Bayesian Personalized Feature Interaction Selection for Factorization Machines

Yifan Chen*
University of Amsterdam
Amsterdam, The Netherlands
y.chen4@uva.nl

Pengjie Ren
University of Amsterdam
Amsterdam, The Netherlands
p.ren@uva.nl

Yang Wang†
Hefei University of Technology
Hefei, China
yangwang@hfut.edu.cn

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands
derijke@uva.nl

ABSTRACT

Factorization Machines (FMs) are widely used for feature-based collaborative filtering tasks, as they are very effective at modeling feature interactions. Existing FM-based methods usually take all feature interactions into account, which is unreasonable because not all feature interactions are helpful: incorporating useless feature interactions will introduce noise and degrade the recommendation performance. Recently, methods that perform Feature Interaction Selection (FIS) have attracted attention because of their effectiveness at filtering out useless feature interactions. However, they assume that all users share the same feature interactions, which is not necessarily true, especially for collaborative filtering tasks. In this work, we address this issue and study Personalized Feature Interaction Selection (P-FIS) by proposing a Bayesian Personalized Feature Interaction Selection (BP-FIS) mechanism under the Bayesian Variable Selection (BVS) theory. Specifically, we first introduce interaction selection variables with hereditary spike and slab priors for P-FIS. Then, we form a Bayesian generative model and derive the Evidence Lower Bound (ELBO), which can be optimized by an efficient Stochastic Gradient Variational Bayes (SGVB) method to learn the parameters. Finally, because BP-FIS can be seamlessly integrated with different variants of FMs, we implement two FM variants under the proposed BP-FIS. We carry out experiments on three benchmark datasets. The empirical results demonstrate the effectiveness of BP-FIS for selecting personalized interactions and improving the recommendation performance.

CCS CONCEPTS

• **Information systems → Recommender systems**;

*Also with Key lab of Information System Engineering, National University of Defense Technology, Changsha, China.
†Corresponding author.

KEYWORDS

Personalized Feature Interaction Selection; Bayesian Variable Selection; Factorization Machines; Recommender Systems

ACM Reference Format:
Yifan Chen, Pengjie Ren, Yang Wang, and Maarten de Rijke. 2019. Bayesian Personalized Feature Interaction Selection for Factorization Machines. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3331184.3331196

1 INTRODUCTION

Factorization Machines (FMs) [39, 40] are a generic supervised learning approach that combines the advantage of Support Vector Machines (SVMs) [48] with factorization models [25]. FMs account for interactions between features with factorized parameters [5, 46]. Feature interactions are crafted as combinations of individual features [12]. For example, in the movie recommendation scenario, the features for the movie "Spider-Man" can be "*comics*", "*marvel*" and "*avengers*". Accordingly, feature interactions can be combinations such as, e.g., "(*comics*, *marvel*)", "(*comics*, *avengers*)", etc.

FMs are widely applied in predictive analytics, ranging from recommendation [41], computational advertising [18], to search ranking [30] and toxicogenomics prediction [59]. FMs have been well studied for recommendations, due to their effective use of historical interactions between users and items [39]. FMs can incorporate additional information associated with users or items, including content descriptions [60], context information [43], social networks [7, 10], sequential dependencies [26]. While the wide availability of auxiliary data provides rich sources that may help reveal user preferences, they also increase the dimensionality of the feature space [9]. The problem of high-dimensionality is particularly severe for FMs, because FMs consider feature interactions. Hence, the complexity of FM models grows exponentially with the order of feature interactions. But not all feature interactions are helpful [11]; incorporating unnecessary feature interactions may bring in noise, which adversely impacts the recommendation accuracy and increases the difficulty of interpreting outcomes [58].

Feature Interaction Selection (FIS) has been proposed to select useful feature interactions and filter out useless feature interactions. Existing FIS methods can be divided into two classes: *wrapper methods* [31] and *embedded methods* [11, 58]. Wrapper methods evaluate

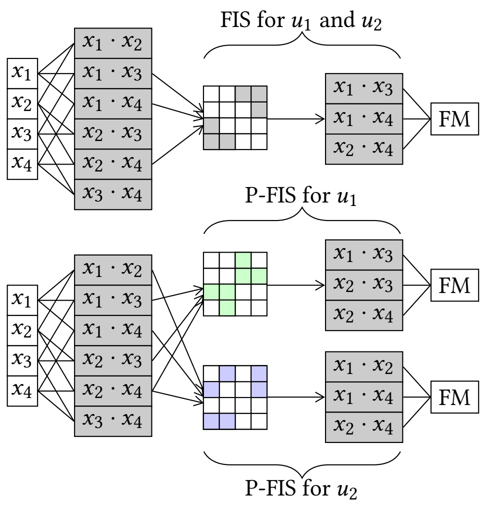

Figure 1: Feature Interaction Selection and Personalized Feature Interaction Selection. $x_1, \ldots, x_4$ are features and $x_1 \cdot x_2, \ldots, x_3 \cdot x_4$ are feature interactions. The 4×4 matrices indicate masks for the selection of feature interactions. $x_i \cdot x_j$ will pass through the grid in the i-th row and j-th column or the j-th row and i-th column. The white grids of the matrices filter out feature interactions. FIS selects identical feature interactions for u_1 and u_2 while P-FIS selects feature interactions for u_1 and u_2 separately.

subsets of feature interactions by training a model with each subset and scoring on a held-out set. Although wrapper methods are more flexible as they do not depend on the recommendation model to use, they suffer from poor scalability. Embedded methods perform interaction selection during model training, which is more efficient and effective. Sparse Factorization Machines (SFMs) [37, 58, 62] are an example of embedded methods; they utilize sparsity regularization [49, 53] to achieve FIS. Existing FIS-based methods, including SFMs, overwhelmingly select a common set of interactions for all users non-personally, on the assumption that the same feature interactions are equally powerful to predict user's behavior. This assumption may not be valid, as it overlooks the individuality and personality of user's behavior, especially for recommendation tasks.

We introduce and study *Personalized Feature Interaction Selection* (P-FIS). As shown in Figure 1, unlike FIS, P-FIS aims to achieve adaptive FIS for each individual user. P-FIS is a more challenging task than non-personalized FIS since we have a limited number of instances associated with each user to select user-specific feature interactions. Although we can train a particular Sparse Factorization Machine for each user, this is problematic for at least two reasons. First, it is both time and storage inefficient since we would need to maintain a model for each user. Second, it is less effective because estimating a model separately for each user fails to take advantage of collaborative filtering. To this end, we propose a *Bayesian Personalized Feature Interaction Selection* (BP-FIS) mechanism for FMs. First, instead of learning expensive and less effective personalized feature embeddings for each user, we estimate a single set of feature embeddings shared by all users to preserve the advantage of collaborative filtering. We achieve P-FIS by introducing personalized interaction selection variables and employ *Bayesian Variable Selection* (BVS) to estimate the selection variables, which allows us to avoid expensive cross-validation required by sparsity regularizations [49]. The widely used sparsity priors [3] for BVS are not ideal since they assign zero probability mass to events associated with weights having zero value [50]. Instead, we propose a Hereditary Spike-and-Slab Prior (HSSP), a variant of the commonly used *spike and slab priors* in BVS [2, 15, 34]. We formulate the BP-FIS as a probabilistic hierarchical generation procedure and derive the Evidence Lower Bound (ELBO). Inspired by Variational Auto-Encoders (VAEs) [23, 28], we use a Stochastic Gradient Variational Bayes (SGVB) estimator to approximate posteriors of the latent variables and propose an efficient algorithm to optimize the model. BP-FIS can be seamlessly integrated into both traditional FMs (linear) and neural FMs (nonlinear).

We summarize the contributions of this paper as follows:

- To the best of our knowledge, we are the first to study *Personalized Feature Interaction Selection* (P-FIS) for FMs to improve recommendation performance.
- We propose *hereditary spike and slab priors* to assign non-zero probabilities to zero values while preserving hereditary relations.
- We formulate the BP-FIS task as a probabilistic hierarchical generation procedure and conduct variational inference to derive the ELBO for optimization.
- We implement two FM variants under our proposed *Bayesian Personalized Feature Interaction Selection* (BP-FIS) mechanism and verify their effectiveness through extensive experiments.

2 PRELIMINARIES

Table 1 summarizes the notation used in this paper.

2.1 Factorization machines

In the recommendation scenario, FMs try to predict the rating of an item based on its feature vector $\boldsymbol{x} \in \mathbb{R}^d$. A general formulation is shown in Eq. (1):

$$\hat{r}(\boldsymbol{x}) = b_0 + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d} \sum_{j=i+1}^{d} w_{ij} x_i x_j, \tag{1}$$

where $\hat{r}(\boldsymbol{x})$ is the predicted rating for $\boldsymbol{x}$; $x_i \in \boldsymbol{x}$ is the i-th feature of $\boldsymbol{x}$; b_0, w_i, w_{ij} are the parameters, where b_0 models the global bias, w_i models the first-order interaction, i.e., the interaction between the feature i and the target, and w_{ij} models the second-order interaction, i.e., the interaction between feature i and j. Instead of learning a fixed interaction parameter w_{ij}, FMs factorize it as $w_{ij} = \boldsymbol{v}_i^T \boldsymbol{v}_j$, where $\boldsymbol{v}_i \in \mathbb{R}^k$ is the embedding of feature i and k is the dimension of the latent space.

2.2 Bayesian variable selection

Variable selection is an important problem in statistical analysis, which selects a subset of variables that should be taken into consideration [49]. Bayesian Variable Selection (BVS) attempts to estimate the marginal posterior of a variable as the probability that the variable should be selected. Depending on the definition of priors, various Bayesian Variable Selection (BVS) methods have

Table 1: Summary of symbols and notation.

	Notation	Description
Sets and numbers	$\mathcal{U}$	Set of users
	$\mathcal{F}$	Set of features
	$\mathcal{X}$	Set of feature vectors
	$\mathcal{R}$	Set of ratings
	m	Number of users, i.e., $m = \lvert\mathcal{U}\rvert$
	d	Number of features, i.e., $d = \lvert\mathcal{F}\rvert$
	n	Number of ratings, i.e., $n = \lvert\mathcal{X}\rvert = \lvert\mathcal{R}\rvert$
	k	Dimension of feature embedding
Bayesian variables	$\boldsymbol{x} \in \mathbb{R}^d$	Feature vector
	$r(\boldsymbol{x}) \in \mathbb{R}$	Rating associated with feature $\boldsymbol{x}$
	w_{ui}	Personalized first-order feature interaction weight of user u for feature i
	w_{uij}	Personalized second-order feature interaction weight of user u between feature i and j
	W	Set of interaction weights, i.e., $W = \{w_{ui}\} \cup \{w_{uij}\}$
	$s_{ui}, \tilde{w}_i$	Variables to reformulate w_{ui}, i.e., $w_{ui} = s_{ui}\tilde{w}_i$
	$s_{uij}, \tilde{w}_{ij}$	Variables to reformulate w_{uij}, i.e., $w_{uij} = s_{uij}\tilde{w}_{ij}$
	$S, \tilde{W}$	Set of variables for reformulation, i.e., $S = \{s_{ui}\} \cup \{s_{uij}\}$ and $\tilde{W} = \{\tilde{w}_i\} \cup \{\tilde{w}_{ij}\}$
Parameters	$\boldsymbol{v}_i \in \mathbb{R}^k$	Embedding for feature i
	$V \in \mathbb{R}^{d\times k}$	Embeddings of all features
	b_u	Parameter for user bias of user u
	π	Variational parameters for S
	ρ	Variational parameters for $\tilde{W}$

been proposed [14]; Spike-and-Slab Priors (SSPs) [55] have been widely studied.

A variable w following SSP is sampled from a linear combination of two distributions:

$$w \sim \pi\mathcal{N}(\mu, \sigma^2) + (1-\pi)\delta_0,$$

where $\mathcal{N}(\mu, \sigma^2)$ is the slab prior, which is modeled using a Gaussian distribution with mean μ and variance σ^2; δ_0 is the spike prior, which is modeled using a Dirac delta mass function centered at zero. The Dirac delta function is a generalized distribution that is used to model the density of an idealized point mass as a function equal to zero everywhere except for zero and whose integral over the entire real line is equal to one. SSP can assign non-zero probability for the event $w = 0$ ($p(w = 0) = 1 - \pi$). Therefore, SSP is the ideal distribution for variable selection. However, the presence of the Dirac delta function δ_0 in the SSP complicates inference. Titsias and Lázaro-Gredilla [50] reformulate SSP as follows:

$$s \sim \mathit{Bernoulli}(\pi), \quad \tilde{w} \sim \mathcal{N}(0,1), \quad w = \tilde{w}\cdot s. \tag{2}$$

This brings two additional variables, $\tilde{w}$ and s, where $\tilde{w}$ represents the weight of the variable and s indicates whether to select the variable. The SSP in Eq. (2) is amenable to approximate inference [50].

3 MODEL DESCRIPTION

We first present a personalized FM framework. Then, we propose a Bayesian Personalized Feature Interaction Selection (BP-FIS) method

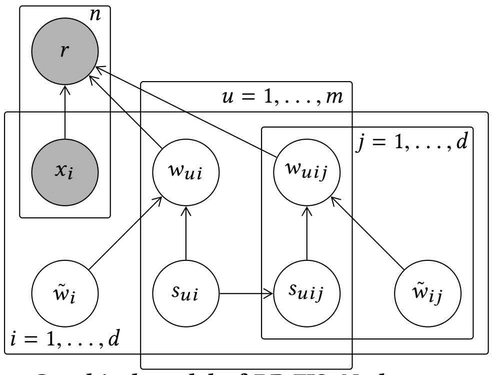

Figure 2: Graphical model of BP-FIS. Nodes represent random variables and edges represent dependencies between variables.

within this framework. To incorporate collaborative filtering into BP-FIS, we propose Hereditary Spike-and-Slab Priors (HSSPs). Finally, we present the loss function of BP-FIS by conducting variational inference.

3.1 Personalized factorization machines

To incorporate P-FIS for FMs, we reformulate Eq. (1) as a personalized FM by introducing personalized feature interaction parameters, as indicated in Eq. (3):

$$\hat{r}(\boldsymbol{x}) = b_u + \sum_{i=1}^{d} w_{ui}x_i + \sum_{i=1}^{d}\sum_{j=i+1}^{d} w_{uij}x_i x_j. \tag{3}$$

Here, b_u, $\{w_{ui}\}$, $\{w_{uij}\}$ are the personalized coefficients of user u, and w_{ui} and w_{uij} reflect the preferences of user u over first- and second-order feature interactions. While the FM in Eq. (1) can also account for personalization by taking user ID as features, it fails to personalize every interactions, which is required by FIS.

3.2 Bayesian personalized feature interaction selection

We formulate a Bayesian generation model, BP-FIS, for Eq. (3). The graphical model of BP-FIS is depicted in Figure 2. According to BP-FIS, each rating in Eq. (3) is generated with the procedure detailed in Algorithm 1. Note that we treat $\{w_{ui}\}$, $\{w_{uij}\}$ as variables and b_u as the parameter.

In the first part of the generation (line 1–10), we generate personalized feature interaction weights w_{ui} and w_{uij} for all users. We reformulate w_{ui} by $s_{ui}\cdot\tilde{w}_i$ and w_{uij} by $s_{uij}\cdot\tilde{w}_{ij}$ as suggested by [50]. Through the reformulation, we can take advantage of collaborative filtering by learning a single set of feature interaction weights $\tilde{W} = \{\tilde{w}_i\} \cup \{\tilde{w}_{ij}\}$ for all users and operationalize P-FIS by learning the personalized interaction selection variable $S = \{s_{ui}\} \cup \{s_{uij}\}$. The generation of s_{uij} is conditioned on s_{ui} and s_{uj}, which captures the hereditary relation between the first- and second-order interactions; see §3.3 for details on s_{ui} and s_{uij}.

In the second part (line 11–13), we calculate the rating prediction by Eq. (3) and generate the rating $r(\boldsymbol{x})$, following $p(r \mid \hat{r}(\boldsymbol{x}))$. The distribution of $r(\boldsymbol{x})$ determines the likelihood function for optimization. For example, if we assume $r(\boldsymbol{x})$ to follow a Gaussian

Algorithm 1 Generation procedure

```
1:  for each feature $i \in \mathcal{F}$ do
2:      draw first-order interaction weight $\tilde{w}_i \sim \mathcal{N}(0, 1)$;
3:      for each user $u \in \mathcal{U}$ do
4:          draw first-order interaction selection variable $s_{ui} \sim Bernoulli(\pi_1)$;
5:          $w_{ui} = s_{ui} \cdot \tilde{w}_i$.
6:  for each feature pair $i, j \in \mathcal{F}$ do
7:      draw second-order interaction weight $\tilde{w}_{ij} \sim \mathcal{N}(0, 1)$;
8:      for each user $u \in \mathcal{U}$ do
9:          draw second-order interaction selection variable $s_{uij} \sim p(s_{uij} \mid s_{ui}, s_{uj}, \pi_2)$;
10:         $w_{uij} = s_{uij} \cdot \tilde{w}_{ij}$.
11: for each feature vector $\boldsymbol{x} \in \mathcal{X}$ do
12:     calculate the rating prediction $\hat{r}(\boldsymbol{x})$ by Eq. (3);
13:     draw $r(\boldsymbol{x}) \sim p(r \mid \hat{r}(\boldsymbol{x}))$.
```

distribution $\mathcal{N}(\hat{r}(\boldsymbol{x}), 1)$, we can derive the Gaussian log-likelihood:

$$\sum_{\boldsymbol{x} \in \mathcal{X}} \frac{1}{2}(r(\boldsymbol{x}) - \hat{r}(\boldsymbol{x}))^2. \tag{4}$$

If $r(\boldsymbol{x})$ follows a Bernoulli distribution $Bernoulli(\sigma(\hat{r}(\boldsymbol{x})))$, the logistic log-likelihood can be derived:

$$\sum_{\boldsymbol{x} \in \mathcal{X}} r(\boldsymbol{x}) \log \sigma(\hat{r}(\boldsymbol{x})) + (1 - r(\boldsymbol{x})) \log(1 - \sigma(\hat{r}(\boldsymbol{x}))). \tag{5}$$

The likelihood functions in Eq. (4) and (5) correspond to the squared loss and cross entropy loss, which are widely used by collaborative filtering methods [28]. Besides, ranking loss can also be derived, e.g., pairwise ranking loss [42] or listwise ranking loss [60]. However, deriving the proper likelihood function for optimization is beyond the scope of this paper. In this work, we employ the Gaussian likelihood of Eq. (4) for optimization.

3.3 Hereditary spike-and-slab prior

Although we can learn the personalized parameters in Eq. (3) for each user separately, there are two disadvantages to this. First, selecting interactions directly based on Eq. (3) fails to take advantage of collaborative filtering. Second, Eq. (3) raises some challenges for optimization due to the large number of parameters ($O(md^2)$).

Therefore, we reformulate w_{ui} by $s_{ui} \cdot \tilde{w}_i$ (line 5) and w_{uij} by $s_{uij} \cdot \tilde{w}_{ij}$ (line 10) in the generation procedure in §3.2. In order to model s_{ui}, s_{uij}, we propose the Hereditary Spike-and-Slab Prior (HSSP), which optimizes over SSP by capturing heredity constraints [13] between first- and second-order interactions. The intuition is that there are natural hereditary constraints among the first- and second-order interactions [32], i.e., feature i and feature j are the "parents" of feature interaction (i, j). The hereditary constraints help to dramatically reduce the number of candidate interactions.

The HSSP is based on two hereditary constraints: *strong heredity* and *weak heredity*, where we define as follows based on FMs.

Definition 3.1 (Strong heredity). Given a FM, *strong heredity* is the constraint that if the first-order interactions x_i and x_j are selected for the FM, the second-order interaction of their combination, i.e., (x_i, x_j) will also be selected.

According to *strong heredity*, we have:

$$\text{if } s_{ui} = 1 \text{ or } s_{uj} = 1 \text{ then } s_{uij} = 1.$$

Definition 3.2 (Weak heredity). Given a FM, *weak heredity* is the constraint that if one of the first-order interactions x_i or x_j is selected for the FM, then the second-order interaction of their combination (x_i, x_j) will have a non-zero probability to be selected.

According to *weak heredity*, we have:

$$\text{if } s_{ui} = 1 \text{ or } s_{uj} = 1 \text{ then } p(s_{uij} = 1) > 0.$$

Based on the definitions of *strong heredity* and *weak heredity*, we derive the HSSP by modifying the priors for s_{uij} of SSP:

$$\begin{aligned} p(s_{ui} = 1) = p(s_{uj} = 1) &= \pi_1 \\ p(s_{uij} = 1 \mid s_{ui}s_{uj} = 1) &= 1 && (\textit{Strong heredity}) \\ p(s_{uij} = 1 \mid s_{ui} + s_{uj} = 1) &= \pi_2 && (\textit{Weak heredity}) \\ p(s_{uij} = 1 \mid s_{ui} + s_{uj} = 0) &= 0, \end{aligned}$$

where $\pi_1, \pi_2 \in [0, 1]$ are constant values.

3.4 Variational inference

To optimize BP-FIS, we need to maximize the posterior $p(\tilde{W}, S \mid \mathcal{R}, \mathcal{X})$, where $S = \{s_{ui}\} \cup \{s_{uij}\}$ and $\tilde{W} = \{\tilde{w}_i\} \cup \{\tilde{w}_{ij}\}$. However, exact inference for $p(\tilde{W}, S \mid \mathcal{R}, \mathcal{X})$ requires an infeasible combinatorial search over $O(2^{md^2})$ possible models. To speed up the process, we conduct variational inference to approximate the posterior $p(\tilde{W}, S \mid \mathcal{R}, \mathcal{X})$ by a variational distribution $q(\tilde{W}, S)$. The closeness of the posterior and the variational distribution is measured by the Kullback Leibler (KL)-divergence, i.e., $\mathbb{KL}(q(\tilde{W}, S) \parallel p(\tilde{W}, S \mid \mathcal{R}, \mathcal{X}))$. The KL-divergence can be derived as follows:

$$\begin{aligned} &\mathbb{KL}(q(\tilde{W}, S) \parallel p(\tilde{W}, S \mid \mathcal{R}, \mathcal{X})) \\ &\quad = -\mathbb{E}_q\left[\log p(\tilde{W}, S, \mathcal{R}, \mathcal{X}) - \log q(\tilde{W}, S)\right] + \log p(\mathcal{R}, \mathcal{X}), \end{aligned} \tag{6}$$

where $\mathcal{L} = \mathbb{E}_q\left[\log p(\tilde{W}, S, \mathcal{R}, \mathcal{X}) - \log q(\tilde{W}, S)\right]$ is the Evidence Lower Bound (ELBO); $\log p(\mathcal{R}, \mathcal{X})$ is the marginal likelihood which does not depend on $q(\tilde{W}, S)$. Eq. (6) states that minimizing the KL-divergence is the same as maximizing the ELBO. To maximize the ELBO, we first discuss the variational distribution $q(\tilde{W}, S)$.

Variational distribution. To ensure the quality of approximation, we assume the hereditary constraints also hold in the variational distributions. Therefore, $q(\tilde{W}, S)$ can be factorized as follows:

$$q(\tilde{W}, S) = \prod_{i=1}^{d} \prod_{j=i+1}^{d} q(\tilde{w}_i) q(\tilde{w}_{ij}) \prod_{u=1}^{m} q(s_{ui}) q(s_{uij} \mid s_{ui}, s_{uj}). \tag{7}$$

The factorized distributions are modeled as follows:

$$\begin{aligned} q(w_i \mid \mu_i, \sigma_i) &= \mathcal{N}(\mu_i, \sigma_i) \\ q(w_{ij} \mid \mu_{ij}, \sigma_{ij}) &= \mathcal{N}(\mu_{ij}, \sigma_{ij}) \\ q(s_{ui} \mid \pi_{ui}) &= Bernoulli(\pi_{ui}) \\ q(s_{uij} \mid s_{ui}s_{uj} = 1) &= s_{uij} \\ q(s_{uij} \mid s_{ui} + s_{uj} = 1, \pi_{uij}) &= Bernoulli(\pi_{uij}) \\ q(s_{uij} \mid s_{ui} + s_{uj} = 0) &= 1 - s_{uij}, \end{aligned}$$

where $\rho = \{\mu_i, \sigma_i\} \cup \{\mu_{ij}, \sigma_{ij}\}$ and $\pi = \{\pi_{ui}\} \cup \{\pi_{uij}\}$ are the variational parameters.

Evidence lower bound. Given the variational distribution $q(\tilde{W}, S \mid \rho, \pi)$, the ELBO can be derived as follows:

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_q \left[\log p(\tilde{W}, S, \mathcal{R}, \mathcal{X}) - \log q(\tilde{W}, S \mid \rho, \pi)\right] \\ &= \sum_{\boldsymbol{x} \in \mathcal{X}} \mathbb{E}_q \left[\log p(r(\boldsymbol{x}) \mid \hat{r}(\boldsymbol{x}))\right] - \mathbb{KL}\left(q(\tilde{W}, S \mid \rho, \pi) \parallel p(\tilde{W}, S)\right),\end{aligned}$$

where $\mathbb{E}_q[\cdot]$ stands for the expectation w.r.t. $q(\tilde{W}, S \mid \rho, \pi)$. In the ELBO, $\mathbb{KL}(q(\tilde{W}, S \mid \rho, \pi) \parallel p(\tilde{W}, S))$ can be analytically derived (see Appendix A.1 for details). However, it is problematic to derive $\mathbb{E}_q[\log p(r(\boldsymbol{x}) \mid \hat{r}(\boldsymbol{x}))]$. Therefore, we approximate the expectation with Monte Carlo estimation as follows:

$$\sum_{\boldsymbol{x} \in \mathcal{X}} \mathbb{E}_q \left[\log p(r(\boldsymbol{x}) \mid \hat{r}(\boldsymbol{x}))\right] \approx \frac{1}{L} \sum_{l=1}^{L} \sum_{\boldsymbol{x} \in \mathcal{X}} \frac{1}{2} \left(r(\boldsymbol{x}) - \hat{r}(\boldsymbol{x})^{(l)}\right)^2, \quad (8)$$

where $\hat{r}(\boldsymbol{x})^{(l)}$ is calculated by Eq. (3) with the l-th sampling $W^{(l)}$. We write $\log p(r(\boldsymbol{x}) \mid \hat{r}(\boldsymbol{x})^{(l)})$ as a Gaussian log-likelihood as we assume $r(\boldsymbol{x})$ to follow $\mathcal{N}(\hat{r}(\boldsymbol{x}), 1)$ (Eq. (4) in §3.2).

Reparameterization. When sampling w_{ui} and w_{uij}, we apply the reparameterization trick [23]:

$$\begin{aligned} \epsilon_1, \epsilon_2 &\sim \mathit{Uniform}(0, 1), \quad \varepsilon_1, \varepsilon_2 \sim \mathcal{N}(0, 1) \\ s_{ui} &= [\![\epsilon_1 \geq \pi_{ui}]\!] \\ s_{uij} &= s_{ui}s_{uj} + (1 - s_{ui}s_{uj})(s_{ui} + s_{uj})[\![\epsilon_2 \geq \pi_{uij}]\!] \\ \tilde{w}_i &= \mu_i + \varepsilon_1 \sigma_i \\ \tilde{w}_{ij} &= \mu_{ij} + \varepsilon_2 \sigma_{ij} \\ w_{ui} &= \tilde{w}_i \cdot s_{ui}, \quad w_{uij} = \tilde{w}_{ij} \cdot s_{uij}. \end{aligned} \quad (9)$$

By doing so, the stochasticity in the sampling process is isolated and the gradient with respect to $\rho = \{\mu_i, \sigma_i\} \cup \{\mu_{ij}, \sigma_{ij}\}$ can be back-propagated through the sampled w_{ui} and w_{uij}. Unfortunately, the above procedure fails to take the gradient of $\pi = \{\pi_{ui}\} \cup \{\pi_{uij}\}$ due to the discrete nature of $S = \{s_{ui}\} \cup \{s_{uij}\}$. We follow [51] by marginalizing out the variable of interest (see Appendix A.2).

Faster inference. To further speed up the variational inference, we factorize the variational parameter π_{uij} as $\pi_{ui} \cdot \pi_{uj}$. In this way, we only need to preserve $\{\pi_{ui}\}$ for each user, decreasing the learning parameters from $O(md^2)$ to $O(md)$. For $\{\tilde{w}_{ij}\}$, we introduce feature embeddings $V \in \mathbb{R}^{d \times k}$ and replace the variational parameters for $\tilde{w}_{ij}$ as follows:

$$\mu_{ij} = \mu(\boldsymbol{v}_i, \boldsymbol{v}_j), \quad \sigma_{ij} = \sigma(\boldsymbol{v}_i, \boldsymbol{v}_j),$$

where $\mu(\cdot)$ and $\sigma(\cdot)$ are the transformation functions, and $\boldsymbol{v}_i, \boldsymbol{v}_j \in \mathbb{R}^k$ are the embeddings for feature i, j, respectively. Note that we can have different definitions for $\mu(\cdot)$ and $\sigma(\cdot)$. Inspired by [17, 57], we define the transformations as follows:

$$\mu(\boldsymbol{v}_i, \boldsymbol{v}_j) = \boldsymbol{\mu}^T f_\phi(\boldsymbol{v}_i \circ \boldsymbol{v}_j), \quad \sigma(\boldsymbol{v}_i, \boldsymbol{v}_j) = \boldsymbol{\sigma}^T f_\phi(\boldsymbol{v}_i \circ \boldsymbol{v}_j), \quad (10)$$

where $\circ$ is the element-wise product operation and $f_\phi(\cdot)$ is the transformation parameterized by ϕ, which transforms a vector from $\mathbb{R}^k$ to $\mathbb{R}^h$; $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are the vectors in $\mathbb{R}^h$. Note that $\mu(\cdot)$ and $\sigma(\cdot)$ can be either linear transformations, e.g., $f_\phi(\boldsymbol{v}) = \boldsymbol{v}$, or non-linear transformations, e.g., $f_\phi(\cdot)$ is a neural network. The variational parameter for $\tilde{W}$ is $\rho = \{\phi, \boldsymbol{\mu}, \boldsymbol{\sigma}, V\}$.

Learning. We propose to learn BP-FIS via Stochastic Gradient Variational Bayes (SGVB). As the reparameterization procedures for estimating gradients of ρ and π are different, we propose to optimize the variational parameter π and ρ alternatively. Specifically, at iteration t, by fixing $\rho^{(t-1)}$, we train $\pi^{(t)}$ to update the probabilities that a user will select the specific feature interactions. Then by fixing $\pi^{(t)}$, rather than training $\rho^{(t)}$ based on $\rho^{(t-1)}$, we train $\rho^{(t)}$ from scratch. This is because when π is updated, we will have different user preferences of feature interactions, where ρ should be optimized accordingly. As we have experimented, adapting ρ fitting for $\pi^{(t-1)}$ to $\pi^{(t)}$ could adversely bias the learning.

Prediction. Once the model has been trained, we can generate predictions via point estimation in Eq. (11):

$$\mathbb{E}[\hat{r}(\boldsymbol{x})] = b_u + \sum_{i=1}^{d} \mathbb{E}[w_{ui}] x_i + \sum_{i=1}^{d} \sum_{j=i+1}^{d} \mathbb{E}\left[w_{uij}\right] x_i x_j, \quad (11)$$

where $\mathbb{E}[w_{ui}] = \pi_{ui}\mu_i$ and $\mathbb{E}\left[w_{uij}\right] = [\pi_{ui}\pi_{uj} + (1 - \pi_{ui}\pi_{uj})(\pi_{ui} + \pi_{uj})\pi_{ui}\pi_{uj}]\mu_{ij}$.

Table 2: Statistics of the datasets.

Dataset	#User	#Item	#Feature	#Rating
MLHt	2,112	5,682	11,945	731,215
LastFM	1,892	17,632	29,200	341,104
Delicious	1,867	69,223	77,735	372,609

4 EXPERIMENTAL SETUP

We evaluate BP-FIS using the top-N recommendation task.

4.1 Research questions

We seek to answer the following research questions: (RQ1) Does P-FIS help to improve the performance of FMs on the top-N recommendation task? Specifically, how well does BP-FIS perform against state-of-the-art FMs? (RQ2) How does the embedding size impact the ability of BP-FIS to improve the performance of FMs? (RQ3) How does the alternating optimization procedure affect the performance of BP-FIS? (RQ4) How can BP-FIS provide explainability for the recommendations generated by the FMs?

4.2 Dataset

We use three benchmark recommendation datasets from HetRec [6] in our experiments. Summary statistics are provided in Table 2.

- MovieLens Hetrec (MLHt) – Extends the MovieLens10M [16] dataset.[1] It links the movies of the MovieLens dataset with their corresponding web pages at the Internet Movie Database (IMDb)[2] and Rotten Tomatoes movie review systems.[3] MLHt only contains users with both rating and tagging information.
- LastFM – Contains social networking, tagging, and music artist listening information from the Last.fm online music system.[4] Each user has a list of most listened music artists, tag assignments, and friend relations within the social network.
- Delicious – Contains social networking, bookmarking, and tagging information from the Delicious social bookmarking system.[5]

[1]http://www.grouplens.org
[2]http://www.imdb.com
[3]http://www.rottentomatoes.com
[4]http://www.last.fm
[5]http://www.delicious.com

Each user has bookmarks, tag assignments, and contact relations within the social network.

For MLHt, users rate each movie with stars, on a scale from 1 to 5. We treat ratings of at least 3 as positive ($r = 1$ if rating ≥ 3) and treat all other ratings as missing ($r = 0$) [20, 56]. For LastFM, we regard user's listening history of artists as implicit ratings, i.e., $r = 1$ if the user listened to a song by the artist. Similarly for Delicious, we regard the bookmarks added by the users as implicit ratings. The side information of these datasets is utilized as additional features, i.e., user tags, movie genres and social networks. Ratings with less than 10 non-zero feature values are discarded. Therefore, our statistics may slightly differ from the original datasets.

4.3 Baselines

To evaluate the effectiveness of BP-FIS, we provide two implementations, namely BP-FM and BP-NFM. They differ in the definition of $f_\phi(\cdot)$ in Eq. (10): $f_\phi(\cdot)$ is an identify transformation for BP-FM and a stack of fully connected network layers for BP-NFM. We implement BP-FM and BP-NFM using PyTorch.[6] We compare BP-FM and BP-NFM with the following baseline methods:

FM The Factorization Machine (FM) is originally proposed by Rendle [39]. The official implementation is specifically optimized for the rating prediction task, whereas we evaluate the performance with top-N recommendation metrics. Therefore, we provide a PyTorch implementation of FM for a fair comparison.

SFM The Sparse Factorization Machine (SFM) is learns sparse first- and second-order interactions [37, 58, 62] study SFMs. We implement SFM, on top of the implementation of FM, to utilize general features for top-N recommendation, where all feature embeddings are penalized by group Lasso regularizations.

AFM The Attentional Factorization Machine (AFM) [57] learns the importance of each feature interaction from data via an attention network. We utilize the tensorflow implementation[7] of AFM released by the authors in our experiments.

NFM The Neural Factorization Machines (NFM) [17] models higher-order interactions through neural networks, which is the state-of-the-art neural extension of factorization machines. We utilize the tensorflow implementation[8] of NFM released by the authors.

4.4 Evaluation

We adopt the leave-one-out evaluation, which has been widely used in the literature [21, 36]. For each user, we randomly hold-out one of her interactions as the test set and utilize the remaining data for training. Since it is too time-consuming to rank all items for every user during evaluation, we follow the common strategy [25, 28] which randomly samples items that are not interacted with by the user, ranking the test item among 100 items. The recommendation quality is measured using Hit Rate (HR) and Average Reciprocal Hit-Rank (ARHR). HR is defined as follows:

$$\text{HR} = \frac{\#Hit}{\#User}, \quad \text{ARHR} = \frac{1}{\#User} \sum_{i=1}^{\#Hit} \frac{1}{p_i},$$

[6]https://pytorch.org/
[7]https://github.com/hexiangnan/attentional_factorization_machine
[8]https://github.com/hexiangnan/neural_factorization_machine

where $\#User$ is the total number of users, and $\#Hit$ is the number of users whose item in the test set is recommended (i.e., a hit) in the size-N recommendation list. p_i is the position of the item in the ranked recommendation list, if an item of a user is among the list. ARHR is a weighted version of HR and it measures how strongly an item is recommended, in which the weight is the reciprocal of the hit position in the recommendation list.

4.5 Implementation details

For a fair comparison, we have the following identical experimental settings for all compared methods. (1) All models are optimized using Adam [22]. Adam computes individual adaptive learning rates for different parameters. Therefore, we do not need to tune the learning rate. In practice, setting the initial learning rate as 0.001 provides a good default value. (2) All models are optimized by the mean square loss, accounting for the fairness of comparison and efficiency of training. (3) We hold-out ratings from the training set for validation. We tune parameters of all methods and select the ones with the best performance. (4) We set the maximum training epochs to 50. We apply early-stop for all methods, where we stop training if no further performance gain is observed for 4 successive epochs. (5) Feature embeddings are randomly initialized according to $\mathcal{N}(0, 0.01)$. (6) We tune the parameter k (the latent dimension of feature embeddings) from $64, 128, 256$.

For BP-FIS, we set $\pi_1 = \pi_2 = 0.5$ as we presume no prior knowledge about the selection. For the baselines, we follow the experimental settings in [17, 57] to tune parameters. We tune the ℓ_2-norm regularization parameter for FM in $1e^{-6}, 5e^{-6}, \ldots, 1e^{-1}$. Similarly, we tune the $\ell_{2,1}$-norm regularization parameter for SFM in $1e^{-6}, 5e^{-6}, \ldots, 1e^{-1}$. We use dropout for NFM and AFM. For NFM, we fix the dropout rate of the hidden layers as 0.8 and tune the dropout rate for Bi-interaction layer in $0.1, 0.2, \ldots, 1.0$. For AFM, we use dropout for the embedding layer, which is searched from $0.1, 0.2, \ldots, 1.0$. As suggested by the author, we tune the ℓ_2-norm regularization parameter for the attention layer in $0, 0.5, 1, 2, 4, 8, 16$.

For the network structure of NFM and BP-NFM, we follow [17] and set identical dimensions for hidden layers. According to [17], NFM with one hidden layer generates the best performance. Therefore, we also use one hidden layer for NFM and BP-NFM in our experiments. We use ReLU as the activation function.

5 EXPERIMENTAL RESULT AND ANALYSIS

We answer the research questions listed in §4.1 in four subsections.

5.1 RQ1: Results

We report the recommendation performance of all models in Table 3, in terms of HR@1, HR@10 and ARHR@10. Table 3 shows that BP-FM and BP-NFM consistently outperform the other methods. This proves the effectiveness of BP-FIS for improving the performance of both linear and non-linear FMs for top-N recommendation.

To answer RQ1, we analyze the results by separating the comparison between linear and non-linear models. Among linear models, SFM outperforms FM and AFM on the LastFM and Delicious datasets. This supports the need for conducting FIS for FMs. However, SFM cannot always improve over FMs. It fails to beat FM on MLHt. In contrast, BP-FM achieves improvements over FM on the

Table 3: Comparison of top-N recommendation methods.

	Method	λ	drop	k	HR@1	HR@10	ARHR@10
MLHt	FM	0.1	–	64	0.2371	0.6028	0.3398
	SFM	0.01	–	64	0.2294	0.6052	0.3351
	AFM	2	0.1	64	0.1138	0.4273	0.1969
	BP-FM	–	–	128	**0.2401***	**0.7070****	**0.3932****
	NFM	–	0.2	256	0.2180	0.6257	0.3389
	BP-NFM	–	–	128	**0.2519****	**0.6831****	**0.3814****
LastFM	FM	0.1	–	64	0.1894	0.6403	0.3215
	SFM	0.05	–	256	0.2118	0.6542	0.3449
	AFM	8	0.7	256	0.2166	0.6126	0.3332
	BP-FM	–	–	256	**0.2209**	**0.6798****	**0.3581****
	NFM	–	0.6	64	0.2150	0.6798	0.3563
	BP-NFM	–	–	256	**0.2257**	**0.6910**	**0.3660**
Delicious	FM	0.1	–	64	0.0202	0.1147	0.0440
	SFM	0.1	–	128	0.0229	0.1212	0.0465
	AFM	2	0.1	64	0.0274	0.1169	0.0494
	BP-FM	–	–	128	**0.0278**	**0.1240****	**0.0509***
	NFM	–	0.1	64	0.0229	0.1065	0.0426
	BP-NFM	–	–	128	**0.0268**	**0.1289****	**0.0504****

Boldface scores indicate best results for linear and non-linear FMs on each metric. We conducted two-sided tests for the null hypothesis that the best and the second best have identical average values. * and ** indicate that the best score is significantly better than the second best score with $p < 0.1$ and $p < 0.05$, respectively.

same dataset. This shows that selecting or weighing a single set of feature interactions for all users might overlook the personalization over features, and conducting P-FIS is required. Except for HR@1 on LastFM and Delicious, the improvement achieved by BP-FM is significant, especially HR@10 on MLHt, where BP-FM improves FM by 17.286%. This demonstrates the efficacy of BP-FIS.

The comparison between non-linear models, i.e., NFM and BP-NFM, shows similar results. BP-NFM steadily achieves better performance than NFM on all datasets and all metrics, which shows that the improvement achieved by P-FIS is orthogonal to the non-linear modeling of interactions.

5.2 RQ2: Impact of embedding size

To answer RQ2, we analyze the performance of all models with different embedding sizes. Specifically, we plot figures to show results w.r.t. HR@1, HR@5, HR@10, ARHR@5 and ARHR@10 of all models on the MLHt dataset, as shown in Figure 3. Generally, we can see that BP-FM and BP-NFM achieve better a performance than the other models. This demonstrates the robustness of BP-FIS as it constantly improves the performance of FMs, regardless of the embedding size and evaluation metric.

Specifically, for the setting $k = 64$, almost all models show a competitive performance. This is because the space for the performance improvement is limited when $k = 64$ on the MLHt dataset. P-FIS has insignificant effect on FMs due to the restricted embedding size. In contrast, for the setting $k = 128$, while FM, SFM and NFM achieve comparable results, BP-FM and BP-NFM significantly outperform them. This means that the effect of P-FIS can be better expressed by

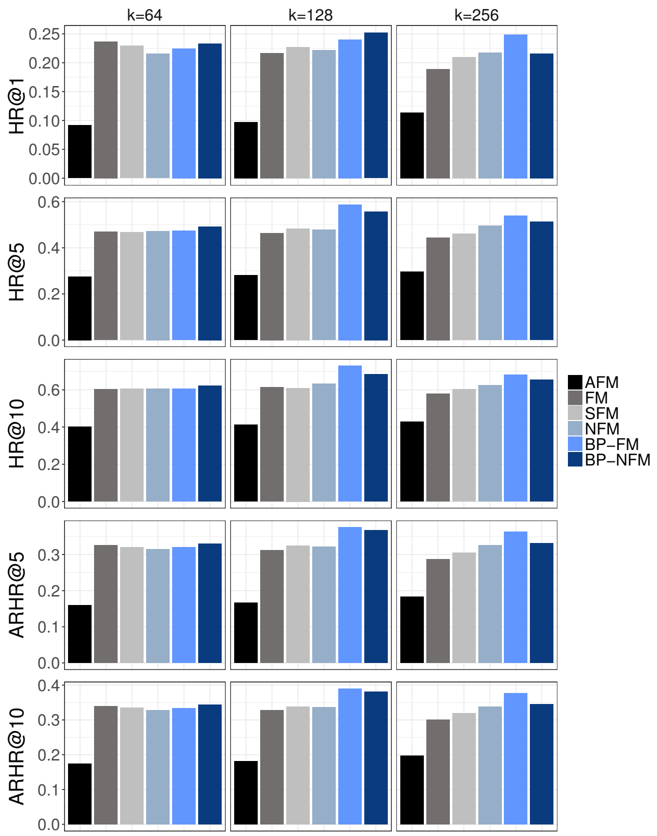

Figure 3: Performance comparison w.r.t. different embedding sizes. ARHR@1 is actually HR@1.

increasing the embedding size for FMs. Interestingly, while BP-FM performs better than BP-NFM for most metrics, the performance gain achieved by BP-NFM is more remarkable on HR@1. Similar performance levels are seen when $k = 256$, except that BP-NFM does not outperform BP-FM. This might be because BP-NFM has far more parameters than BP-FM when the embedding size is large, which brings extra difficulty to avoid overfitting.

5.3 RQ3: Impact of training

To analyze alternative training procedure of BP-FIS, we show the performance of BP-FM and BP-NFM after each iteration in Figure 4. A general trend could be revealed by Figure 4 that the recommendation performance of both BP-FM and BP-NFM grows initially and fluctuate successively. Although BP-NFM can mostly achieve better performance than BP-FM, it also shows higher variation.

When $k = 64$, BP-FM outperforms BP-NFM w.r.t. HR@1 and performs competitively with BP-NFM w.r.t. HR@5 and HR@10. When $k = 128$, we can witness a relatively stable growth in BP-FM, especially for HR@5 and HR@10. While a certain level of convergence can be witnessed for HR@5 and HR@10, BP-NFM still fluctuates more than BP-FM during the training procedure. These observations might suggest that the training of BP-FIS shows better robustness for linear FMs than non-linear ones. BP-NFM is more unstable in terms of HR@5 and HR@10 when $k = 256$, the performance of which drops sharpely during the 9-th iteration. Thus,

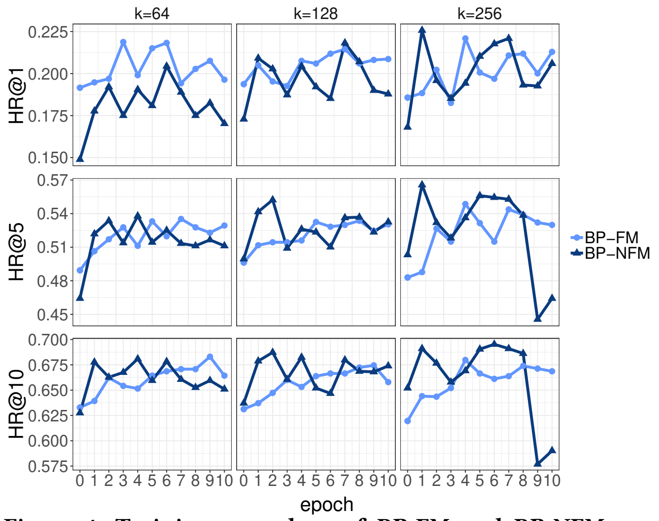

Figure 4: Training procedure of BP-FM and BP-NFM on LastFM dataset.

Table 4: Case study.

	48 Hrs.	Spider-Man	Lethal Weapon	True Lies
action (a)	√	√	√	
buddy (b)	√		√	√
clever (cl)	√			√
comedy (co)	√		√	√
funny (f)		√		√
sequel (se)		√	√	
special effects (sp)		√		√
user 1	–	top-5	top-1	top-10
user 2	–	top-5	top-5	top-10
user 3	–	–	–	–

Ticks indicate whether a movie has the feature. We also indicate for each user whether the movie is within the top-1, top-5 or top-10 recommendation.

more iterations do not help a lot for improving the performance but might adversely harm the performance. Another observation is that the training procedure of BP-FM and BP-NFM varies with different embedding sizes. Training BP-FM and BP-NFM with $k = 128$ provides the most stable procedure. This shows that a proper selection of embedding size can further extend the potential of BP-FIS.

5.4 RQ4: Explainability for recommendation

To answer RQ4, we provide a case study based on the experiments on the MLHt dataset. We select four movies ("48 Hrs.", "Spider-Man", "Lethal Weapon" and "True Lies") with seven shared features ("action", "buddy", "clever", "comedy", "funny", "sequel" and "special effects"). We generate the recommendation results with BP-FM for three users ("user 1", "user 2" and "user 3") and check if the four movies are in the top-1, top-5 or top-10 list. The results are shown in Table 4.

We also visualize the selection probability of second-order feature interactions ($p(s_{uij} = 1)$) in Figure 5. The color depth indicates how likely the corresponding feature interaction will be selected for the user. The probability indicates the predictive power of the feature interaction for the specific user.

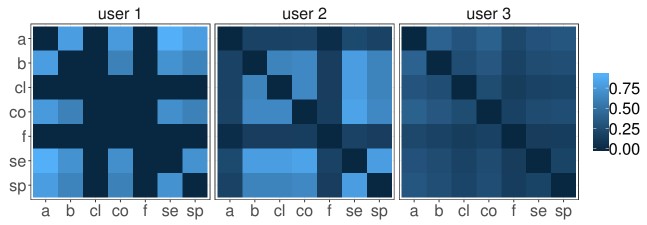

Figure 5: Visualization of second-order feature interaction selection w.r.t. the case studies in Table 4. The color depth corresponds to the probability of selecting the corresponding interaction. (a, b, cl, co, f, se, sp) are the abbreviations of the features listed in Table 4.

Figure 5 shows the diverse selections of feature interactions for the three users. All interactions are unlikely to be selected for user 3. Therefore, the four movies sharing these features are not recommended to her. In comparison, several interactions have high probabilities to be selected for user 1 and user 2. Some interactions are predictive for both users, e.g., (sequel, special effects), (comedy, sequel), (buddy, comedy), and some are useful only for a specific user, e.g., (action, buddy), (action, comedy), (action, sequel) for user 1 and (buddy, clever), (buddy, comedy) for user 2.

The commonality of feature interactions for user 1 and user 2 explains the recommendation of "Spider-Man" and "True-Lies". "Spider-Man" has (sequel, special effects) and "True-Lies" has (buddy, commedy), which have been selected for both users. The personalization of feature interactions explains the recommendation of "Lethal Weapon". While the movie has the interaction (buddy, comedy) that was selected for both users, it also has the specific interaction (action, comedy) that is only selected for user 1. Therefore, "Lethal Weapon" is the top-1 item for user 1 and the top-5 item for user 2. On the other hand, "48Hrs." is not recommended to any user. Although the movie has (action, buddy) and (action, comedy) selected for user 1 or (buddy, clever) and (buddy, comedy) selected for user 2, it has no interactions like (comedy, sequel) or (sequel, special effects), which might account largely for the recommendation.

6 RELATED WORK

In this section, we survey related work on factorization machines and feature (interaction) selection, respectively.

6.1 Factorization machines

FMs have been widely studied and are commonly used in practical systems for their effectiveness and flexibility. Early studies [39, 43] show the generality of FMs. In contrast to Matrix Factorization (MF) [47] or Tensor Factorization (TF) [44], which model interactions between categorical variables only, FMs provides a generic way to model interactions between any real valued features. Rendle [39] show that FMs can mimic many of the most successful factorization models (including MF, parallel factor analysis, and SVD++ [25]).

Recently, successive variants of FMs have been developed [1, 4, 5, 19, 29, 30]. They have achieved promising performance in different recommendation scenarios [17, 35, 37, 38, 60]. Juan et al. [19] propose the Field-Aware Factorization Machine (FFM) to factorize the interactions of fields (the category of features). Blondel et al. [4] present the Higher-Order Factorization Machine (HOFM), which provides an efficient algorithm to train FMs with higher-order interactions. Besides rating prediction, FMs have also been optimized for the top-N recommendation task. Yuan et al. [60] introduce Boosted Factorization Machiness (BoostFMs) to incorporate contextual information into FMs for Context-Aware Recommendation (CAR) [33, 35, 43]. Xiao et al. [57] propose the Attentional Factorization Machine (AFM), which uses an attention network to learn the importance of each feature interaction. To investigate the linear expressiveness limitation of FMs, He and Chua [17] propose Neural Factorization Machiness (NFMs), which perform non-linear transformations on the latent space of second-order feature interactions.

Despite the effectiveness of modeling various feature interactions, existing FM variants suffer from the high-dimensionality issue which limits their application in high-dimensional scenarios.

6.2 Feature selection

An effective approach to alleviate the high-dimensionality issue of FMs is feature selection. Cheng et al. [11] propose to select feature interactions though a greedy interaction feature selection algorithm based on gradient boosting. Xu et al. [58] apply group Lasso [61] to user and item feature embeddings to select interactions between user and item features. Zhao et al. [62] propose to select meta-graph based features. Similarly, they also apply group Lasso for feature selection. Mao et al. [31] propose to select context features for FMs. They first choose features based on predictive power. Then, they subsample the set of features selected in the first step.

Feature selection has also been well investigated for Feature-based Recommender Systems (FRSs), which often utilize high-dimensional auxiliary information. Ronen et al. [45] propose to select content features. Their algorithm selects the most informative features by computing relevance scores based on pluggable feature similarity functions. Koenigstein and Paquet [24] propose to select content features for Xbox movies by incorporating sparsity priors on feature parameters. Different feature weighing methods have also been proposed to select context features [63]. Li et al. [27] study personalized feature selection for unsupervised learning; they learn a specific model for each user, which is only applicable with a limited number of users.

The differences between our method and the above methods are at least three-fold. First, we target personalized feature interaction selection, which better captures a user's personalized preferences over different features. Second, we provide a generic way to achieve feature selection, which can be seamlessly integrated to different FM variants. Third, we opt for Bayesian Variable Selection (BVS) with spike and slab priors, rather than the sparsity induced regularizations that have previously been considered.

7 CONCLUSION

We propose a Bayesian Personalized Feature Interaction Selection (BP-FIS) method to address the Personalized Feature Interaction Selection (P-FIS) task for Factorization Machines (FMs) in this paper. BP-FIS fuses Hereditary Spike-and-Slab Prior (HSSP) to achieve P-FIS while taking advantage of collaborative filtering. We conduct variational inference and propose a Stochastic Gradient Variational Bayes (SGVB) method to optimize BP-FIS. Experimental results show that BP-FIS significantly improves the performance of both linear and non-linear FMs. Further analytical experiments show that: (1) BP-FIS is effective for both linear and non-linear FMs; (2) BP-FIS is robust for performance gain, regardless of the embedding size; and (3) it is preferable to train BP-FIS with a limited number of iterations.

A limitation of BP-FIS is that it needs more time to train than other FMs because we need to optimize the parameters specifically for each user. In future work, we hope to improve BP-FIS in two ways: (1) extend BP-FIS to higher-order interactions or multi-view and multimodal factorizations [52, 54]; and (2) consider group-level personalization by clustering users to speed up training.

ACKNOWLEDGMENTS

Yifan Chen is supported by the China Scholarship Council. Yang Wang is supported by National Natural Science Foundation of China with Grant No. 61806035. Maarten de Rijke is supported by Ahold Delhaize, the Association of Universities in the Netherlands (VSNU), and the Innovation Center for Artificial Intelligence (ICAI). All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

[1] Michal Aharon, Natalie Aizenberg, Edward Bortnikov, Ronny Lempel, Roi Adadi, and etc. 2013. OFF-set: One-pass Factorization of Feature Sets for Online Recommendation in Persistent Cold Start Settings. In *RecSys*. ACM, 375–378.

[2] Taha Alshaybawee, Rahim Alhamzawi, Habshah Midi, and Intisar Ibrahim Allyas. 2018. Bayesian Variable Selection and Coefficient Estimation in Heteroscedastic Linear Regression Model. *Journal of Applied Statistics* 45, 14 (2018), 2643–2657.

[3] Cédric Archambeau and Francis R. Bach. 2009. Sparse Probabilistic Projections. In *NIPS*. 73–80.

[4] Mathieu Blondel, Akinori Fujino, Naonori Ueda, and Masakazu Ishihata. 2016. Higher-Order Factorization Machines. In *NIPS*. 3351–3359.

[5] Mathieu Blondel, Masakazu Ishihata, Akinori Fujino, and Naonori Ueda. 2016. Polynomial Networks and Factorization Machines: New Insights and Efficient Training Algorithms. In *ICML*. ACM, 850–858.

[6] Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik. 2011. 2nd Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011). In *RecSys*. ACM.

[7] Kailong Chen, Tianqi Chen, Guoqing Zheng, Ou Jin, Enpeng Yao, and Yong Yu. 2012. Collaborative Personalized Tweet Recommendation. In *SIGIR*. ACM, 661–670.

[8] Yifan Chen, Yang Wang, Xiang Zhao, Hongzhi Yin, Ilya Markov, and Maarten de Rijke. 2019. Local Variational Feature-based Similarity Models for Recommending Top-N New Items. *ACM Transactions on Information Systems* (2019). To appear.

[9] Yifan Chen, Xiang Zhao, and Maarten de Rijke. 2017. Top-N Recommendation with High-dimensional Side Information via Locality Preserving Projection. In *SIGIR*. ACM, 985–988.

[10] Yifan Chen, Xiang Zhao, Xuemin Lin, Yang Wang, and Deke Guo. 2019. Efficient Mining of Frequent Patterns on Uncertain Graphs. *IEEE Trans. Knowl. Data Eng.* 31, 2 (2019), 287–300.

[11] Chen Cheng, Fen Xia, Tong Zhang, Irwin King, and Michael R. Lyu. 2014. Gradient Boosting Factorization Machines. In *RecSys*. ACM, 265–272.

[12] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & Deep Learning for Recommender Systems. In *DLRS*. 7–10.

[13] Nam Hee Choi, William Li, and Ji Zhu. 2010. Variable Selection with the Strong Heredity Constraint and its Oracle Property. *J. Amer. Statist. Assoc.* 105, 489 (2010), 354–364.

[14] Petros Dellaportas, Jonathan J. Forster, and Ioannis Ntzoufras. 2002. On Bayesian Model and Variable Selection using MCMC. *Statistics and Computing* 12, 1 (2002), 27–36.

[15] Cédric Févotte and Simon J. Godsill. 2006. Sparse Linear Regression in Unions of Bases via Bayesian Variable Selection. *IEEE Signal Processing Letters* 13, 7 (2006), 441–444.

[16] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Trans. Inter. Intel. Syst.* 5, 4 (2015), 19:1–19:19.

[17] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. In *SIGIR*. ACM, 355–364.

[18] Yuchin Juan, Damien Lefortier, and Olivier Chapelle. 2017. Field-aware Factorization Machines in a Real-world Online Advertising System. In *WWW*. 680–688.

[19] Yu-Chin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware Factorization Machines for CTR Prediction. In *RecSys*. ACM, 43–50.

[20] Santosh Kabbur, Xia Ning, and George Karypis. 2013. FISM: Factored Item Similarity Models for Top-N Recommender Systems. In *SIGKDD*. ACM, 659–667.

[21] George Karypis. 2001. Evaluation of Item-Based Top-N Recommendation Algorithms. In *CIKM*. ACM, 247–254.

[22] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

[23] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *ICLR*.

[24] Noam Koenigstein and Ulrich Paquet. 2013. Xbox Movies Recommendations: Variational Bayes Matrix Factorization with Embedded Feature Selection. In *RecSys*. ACM, 129–136.

[25] Yehuda Koren. 2008. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. In *SIGKDD*. ACM, 426–434.

[26] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural Attentive Session-based Recommendation. In *CIKM*. ACM, 1419–1428.

[27] Jundong Li, Liang Wu, Harsh Dani, and Huan Liu. 2018. Unsupervised Personalized Feature Selection. In *AAAI*. 3514–3521.

[28] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *WWW*. 689–698.

[29] Xiao Lin, Wenpeng Zhang, Min Zhang, Wenwu Zhu, Jian Pei, Peilin Zhao, and Junzhou Huang. 2018. Online Compact Convexified Factorization Machine. In *WWW*. 1633–1642.

[30] Chun-Ta Lu, Lifang He, Weixiang Shao, Bokai Cao, and Philip S. Yu. 2017. Multilinear Factorization Machines for Multi-Task Multi-View Learning. In *WSDM*. ACM, 701–709.

[31] Xueyu Mao, Saayan Mitra, and Viswanathan Swaminathan. 2017. Feature Selection for FM-Based Context-Aware Recommendation Systems. In *ISM*. IEEE, 252–255.

[32] Peter McCullagh and John A Nelder. 1989. *Generalized Linear Models*. Vol. 37. CRC press.

[33] Lei Mei, Pengjie Ren, Zhumin Chen, Liqiang Nie, Jun Ma, and Jian-Yun Nie. 2018. An Attentive Interaction Network for Context-aware Recommendations. In *CIKM*. ACM, 157–166.

[34] Toby J. Mitchell and John J. Beauchamp. 1988. Bayesian Variable Selection in Linear Regression. *J. Amer. Statist. Assoc.* 83, 404 (1988), 1023–1032.

[35] Trung V. Nguyen, Alexandros Karatzoglou, and Linas Baltrunas. 2014. Gaussian Process Factorization Machines for Context-aware Recommendations. In *SIGIR*. ACM, 63–72.

[36] Xia Ning and George Karypis. 2011. SLIM: Sparse Linear Methods for Top-N Recommender Systems. In *ICDM*. IEEE, 497–506.

[37] Zhen Pan, Enhong Chen, Qi Liu, Tong Xu, Haiping Ma, and Hongjie Lin. 2016. Sparse Factorization Machines for Click-through Rate Prediction. In *ICDM*. IEEE, 400–409.

[38] Rajiv Pasricha and Julian McAuley. 2018. Translation-based Factorization Machines for Sequential Recommendation. In *RecSys*. ACM, 63–71.

[39] Steffen Rendle. 2010. Factorization Machines. In *ICDM*. IEEE, 995–1000.

[40] Steffen Rendle. 2012. Factorization Machines with libFM. *ACM Trans. Intel. Syst. & Tech.* 3, 3 (2012), 57:1–57:22.

[41] Steffen Rendle. 2012. Learning Recommender Systems with Adaptive Regularization. In *WSDM*. ACM, 133–142.

[42] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*. 452–461.

[43] Steffen Rendle, Zeno Gantner, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2011. Fast Context-aware Recommendations with Factorization Machines. In *SIGIR*. ACM, 635–644.

[44] Steffen Rendle and Lars Schmidt-Thieme. 2010. Pairwise Interaction Tensor Factorization for Personalized Tag Recommendation. In *WSDM*. ACM, 81–90.

[45] Royi Ronen, Noam Koenigstein, Elad Ziklik, and Nir Nice. 2013. Selecting Content-based Features for Collaborative Filtering Recommenders. In *RecSys*. ACM, 407–410.

[46] Ying Shan, T. Ryan Hoens, Jian Jiao, Haijing Wang, Dong Yu, and J. C. Mao. 2016. Deep Crossing: Web-Scale Modeling without Manually Crafted Combinatorial Features. In *SIGKDD*. ACM, 255–262.

[47] Nathan Srebro, Jason D. M. Rennie, and Tommi S. Jaakkola. 2004. Maximum-Margin Matrix Factorization. In *NIPS*. 1329–1336.

[48] Ingo Steinwart and Andreas Christmann. 2008. *Support Vector Machines*. Springer Science & Business Media.

[49] Robert Tibshirani. 1996. Regression Shrinkage and Selection via the Lasso. *J. Royal Statist. Soci.* (1996), 267–288.

[50] Michalis K. Titsias and Miguel Lázaro-Gredilla. 2011. Spike and Slab Variational Inference for Multi-Task and Multiple Kernel Learning. In *NIPS*. 2339–2347.

[51] Seiya Tokui and Issei Sato. 2016. Reparameterization Trick for Discrete Variables. *CoRR* abs/1611.01239 (2016). arXiv:1611.01239

[52] Meng Wang, Hao Li, Dacheng Tao, Ke Lu, and Xindong Wu. 2012. Multimodal Graph-Based Reranking for Web Image Search. *IEEE Trans. Image Processing* 21, 11 (2012), 4649–4661.

[53] Meng Wang, Xueliang Liu, and Xindong Wu. 2015. Visual Classification by ℓ_1-Hypergraph Modeling. *IEEE Trans. Knowl. Data Eng.* 27, 9 (2015), 2564–2574.

[54] Yang Wang, Xuemin Lin, Lin Wu, Wenjie Zhang, Qing Zhang, and Xiaodi Huang. 2015. Robust Subspace Clustering for Multi-View Data by Exploiting Correlation Consensus. *IEEE Trans. Image Processing* 24, 11 (2015), 3939–3949.

[55] Mike West. 2003. Bayesian Factor Regression Models in the "Large p, Small n" Paradigm. In *Bayesian Statistics*. Oxford University Press, 723–732.

[56] Yao Wu, Christopher DuBois, Alice X. Zheng, and Martin Ester. 2016. Collaborative Denoising Auto-Encoders for Top-N Recommender Systems. In *WSDM*. ACM, 153–162.

[57] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional Factorization Machines: Learning the Weight of Feature Interactions via Attention Networks. In *IJCAI*. 3119–3125.

[58] Jianpeng Xu, Kaixiang Lin, Pang-Ning Tan, and Jiayu Zhou. 2016. Synergies that Matter: Efficient Interaction Selection via Sparse Factorization Machine. In *SDM*. SIAM, 108–116.

[59] Makoto Yamada, Wenzhao Lian, Amit Goyal, Jianhui Chen, Kishan Wimalawarne, and etc. 2017. Convex Factorization Machine for Toxicogenomics Prediction. In *SIGKDD*. ACM, 1215–1224.

[60] Fajie Yuan, Guibing Guo, Joemon M. Jose, Long Chen, Haitao Yu, and Weinan Zhang. 2017. BoostFM: Boosted Factorization Machines for Top-N Feature-based Recommendation. In *IUI*. 45–54.

[61] Ming Yuan and Yi Lin. 2006. Model Selection and Estimation in Regression with Grouped Variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68, 1 (2006), 49–67.

[62] Huan Zhao, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. 2017. Meta-Graph Based Recommendation Fusion over Heterogeneous Information Networks. In *SIGKDD*. ACM, 635–644.

[63] Yong Zheng, Robin D. Burke, and Bamshad Mobasher. 2013. Recommendation with Differential Context Weighting. In *UMAP*. 152–164.

A APPENDIX

A.1 Solution of $\mathbb{KL}\left(q(\tilde{W}, S \mid \rho, \pi) \parallel p(\tilde{W}, S)\right)$

$$\mathbb{KL}\left(q(\tilde{W}, S \mid \rho, \pi) \parallel p(\tilde{W}, S)\right) = \sum_{i=1}^{d} \mathbb{KL}\left(q(\tilde{w}_i) \parallel q(\tilde{w}_i)\right) \sum_{u=1}^{d} \mathbb{KL}\left(q(s_{ui}) \parallel p(s_{ui})\right) +$$
$$\sum_{i=1}^{d} \sum_{j=i+1}^{d} \mathbb{KL}\left(q(\tilde{w}_{ij}) \parallel q(\tilde{w}_{ij})\right) \sum_{u=1}^{m} \mathbb{KL}\left(q(s_{uij} \mid s_{ui}, s_{uj}) \parallel p(s_{uij} \mid s_{ui}, s_{uj})\right),$$

where

$$\mathbb{KL}\left(q(\tilde{w}_i) \parallel p(\tilde{w}_i)\right) = \frac{1}{2}\left(1 + \log \sigma_i^2 - \mu_i^2 - \sigma_i^2\right),$$
$$\mathbb{KL}\left(q(\tilde{w}_{ij}) \parallel p(\tilde{w}_{ij})\right) = \frac{1}{2}\left(1 + \log \sigma_{ij}^2 - \mu_{ij}^2 - \sigma_{ij}^2\right),$$
$$\mathbb{KL}\left(q(s_{ui}) \parallel p(s_{ui})\right) = \pi_{ui} \log \frac{\pi_{ui}}{\pi_1} + (1 - \pi_{ui}) \log \frac{1 - \pi_{ui}}{1 - \pi_1},$$
$$\mathbb{KL}\left(q(s_{uij} \mid s_{ui}, s_{uj}) \parallel p(s_{uij} \mid s_{ui}, s_{uj})\right) =$$
$$(\pi_{ui} + \pi_{uj} - 2\pi_{uij}) \left(\pi_{uij} \log \frac{\pi_{uij}}{\pi_2} + (1 - \pi_{uij}) \log \frac{1 - \pi_{uij}}{1 - \pi_2}\right).$$

A.2 Taking gradients of $\{\pi_{ui}\}$ and $\{\pi_{uij}\}$

We follow [51] to take gradients specifically for discrete variables. To take the gradient for π_{ui}, we marginalize out π_{ui}:

$$\mathbb{E}_q\left[\log p(r(\boldsymbol{x}) \mid \hat{r}(\boldsymbol{x}))\right] = \pi_{ui} \mathbb{E}_{q \setminus \{s_{ui}\}}\left[\log p(r(\boldsymbol{x}) \mid \hat{r}(\boldsymbol{x}), s_{ui} = 1)\right] +$$
$$(1 - \pi_{ui}) \mathbb{E}_{q \setminus \{s_{ui}\}}\left[\log p(r(\boldsymbol{x}) \mid \hat{r}(\boldsymbol{x}), s_{ui} = 0)\right].$$

The gradient of $\log p(r(\boldsymbol{x}) \mid \hat{r}(\boldsymbol{x})^{(l)} \setminus \{\pi_{ui}\})$ over π_{ui} is:

$$\nabla_{\pi_{ui}} = \mathbb{E}_{q \setminus \{s_{ui}\}}\left[\log p(r(\boldsymbol{x}) \mid \hat{r}(\boldsymbol{x}), s_{ui} = 1)\right] -$$
$$\mathbb{E}_{q \setminus \{s_{ui}\}}\left[\log p(r(\boldsymbol{x}) \mid \hat{r}(\boldsymbol{x}), s_{ui} = 0)\right]$$
$$= \frac{1}{L} \sum_{l=1}^{L} \log p(r(\boldsymbol{x}) \mid \hat{r}(\boldsymbol{x})^{(l)}, s_{ui} = 1) - \log p(r(\boldsymbol{x}) \mid \hat{r}(\boldsymbol{x})^{(l)}, s_{ui} = 0).$$

The gradient of π_{uij} can be computed in a similarly way.