

Hierarchical Bayesian Personalized Recommendation: A Case Study and Beyond

Zhexuan Xu, Zitao Liu, Jincheng Wang, Yan Yan, Yongjun Bao, Weipeng Yan

Intelligent Advertising Lab, JD.COM
675 E Middlefield Rd.,
Mountain View, CA, 94043

Abstract

Items in modern recommender systems are often organized in hierarchical structures. These hierarchical structures and the data within them provide valuable information for building personalized recommendation systems. In this paper, we propose a general hierarchical Bayesian learning framework, i.e., *HBayes*, to learn both the structures and associated latent factors. Furthermore, we develop a variational inference algorithm that is able to learn model parameters with fast empirical convergence rate. The proposed *HBayes* is evaluated on two real-world datasets from different domains. The results demonstrate the benefits of our approach on item recommendation tasks, and show that it can outperform the state-of-the-art models in terms of precision, recall, F1 measurement and normalized discounted cumulative gain.

Introduction

Real-world organizations in business domains operate in a multi-item, multi-level environment. Items and their corresponding information collected by these organizations often reflect a hierarchical structure. For examples, products in retail stores are usually stored in hierarchical inventories. News on web pages is created and placed hierarchically in most websites. These hierarchical structures and the data within them provide a large amount of information when building effective recommendation systems. Especially in e-commerce domain, all products are displayed in a site-wide hierarchical catalog and how to build an accurate recommendation engine on top of it becomes one of the keys to majority companies' business success.

However, how to utilize the rich information behind hierarchical structures to make personalized and accurate product recommendations still remains challenging due to the unique characteristics of hierarchical structures and the modeling trade-offs arising from them. Briefly, most well-established recommendation algorithms cannot naturally take hierarchical structures as additional inputs and flattening hierarchical structures usually doesn't work well. It will not only blow up the entire feature space but introduce noise when training the recommendation models. On the other

hand, completely ignoring hierarchies will lead to recommendation inaccuracies. The most common way to alleviate this problem is to feed every piece of data from the hierarchy into a complex deep neural network and hope the network itself can figure out a way to use the hierarchical knowledge. However, such approaches usually behave more like black boxes. They are difficult to debug and cannot provide any interpretation of their intermediate results and outcomes.

In this work, we propose and develop a hierarchical Bayesian, a.k.a., *HBayes*, modeling framework that is able to flexibly capture various relations between items in hierarchical structures from different recommendation scenarios. By introducing latent variables, all hierarchical structures are encoded as conditionally independences in *HBayes* graphical models. Moreover, we develop a variational inference algorithm for learning parameters of *HBayes*.

To illustrate the modeling power of the proposed *HBayes* approach, we introduce *HBayes* by using a real-world apparel recommendation problem as an example. As an illustration, we form apparel styles, product brands and apparel items into a three-level hierarchy. We add additional latent variables as the apparel style membership variables to capture the diverse and hidden style properties of each brand. Furthermore, we include user-specific features into *HBayes* and extend *HBayes* into the supervised learning setting where user feedback events such as clicks and conversions are incorporated. Please note that our *HBayes* framework is not only limited to apparel recommendation. In the end, we show its flexibility and effectiveness on a music recommendation problem as well.

Overall this paper makes the following contributions:

- It presents a generalized hierarchical Bayesian learning framework to learn models from rich data with hierarchies.
- It provides a variational inference algorithm that can learn the model parameters with few iterations.
- We evaluate our *HBayes* and its benefits comprehensively in tasks of apparel recommendation on a real-world data set.
- We test our framework in different recommendation scenarios to show its generalization and applicability.

The remainder of the paper is organized as follows: Section provides a review of existing recommendation al-

gorithms and their extensions in hierarchal learning settings. Section introduces the notations and our generalized HBayes learning framework and its variational inference algorithm. In Section , we conduct experiments in a real-world e-commerce data set to show the effectiveness of our proposed recommendation algorithm in different aspects. In addition, we test our model on a music recommendation data set to illustrate the generalization and extended ability of HBayes. We summarize our work and outline potential future extensions in Section .

Related Work

Recommender Systems

With the explosive growth of digital information over the Internet, the recommender system is considered to be one of the most effective approaches to overcome such information overload. Traditional recommendation algorithms such as item-based approaches learn interactions between users and items and recommend items to users who share similar historical behaviors: collaborative filtering (Sarwar et al. 2001; Su and Khoshgoftaar 2009) and matrix factorization (Rendle, Freudenthaler, and Schmidt-Thieme 2010) are both effective approaches under this category. Content-based approaches including (Lops, de Gemmis, and Semeraro 2011; Liu et al. 2011; Yuan et al. 2015) take use of the auxiliary information of both users and items for recommending items to users that are close in the content space. Furthermore, session based recommender systems are developed by analyzing user session information and user visiting patterns (Gultekin and Paisley 2014; Tang, Hu, and Liu 2013). These particular recommendation techniques take *time* as an additional input dimension, aiming at explicitly modeling user interests over time. Koren (Koren 2010) develops a collaborative filtering type of approach with predictions from static average values combining with a dynamic changing factor. Yin et.al. (Yin et al. 2011) proposes a user-tag-specific temporal interest model to track user interests over time by maximizing the time weighted data likelihood.

Bayesian Approaches for Recommendation

Recently, there are works using Bayesian inferencing for recommendation tasks. Rendle et.al. (Rendle et al. 2009) combines the Bayesian inference and matrix factorization together for learning users implicit feedbacks (click & purchase) that is able to directly optimize the recommendation ranking results. Ben-Elazar et.al. (Ben-Elazar et al. 2017) and Zhang et.al. (Zhang and Koren 2007) take user preference consistency into account and develop a variational Bayesian personalized ranking model for better music recommendation. However, these approaches do not leverage the item structural information when building their Bayesian models. Given the fact that the hierarchical structural information widely exists in real-world recommendation scenarios such as e-commerce, social network, music, etc. failing to utilize such information makes these Bayesian approaches inefficient and inaccurate.

Hierarchical Recommender Systems

Hierarchical information for recommender systems, as mentioned, is a natural yet powerful entity structure that encodes human knowledge by means of tree-based dependency constraints between entities defined by a group of latent parameters. Recommender system hierarchies, in particular, could be either explicit or implicit; either approximate or exact. There are recommendation approaches take use of such information in order to promote items to users who have explicitly visited hierarchically related items or have shown preferences to items that belong to the same sub-categories. In social networks, Shepitsen et.al. (Shepitsen et al. 2008) rely on the hierarchies generated by user-taggings, a.k.a., *folksonomies*, to build a better personalized recommender system. In e-commerce, Wang et.al. (Wang et al. 2018) introduce a hierarchical matrix factorization approach that exploits the intrinsic structural information to alleviate cold-start and data sparsity problems. Despite the fact that these hierarchical recommender systems have received some success, there are still challenges such as: 1. how to infer the hierarchical structure efficiently and accurately if it is not explicit; 2. how to better understand the hierarchical topologies discovered by recommendation approaches; 3. how to utilize the inferred hierarchical information for precise data explanation all remain unsolved.

The HBayes Framework

In this work, we develop our generalized hierarchical Bayesian modeling framework that is able to capture the hierarchical structural relations and latent relations in the real-world recommendation scenarios. In our framework, we incorporate both user specific features and user events into model learning so that more personalized recommendation results can be achieved. Furthermore, we present a variational inference algorithm for the HBayes framework to provide fast parameter estimation.

In the following, we will describe how HBayes works by using apparel recommendation as an illustrative example. Using a real-world example helps explain the latent variables, hierarchy structural relations, and the conditional independence assumptions implied in HBayes. Please note that even we explain HBayes in apparel recommendation, the framework itself can be generalized into other recommendation scenarios with little modification.

Generative Process

In the real-world scenario, each item or product has to come with a brand and a brand may have more than one items in the hierarchical structures. Therefore, we denote each event t as a 4-tuple (Item, Brand, User, IsClick), i.e., $(\mathbf{X}_t, b_t, u_t, y_t)$. \mathbf{X}_t represents the item features associated with event t and y_t is the binary label that indicates whether user u_t has clicked \mathbf{X}_t or not. b_t is the brand of item \mathbf{X}_t .

Furthermore, we expand the hierarchy by a hidden factor, i.e., “style”. Products from each brand b_t tend to exhibit different styles or tastes, which are unknown but exist. In this paper, brands are represented as random mixtures over

Optimization

Since both \mathbf{Z} and Θ defined by the HBayes are unobserved, we cannot learn our HBayes directly. Instead, we infer the expectations of these latent variables and compute the expected log likelihood of the log joint probability with respect to the latent variables distribution, i.e., \mathcal{Q} function defined in eq.(3). In the following, we omit the explicit conditioning on \mathcal{H} for notational brevity.

$$\mathcal{Q} = \int_{\Theta} \sum_{\mathbf{Z}} p(\mathbf{Z}, \Theta | \mathcal{D}) \log(p(\mathcal{D}, \mathbf{Z}, \Theta)) d\Theta \quad (3)$$

From the Bayes rule, we can see that the posteriors distribution of \mathbf{Z} and Θ can be represented by $p(\mathbf{Z}, \Theta | \mathcal{D}) = \frac{p(\mathcal{D}, \mathbf{Z}, \Theta)}{p(\mathcal{D})}$. However, this above distribution is intractable to compute in general (Dickey 1983). To tackle this problem, a wide variety of approximate inference algorithms are developed, such as Laplace approximation (Rue, Martino, and Chopin 2009), variational approximation (Bishop 2006), and Markov Chain Monte Carlo (MCMC) approach (Blei, Ng, and Jordan 2003), etc.

In this work, we choose to solve this problem by using variational Bayes approximation (Bishop 2006). More specifically, we approximate the original posterior distribution $p(\mathbf{Z}, \Theta | \mathcal{D})$ with a tractable distribution $q(\mathbf{Z}, \Theta)$ such that instead of maximizing the \mathcal{Q} function defined in eq.(3), we maximize the variational free energy defined as

$$\mathcal{Q}'(q) = \int_{\Theta} \sum_{\mathbf{Z}} q(\mathbf{Z}, \Theta) \log \frac{p(\mathcal{D}, \mathbf{Z}, \Theta)}{q(\mathbf{Z}, \Theta)} d\Theta \quad (4)$$

which is also equal to minimize the KL divergence of $p(\mathbf{Z}, \Theta | \mathcal{D})$ and $q(\mathbf{Z}, \Theta)$.

Here we choose to apply *Mean Field* approximation technique to approximate $p(\mathbf{Z}, \Theta | \mathcal{D})$, where we assume independence among all different variables (\mathbf{Z} and Θ) and define $q(\mathbf{Z}, \Theta)$ as follows:

$$q(\mathbf{Z}, \Theta) = q(\mathbf{Z}) \cdot \prod_{k=1}^K q(\mathbf{U}_k) \cdot \prod_{j=1}^S q(\mathbf{S}_j) \cdot \prod_{i=1}^B q(\mathbf{B}_i) \cdot q(\mathbf{w}) \cdot q(\boldsymbol{\theta}) \cdot q(\delta_u) \cdot q(\delta_b) \cdot q(\delta_s) \cdot q(\delta_w) \quad (5)$$

where q denotes different distribution functions for notation brevity. Details of choices of different distributions will be discussed in Section .

Sigmoid Approximation The Gaussian priors from our log joint probability (see eq.(2)) are not conjugate to the data likelihood due to the fact that our events are modeled by a sigmoid function (see eq.(1)). In order to conduct tractable inference on $\mathcal{Q}'(q)$, we apply a variational lower bound approximation on eq.(1) that has the ‘‘squared exponential’’ form. Therefore, they are conjugate to the Gaussian priors.

$$\sigma(h_t) \geq \sigma(\xi_t) \exp \left\{ \frac{1}{2} (h_t - \xi_t) - \lambda_t (h_t^2 - \xi_t^2) \right\}$$

where $\lambda_t = \frac{1}{2\xi_t} [\sigma(\xi_t) - \frac{1}{2}]$ and ξ_t is a variational parameter. This lower bound is derived using the convex inequality. The similar problem was discussed in (Jaakkola and Jordan 1997; Jordan et al. 1999).

Therefore, each event likelihood can be expressed as follows:

$$\begin{aligned} [\sigma(h_t)]^{y_t} \cdot [1 - \sigma(h_t)]^{1-y_t} &= \exp(y_t h_t) \sigma(-h_t) \\ &\geq \sigma(\xi_t) \exp(y_t h_t - \frac{1}{2} (h_t + \xi_t) - \lambda_t (h_t^2 - \xi_t^2)) \end{aligned} \quad (6)$$

By using the sigmoid approximation in eq.(6), our variational free energy $\mathcal{Q}'(q)$ (eq.(4)) can be bounded as:

$$\mathcal{Q}'(q) \geq \mathcal{Q}'_{\xi}(q) = \int_{\Theta} \sum_{\mathbf{Z}} q(\mathbf{Z}, \Theta) \log \frac{p_{\xi}(\mathcal{D}, \mathbf{Z}, \Theta)}{q(\mathbf{Z}, \Theta)} d\Theta \quad (7)$$

In the following, we will maximize the lower bound of the variational free energy $\mathcal{Q}'_{\xi}(q)$ for parameter estimation.

Parameter Estimation We develop a Variational Bayes (VB) algorithm for HBayes parameter estimation, where in the E-step, we compute the expectation of the hidden variables \mathbf{Z} and in the M-step, we try to find Θ that maximizes lower bound of the variational free energy $\mathcal{Q}'_{\xi}(q)$ (eq.(7)). In the VB algorithm, we use coordinate ascent variational inference (CAVI) (Bishop 2006) to optimize $\mathcal{Q}'_{\xi}(q)$. CAVI iteratively optimizes each factor of the mean field variational distribution, while holding the others fixed.

update expectation of \mathbf{Z} : We assume each brand’s style membership latent variable is independent and therefore, $q(\mathbf{Z}) = \prod_{i=1}^B q(\mathbf{z}_i)$. For each \mathbf{z}_i , we parameterize $q(\mathbf{z}_i)$ as a multinomial distribution, i.e.,

$$q(\mathbf{z}_i) = \prod_{j=1}^S \mu_{i,j}^{\mathbb{I}(z_{i,j}=1)}$$

and the update rule for $\mu_{i,j}$ is

$$\mu_{i,j} = \frac{\rho_{i,j}}{\sum_{p=1}^S \rho_{i,p}} \quad (8)$$

$$\ln(\rho_{i,j}) = \mathbb{E}[\ln(\theta_j)] + \frac{d}{2} \mathbb{E}[\ln(\delta_b)] - \frac{d}{2} \ln(2\pi) - \frac{1}{2} \mathbb{E}[\delta_b(\mathbf{B}_i - \mathbf{S}_j)^T (\mathbf{B}_i - \mathbf{S}_j)] \quad (9)$$

where the expectation $\mathbb{E}[\cdot]$ is with respect to the (currently fixed) variational density over Θ i.e., $\mathbb{E}[\cdot] = \mathbb{E}_{-\mathbf{Z}}[\cdot]$ in this part. Furthermore, in the following, we note that:

$$\mathbb{E}[z_{i,j}] = \mu_{i,j} \quad (10)$$

Parametrization and update rule of $q(\theta)$: For the style proportion distribution θ , we parameterize $q(\theta)$ as a Dirichlet distribution, i.e., $q(\theta) = \text{Dir}(\theta; \gamma)$, and the update rule for γ are

$$\gamma_j = \gamma_j + \sum_{i=1}^B \mu_{i,j}, j = 1, \dots, S \quad (11)$$

Parametrization and update rule of $q(\mathbf{U}_k)$: For each user $k, k = 1, \dots, U$, we parameterize $q(\mathbf{U}_k)$ as a multivariate normal distribution, i.e., $q(\mathbf{U}_k) = \mathcal{N}(\mathbf{U}_k; \boldsymbol{\mu}_k^u, \boldsymbol{\Sigma}_k^u)$, and the update rule for $\boldsymbol{\mu}_k^u, \boldsymbol{\Sigma}_k^u$ are

$$\boldsymbol{\Sigma}_k^u = [\delta_u \mathbf{I} + \sum_{t=1}^N \mathbb{I}(u_t = k) 2\lambda_t \mathbf{X}_t \mathbf{X}_t^T]^{-1} \quad (12)$$

$$\boldsymbol{\mu}_k^u = \boldsymbol{\Sigma}_k^u \left[\sum_{t=1}^N \mathbb{I}(u_t = k) \left(y_t - \frac{1}{2} - 2\lambda_t \mathbf{X}_t^T \mathbb{E}[\mathbf{B}_{b_t}] \right) \mathbf{X}_t \right] \quad (13)$$

Parametrization and update rule of $q(\mathbf{B}_i)$: For each brand $i, i = 1, \dots, B$, we parameterize $q(\mathbf{B}_i)$ as a multivariate normal distribution, i.e., $q(\mathbf{B}_i) = \mathcal{N}(\mathbf{B}_i; \boldsymbol{\mu}_i^b, \boldsymbol{\Sigma}_i^b)$, and the update rule for $\boldsymbol{\mu}_i^b, \boldsymbol{\Sigma}_i^b$ are

$$\boldsymbol{\Sigma}_i^b = \left[\delta_b \sum_{j=1}^S \mu_{i,j} \mathbf{I} + \sum_{t=1}^N \mathbb{I}(b_t = i) 2\lambda_t \mathbf{X}_t \mathbf{X}_t^T \right]^{-1} \quad (14)$$

$$\boldsymbol{\mu}_i^b = \boldsymbol{\Sigma}_i^b \left[\delta_b \sum_{j=1}^S \mu_{i,j} \mathbb{E}[\mathbf{S}_j] + \sum_{t=1}^N \mathbb{I}(b_t = i) \left(y_t - \frac{1}{2} - 2\lambda_t \mathbf{X}_t^T \mathbb{E}[\mathbf{U}_{u_t}] \right) \mathbf{X}_t \right] \quad (15)$$

Parametrization and update rule of $q(\mathbf{S}_j)$: For each style $j, j = 1, \dots, S$, we parameterize $q(\mathbf{S}_j)$ as a multivariate normal distribution, i.e., $q(\mathbf{S}_j) = \mathcal{N}(\mathbf{S}_j; \boldsymbol{\mu}_j^s, \boldsymbol{\Sigma}_j^s)$, and the update rule for $\boldsymbol{\mu}_j^s, \boldsymbol{\Sigma}_j^s$ are

$$\boldsymbol{\Sigma}_j^s = [\delta_s + \delta_b \sum_{i=1}^B \mu_{i,j}]^{-1} \mathbf{I} \quad (16)$$

$$\boldsymbol{\mu}_j^s = \boldsymbol{\Sigma}_j^s \left[\delta_s \mathbb{E}[\mathbf{w}] + \delta_b \sum_{i=1}^B \mu_{i,j} \mathbb{E}[\mathbf{B}_i] \right] \quad (17)$$

Parametrization and update rule of $q(\mathbf{w})$: For the mean variable of style prior \mathbf{w} , we parameterize $q(\mathbf{w})$ as a multivariate normal distribution, i.e., $q(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}^w, \boldsymbol{\Sigma}^w)$, and the update rule for $\boldsymbol{\mu}^w, \boldsymbol{\Sigma}^w$ are

$$\boldsymbol{\Sigma}^w = [\delta_w + \delta_s \cdot S]^{-1} \mathbf{I} \quad (18)$$

$$\boldsymbol{\mu}^w = \boldsymbol{\Sigma}^w \left[\delta_s \sum_{j=1}^S \mathbb{E}[\mathbf{S}_j] \right] \quad (19)$$

Parametrization and update rule of $q(\delta_u), q(\delta_b), q(\delta_s)$ and $q(\delta_w)$: For all the precision parameters' distributions, we parameterize them as a Gamma distribution, i.e., $p(\delta_*) = \mathcal{G}(\delta_*; \alpha_*, \beta_*)$, where $\delta_* \in \{\delta_w, \delta_s, \delta_u, \delta_b\}$ and the update rule for α and β are

$$\alpha_u = \alpha + \frac{dU}{2}, \quad \beta = \beta + \frac{1}{2} \sum_{k=1}^U \mathbb{E}[\mathbf{U}_k^T \mathbf{U}_k] \quad (20)$$

$$\alpha_b = \alpha + \frac{dB}{2}, \quad \beta = \beta + \frac{1}{2} \sum_{i=1}^B \sum_{j=1}^S \mu_{i,j} \mathbb{E}[(\mathbf{B}_i - \mathbf{S}_j)^T (\mathbf{B}_i - \mathbf{S}_j)] \quad (21)$$

$$\alpha_s = \alpha + \frac{dS}{2}, \quad \beta = \beta + \frac{1}{2} \sum_{j=1}^S \mathbb{E}[(\mathbf{S}_j - \mathbf{w})^T (\mathbf{S}_j - \mathbf{w})] \quad (22)$$

$$\alpha_w = \alpha + \frac{d}{2}, \quad \beta = \beta + \frac{1}{2} \mathbb{E}[\mathbf{w}^T \mathbf{w}] \quad (23)$$

Update rule of ξ : For the variational parameters $\xi_t, t = 1, \dots, N$, in order to maximize $\mathcal{Q}'_\xi(q)$ such that the bound on $\mathcal{Q}'(q)$ is tight (Bishop 2006), the update rule is:

$$\xi_t = \sqrt{\mathbb{E}[(\mathbf{X}_t^T (\mathbf{B}_{b_t} + \mathbf{U}_{u_t}))^2]} \quad (24)$$

Summary The parameter estimation method for the HBayes is summarized by Algorithm 1.

Algorithm 1 Parameter Estimation in HBayes

```

1: INPUT:
2: Hyper-parameters  $\mathcal{H}$ :  $\mathcal{H} = \{\alpha, \beta, \gamma\}$ 
3: Data samples  $\mathcal{D}$ :  $(\mathbf{X}_t, b_t, u_t, y_t), t = 1, \dots, N$ 
4: procedure LEARNING HBAYES
5:   repeat
6:     E-step: compute expectation of  $\mathbf{Z}$  by eq.(10).
7:     M-step: estimate  $\{\mathbf{U}_k\}, \{\mathbf{B}_i\}, \{\mathbf{S}_j\}, \mathbf{w}, \boldsymbol{\theta}, \delta_u, \delta_b, \delta_s, \delta_w \xi_t$  by eq.(11)
      - eq.(24).
8:   until Convergence
9:   return  $\Theta$ 

```

Prediction

In the recommender system, the task is to generate top K products list for each user. So, given the user u^* , it's natural to expose top M products based on the probability of the positive outcome. For the m^{th} item, the probability is calculated as:

$$\begin{aligned} \hat{y}_m &= p(y_m = 1 | \mathbf{X}_m, \mathcal{D}, \mathcal{H}) \approx \int \sigma(h_m) q(\mathbf{Z}, \Theta) d\Theta \\ &= \int \sigma(h_m) \mathcal{N}(h_m | \mu_m, \sigma_m^2) dh_m \approx \sigma\left(\frac{\mu_m}{\sqrt{1 + \pi \sigma_m^2 / 8}}\right) \end{aligned}$$

where h_m is a random variable with Gaussian distribution:

$$\begin{aligned} h_m &= \mathbf{X}_m^T (\mathbf{B}_{b_m} + \mathbf{U}_{u^*}) \sim \mathcal{N}(h_m; \mu_m, \sigma_m^2) \\ \mu_m &= \mathbb{E}[\mathbf{X}_m^T (\mathbf{B}_{b_m} + \mathbf{U}_{u^*})] \\ \sigma_m^2 &= \mathbb{E}[(\mathbf{X}_m^T (\mathbf{B}_{b_m} + \mathbf{U}_{u^*}) - \mu_m)^2] \end{aligned}$$

Experiment

In this section, we conduct several experiments on two data sets: (1) our case study: the real-world e-commerce apparel data set; (2) the publicly available music data set. For both data sets, we compare HBayes against several other *state-of-the-art* recommendation approaches which are briefly mentioned as follows:

HSR (Wang et al. 2015) is an item-based recommendation approach that employs a special non-negative matrix factorization for exploring the implicit hierarchical structure of users and items so the user preference towards certain products is better understood.

HPF (Gopalan, Hofman, and Blei 2015) generates a hierarchical *Poisson* factorization model for better modeling user ratings towards certain items based upon each user’s latent preference. Unlike proposed HBayes, HPF does not leverage the entity content feature for constructing the hierarchical structure.

SVD++ (Mnih and Salakhutdinov 2008; Koren 2008) combines the collaborative filtering and latent factor approaches, so to provide the more accurate neighboring based recommendation results.

CoClustering (George and Merugu 2005) is a collaborative filtering approach based on weighted co-clustering improvements that simultaneously cluster users and items.

Factorization Machine (FM) (Rendle 2010; 2012) combines support vector machines (SVM) with factorization models. It takes the advantage of SVM meanwhile overcomes the feature sparsity issues. In this paper, we adopt the LibFM implementation mentioned in (Rendle 2012) specifically as another baseline.

LambdaMART (Burgess 2010) is the boosted tree version of LambdaRank (Donmez, Svore, and Burgess 2009), which is based on RankNet (Burgess et al. 2005). LambdaMART proves to be a very successful approach for ranking as well as recommendation.

Evaluation Metrics

Throughout the experiments, we compare HBayes against other baselines on the testing held-out dataset under the 5-fold cross-validation settings. For each fold, after fitting the model on the training set, we rank on the testing set by each model, and generate the top K samples with maximal ranking scores for recommendation. Regarding metrics, we adopt precisions, recalls as well as F1-scores for evaluating the item retrieval quality and normalized discounted information gain (NDCG) for evaluating the recommendation ranking quality. More specifically, each metric mentioned is defined as:

$$\begin{aligned} \text{Precision@K} &= \frac{\# \text{ of products clicked in top K}}{K} \\ \text{Recall@K} &= \frac{\# \text{ of products clicked in top K}}{\# \text{ of products the user clicked}} \\ \text{F1-score@K} &= 2 \cdot \frac{\text{Precision@K} \cdot \text{Recall@K}}{\text{Precision@K} + \text{Recall@K}} \end{aligned}$$

discounted cumulative gain (DCG) measures the ranking quality based on the item positions from the resulting list,

which is defined as:

$$\text{DCG@K} = \sum_{i=1}^K \frac{r_i}{\log_2(i+1)} = r_1 + \sum_{i=2}^K \frac{r_i}{\log_2(i+1)}$$

re-rank all potential products to produce the ideal maximal possible DCG through out first K items, and its corresponding DCG score is called the ideal discounted cumulative gain (IDCG). By using both DCG and IDCG, the normalized discounted cumulative gain (NDCG) is defined as:

$$\text{NDCG@K} = \frac{\text{DCG@K}}{\text{IDCG@K}}$$

Recommendation on Apparel Data

The first apparel data set is collected from a large e-commerce company. In this dataset, each sample represents a particular apparel product which is recorded by various features including: categories, titles, and other properties, etc. Meanwhile, the user click information is also recorded and translated into data labels. Throughout the experiment, positive labels indicate that certain recommended products are clicked by the user, whereas negative samples indicate that the recommended products are skipped by the user which usually implies that the user is ‘lack of interest’ towards that certain item. By data cleaning and preprocessing: (1) merging duplicated histories; (2) removing users of too few records, the post-processed data set ends up with **895** users, **81223** products, **5535** brands with **380595** uniquely observed user-item pairs. In average, each user has **425** products records, ranging from **105** to **2048**, and **61.2%** of the users have fewer than **425** product clicking records. For each item, we encode the popularity and category features into a **20** dimensional feature vector; title and product property into a **50** dimensional feature vector. Combining with all features, the total dimension of each sample ends up with **140**.

Feature Analysis The apparel data are composed of four types of features: (1) product popularity; (2) product category; (3) product title; (4) product properties. We briefly explain each feature’s physical meaning and how we process the data as follows:

Product Popularity (POP): product popularity is a measure of the prevalence of certain items in the dataset. In general, customers have preference for a particular product during a period of time. This phenomenon is pretty common for apparel products (*e.g.* apparels’ popularity in certain styles may be affected by certain people or events, especially by those important public figures). For a particular product i , the popularity is defined as: $\text{POP}_i := \frac{n_{x_i}}{\mathcal{N}_{\mathbf{x}}}$, where n_{x_i} are the number of orders or the contribution of gross merchandise volume (GMV) for product i , and $\mathcal{N}_{\mathbf{x}} = \sum_{\forall x_i} n_{x_i}$ is the summation of n_{x_i} across all products in the dataset.

Product Category (CID): In e-commerce, items are clustered into different groups based on the alliance/ similarity of the item functionalities and utilities. In e-commerce

websites, such an explicit hierarchy usually exists. We encode each item’s category into a high dimensional vector via one-hot encoding and adjust the feature weights by the popularity of such category.

Product Title (TITLE): product titles are created by vendors and they are typically in the forms of natural languages indicating the item functionality and utility. Examples could be like ‘*INMAN short sleeve round neck triple color block stripe T-shirt 2017*’. We preprocess the product titles by generating the sentence embedding based on (De Boom et al. 2016). The main idea is to average the wording weights in the title sentence based on the inverse document frequency (IDF) value of each individual word involved.

Product Property Features (PROP): other product metadata features are also provided in the apparel dataset. For instance, the color feature of items takes values such like: ‘black’, ‘white’, ‘red’, etc, and the sizing feature takes values such like: ‘S’, ‘M’, ‘L’, ‘XL’, etc. Similar to category features, product properties are first encoded into binary vectors $x_i \in \{0,1\}^{|N|}$, where N denotes the set of all possible product values. Then the property binary vectors are hashed into fixed length (50) vectors.

On one hand, by utilizing more features HBayes in general reaches better performance in terms of precision-recall metrics. We report PR-AUC in Table (1) to prove this argument; on the other hand, more features need more training time. Figure (8) reports the changes in likelihood against training time cost (in minutes) for each feature use case ¹. It is shown that by only taking POP features, model spends less than 5 minutes to converge while taking POP+CID+TITLE+PROP features, the model needs more than 4 hours in training.

| Features | PR AUC |
|--------------------|--------|
| POP | 0.0406 |
| POP+CID | 0.0414 |
| POP+CID+TITLE | 0.0489 |
| POP+CID+TITLE+PROP | 0.0491 |

Table 1: Model performance under different feature combinations in terms of PR AUC

Performance Comparison We first report the model performance regarding precision, recall, as well as F1-score for HBayes against other baselines in Figure (2,3,4). As shown, when each method recommends fewer number of products ($K = 5$), HBayes does not show the superiority regarding with recalls, with the increment of recommended items, the recall for HBayes becomes much better against others, which implies HBayes is really efficient in terms of finding items that people tend to take interest in. In the sense of precision, HBayes is consistently better than other baseline

¹The experiment is conducted via the same Linux Qual core 2.8 GHz Intel Core i7 MacBook with 16 Gigabytes of memory

| Method | NDCG@5 | NDCG@10 | NDCG@25 | NDCG@50 |
|--------------|---------------|---------------|---------------|---------------|
| CoClustering | 0.1288 | 0.1637 | 0.2365 | 0.3050 |
| NMF | 0.1249 | 0.0156 | 0.2272 | 0.3020 |
| SVD++ | 0.1138 | 0.1487 | 0.2287 | 0.3073 |
| HSR | 0.1266 | 0.1603 | 0.2354 | 0.3107 |
| HPF | 0.1412 | 0.1757 | 0.2503 | 0.3229 |
| FM | 0.1363 | 0.1592 | 0.2291 | 0.3117 |
| LambdaMART | 0.1287 | 0.1585 | 0.2304 | 0.3123 |
| HBayes | 0.1557 | 0.1974 | 0.2871 | 0.3590 |

Table 2: NDCG on apparel recommendations

methods which implies HBayes is much more accurate in terms of item classification under different K . Given the performance of precisions and recalls, HBayes is much better regarding F1-score with different K for apparel recommendation.

Regarding the ranking quality, we use NDCG to report each method’s performance in Table (2). HBayes is superior against other baseline methods through out different K recommended. Specially, HBayes beats the second best HPF at $K = 5$ by 10.3%, at $K = 10$ by 12.4%, at $K = 25$ by 14.7% and at $K = 50$ by 11.2%.

Model Learning Analysis Like mentioned in Section , HBayes learns the latent style clusters, and group different brands of products based on their different hidden style representations. Figure (9) shows the tSNE (Maaten and Hinton 2008) representations of different apparel clusters learned by HBayes and we randomly pick 4 samples out of each cluster and display each product image at the right subfigure. As shown, cluster one which takes the majority proportion of apparel items seems about stylish female youth garment. This intuitively makes sense because the majority of apparel customers are young females for e-commerce websites; as a result, most apparels are focusing on the young female audience as well. The second cluster seems about senior customers who are elder in age. Interestingly, the third cluster and the fourth cluster that are closely tied up are both about young male customers. However, the third cluster seems focusing more on office business garment while the fourth cluster seems more about Korean-pop street styles. This indicates us that the HBayes indeed learns the meaningful intrinsic garment styles from apparel items by leveraging customer behavior data.

Recommendation on Last.fm Music Data

The second data set is collected from Last.fm dataset (Celma 2010) and Free Music Archive (FMA) (Defferrard et al. 2017). Last.fm is a publicly available dataset which contains the whole listening habits (till May, 5th 2009) for 1000 users. FMA is an open and easily accessible dataset providing 917 GiB and 343 days of Creative Commons-licensed audio from 106574 tracks, 16341 artists and 14854 albums, arranged in a hierarchical taxonomy of 161 genres. It also provides full-length and high-quality audios with precomputed features. In our experiment, tracks in Last.fm dataset were further intersected with FMA dataset for better feature generation. The resulting dataset contains 500 users,

■ CoClustering ■ NMF ■ SVD++ ■ HSR ■ HPF ■ FM ■ LambdaMART ■ HBayes

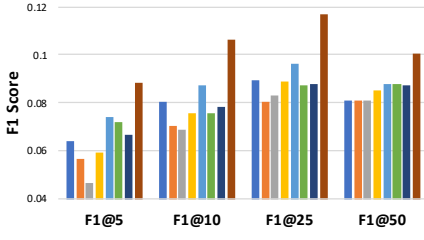


Figure 2: F1@K on *Apparel* data.

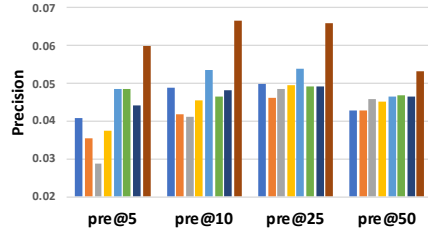


Figure 3: Precision@K on *Apparel* data.

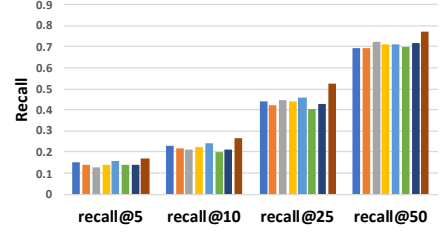


Figure 4: Recall@K on *Apparel* data.

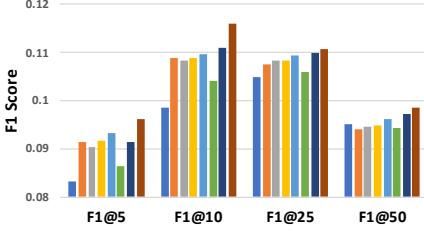


Figure 5: F1@K on *Music* data.

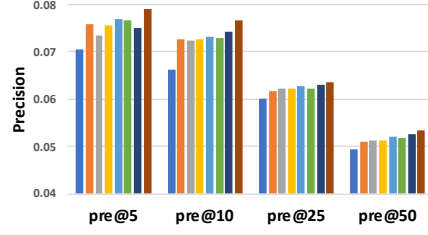


Figure 6: Precision@K on *Music* data.

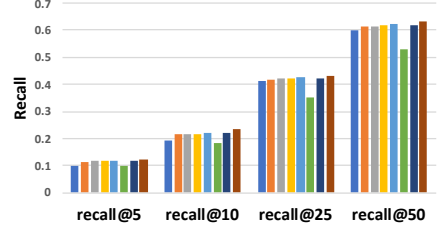


Figure 7: Recall@K on *Music* data.

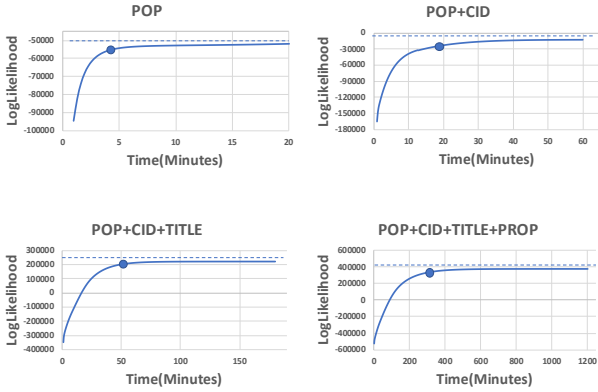


Figure 8: Training time under different feature combinations on e-commerce recommendations

16328 tracks and 36 genres.

Performance Comparison We conduct similar experiments as we do for apparel dataset and report precisions, recalls as well as F1-scores in Figure (5,6,7). Although HPF and HBayes share similar performance regarding recalls along with different K, HBayes is dominant for precisions at different K, especially when K is small (5, 10), which indicates HBayes is very efficient and precise for helping users pick up the songs they prefer even when the recommended item lists are short. Combining the two, HBayes is superior in terms of F1-scores for different K recommended.

For ranking qualities, we report the NDCG performance in Table (3). Similar as e-commerce apparel data, HBayes is the best approach and HPF is the second best one throughout

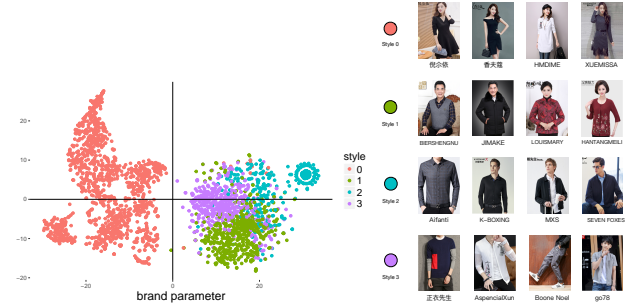


Figure 9: tSNE of four latent apparel style clusters

different K items recommended. Specifically, HBayes beats HPF for 2.7% at K = 5, 4.3% at K = 10, 2.1% at K = 25, and 2.5% at K = 50 separately.

| Method | NDCG@5 | NDCG@10 | NDCG@25 | NDCG@50 |
|--------------|---------------|---------------|---------------|---------------|
| CoClustering | 0.2415 | 0.2314 | 0.2289 | 0.2349 |
| NMF | 0.2556 | 0.2494 | 0.2368 | 0.2431 |
| SVD++ | 0.2493 | 0.2478 | 0.2381 | 0.2439 |
| HSR | 0.2544 | 0.2495 | 0.2384 | 0.2448 |
| HPF | 0.2584 | 0.2513 | 0.2405 | 0.2474 |
| FM | 0.2527 | 0.2453 | 0.2284 | 0.2333 |
| LambdaMART | 0.2372 | 0.2337 | 0.2272 | 0.2218 |
| HBayes | 0.2655 | 0.2620 | 0.2455 | 0.2537 |

Table 3: NDCG on Last.fm recommendations

Conclusion

In this paper, we propose a novel generalized learning framework that learns both the entity hierarchical structure and its latent factors for building a personalized Bayesian recommender system, *HBayes*. By utilizing variational Bayesian inference approach, *HBayes* is able to converge efficiently with few iterations. In empirical studies, we walk through a practical case study of e-commerce apparel data and another publicly available music recommendation data. Experiment results show that *HBayes* beats other *state-of-the-art* recommendation approaches in the sense of precisions, recalls, F1-score, as well as NDCG metrics, due to the fact that *HBayes* is able to extract the data characteristics as well as capture user hidden preferences.

References

- Ben-Elazar, S.; Lavee, G.; Koenigstein, N.; Barkan, O.; Berezin, H.; Paquet, U.; and Zaccai, T. 2017. Groove radio: A bayesian hierarchical model for personalized playlist generation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM '17*, 445–453. New York, NY, USA: ACM.
- Bishop, C. M. 2006. Pattern recognition and machine learning (information science and statistics) springer-verlag new york. *Inc. Secaucus, NJ, USA*.
- Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3(Jan):993–1022.
- Burges, C.; Shaked, T.; Renshaw, E.; Lazier, A.; Deeds, M.; Hamilton, N.; and Hullender, G. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, 89–96. ACM.
- Burges, C. J. 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning* 11(23-581):81.
- Celma, O. 2010. *Music Recommendation and Discovery in the Long Tail*. Springer.
- De Boom, C.; Van Canneyt, S.; Demeester, T.; and Dhoedt, B. 2016. Representation learning for very short texts using weighted word embedding aggregation. *Pattern Recognition Letters* 80:150–156.
- Defferrard, M.; Benzi, K.; Vandergheynst, P.; and Bresson, X. 2017. Fma: A dataset for music analysis. In *18th International Society for Music Information Retrieval Conference*.
- Dickey, J. M. 1983. Multiple hypergeometric functions: Probabilistic interpretations and statistical uses. *Journal of the American Statistical Association* 78(383):628–637.
- Donmez, P.; Svore, K. M.; and Burges, C. J. 2009. On the local optimality of lambdarank. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, 460–467. ACM.
- George, T., and Merugu, S. 2005. A scalable collaborative filtering framework based on co-clustering. In *Data Mining, Fifth IEEE international conference on*, 4–pp. IEEE.
- Gopalan, P.; Hofman, J. M.; and Blei, D. M. 2015. Scalable recommendation with hierarchical poisson factorization. In *UAI*, 326–335.
- Gultekin, S., and Paisley, J. 2014. A collaborative kalman filter for time-evolving dyadic processes. In *Proceedings of the 2014 IEEE International Conference on Data Mining, ICDM '14*, 140–149. Washington, DC, USA: IEEE Computer Society.
- Jaakkola, T., and Jordan, M. 1997. A variational approach to bayesian logistic regression models and their extensions. In *Sixth International Workshop on Artificial Intelligence and Statistics*, volume 82, 4.
- Jordan, M. I.; Ghahramani, Z.; Jaakkola, T. S.; and Saul, L. K. 1999. An introduction to variational methods for graphical models. *Machine learning* 37(2):183–233.
- Koren, Y. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 426–434. ACM.
- Koren, Y. 2010. Collaborative filtering with temporal dynamics. *Commun. ACM* 53(4):89–97.
- Liu, Y.; Miao, J.; Zhang, M.; Ma, S.; and Ru, L. 2011. How do users describe their information need: Query recommendation based on snippet click model. *Expert Systems with Applications* 38(11):13847 – 13856.
- Lops, P.; de Gemmis, M.; and Semeraro, G. 2011. *Content-based Recommender Systems: State of the Art and Trends*. 73.
- Maaten, L. v. d., and Hinton, G. 2008. Visualizing data using t-sne. *Journal of machine learning research* 9(Nov):2579–2605.
- Mnih, A., and Salakhutdinov, R. R. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*, 1257–1264.
- Rendle, S.; Freudenthaler, C.; Gantner, Z.; and Schmidt-Thieme, L. 2009. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, 452–461. AUAI Press.
- Rendle, S.; Freudenthaler, C.; and Schmidt-Thieme, L. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, 811–820. New York, NY, USA: ACM.
- Rendle, S. 2010. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, 995–1000. IEEE.
- Rendle, S. 2012. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)* 3(3):57.
- Rue, H.; Martino, S.; and Chopin, N. 2009. Approximate bayesian inference for latent gaussian models by using integrated nested laplace approximations. *Journal of the royal statistical society: Series b (statistical methodology)* 71(2):319–392.
- Sarwar, B.; Karypis, G.; Konstan, J.; and Riedl, J. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference*

on *World Wide Web*, WWW '01, 285–295. New York, NY, USA: ACM.

Shepitsen, A.; Gemmell, J.; Mobasher, B.; and Burke, R. 2008. Personalized recommendation in social tagging systems using hierarchical clustering. In *Proceedings of the 2008 ACM conference on Recommender systems*, 259–266. ACM.

Su, X., and Khoshgoftaar, T. M. 2009. A survey of collaborative filtering techniques. *Adv. in Artif. Intell.* 2009:4:2–4:2.

Tang, J.; Hu, X.; and Liu, H. 2013. Social recommendation: a review. *Social Network Analysis and Mining* 3(4):1113–1133.

Wang, S.; Tang, J.; Wang, Y.; and Liu, H. 2015. Exploring implicit hierarchical structures for recommender systems. In *IJCAI*, 1813–1819.

Wang, S.; Tang, J.; Wang, Y.; and Liu, H. 2018. Exploring hierarchical structures for recommender systems. *IEEE Transactions on Knowledge and Data Engineering*.

Yin, D.; Hong, L.; Xue, Z.; and Davison, B. D. 2011. Temporal dynamics of user interests in tagging systems. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, AAAI'11, 1279–1285. AAAI Press.

Yuan, Q.; Cong, G.; Zhao, K.; Ma, Z.; and Sun, A. 2015. Who, where, when, and what: A nonparametric bayesian approach to context-aware recommendation and search for twitter users. *ACM Trans. Inf. Syst.* 33(1):2:1–2:33.

Zhang, Y., and Koren, J. 2007. Efficient bayesian hierarchical user modeling for recommendation system. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, 47–54. ACM.