbol{R}, \boldsymbol{S}) \\ & = P(\boldsymbol{W}, \boldsymbol{H}, \boldsymbol{Z}, \boldsymbol{E}, \boldsymbol{R}, \boldsymbol{S}) / P(\boldsymbol{R}, \boldsymbol{S}) \\ & \propto P(\boldsymbol{W}, \boldsymbol{H}, \boldsymbol{Z}, \boldsymbol{E}, \boldsymbol{R}, \boldsymbol{S}) \\ & = P(\boldsymbol{R} \mid \boldsymbol{W}, \boldsymbol{H}, \boldsymbol{E}) P(\boldsymbol{S} \mid \boldsymbol{H}, \boldsymbol{Z}) P(\boldsymbol{W}) \\ & \times P(\boldsymbol{H} \mid \boldsymbol{E}) P(\boldsymbol{H}) P(\boldsymbol{Z}) P(\boldsymbol{E}). \end{aligned} \tag{7}$$

Correspondingly, maximizing the log-posterior distribution over the hidden variables with hyper-parameters kept fixed (*i.e.,* $\alpha, \beta, \gamma, \lambda_w, \lambda_h, \lambda_z$ and λ_e) is equivalent to minimizing the following sum-of-squared-errors objective function with quadratic regularization terms:

$$\begin{aligned} & \mathcal{L}(\boldsymbol{R}, \boldsymbol{S}, \boldsymbol{W}, \boldsymbol{H}, \boldsymbol{Z}, \boldsymbol{E}) \\ & = \frac{1}{2} \sum\nolimits_{u=1}^{m} \sum\nolimits_{j=1}^{n} C_{uj}^{r} (r_{uj} - \boldsymbol{w}_u^\top \boldsymbol{h}_j)^2 + \frac{\lambda_w}{2} \|\boldsymbol{W}\|_F^2 \\ & + \frac{\alpha}{2} \sum\nolimits_{j=1}^{n} \sum\nolimits_{i=1}^{n} C_{ji}^{s} (s_{ji} - \boldsymbol{h}_j^\top \boldsymbol{z}_i)^2 + \frac{\lambda_h}{2} \|\boldsymbol{H}\|_F^2 \\ & + \frac{\beta}{2} \sum\nolimits_{j=1}^{n} \|\boldsymbol{h}_j - \boldsymbol{E}^\top \boldsymbol{f}_j\|_2^2 + \frac{\lambda_z}{2} \|\boldsymbol{Z}\|_F^2 + \frac{\lambda_e}{2} \|\boldsymbol{E}\|_F^2. \end{aligned} \tag{8}$$

4.4 An Optimization Algorithm

In recommender system applications, ALS and SGD have attracted much attention and are widely utilized for MF [14], [33]. ALS works by iteratively optimizing one parameter, while keeping the others fixed. Due to the cubic time complexity in the target rank [13], the standard ALS (*i.e.,* vector-wise ALS) is not efficient for solving the minimization problem in Eq. (8). SGD is a more generic solver for a wide range of models and is a suitable choice for online learning. But as mentioned in Section 1, it requires more iterations to converge [17], and more important, its accuracy is highly sensitive to the choice of learning rate and negative examples [18]. Towards this end, we design an element-wise ALS algorithm to efficiently solve the optimization problem in Eq. (8). Similar to the standard ALS [13] with respect to the updates, we cyclically optimize the model parameters. The overall update sequence per epoch is

$$\begin{aligned} & \overbrace{\underbrace{w_{11} \cdots w_{1k}}_{\boldsymbol{w}_1} \cdots \underbrace{w_{m1} \cdots w_{mk}}_{\boldsymbol{w}_m}}^{\boldsymbol{W}}, \overbrace{\underbrace{h_{11} \cdots h_{1k}}_{\boldsymbol{h}_1} \cdots \underbrace{h_{n1} \cdots h_{nk}}_{\boldsymbol{h}_n}}^{\boldsymbol{H}}, \\ & \overbrace{\underbrace{z_{11} \cdots z_{1k}}_{\boldsymbol{z}_1} \cdots \underbrace{z_{n1} \cdots z_{nk}}_{\boldsymbol{z}_n}}^{\boldsymbol{Z}}, \overbrace{\underbrace{e_{11} \cdots e_{c1}}_{\boldsymbol{e}_1} \cdots \underbrace{e_{1k} \cdots e_{ck}}_{\boldsymbol{e}_k}}^{\boldsymbol{E}}. \end{aligned} \tag{9}$$

Next, we elaborate how we optimize a single variable at a time while leaving others fixed. To simplify notations, we use $\mathcal{L}$ to denote the objective function in Eq. (8).

Update w_{ud} and z_{id}

First, we get the partial derivative of $\mathcal{L}$ *w.r.t.* w_{ud} as:

$$\frac{\partial \mathcal{L}}{\partial w_{ud}} = -\sum\nolimits_{j=1}^{n} C_{uj}^{r} (r_{uj} - \hat{r}_{uj}) h_{jd} + \lambda_w w_{ud}, \tag{10}$$

where $\hat{r}_{uj} = \boldsymbol{w}_u^\top \boldsymbol{h}_j$. The optimal strategy for updating w_{ud} can be achieved by setting the partial derivative to 0, *i.e.,*

$$\begin{aligned} w_{ud} &= \frac{\sum_{j=1}^{n} C_{uj}^{r} (r_{uj} - \hat{r}_{uj}) h_{jd}}{\lambda_w + \sum_{j=1}^{n} C_{uj}^{r} h_{jd}^2} \\ &= \frac{\sum_{j \in \mathcal{R}_u} C_{uj}^{r} (r_{uj} - \tilde{r}_{uj}) h_{jd} - \sum_{j \notin \mathcal{R}_u} \tilde{r}_{uj} h_{jd}}{\lambda_w + \sum_{j \in \mathcal{R}_u} C_{uj}^{r} h_{jd}^2 + \sum_{j \notin \mathcal{R}_u} h_{jd}^2}, \end{aligned} \tag{11}$$

where $\tilde{r}_{uj} = \hat{r}_{uj} - w_{ud} h_{jd}$, *i.e.,* the prediction without the component of latent feature dimension d. We can see that the main computational bottleneck lies in the massive repeated computations over the missing data part (*i.e.,* the $\sum_{j \notin \mathcal{R}_u}$ term), which requires iterating over the whole negative space. Here, we first concentrate on compute $\sum_{j \notin \mathcal{R}_u} \tilde{r}_{uj} h_{jd}$.

$$\begin{aligned} \sum_{j \notin \mathcal{R}_u} \tilde{r}_{uj} h_{jd} &= \sum_{j=1}^{n} h_{jd} \sum_{v \neq d} w_{uv} h_{jv} - \sum_{j \in \mathcal{R}_u} \tilde{r}_{uj} h_{jd} \\ &= \sum_{v \neq d} w_{uv} \sum_{j=1}^{n} h_{jd} h_{jv} - \sum_{j \in \mathcal{R}_u} \tilde{r}_{uj} h_{jd}. \end{aligned} \tag{12}$$

By the above reformulation, we find that the major computation — the $\sum_{j=1}^{n} h_{jd} h_{jv}$ term that iterates over all products — is independent of u. As such, a significant speedup can be achieved by memorizing the term for all users. Now we define $\boldsymbol{X}^h = \boldsymbol{H}^\top \boldsymbol{H}$, which can be pre-computed and applied to the update of latent features for all users. Correspondingly, Eq. (12) can be rewritten as:

$$\sum_{j \notin \mathcal{R}_u} \tilde{r}_{uj} h_{jd} = \sum_{v \neq d} w_{uv} x_{dv}^h - \sum_{j \in \mathcal{R}_u} \tilde{r}_{uj} h_{jd}. \tag{13}$$

which can be computed in $O(k + |\mathcal{R}_u|)$ time.

Analogously, the calculation of denominator can be speeded up by applying the cache technology:

$$\sum_{j \notin \mathcal{R}_u} h_{jd}^2 = \sum_{j=1}^{n} h_{jd}^2 - \sum_{j \in \mathcal{R}_u} h_{jd}^2 = x_{dd}^h - \sum_{j \in \mathcal{R}_u} h_{jd}^2. \tag{14}$$

To summarize the aforementioned acceleration strategy, we can derive the update rule for w_{ud} by using the $\boldsymbol{X}^h$ cache:

$$w_{ud} = \frac{\sum_{j\in\mathcal{R}_u}[C^r_{uj}r_{uj} - (C^r_{uj}-1)\tilde{r}_{uj}]h_{jd} - \sum_{v\neq d} w_{uv}x^h_{dv}}{\lambda_w + \sum_{j\in\mathcal{R}_u}(C^r_{uj}-1)h^2_{jd} + x^h_{dd}}. \quad (15)$$

Analogously, we can give the update rule for z_{id}:

$$z_{id} = \frac{\alpha\sum_{j\in\mathcal{S}_i}[C^s_{ji}s_{ji} - (C^s_{ji}-1)\tilde{s}_{ji}]h_{jd} - \alpha\sum_{v\neq d} z_{iv}x^h_{dv}}{\lambda_z + \alpha\sum_{j\in\mathcal{S}_i}(C^s_{ji}-1)h^2_{jd} + \alpha x^h_{dd}}, \quad (16)$$

where $\hat{s}_{ji} = \boldsymbol{h}_j^\top \boldsymbol{z}_i$ and $\tilde{s}_{ji} = \hat{s}_{ji} - h_{jd}z_{id}$. x^h_{dv} represents the $(d,v)^{th}$ element of the $\boldsymbol{X}^h$ cache, defined as $\boldsymbol{X}^h = \boldsymbol{H}^\top\boldsymbol{H}$.

Update h_{jd}

The partial derivative of $\mathcal{L}$ *w.r.t.* h_{jd} is given as

$$\frac{\partial\mathcal{L}}{\partial h_{jd}} = -\sum_{j=1}^n C^r_{uj}(r_{uj} - \hat{r}_{uj})h_{jd} + \beta y_{jd} + \lambda_h h_{jd} - \alpha\sum_{j=1}^n C^s_{ji}(s_{ji} - \hat{s}_{ji})z_{id}. \quad (17)$$

Direct computation of h_{jd} via Eq. (8) repeatedly computes $\boldsymbol{y}_j = \boldsymbol{E}^\top\boldsymbol{f}_j$, when updating the laten factors for different d. Clearly, we can accelerate the computation by defining the $\boldsymbol{Y}$ cache as $\boldsymbol{Y} = \boldsymbol{F}\boldsymbol{E}$. Therefore, we can get the solver for a product latent factor:

$$\begin{aligned} T_1 &= \sum_{u=1}^m C^r_{uj}(r_{uj} - \tilde{r}_{uj})w_{ud}, \\ T_2 &= \sum_{j=1}^n C^s_{ji}(s_{ji} - \tilde{s}_{ji})z_{id}, \\ h_{jd} &= \frac{T_1 + \alpha T_2 + \beta y_{jd}}{\lambda_h + \beta + \sum_{u=1}^m C^r_{uj}w^2_{ud} + \alpha\sum_{j=1}^n C^s_{ji}z^2_{id}}. \end{aligned} \quad (18)$$

Similarly, we can apply the cache to accelerate the calculation of numerator and denominator. Then, we can derive the update rule for h_{jd}:

$$\begin{aligned} T_3 &= \sum_{u\in\mathcal{R}_j}[C^r_{uj}r_{uj} - (C^r_{uj}-1)\tilde{r}_{uj}]w_{ud} - \sum_{v\neq d} h_{jv}x^w_{dv}, \\ T_4 &= \sum_{i\in\mathcal{S}_j}[C^s_{ji}s_{ji} - (C^s_{ji}-1)\tilde{s}_{ji}]z_{id} - \sum_{v\neq d} h_{jv}x^z_{dv}, \\ T_5 &= x^w_{dd} + \sum_{u\in\mathcal{R}_j}(C^r_{uj}-1)w^2_{ud}, \\ T_6 &= x^z_{dd} + \sum_{i\in\mathcal{S}_j}(C^s_{ji}-1)z^2_{id}, \\ h_{jd} &= \frac{T_3 + \alpha T_4 + \beta y_{jd}}{\lambda_h + \beta + T_5 + \alpha T_6}, \end{aligned} \quad (19)$$

where x^w_{dv} represents the $(d,v)^{th}$ element of the $\boldsymbol{X}^w = \sum_{u=1}^m \boldsymbol{w}_u\boldsymbol{w}_u^\top$ cache, and x^z_{dv} denotes the $(d,v)^{th}$ element of the $\boldsymbol{X}^z = \sum_{i=1}^n \boldsymbol{z}_i\boldsymbol{z}_i^\top$ cache.

Update e_{td}

To update the e_{td} parameter, we fix $\boldsymbol{W}$, $\boldsymbol{H}$ and $\boldsymbol{Z}$ and remove terms irrelevant to e_{td}, then obtain the partial derivative of $\mathcal{L}$ *w.r.t.* e_{td} as:

$$\frac{\partial\mathcal{L}}{\partial e_{td}} = -\beta\sum_{j\in\mathcal{F}_t}(h_{jd} - \tilde{y}_{jd} - e_{td}f_{jt})f_{jt} + \lambda_e e_{td}, \quad (20)$$

where $\tilde{y}_{jd} = \boldsymbol{e}_d^\top\boldsymbol{f}_j - e_{td}f_{jt}$. Let us denote the t^{th} column of the matrix $\boldsymbol{F}$ by $\boldsymbol{f}_t$, $\mathcal{F}_t$ is the set of non-zero elements of the vector $\boldsymbol{f}_t$. We define the $\boldsymbol{b} \in \mathbb{R}^c$ cache as $b_t = \sum_{j=1}^n f_{jt}f_{jt}$, which can be pre-computed and used in updating e_{td} for each iteration. The optimal e_{td} can be computed by:

$$e_{td} = \frac{\beta\sum_{j\in\mathcal{F}_t}(h_{jd} - \tilde{y}_{jd})f_{jt}}{\beta b_t + \lambda_e}. \quad (21)$$

Algorithm 1 summarizes the designed algorithm for our proposed model. Since our designed optimization algorithm greatly differs from the traditional learning algorithms (*i.e.*, ALS [13] and SGD), we provide theoretical analysis for Algorithm 1 in Appendix A.

Algorithm 1: Fast ALS-based (FALS) learning algorithm for the VFPMF model.

Input: $\boldsymbol{R}$, $\boldsymbol{F}$, $\boldsymbol{S}$, k, λ_b, λ_w, λ_h, α and β;
Output: $\boldsymbol{W}$, $\boldsymbol{H}$, $\boldsymbol{Z}$ and $\boldsymbol{E}$;
1 Randomly initialize $\boldsymbol{W}$, $\boldsymbol{H}$, $\boldsymbol{Z}$ and $\boldsymbol{E}$;
2 **for** $(u,j)\in\mathcal{R}$ **do** $\hat{r}_{uj} \leftarrow \boldsymbol{w}_u^\top\boldsymbol{h}_j$; ▷ $O(|\mathcal{R}|k)$
3 **for** $(j,i)\in\mathcal{S}$ **do** $\hat{s}_{ji} \leftarrow \boldsymbol{h}_j^\top\boldsymbol{z}_i$; ▷ $O(|\mathcal{S}|k)$
4 $x_t = \sum_{j=1}^n f_{jt}f_{jt}$
5 **while** *Stopping criteria is not met* **do**
6 $\quad\boldsymbol{X}^h = \boldsymbol{H}^\top\boldsymbol{H}$; ▷ $O(nk^2)$
7 $\quad$**for** $u \leftarrow 1$ **to** m **do** ▷ $O(mk^2 + |\mathcal{R}|k)$
8 $\quad\quad$**for** $d \leftarrow 1$ **to** k **do**
9 $\quad\quad\quad$**for** $j \in \mathcal{R}_u$ **do** $\tilde{r}_{uj} \leftarrow \hat{r}_{uj} - w_{ud}h_{jd}$;
10 $\quad\quad\quad w_{ud} \leftarrow$ Eq. (15) ; ▷ $O(k + |\mathcal{R}_u|)$
11 $\quad\quad\quad$**for** $j \in \mathcal{R}_u$ **do** $\hat{r}_{uj} \leftarrow \tilde{r}_{uj} + w_{ud}h_{jd}$;
12 $\quad\quad$**end**
13 $\quad$**end**
14 $\quad\boldsymbol{X}^w = \boldsymbol{W}^\top\boldsymbol{W}$, $\boldsymbol{X}^z = \boldsymbol{Z}^\top\boldsymbol{Z}$; ▷ $O(mk^2 + nk^2)$
15 $\quad$**for** $j \leftarrow 1$ **to** n **do** ▷ $O(nk^2 + |\mathcal{R}|k + |\mathcal{S}|k)$
16 $\quad\quad$**for** $d \leftarrow 1$ **to** k **do**
17 $\quad\quad\quad$**for** $u \in \mathcal{R}_j$ **do** $\tilde{r}_{uj} \leftarrow \hat{r}_{uj} - w_{ud}h_{jd}$;
18 $\quad\quad\quad$**for** $i \in \mathcal{S}_j$ **do** $\tilde{s}_{ji} \leftarrow \hat{s}_{ji} - h_{jd}z_{id}$;
19 $\quad\quad\quad h_{jd} \leftarrow$ Eq. (19) ; ▷ $O(k + |\mathcal{R}_j| + |\mathcal{S}_j|)$
20 $\quad\quad\quad$**for** $u \in \mathcal{R}_j$ **do** $\hat{r}_{uj} \leftarrow \tilde{r}_{uj} + w_{ud}h_{jd}$;
21 $\quad\quad\quad$**for** $i \in \mathcal{S}_j$ **do** $\hat{s}_{ji} \leftarrow \tilde{s}_{ji} + h_{jd}z_{id}$;
22 $\quad\quad$**end**
23 $\quad$**end**
24 $\quad$Similar to line 6-12, update $\boldsymbol{Z}$ based on Eq. (16);
25 $\quad$Similar to line 7-12, update $\boldsymbol{E}$ based on Eq. (21);
26 **end**
27 **return** $\boldsymbol{W}$, $\boldsymbol{H}$, $\boldsymbol{Z}$ *and* $\boldsymbol{E}$

4.5 Online Update

The offline training of a recommender model provides top-N recommendation based on historical data. To support real-time personalization, an important aspect for commercial recommender systems is how to efficiently refresh model parameters when a new interaction streams in. For example, soon after a user watches a video, he may want to explore more videos to watch next. Since re-training the whole model in real-time is impractical, it is better to perform local updates on model parameters based on the signal in the new interaction only. Here, we devise an online updating strategy to adapt FALS to real-time streaming data.

Incremental Updating. Let (u,j) represent the new interaction streamed in, and $\hat{\boldsymbol{W}}$, $\hat{\boldsymbol{H}}$, $\hat{\boldsymbol{Z}}$ and $\hat{\boldsymbol{E}}$ represent the

Algorithm 2: Online Incremental Updates for FALS.

Input: $\hat{\boldsymbol{W}}$, $\hat{\boldsymbol{H}}$, $\hat{\boldsymbol{Z}}$, $\hat{\boldsymbol{E}}$ and new interaction (u, j)
Output: Refreshed parameters $\boldsymbol{W}$, $\boldsymbol{H}$, $\boldsymbol{Z}$ and $\boldsymbol{E}$;
$\boldsymbol{W} \leftarrow \hat{\boldsymbol{W}}$; $\boldsymbol{H} \leftarrow \hat{\boldsymbol{H}}$; $\boldsymbol{Z} \leftarrow \hat{\boldsymbol{Z}}$; $\boldsymbol{E} \leftarrow \hat{\boldsymbol{E}}$;
If u is a new user **do** Randomly initialize $\boldsymbol{w}_u$;
If j is a new product **do** Randomly initialize $\boldsymbol{h}_j$;
$r_{uj} \leftarrow 1$; $\hat{r}_{uj} \leftarrow \boldsymbol{w}_u^\top \boldsymbol{h}_j$;
while *Stopping criteria is not met* **do**
  `// Line 8-12 of Algorithm 1`
  update $\boldsymbol{w}_u$; ▷ $O(k^2 + |\mathcal{R}_u|k)$
  update cache $(u, \boldsymbol{X}^w)$; ▷ $O(k^2)$
  if j *is not a new product* **then**
    `// Line 16-22 of Algorithm 1`
    update $\boldsymbol{h}_j$; ▷ $O(k^2 + |\mathcal{R}_j|k + |\mathcal{S}_j|k)$
    update cache $(j, \boldsymbol{X}^h)$; ▷ $O(k^2)$
    `// Line 24 of Algorithm 1`
    update $\boldsymbol{z}_j$; ▷ $O(k^2 + |\mathcal{S}_j^{'}|k)$
    update cache $(j, \boldsymbol{X}^z)$; ▷ $O(k^2)$
  else
    update $\boldsymbol{h}_j \leftarrow \frac{T_3 + \beta y_{jd}}{\lambda_h + \beta + T_5}$; ▷ $O(k^2 + |\mathcal{R}_j|k)$
    update cache $(j, \boldsymbol{X}^h)$; ▷ $O(k^2)$
  end
  update $e_{td} \leftarrow (h_{jd} - \tilde{y}_{jd}) f_{jt} / (f_{jt}^2 + \lambda_e)$; ▷ $O(ck)$
end
return $\boldsymbol{W}$, $\boldsymbol{H}$, $\boldsymbol{Z}$ *and* $\boldsymbol{E}$

model parameters learnt from offline training. Our strategy is to perform updates on parameters that are relevant to the prediction of r_{uj} only, *i.e.*, $\hat{\boldsymbol{w}}_u$, $\hat{\boldsymbol{h}}_j$, $\hat{\boldsymbol{z}}_j$ and $\hat{\boldsymbol{E}}$ (note that if j is a new product, the features for $\hat{\boldsymbol{w}}_u$, $\hat{\boldsymbol{h}}_j$ and $\hat{\boldsymbol{E}}$ are updated, and the following is the same). The underlying assumption is that, given the new consumption, the factors $\hat{\boldsymbol{w}}_u$, $\hat{\boldsymbol{h}}_j$, $\hat{\boldsymbol{z}}_j$ and $\hat{\boldsymbol{E}}$ are changed significantly, while other features in $\hat{\boldsymbol{W}}$, $\hat{\boldsymbol{H}}$ and $\hat{\boldsymbol{Z}}$ should remain largely stable since they are not directly affected. The incremental learning process for FALS is summarized in Algorithm 2.

Time Complexity. If j is a new product, the computational complexity of online updates for FALS in Algorithm 2 is $O(k^2 + (|\mathcal{R}_u| + |\mathcal{R}_j| + c)k)$, otherwise $O(k^2 + (|\mathcal{R}_u| + |\mathcal{R}_j| + |\mathcal{S}_j| + |\mathcal{S}_j^{'}| + c)k)$. We can see that the complexity depends on the number of observed interactions for u and j, while being independent with the number of total consumptions, users and products. In fact, our empirical study shows that one iteration is sufficient for the online updating strategy to achieve good performance and more iterations are not beneficial (*cf.* Figure 7). As such, we can analytically conclude that the online learning algorithm is capable of serving the needs of industrial scenario.

5 Experiments

In this section, we conduct extensive experiments to explore our method on four real-world datasets. The experiments are designed to answer the following research questions:

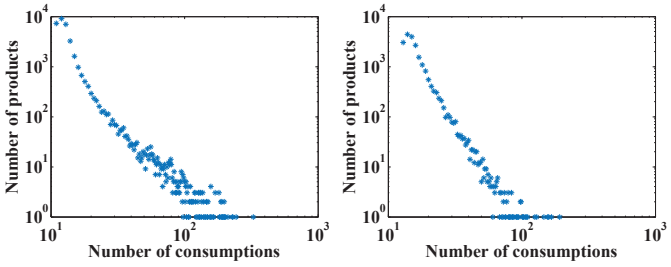
RQ1 How does VFPMF perform when compared with other state-of-the-art competitors?
RQ2 How does the performance of VFPMF on users and products of different sparsity levels?
RQ3 How is the efficiency of our accelerated FALS as compared to standard ALS?
RQ4 How does VFPMF perform for online learning scenarios?

5.1 Experimental Settings

TABLE 2: Statistics of the four Amazon datasets.

Dataset	Office Product	Auto motive	Amazon Men	Amazon Women
# of Users	16,772	34,892	27,627	19,200
# of Products	37,956	129,108	88,112	22,489
# of Consumptions	145,141	284,101	205,877	136,718
# of Relations	6,913	65,583	19,996	14,109
Density C/(U*P)	0.0227%	0.0036%	0.0084%	0.0316%
Avg C/U	8.65	8.14	7.45	7.12
Avg C/P	3.82	2.20	2.34	6.08

Data Description. To comprehensively demonstrate the performance of VFPMF in terms of top-N recommendation, we choose four real-world user-product consumption datasets for our experiments, which are obtained from Amazon[4]. These datasets consist of user ratings, product images, a list of "*frequently-bought-together*" products, and etc. Among these metadata, we use "*frequently-bought-together*" information to construct *functional characteristic matrix* $\boldsymbol{S}$ [9], [44]. Aiming at demonstrating the benefit of our method in general various domains, we not only use two datasets (*i.e.,* Office Product and Automotive) from visually non-aware product domain, where the functional characteristic is very important, but also two datasets (*i.e.,* Amazon Men and Amazon Women) from visually-aware product domain, where the visual characteristic plays a more crucial role. For all datasets, we eliminate users that have less than five consumptions. As a result, Table 2 shows the characteristics of the final datasets.



(a) Office Product **(b) Amazon Women**

Fig. 3: Dataset statistics *w.r.t.* product.

To gain a better insight into the data *w.r.t.* products, some statistical analysis is now presented. We plotted the product distribution *w.r.t.* the number of consumptions and the number of products in Figure 3 (log-log plot). We focus on the Office Product and Amazon Women datasets as it shows similar trends on other datasets. As we can see, products show a long-tail distribution, which refers to the fact that a disproportionally large number of consumptions are condensed in a small fraction of products (popular products) and most products was bought by few users. Popular products tend to dominate the recommendation

4. http://jmcauley.ucsd.edu/data/amazon/

results, making it difficult to provide diverse and novel recommendations.

In our experiments, we transform observed ratings into binary implicit feedback as ground truth, whether the user rated the product. To evaluate the quality of product recommendation, we use two different strategies to split a dataset for different testing scenarios. (1) *Offline Protocol.* For each user, we randomly select 80% of historical actions to constitute the training set, and use all remaining actions for testing. From the training set, we randomly select 10% of consumption records as validation set to tune hyper-parameters. (2) *online Protocol.* To simulate a realistic recommendation scenario, all consumption records are sorted by time, the first 90% of the interactions are utilized for training with the optimal hyper-parameter settings demonstrated by the offline evaluation, and the latest 10% of the consumption records are used for testing.

Evaluation Metrics. In this product recommendation setting, we present each user with the top N products sorted by their predicted preference, and evaluate different methods based on which of these products were actually purchased by the user in the test data. Following previous work [45], [46], [47], [48], four commonly ranking-based metrics, *i.e.,* Precision@N, Recall@N, MAP@N and NDCG@N, are used for evaluating the performance of recommender systems. Formally, their definitions are shown as follows:

$$\begin{aligned}
\text{Precision@N} &= \frac{1}{m}\sum_{u=1}^{m}\frac{|l_{\text{rec}}^{u}\bigcap l_{\text{tes}}^{u}|}{\text{N}},\\
\text{Recall@N} &= \frac{1}{m}\sum_{u=1}^{m}\frac{|l_{\text{rec}}^{u}\bigcap l_{\text{tes}}^{u}|}{|l_{\text{tes}}^{u}|},\\
\text{MAP@N} &= \frac{1}{m}\sum_{u=1}^{m}\frac{\sum_{a=1}^{\text{N}}P_u(a)\times\delta_u(a)}{\min\{\text{N},|l_{\text{tes}}^{u}|\}},\\
\text{NDCG@N} &= \frac{1}{m}\sum_{u=1}^{m}\frac{\sum_{a=1}^{\text{N}}\frac{2^{\delta_u(a)}-1}{\log_2(a+1)}}{1/\log_2(a+1)},
\end{aligned} \tag{22}$$

where l_{rec}^{u} is the set of top-N unobserved products recommended to user u excluding those products in the training data, and l_{tes}^{u} is the set of products that user u has purchased in the test data. $P_u(a)$ is the precision at the a^{th} position in the top-N list l_{rec}^{u}, and $\delta_u(a)$ is a binary indicator function that equals to 1 if the product at rank a is purchased in the test data, otherwise equals to 0.

Baselines. Matrix factorization based models are currently state-of-the-arts for top-N recommendation. To comparatively demonstrate the effectiveness of our proposed method, we mainly compare against state-of-the-art MF competitors and neural collaborative filtering, including both pairwise and pointwise models.

- **POP:** This baseline ranks products according to their popularity and is non-personalised method.
- **BPR [20]:** This baseline is a state-of-the-art method for personalised product recommendation on implicit feedback datasets, which uses pairwise log-sigmoid loss. It models the pairwise ranking for each pair of the unobserved and observed products, and employs SGD with bootstrap sampling for optimization.
- **VBPR [10]:** The Visual Bayesian Personalized Ranking model is a state-of-the-art visually-aware method for top-N recommendation, which is based on visual contents of the products.
- **ACF [11]:** The attentive collaborative filtering model is a state-of-the-art visually-aware neural network method for top-N recommendation, which models the item- and component-level implicitness with a hierarchical attention network.
- **AoBPR [49]:** This is an extension of BPR, which uses adaptive sampling for optimization. The products are re-scored after every few epoches, and sampled proportional to their current score.
- **WARP [50]:** A pair-wise MF model from Weston, which is optimized for a weighted approximate ranking loss.
- **WRMF [13]:** This baseline is a state-of-the-art pointwise model for implicit feedback. It treats user preference as a binary value and minimizes the square error loss by assigning both observed and unobserved products with different confidence levels based on MF.
- **VPMF:** A visually-aware probabilistic matrix factorization model that only considers products' visual characteristics, and its objective function is written as

$$\begin{aligned}
\mathcal{L}(\boldsymbol{R},\boldsymbol{W},\boldsymbol{H},\boldsymbol{E}) &= \frac{1}{2}\sum_{u=1}^{m}\sum_{j=1}^{n}C_{uj}^{r}(r_{uj}-\boldsymbol{w}_u^{\top}\boldsymbol{h}_j)^2+\\
&\frac{\beta}{2}\sum_{j=1}^{n}\|\boldsymbol{h}_j-\boldsymbol{E}^{\top}\boldsymbol{f}_j\|_2^2+\frac{\lambda_w}{2}\|\boldsymbol{W}\|_F^2+\frac{\lambda_e}{2}\|\boldsymbol{E}\|_F^2.
\end{aligned} \tag{23}$$

- **FPMF:** This baseline models product's functional characteristic using the technique we introduced in Section 4.2, but does not account for product's visual characteristic.

$$\begin{aligned}
\mathcal{L}(\boldsymbol{R},\boldsymbol{S},\boldsymbol{W},\boldsymbol{H},\boldsymbol{Z}) &= \frac{\lambda_w}{2}\|\boldsymbol{W}\|_F^2+\frac{\lambda_h}{2}\|\boldsymbol{H}\|_F^2+\\
&\frac{1}{2}\sum\nolimits_{u=1}^{m}\sum\nolimits_{j=1}^{n}C_{uj}^{r}(r_{uj}-\boldsymbol{w}_u^{\top}\boldsymbol{h}_j)^2+\frac{\lambda_z}{2}\|\boldsymbol{Z}\|_F^2+\\
&\frac{\alpha}{2}\sum\nolimits_{j=1}^{n}\sum\nolimits_{i=1}^{n}C_{ji}^{s}(s_{ji}-\boldsymbol{h}_j^{\top}\boldsymbol{z}_i)^2.
\end{aligned}$$

In summary, these baselines are designed to demonstrate: (1) the performance of the current state-of-the-art pairwise method with uniformly sampling (*i.e.,* BPR), (2) the influence of using non-uniformly sampling (*i.e.,* WARP and AoBPR) (3) the strength of state-of-the-art pointwise approach merely based on user-product interactions (*i.e.,* WRMF), (4) the effect of using product's visual characteristic (*i.e.,* VBPR, ACF and VPMF), and (5) the improvement of using product's functional characteristic to improve traditional MF model in a relatively straightforward manner (*i.e.,* FPMF).

Hyper-parameter Settings. Most baselines are from LibRec [51], Our implementation of VFPMF is publicly available at Github[5]. For ACF[6], we adopt the authors' released source code and tune its hyper-parameters in the same way as [11]. For a fair comparison, our experiments randomly initialize all involved model parameters normal distribution, where the mean and standard deviation is 0 and 0.01, respectively. For pairwise methods

5. https://github.com/wubinzzu/VFPMF
6. https://github.com/ChenJingyuan91/ACF

TABLE 3: Recommendation performance (%) of different methods on four real-world datasets with N=10. Where row 'Improve' indicates the percentage of improvements that VFPMF achieves relative to the '*' results.

Dataset	Method	k=10				k=30			
		Precision	Recall	MAP	NDCG	Precision	Recall	MAP	NDCG
Office Product	POP	0.3286	1.0463	0.3924	0.7279	0.3286	1.0463	0.3924	0.7279
	BPR	0.5145	2.2221	0.7868	1.3602	0.8179	3.7897	1.6316	2.5379
	VBPR	0.5671	2.5846	1.0342	1.6761	0.8499	3.8344	1.7453	2.6423
	ACF	0.6205	2.6667	1.0752	1.7318	0.8766	3.9110	1.7938	2.7455
	WARP	0.7485	3.0936	1.2435	2.0573	0.9315	3.8977	1.7371	2.7078
	AoBPR	0.6441	2.7580	1.0603	1.7815	0.9661	4.0212	1.8061	2.8649
	WRMF	0.8911*	3.3652*	1.3659*	2.2902*	1.0138*	4.1880*	1.8190*	2.9017*
	VPMF	0.9193	3.5023	1.4021	2.3851	1.0283	4.3195	1.8357	2.9403
	FPMF	0.9410	3.7804	1.4368	2.4692	1.2018	5.1685	2.2202	3.5705
	VFPMF	**0.9559**	**3.8752**	**1.4472**	**2.4944**	**1.2326**	**5.3855**	**2.2292**	**3.5783**
	Improve	7.27%	15.16%	5.95%	8.92%	21.58%	28.59%	22.55%	23.31%
Automotive	POP	0.2163	1.1014	0.4489	0.7092	0.2163	1.1014	0.4489	0.7092
	BPR	0.2299	1.1710	0.4594	0.7362	0.4014	2.2296	0.9748	1.4624
	VBPR	0.2773	1.3979	0.5847	0.9131	0.4303	2.2597	0.9921	1.4969
	ACF	0.2978	1.5040	0.6017	0.9552	0.4434	2.2973	1.0378	1.5433
	WARP	0.3345	1.6544	0.6608	1.0557	0.4522	2.3154	0.9632	1.5391
	AoBPR	0.3265	1.6711	0.5932	0.9854	0.4791	2.4456	1.0802	1.6546
	WRMF	0.4411*	2.1793*	0.8463*	1.3783*	0.4901*	2.4743*	1.0836*	1.6554*
	VPMF	0.4478	2.1885	0.8508	1.3886	0.5268	2.6458	1.1329	1.7496
	FPMF	0.4627	2.3160	0.9075	1.4648	0.5840	2.8750	1.1676	1.8601
	VFPMF	**0.4792**	**2.3644**	**0.9094**	**1.4858**	**0.6176**	**3.0804**	**1.2272**	**1.9533**
	Improve	8.63%	8.49%	7.45%	7.79%	26.01%	24.50%	13.25%	17.99%
Amazon Men	POP	0.1338	0.7827	0.2473	0.4247	0.1338	0.7827	0.2473	0.4247
	BPR	0.1470	0.7907	0.2681	0.4546	0.2330	1.2089	0.4831	0.7690
	VBPR	0.1725	0.9190	0.3324	0.5411	0.2879	1.5289	0.5947	0.9533
	ACF	0.1829	0.9847	0.3562	0.6088	0.2931*	1.5419*	0.6104*	0.9669
	WARP	0.2052	1.0568	0.3683	0.6264	0.2402	1.2214	0.4866	0.7723
	AoBPR	0.1659	0.8612	0.2980	0.5049	0.2430	1.2758	0.5007	0.7987
	WRMF	0.2548*	1.3071*	0.5173*	0.8267*	0.2926	1.5152	0.6044	0.9696*
	VPMF	0.3025	1.5634	0.5973	0.9643	0.3489	1.8336	0.7093	1.1404
	FPMF	0.2836	1.4443	0.5621	0.9041	0.3342	1.7247	0.6956	1.1218
	VFPMF	**0.3091**	**1.6034**	**0.6005**	**0.9797**	**0.3635**	**1.9217**	**0.7738**	**1.2187**
	Improve	21.31%	22.66%	16.08%	18.50%	24.02%	24.63%	26.76%	25.69%
Amazon Women	POP	0.1950	0.9282	0.3055	0.5452	0.1950	0.9282	0.3055	0.5452
	BPR	0.2620	1.2170	0.4297	0.7409	0.4157	1.9024	0.7216	1.2054
	VBPR	0.3276	1.5177	0.6538	1.0227	0.4653	2.1267	0.7788	1.3220
	ACF	0.3452	1.6011	0.6894	1.2920	0.4768	2.2272	0.8275	1.3591
	WARP	0.3423	1.5775	0.5936	0.9952	0.4081	1.8910	0.8030	1.2608
	AoBPR	0.3218	1.4637	0.5932	0.8922	0.4578	2.1449	0.8572	1.3743
	WRMF	0.4403*	2.0447*	0.8015*	1.3177*	0.5147*	2.4201*	1.0653*	1.6524*
	VPMF	0.4754	2.2296	0.8408	1.4199	0.5754	2.9109	1.2198	1.9615
	FPMF	0.4621	2.1667	0.8532	1.3965	0.5619	2.9034	1.1913	1.9013
	VFPMF	**0.5024**	**2.3242**	**0.9249**	**1.5110**	**0.6596**	**3.1551**	**1.2678**	**2.0423**
	Improve	14.10%	13.67%	15.39%	14.67%	28.15%	30.37%	19.01%	23.60%

(*i.e.,* BPR, AoBPR, VBPR and WARP), the learning ratio is selected from $\{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1\}$. For all methods, the L_2 regularization coefficients are tuned in $\{0.0001, 0.001, 0.01, 0.1, 1\}$. The Geometric distribution hyper-parameter ρ in AOBPR is chosen from $\{0.01, 0.03, 0.005, 0.007, 0.1, 0.3, 0.5, 0.7\}$. For VPMF,VBPR and VFPMF, λ_e is tuned in $\{0.1, 1, 10, 100, 1000\}$. We tune the hyper-parameter α in $\{0.01, 0.1, 1, 3, 5, 7, 10\}$, β in $\{1, 10, 100, 300, 500, 1000\}$, and γ in $\{1, 3, 5, 7, 10\}$. All methods except ACF are implemented in Java 7 and running on a server equipped with Intel Xeon E5-2699 CPU @ 2.30GHz on 128GB RAM, 36 cores and 64-bit Windows Server 2016 operating system.

5.2 Comparison with State-of-The-Arts (RQ1)

We now evaluate the performance of VFPMF by comparing the results on four real-world datasets with its competitors. Note that visually-aware product domain denotes Amazon Women and Amazon Men, and visually non-aware product domain denotes Office Product and Automotive. Table 3 shows recommendation accuracy in four metrics *w.r.t.* the dimension of latent features. Moreover, we also conduct the paired two-sample t-test experiments, showing that the improvements are statistically stable and non-contingent (p-value < 0.01) in four metrics. For the sake of space, they are omitted. We make a few comparisons to better explain and understand our findings as follows:

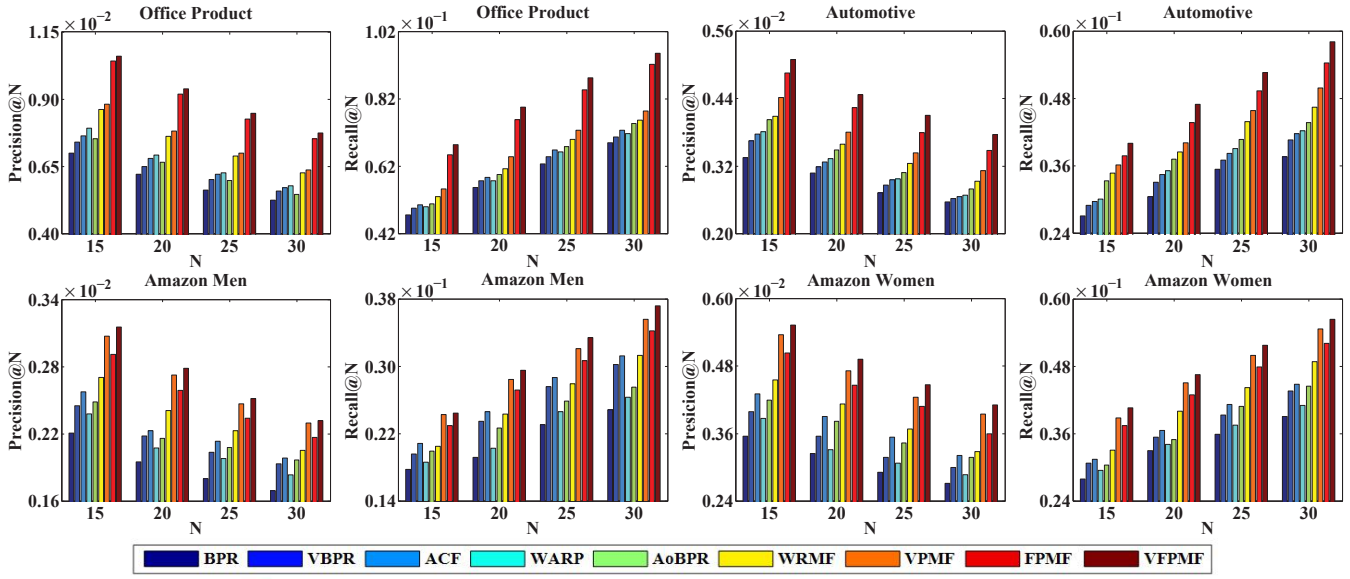


Fig. 4: Performance comparisons for different values of N on four real-world datasets

POP *vs.* BPR. BPR outperforms POP, and this observation is consistent with that in [20]. The long-tail effects may be one reason why it has the worst performance.

BPR *vs.* AoBPR, WARP. AoBPR and WARP perform slightly better than BPR. Instead of bootstrap sampling negative products, AoBPR and WARP sample with an adaptive manner, and this observation is consistent with that in [49], [52].

BPR *vs.* VBPR. In all datasets, VBPR outperforms BPR, while the improvement is more significant in visually-aware product domain. This is because the user-product interaction matrix is extremely sparse, and product's extrinsic characteristic can provide additional information to alleviate the data sparsity problem.

VBPR *vs.* ACF. For each dataset, ACF has a higher performance than VBPR, which implies that combining the item-and component-level implicitness with a hierarchical attention network have a significant impact on the recommendation accuracy.

VPMF *vs.* FPMF. (1) In visually-aware product domain, the recommendation accuracy of VPMF outperforms FPMF, it indicates that product's visual characteristic plays a very important role in modeling user decision making when selecting products (*i.e.*, Amazon Men and Amazon Women), and (2) FPMF outperforms IPMF in visually non-aware product domain, it shows that product's functional characteristic plays a more crucial role in modeling user preference for non-visual products (*i.e.*, Office Product and Automotive).

VPMF, FPMF *vs.* VFPMF. VFPMF consistently outperforms both VPMF and FPMF in visually-aware and visually non-aware product domain. This can be explained that our method utilizes product's visual and functional characteristics simultaneously to better model the product latent feature space.

ACF *vs.* VFPMF. VFPMF considerably outperforms ACF. For example, in most cases, VFPMF can even significantly better performance by using only an embedding size of $k = 10$ as compared to ACF using $k = 30$. It further justifies that it is necessary to fully exploit product's visual and functional characteristics in a unified manner.

Note that for all results presented so far, the size of the top-N list chosen is 10 (*i.e.*, N = 10). Next, we evaluate the overall performance achieved by various methods with varying the size of the top-N list. Due to the similar conclusions on four metrics, we only report the Precision and Recall of VFPMF versus baselines for different values of N (*i.e.*, 15, 20, 25 and 30) in Figure 4. From the figure, the main findings are summarized as follows:

- All methods perform consistently with different metrics. Precisely, as N increases, the Recall increases while the Precision decreases.
- WRMF has significant improvement over BPR, which contributes to the involvement of all missing data for optimizing the ranking problem.
- In visually-aware product domain, VPMF is much superior than WRMF. This further clarifies that the user decision making process is more influenced by product's visual characteristic.
- VFPMF achieves the best performance in all datasets with two metrics, indicating that our model tends to more relevant products higher than the other baselines.

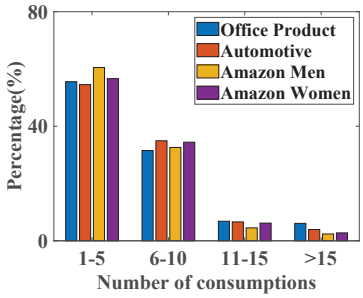


Fig. 5: User distribution.

5.3 Performance under Different Data Sparsity (RQ2)

In the above subsection, we reveal VFPMF overall outperforms WRMF method from the results, but it is not sure

TABLE 4: Comparison between VFPMF and WRMF for a candidate user in Office Product data. On the left, we show a subset of 15 preferred training products for the user. The number in the parenthesis after the product title is the number of users who have bought this product in the training set. The top-10 recommendations by WRMF and VFPMF are shown as the middle column. Bolded entries are actually bought by this user in the test set, which is shown as the last column.

User's purchase history	WRMF recommendations	VFPMF recommendations	Preferred test products
• Shipping Labels (135)	• Pendaflex File Folders (249)	• **Self-Stick Note Pad Holders(143)**	• Scotch Magic Tape(146)
• Smart Notebook (94)	• Survivor R1460 Tyvek Mailer (212)	• Reveal-N-Seal Envelope (66)	• Self-Stick Note Pad Holders(143)
• Rectangle Address Labels (91)	• Postal Scales (155)	• **Mesh Desk Organizer(15)**	• Mesh Desk Organizer(15)
• File Folder Labels (84)	• AmazonBasics Hanging File Folders (206)	• Office Tape Dispensers (55)	
• Pencil Sharpener (68)	• SuperTab Heavyweight File Folder (139)	• Scotch Gift Wrap Tape (122)	
• Ballpoint Pens (61)	• **Scotch Magic Tape(146)**	• Desktop Staplers (56)	
• Mechanical Pencils (55)	• Removable Label Padst (166)	• **Scotch Magic Tape(146)**	
• White Pricing Labels (36)	• Scotch Gift Wrap Tape (122)	• Back & Seat Cushions (10)	
• Dry Erase Learning Board (8)	• Wolf Beaded Bookmark (111)	• Triangular Scales (7)	
• Jet-Bow Speed Compass (3)	• Reveal-N-Seal Envelope (66)	• Lead Refills (3)	

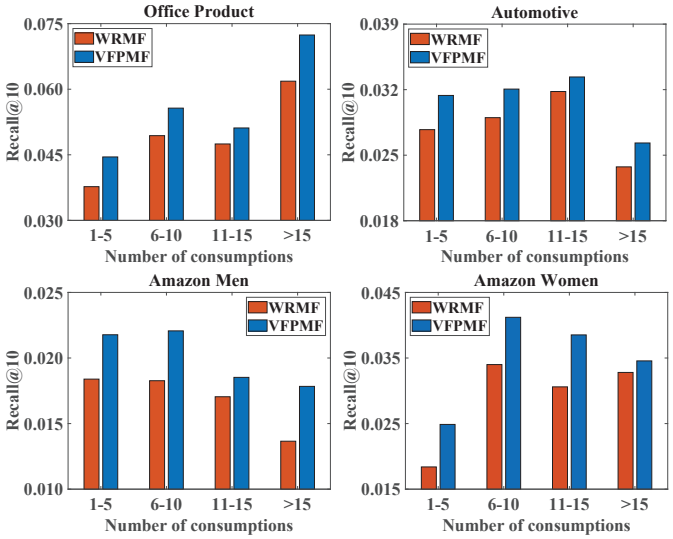


Fig. 6: Performance analysis *w.r.t.* users with different scale of consumption records.

whether VFPMF is consistently superior to WRMF on users and products of different sparsity levels. In order to answer this question, we first group all users into several segments based on their consumptions in the training dataset, and then measure the performance over each segment. Users are grouped into 4 segments: "[1, 5]", "[6, 10]", "[11, 15]", and ">15". Figure 5 summarizes the user segment distribution of the training data. Clearly, These datasets are extremely sparse since more than 55% of users have only a few consumptions (1-5). Figure 6 illustrates the experimental results of both methods in terms of Recall@10. Results for other metrics show similar conclusions but is omitted for space reasons. From the figure, we have the following observations:

- VFPMF clearly outperforms WRMF for inactive user (the leftmost bar pairs in each plot). VFPMF explicitly makes use of the additional information associated with the products (*a.k.a.* the intrinsic and extrinsic characteristics), which has a great influence on alleviating the data sparsity problem.
- With the increasing of the number of consumptions for users, the performance of both methods is getting closer, but VFPMF consistently performs better than WRMF. This demonstrates that the means of integrating visual and functional characteristics into matrix factorization model is realistic and reasonable.

Apart from the superior recommendation performance on users of different sparsity levels, another key advantage of VFPMF is its ability in alleviating the long-tail problem (see section 5.1). To demonstrate this, we now provide a qualitative evaluation of the aforementioned accuracy improvement. On the Office Product dataset, we compare the top 10 recommendations generated by WRMF and VFPMF for a candidate user who has purchased a series of popular products (*e.g.*, "Shipping Labels" and " Smart Notebook"), as well as some rare office products (*e.g.*, "Dry Erase Learning Board" and " Jet-Bow Speed Compass"). From Table 4, we summarize the main findings as follows:

- For the candidate user, precisions of the top-10 products for WRMF and VFPMF are 10% and 30%, respectively.
- WRMF highly ranks popular products and places them on top of its recommendations. By contrast, VFPMF successfully balances between the popular products and relevant (but comparatively rare) products in its recommendations.

With the evidence from Figure 6 and Table 4, we can draw an answer to the second question — VFPMF is resistant to the data sparsity and long-tail problems by exploiting product's visual and functional characteristics simultaneously.

5.4 Training Efficiency (RQ3)

Analytically, the time complexity of ALS-based WRMF method is $O((m+n)k^3 + |\mathcal{R}|k^2)$, while FALS's time complexity is $O((|\mathcal{R}|+|\mathcal{S}|+|\mathcal{F}|)k+(m+n)k^2)$. To demonstrate the efficiency of FALS, we show the training time per epoch in Table 5. It is of interest to see that, though using the additional information associated with the products, VFPMF takes much less time than WRMF. The efficiency of VFPMF is owing to its fast learning algorithm. Specifically, when k is 512, WRMF requires 2.32 hours for one epoch on Office Product, while VFPMF only takes 12.83 minutes. Hence, VFPMF is capable of attaining an optimal balance between accuracy and runtime.

TABLE 5: Training time per epoch of different MF methods with varying k.

	Office Product		Automotive		Amazon Men		Amazon Women	
k	ALS	FALS	ALS	FALS	ALS	FALS	ALS	FALS
32	1m8s	38s	4m40s	2m23s	2m24s	1m24s	34s	20s
64	2m12s	1m16s	16m58s	5m19s	5m27s	3m8s	1m18s	47s
128	5m2s	2m34s	38m45s	11m59s	22m19s	6m17s	3m32s	1m53s
256	11m35s	6m54s	1h25m38s	23m43s	1h6m26s	29m3s	15m31s	7m6s
512	2h19s	12m50s	6h181m4s	1h42m8s	4h6m57s	50m1s	1h29m12s	14m34s
1024	13h9m50s	1h38m21s	42h24m55s	6h12m44s	29h1m59s	3h59m45s	10h22m20s	1h12m48s

h, m, and s denote hours, minutes and seconds, respectively.

5.5 Incremental Updates for Online Environment (RQ4)

To study the effectiveness of incremental updates, we used the latest 10% consumption records as the test data, the remaining 90% data for training with the optimal hyper-parameter settings demonstrated by the offline evaluation. In this section, we study the influence of the number of online epochs required for FALS to convergence on four real-world datasets. Due to the limited space and the similar conclusions in different metrics, we only report the recommendation performance of online learning algorithm in terms of NDCG@10 and Recall@10. Results at 0^{th} epoch show the accuracy of offline learning. From Figure 7, we summarize the main observations as follows:

- The accuracy of offline trained model is notably worse than the corresponding accuracy of incremental updates. This indicates that it's very important to refresh offline trained model for online environment.
- After the first iteration, the performance is incredibly improved. With more epoches, the accuracy is not further improved.

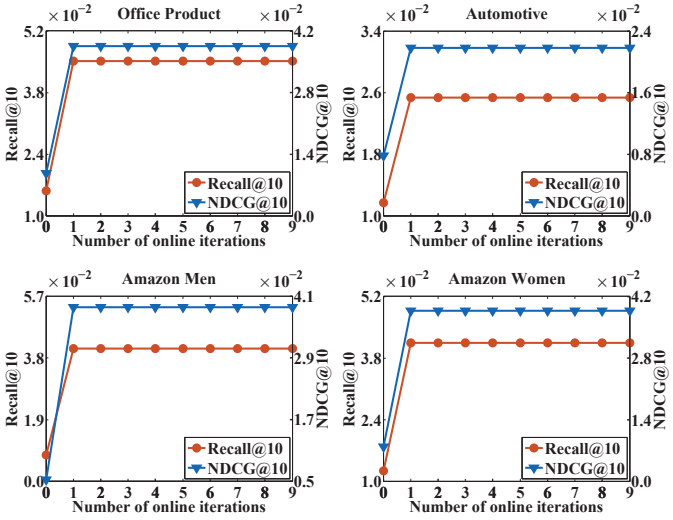


Fig. 7: Impact of online iteration on FALS($k = 10$).

6 Conclusion

In this paper, a novel probabilistic model (VFPMF) was proposed to exploit both product's visual and functional characteristics for product recommendation. In sharp contrast to previous work in which SGD or ALS was applied to learn parameters of MF models, we designed an efficient learning algorithm (FALS) for optimizing the proposed model and proved its convergence in theory. To support online learning, we designed an online-update strategy to adapt FALS to real-time streaming data. To evaluate our proposed method in both offline and online protocols, we conducted extensive experiments with four evaluation metrics and a series of state-of-the-art methods in different product domains. Experimental results have demonstrated both the effectiveness and efficiency of our method. One insight is that, we found the visual characteristic was very important in visually-aware product domain, while the functional characteristic played a more crucial role in visually non-aware product domain.

As a future direction, we plan to examine parallel implementation of our algorithm so that it can be attractive for large-scale industrial deployment. We also plan to explore the utility of social network [53], [54], [55], [56] and textual reviews [57], [58] for understanding user decision making process. For instance, by integrating social network with VFPMF, we can study how social influence affects user preference; by fusing textual reviews, we can build a more effective and explainable recommender system.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (61772475, 61502434, 61532018, U19A2079, 61972069, 61836007, 61832017), the Thousand Youth Talents Program 2018, and the Program for Science and Technology in Henan Province of China under Grant No. 172102210011. This work is also supported by the National Research Foundation, Prime Minister's Office, Singapore under its IRC@Singapore Funding Initiative.

Appendix A

Lemma 1. *Assume FALS generates a sequence of* $\{\boldsymbol{W},\boldsymbol{H},\boldsymbol{Z},\boldsymbol{E}\}$ *tuples:*

$$\begin{aligned}&\{\boldsymbol{W}^{(1)},\boldsymbol{H}^{(1)},\boldsymbol{Z}^{(1)},\boldsymbol{E}^{(1)}\},\{\boldsymbol{W}^{(2)},\boldsymbol{H}^{(1)},\boldsymbol{Z}^{(1)},\boldsymbol{E}^{(1)}\},\\&\{\boldsymbol{W}^{(2)},\boldsymbol{H}^{(2)},\boldsymbol{Z}^{(1)},\boldsymbol{E}^{(1)}\},\{\boldsymbol{W}^{(2)},\boldsymbol{H}^{(2)},\boldsymbol{Z}^{(2)},\boldsymbol{E}^{(1)}\},\\&\{\boldsymbol{W}^{(2)},\boldsymbol{H}^{(2)},\boldsymbol{Z}^{(2)},\boldsymbol{E}^{(2)}\},\dots\end{aligned} \tag{24}$$

Suppose there exists a limit point $\{\bar{\boldsymbol{W}},\bar{\boldsymbol{H}},\bar{\boldsymbol{Z}},\bar{\boldsymbol{E}}\}$*. Then the function value converges to some value:*

$$\lim_{t\to\infty}\mathcal{L}(\boldsymbol{W}^{(t)},\boldsymbol{H}^{(t)},\boldsymbol{Z}^{(t)},\boldsymbol{E}^{(t)})=\mathcal{L}(\bar{\boldsymbol{W}},\bar{\boldsymbol{H}},\bar{\boldsymbol{Z}},\bar{\boldsymbol{E}}). \tag{25}$$

Proof: It is easy to see that $\{\boldsymbol{W}^{(t)},\boldsymbol{H}^{(t)},\boldsymbol{Z}^{(t)},\boldsymbol{E}^{(t)}\}$ generated by Algorithm 1 Step 10, 19, 24 and 25 monotonically decreases the objective function in Eq. (8).

Theorem 1. *if a sequence* $\{\boldsymbol{W}^{(t)},\boldsymbol{H}^{(t)},\boldsymbol{Z}^{(t)},\boldsymbol{E}^{(t)}\}$ *is generated by FALS, then every limit point of this sequence is a stationary point.*

Proof: Eq. (8) is denoted by $\mathcal{L}(a)$, where a represents $\{w_{11}, \ldots, w_{ud}, \ldots\}$. It is very easy to validate that w_{ud} can be rewritten as:

$$a^{(t+1)} = a^{(t)} - \frac{\mathcal{L}'(a^{(t)})}{\mathcal{L}''(a^{(t)})}, t = 0, 1, 2, \ldots \quad (26)$$

The partial derivative and the second derivative are:

$$\mathcal{L}'(a^{(t)}) = \frac{\partial \mathcal{L}}{\partial a}, \mathcal{L}''(a^{(t)}) = \lambda_w + \sum\nolimits_{j=1}^{n} C_{uj}^r h_{jd}^2 > 0. \quad (27)$$

Since $\mathcal{L}''(a^{(t)})$ is positive, based on the Lagrange mean value theorem, we have that $\mathcal{L}'(a^{(t)})$ is Lipchitz function. Hence,

$$\mathcal{L}'(a^+) \leq \mathcal{L}'(a) + (a^+ - a)\mathcal{L}''(a), \forall a, a^+ \geq 0. \quad (28)$$

By setting $a^+ = a - \mathcal{L}'(a)/\mathcal{L}''(a)$ we have $\mathcal{L}'(a - \mathcal{L}'(a)/\mathcal{L}''(a)) \leq 0$. Moreover, from Eq. (27)we have $\mathcal{L}'(0) \leq 0$. Therefore after one step update, the sequence $\{a^{(t)}\}_{t\geq 1}$ has the property that $\mathcal{L}'(a^{(t)}) < 0$ and $-\mathcal{L}'(a^{(t)})/\mathcal{L}''(a^{(t)}) > 0$. Thus the sequence $\{a^{(t)}\}_{t\geq 1}$ is monotonically increasing in a compact set $\mathcal{L}'(a) \leq 0$. $\{a^{(t)}\}$ then converges to a limit point, say a^*. $\mathcal{L}'(a^*)$ can be either zero or negative. If $\mathcal{L}'(a^*) < 0$ and $t \to \infty$, we have the following two properties by continuity of first and second derivatives:

$$\frac{\mathcal{L}'(a^{(t)})}{\mathcal{L}''(a^{(t)})} < \frac{\mathcal{L}'(a^*)}{2\mathcal{L}''(a^*)}, \quad (29)$$

$$a^* - a^{(t)} < \frac{\mathcal{L}'(a^*)}{-2\mathcal{L}''(a^*)}. \quad (30)$$

By combining these two equations we have $a^* - a^{(t+1)} = a^* - a^{(t)} + \mathcal{L}'(a^t)/\mathcal{L}''(a^{(t)}) < 0$, which raises the contradiction with $a^{(t+1)} - a^*$. The algorithm converges to the optimal solution $\mathcal{L}'(a^*) = 0$. Similarly, we have $\mathcal{L}'(h_{jd}^*) = 0$, $\mathcal{L}'(z_{id}^*) = 0$ and $\mathcal{L}'(e_{td}^*) = 0$. Therefore, the theorem is proved. Because $\mathcal{L}$ is not jointly convex *w.r.t.* $\boldsymbol{W}, \boldsymbol{H}, \boldsymbol{Z}$ and $\boldsymbol{E}$, the solution is a local optimum. □

Theorem 2. *The time complexity is linear in the number of interactions, intrinsic and extrinsic characteristics.*

Proof: The training time of Algorithm 1 is mainly taken by updating various variables. The computational complexities for updating $\boldsymbol{W}, \boldsymbol{H}, \boldsymbol{Z}$ and $\boldsymbol{E}$ are $O(mk^2 + k|\mathcal{R}|)$, $O(nk^2 + |\mathcal{R}|k + |\mathcal{S}|k)$, $O(nk^2 + |\mathcal{S}|k)$ and $O(|\mathcal{F}|k)$. Hence, the overall time complexity per epoch is $O((m+n)k^2 + (|\mathcal{R}| + |\mathcal{S}| + |\mathcal{F}|)k)$. Suppose the entire algorithm requires $\#it$ iterations for convergence, the overall time complexity of the learning algorithm is $O(\#it((m+n)k^2 + (|\mathcal{R}| + |\mathcal{S}| + |\mathcal{F}|)k))$. It follows that our algorithm has the potential to be scaled up to large-scale datasets. □

REFERENCES

[1] Y. Ren, M. Tomko, F. Salim, J. Chan, C. L. A. Clarke, and M. Sanderson, "A location-query-browse graph for contextual recommendation," *IEEE TKDE*, vol. 30, no. 2, pp. 204–218, 2018.

[2] H. Zhang, F. Shen, W. Liu, X. He, H. Luan, and T.-S. Chua, "Discrete collaborative filtering," in *SIGIR*, 2016, pp. 325–334.

[3] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua, "Fast matrix factorization for online recommendation with implicit feedback," in *SIGIR*, 2016, pp. 549–558.

[4] B. Wu, X. He, Z. Sun, L. Chen, and Y. Ye, "ATM: an attentive translation model for next-item recommendation," *IEEE TII*, vol. 16, no. 3, pp. 1448–1459, 2020.

[5] L. Wu, Q. Liu, E. Chen, N. Yuan, G. Guo, and X. Xie, "Relevance meets coverage: a unified framework to generate diversified recommendations," *ACM TIST*, vol. 7, no. 3, pp. 1–30, 2016.

[6] H. Wang, F. Zhang, X. Xie, and M. Guo, "Dkn: deep knowledge-aware network for news recommendation," in *WWW*, 2018, pp. 1835–1844.

[7] D. Lian, Y. Ge, F. Zhang, N. Yuan, X. Xie, T. Zhou, and Y. Rui, "Scalable content-aware collaborative filtering for location recommendation," *IEEE TKDE*, vol. 30, no. 6, pp. 1122–1135, 2018.

[8] H. Lee, "Intrinsic and extrinsic motivations affecting impulse-buying tendency in mobile shopping," *SBP Journal*, vol. 46, no. 4, pp. 683–694, 2018.

[9] Z. Wang, Z. Jiang, Z. Ren, J. Tang, and D. Yin, "A path-constrained framework for discriminating substitutable and complementary products in e-commerce," in *WWW*, 2018, pp. 619–627.

[10] R. He and J. McAuley, "Vbpr: visual bayesian personalized ranking from implicit feedback," in *AAAI*, 2016, pp. 144–150.

[11] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, and T.-S. Chua, "Attentive collaborative filtering: multimedia recommendation with item- and component-level attention," in *SIGIR*, 2017, pp. 335–344.

[12] W. Niu, J. Caverlee, and H. Lu, "Neural personalized ranking for image recommendation," in *WSDM*, 2018, pp. 423–431.

[13] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *ICDM*, 2008, pp. 263–272.

[14] D. Liang, J. Altosaar, L. Charlin, and D. M. Blei, "Factorization meets the item embedding: regularizing matrix factorization with item co-occurrence," in *RecSys*, 2016, pp. 59–66.

[15] J. Yang, C. Liu, M. Teng, J. Chen, and H. Xiong, "A unified view of social and temporal modeling for b2b marketing campaign recommendation," *IEEE TKDE*, vol. 30, no. 5, pp. 810–823, 2018.

[16] L. Wu, Q. Liu, R. Hong, E. Chen, Y. Ge, X. Xie, and M. Wang, "Product adoption rate prediction in a competitive market," *IEEE TKDE*, vol. 30, no. 2, pp. 325–338, 2018.

[17] I. Bayer, X. He, B. Kanagal, and S. Rendle, "A generic coordinate descent framework for learning from implicit feedback," in *WWW*, 2017, pp. 1341–1350.

[18] J. Ding, F. Feng, X. He, G. Yu, Y. Li, and D. Jin, "An improved sampler for bayesian personalized ranking by leveraging view data," in *WWW*, 2018, pp. 13–14.

[19] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. Chua, "Neural collaborative filtering," in *WWW*, 2017, pp. 173–182.

[20] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: bayesian personalized ranking from implicit feedback," in *UAI*, 2009, pp. 452–461.

[21] X. Wang, X. He, L. Nie, and T.-S. Chua, "Item silk road: recommending items from information domains to social users," in *SIGIR*, 2017, pp. 185–194.

[22] H. Li, Y. Ge, D. Lian, and H. Liu, "Learning users intrinsic and extrinsic interests for point-of-interest recommendation: A unified approach," in *IJCAI*, 2017, pp. 2117–2123.

[23] H. Ma, H. Yang, M. R. Lyu, and I. King, "Sorec: social recommendation using probabilistic matrix factorization," in *CIKM*, 2008, pp. 931–940.

[24] H. Ma, T. C. Zhou, M. R. Lyu, and I. King, "Improving recommender systems by incorporating social contextual information," *ACM TOIS*, vol. 29, no. 2, pp. 9:1–9:23, 2011.

[25] M. Jiang, P. Cui, F. Wang, W. Zhu, and S. Yang, "Scalable recommendation with social contextual information," *IEEE TKDE*, vol. 26, no. 11, pp. 2789–2802, 2014.

[26] J. Tang, X. Hu, and H. Liu, "Social recommendation: a review," *Social Netw. Analys. Mining*, vol. 3, no. 4, pp. 1113–1133, 2013.

[27] L. Wu, L. Chen, R. Hong, Y. Fu, X. Xie, and M. Wang, "A hierarchical attention model for social contextual image recommendation," *IEEE TKDE*, pp. 1–1, 2019.

[28] R. Yin, K. Li, J. Lu, and G. Zhang, "Enhancing fashion recommendation with visual compatibility relationship," in *WWW*, 2019, pp. 3434–3440.

[29] M. Jian, T. Jia, X. Yang, L. Wu, and L. Huo, "Cross-modal collaborative manifold propagation for image recommendation," in *ICMR*, 2019, pp. 344–348.

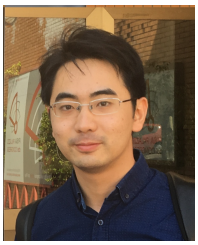
[30] W. Kang, C. Fang, Z. Wang, and J. McAuley, "Visually-aware fashion recommendation and design with generative image models," in *ICDM*, 2017, pp. 207–216.

[31] Q. Liu, S. Wu, and L. Wang, "Deepstyle: learning user preferences for visual recommendation," in *SIGIR*, 2017, pp. 841–844.

[32] W. Yu, H. Zhang, X. He, X. Chen, L. Xiong, and Z. Qin, "Aesthetic-based clothing recommendation," in *WWW*, 2018, pp. 649–658.

[33] S. Wang, Y. Wang, J. Tang, K. Shu, S. Ranganath, and H. Liu, "What your images reveal: exploiting visual contents for point-of-interest recommendation," in *WWW*, 2017, pp. 391–400.
[34] S. Rendle and L. S.-Thieme, "Online-updating regularized kernel matrix factorization models for large-scale recommender systems," in *RecSys*, 2008, pp. 251–258.
[35] J. Zhang, C. Chow, and J. Xu, "Enabling kernel-based attribute-aware matrix factorization for rating prediction," *IEEE TKDE*, vol. 29, no. 4, pp. 798–812, 2017.
[36] G. Ling, H. Y. andI. King, and M. R. Lyu, "Online learning for collaborative filtering," in *IEEE WCCI*, 2012, pp. 1–8.
[37] P. Rebecca and B. Martin, "Effects of intrinsic and extrinsic motivation on user-generated content," *Journal of Strategic Marketing*, vol. 23, no. 4, pp. 305–317, 2015.
[38] R. Kuvykaite, A. Dovaliene, and L. Navickiene, "Impact of package elements on consumer's purchase decision," *Economics and management*, vol. 14, pp. 441–447, 2009.
[39] T. Chen, X. He, and M.-Y. Kan, "Context-aware image tweet modelling and recommendation," in *ACM MM*, 2016, pp. 1018–1027.
[40] X. Geng, H. Zhang, J. Bian, and T.-S. Chua, "Learning image and user features for recommendation in social networks," in *ICCV*, 2015, pp. 4274–4282.
[41] R. He, C. Lin, J. Wang, and J. McAuley, "Sherlock: sparse hierarchical embeddings for visually-aware one-class collaborative filtering," in *IJCAI*, 2016, pp. 3740–3746.
[42] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell, "Caffe: convolutional architecture for fast feature embedding," in *ACM MM*, 2014, pp. 675–678.
[43] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *NIPS*, 2008, pp. 1257–1264.
[44] B. Wu, Y. Ye, and Y. Chen, "Visual appearance or functional complementarity: Which aspect affects your decision making?" *Inf. Sci.*, vol. 476, pp. 19–37, 2019.
[45] S. Zhang, L. Yao, B. Wu, X. Xu, X. Zhang, and L. Zhu, "Unraveling metric vector spaces with factorization for recommendation," *IEEE TII*, vol. 16, no. 2, pp. 732–742, 2020.
[46] Z. Sun, B. Wu, Y. Wu, and Y. Ye, "APL: adversarial pairwise learning for recommender systems," *Expert Syst. Appl.*, vol. 118, pp. 573–584, 2019.
[47] F. Cai and M. de Rijke, "A survey of query auto completion in information retrieval," *FnTIR*, vol. 10, no. 4, pp. 273–363, 2016.
[48] F. Cai, R. Reinanda, and M. de Rijke, "Diversifying query auto-completion," *ACM TOIS*, vol. 34, no. 4, pp. 25:1–25:33, 2016.
[49] S. Rendle and C. Freudenthaler, "Improving pairwise learning for item recommendation from implicit feedback," in *WSDM*, 2014, pp. 273–282.
[50] J. Weston, S. Bengio, and N. Usunier, "Wsabie: scaling up to large vocabulary image annotation," in *IJCAI*, 2011, pp. 2764–2770.
[51] G. Guo, J. Zhang, Z. Sun, and N. Yorke-Smith, "Librec: A java library for recommender systems," in *In Posters, Demos, Late-breaking Results and Workshop Proceedings of the 23rd International Conference on User Modeling, Adaptation and Personalization*, 2015.
[52] T. Zhao, J. McAuley, and N. King, "Improving latent factor models via personalized feature projection for one class recommendation," in *CIKM*, 2015, pp. 821–830.
[53] L. Wu, Y. Ge, Q. Liu, E. Chen, R. Hong, J. Du, and M. Wang, "Modeling the evolution of users' preferences and social links in social networking services," *IEEE TKDE*, vol. 29, no. 6, pp. 1240–1253, 2017.
[54] S. Yan, K. Lin, X. Zheng, W. Zhang, and X. Feng, "An approach for building efficient and accurate social recommender systems using individual relationship networks," *IEEE TKDE*, vol. 29, no. 10, pp. 2086–2099, 2017.
[55] G. Guo, J. Zhang, and N. Yorke-Smith, "A novel recommendation model regularized with user trust and item ratings," *IEEE TKDE*, vol. 28, no. 7, pp. 1607–1620, 2016.
[56] X. Lin, M. Zhang, Y. Zhang, Y. Liu, and S. Ma, "Learning and transferring social and item visibilities for personalized recommendation," in *CIKM*, 2017, pp. 337–346.
[57] Y. Zhang, Q. Ai, X. Chen, and W. Croft, "Joint representation learning for top-n recommendation with heterogeneous information sources," in *CIKM*, 2017, pp. 1449–1458.
[58] W. Zhao, J. Wang, Y. He, J. Wen, E. Chang, and X. Li, "Mining product adopter information from online reviews for improving product recommendation," *ACM TKDD*, vol. 10, no. 3, pp. 1–23, 2016.

Bin Wu is currently a PhD student with the School of Information Engineering at Zhengzhou University of China (ZZU). He received the MS degree in Computer Science from ZZU. His research interests include machine learning, recommender systems, social network analysis, and multimedia.



Xiangnan He is currently a senior research fellow with School of Computing, National University of Singapore (NUS). He received his Ph.D. in Computer Science from NUS. His research interests span recommender system, information retrieval, and multi-media processing. He has over 40 publications appeared in several top conferences such as SIGIR, WWW, MM, CIKM, and IJCAI, and journals including TKDE, TOIS, and TMM. His work on recommender system has received the Best Paper Award Honorable Mention of ACM SIGIR 2016 and WWW 2018. Moreover, he has served as the PC member for the prestigious conferences including SIGIR, WWW, MM, KDD, and IJCAI, and the regular reviewer for prestigious journals including TKDE, TOIS, TKDD, TMM etc.

Yun Chen is currently working toward the master's degree from the School of Information Engineering at Zhengzhou University of China. Her areas of interest include recommender systems, data mining, and information retrieval.

Liqiang Nie is currently a professor with the School of Computer Science and Technology, Shandong University. Meanwhile, he is the adjunct dean with the Shandong AI institute. He received his B.Eng. and Ph.D. degree from Xi'an Jiaotong University in July 2009 and National University of Singapore (NUS) in 2013, respectively. After PhD, Dr. Nie continued his research in NUS as a research follow for more than three years. His research interests lie primarily in information retrieval, multimedia computing and their applications in the field of healthcare. Dr. Nie has co-/authored more than 100 papers, received more than 2000 Google Scholar citations as of May 2018. He is an AE of Information Science, an area chair of ACM MM 2018, a special session chair of PCM 2018, a PC chair of ICIMCS 2017. Meanwhile, he is supported by the program of "Thousand Youth Talents Plan 2016" and "Qilu Scholar 2016". In 2017, he co-founded "Qilu Intelligent Media Forum".

Kai Zheng is a Professor of Computer Science with University of Electronic Science and Technology of China. He received his PhD degree in Computer Science from The University of Queensland in 2012. He has been working in the area of spatial-temporal databases, uncertain databases, social-media analysis, in-memory computing and blockchain technologies.

Yangdong Ye is a Professor in Zhengzhou University of School of Information Engineering. He received his Ph.D. degree in China Academy of Railway Sciences. He worked one year as a senior visiting scholar in Deakin University, Australia. His research interests include machine learning, pattern recognition, knowledge engineering, and intelligent system. He has published some papers in peer-reviewed prestigious journals and conferences, such as IEEE TMM, IEEE MultiMedia, Neural Networks, CVPR, IJCAI and ACM MM.