Hierarchical Negative Binomial Factorization for Recommender Systems on Implicit Feedback

Li-Yen Kuo, Ming-Syan Chen
National Taiwan University
No. 1, Sec. 4, Roosevelt Rd.
Taipei 10617, Taiwan
lykuo@arbor.ee.ntu.edu.tw, mschen@ntu.edu.tw

Abstract

When exposed to an item in a recommender system, a user may consume it (known as success exposure) or neglect it (known as failure exposure). The recently proposed methods that consider both success and failure exposure merely regard failure exposure as a constant prior, thus being capable of neither modeling various user behavior nor adapting to overdispersed data. In this paper, we propose a novel model, hierarchical negative binomial factorization, which models data dispersion via a hierarchical Bayesian structure, thus alleviating the effect of data overdispersion to help with performance gain for recommendation. Moreover, we factorize the dispersion of zero entries approximately into two low-rank matrices, thus reducing the updating time linear to the number of nonzero entries. The experiment shows that the proposed model outperforms state-of-the-art Poisson-based methods merely with a slight loss of inference speed.

Introduction

In personalized recommender systems, a principal goal is to recommend each user a set of previously unseen items that she likes and will likely consume in the future. We use historical user feedback, which often appears in the form of user-item relationship, to infer user's preference, and then suggest items according to the inferred preference.

In general, recommendation can be categorized into two modes: explicit rating and implicit feedback. In the first mode, an explicit rating can be regarded as a positive or negative response from a user to an item, which may represent that a user likes an item or not, and then we can use ratings to infer user's preference to predict the missing ratings. Practically, explicit rating is of limited use due to data acquisition difficulty since it needs users to take part in rating directly. In the second mode, implicit feedback expresses one or a set of binary decisions of item consuming (also called implicit count), e.g., clicking, purchasing, and playing, and then we aim to predict which previously unseen items she would like. Compared with its counterpart, implicit feedback, which is obtained from the record of user behavior, is much more easily accessible but difficult to exploit for recommendation.

The current challenges of recommendation on implicit feedback include recommendation performance and computational cost. The Gaussian-based models that model user exposure (Hu, Koren, and Volinsky 2008; Liang et al. 2016) achieve better performance. However, they are not feasible for large datasets with limited computational resources since they require much time and memory to compute an inverse matrix for each user and each item per iteration. By contrast, Poisson-based models (Cemgil 2009; Gopalan, Hofman, and Blei 2015) can efficiently update with time linear to the number of nonzero entries despite sacrificing performance and thus widely applied to various applications, such as recommender systems and topic modeling (Charlin et al. 2015; Schein et al. 2015; Hosseini et al. 2018; da Silva, Langseth, and Ramampiaro 2017). To the best of our knowledge, there have been no works yet to bridge the gap between performance and computational cost, which inspires us to find a trade-off from the perspective of the *negative binomial distribution* (NB) since modeling implicit count as Poisson suffers the following problems.

Firstly, real-world implicit data are often overdispersed (Basbug and Engelhardt 2016; Kuo, Chou, and Chen 2018; Gouvert, Oberlin, and Févotte 2018) since a user may always consume a handful few items she likes (Gopalan, Hofman, and Blei 2015), which causes substantial values of the corresponding entries in a utility matrix. Kuo et al. (Kuo, Chou, and Chen 2018) address that these entries (called outliers in their paper) may disturb optimizing a PF model and the effect of outliers is unavoidable since the consuming distribution follows a power-law distribution approximately in real-world data. Since the problem is attributable to the limited variance of the Poisson distribution, the effect of data overdispersion could be mitigated as long as the variance assumed by the model is larger than the variance of data.

Moreover, the failure exposure, which denotes the event that a user saw an item but neglected it, is difficult to be modeled. The value of an entry in a utility matrix denotes the number of times by which a user consumed an item, which is called *success exposure count* (SEC) in the paper. By contrast, the number of times a user neglected an exposed item is called *failure exposure count* (FEC). To model FEC, prior works (Gouvert, Oberlin, and Févotte 2018; Zhou 2018; Gouvert, Oberlin, and Févotte 2019) assume an entry in a utility matrix to be a negative binomial distribution (NB). However, in those NB-based models, the structure for modeling FEC is single-layered to avoid the non-conjugacy of the Bayesian structure and simplify the computation, thus leading to unsatisfying performance. Since FEC varies with user-item consuming behavior, a hierarchical structure, where FEC is modeled as a latent variable rather than a constant, is required.

In this paper, we propose a novel model, hierarchical negative binomial factorization (HNBF)[1], where a hierarchical Gamma-NB structure is introduced to estimate data dispersion. To the best of our knowledge, HNBF is the first model utilizing hierarchical structure to estimate data dispersion for implicit feedback. By contrast, all the previous works (Basbug and Engelhardt 2016, 2017; Gouvert, Oberlin, and Févotte 2018, 2019) merely utilize a single-layered dispersion model, which is doubted to be adaptive to various user-item behaviors. Our contributions are listed as follows.

- To address the fact that various user-item consuming behaviors lead to different FECs, we propose HNBF, the first model utilizing hierarchical factorized structure to estimate FEC, thus mitigating data overdispersion.
- Even if the hierarchical NB-based structure does not follow conjugate prior, we show it can be updated via variational inference.
- Since the proposed Gamma-NB structure is easily implemented and essentially generic in Bayesian learning, the structure can also be used in many other domains with overdispersed count data, such as ads prediction, infection prediction, and insurance demand analysis, including but not limited to recommendation.
- We discuss the performance and speed gap between Gaussian-based and Poisson-based models. After observing the data property adaptive to Poisson-based methods, we explain why Gaussian-based outperforms better in most cases.

The remainder of the paper is organized as follows. We firstly discuss the related works, including matrix factorization for recommendation and exposure modeling in Section 2. In Section 3, we propose HNBF and FastHNBF, followed by the updating algorithm based on variational inference. In Section 4, a comprehensive empirical analysis is conducted, followed by our conclusion in Section 5.

Related Works

Nonnegative matrix factorizations (NMF) (Lee and Seung 1999, 2001) are widely-used for recommendation on implicit feedback. Compared with Gaussian (Schmidt, Winther, and Hansen 2009), Poisson distribution can represent non-negativity naturally without imposing constraints (Cemgil 2009). Hierarchical Poisson factorization (HPF) (Gopalan et al. 2014; Gopalan, Hofman, and Blei 2015) models the user-item consumption by assuming each entry to be a factorized Poisson. Poisson factorization has several merits: down-weighting the effect of matrix sparsity, modeling the long-tail of users and items, and fast inference.

Despite these merits, HPF performs worse on overdispersed data. To tackle this, Kuo et al. (Kuo, Chou, and Chen 2018) apply pair-wise learning to rank, which merely considers the item permutation for each user to avoid data overdispersion. Apart from personalized ranking, one can also estimate the data dispersion of entries. Basbug et al. (Basbug and Engelhardt 2016, 2017) introduce compound Poisson factorization, which captures the missing-data mechanism by coupling HPF with an arbitrary data-generating model. Gouvert et al. (Gouvert, Oberlin, and Févotte 2018) assume data to be NB and propose negative binomial factorization (NBF) with the high computational cost. Gouvert et al. (Gouvert, Oberlin, and Févotte 2019) also propose a compound PF with a dispersion model to represent user behavior in listening sessions. All the previous compound PF-based methods merely utilize a single-layered dispersion model to be accessible to various distribution assumptions of dispersion for generality, which is doubted to adaptive to various user behaviors. Though negative binomial factor analysis (NBFA) (Zhou 2018) has a hierarchical structure, where an item is represented by a beta variable, we argue that NBFA is not adaptive to the recommender systems since the success probability of NB is related to user-item consuming behavior and should have been modeled in a factorized manner.

Exposure modeling, which tries to clarify the ambiguity associated with zero values, is related to data dispersion estimation. BPR (Rendle et al. 2009) regards user consumption as pair-wise learning to rank problem, thus leading to good performance on a sparse matrix. WMF (Hu, Koren, and Volinsky 2008) considers nonzeros as positive preference feedback and zeros as negative preference feedback to indicate varying confidence levels. Logistic matrix factorization (Johnson 2014) models the probability of entries as the indication via the logistic function. ExpoMF (Liang et al. 2016) directly incorporates user exposure to items into collaborative filtering. Despite the outstanding performance, these methods need much time and memory for updating, and thus they may not be feasible to large data with limited computing resources.

Methodology

In this section, we introduce NB briefly, followed by HNBF and its speed-up version, FastHNBF, where a factorized Poisson-gamma mixture is introduced to approximate the dispersion of zero entries. Then we propose the updating algorithm based on variational inference for FastHNBF and analyze the computational cost.

Preliminary

Given user set $\mathcal{U}$ and item set $\mathcal{I}$, where $|\mathcal{U}| = M$ and $|\mathcal{I}| = N$, consider a nonnegative *utility matrix* $\mathbf{X} \in \mathbb{R}^{M \times N}$, where the value of entry (u, i), denoted by x_{ui}, represents the relationship between u and i, such as song play count and rating. Given two K-dimensional latent factors $\boldsymbol{\theta}_u$ and $\boldsymbol{\beta}_i$, ground truth value x_{ui} of entry (u, i) can be approximated via maximizing probability $p(x_{ui}|\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i)$. We can stack the latent factors $\{\boldsymbol{\theta}_u | u \in \mathcal{U}\}$ and $\{\boldsymbol{\beta}_i | i \in \mathcal{I}\}$ to form matrices $\boldsymbol{\theta} \in \mathbb{R}^{M \times K}$ and $\boldsymbol{\beta} \in \mathbb{R}^{N \times K}$, respectively. The inference score $\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$ can be seen as the estimation of x_{ui}.

NB in this paper is defined as follows. Assume that user u has consumed item i for $\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$ times and neglected i for r_{ui} times. Considering a binary event that denotes whether user u consumes item i or not, the number of successes (SEC) is

[1]Source code and supplementary materials can be downloaded at https://github.com/iankuoli/HNBF

denoted by $\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$, and the number of failures (FEC) is denoted by r_{ui}. As such, user u has been exposed to item i for $(r_{ui} + \boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i)$ times. Thus, the *success probability* is defined by $p_{ui} = \frac{\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i}{r_{ui}+\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i}$ In NBF, observation x_{ui} of entry (u, i), regarded as ground-truth success exposure count, is sampled from the generative process $x_{ui} \sim \mathrm{NB}(r_{ui}, p_{ui})$, where mean is $\mathbb{E}[x_{ui}] = \boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$ and variance is $\mathrm{Var}[x_{ui}] = \boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i(1 + \boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i r_{ui}^{-1})$. When $\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$ is fixed, the variance is controlled by FEC r_{ui}. With the increasing of r_{ui}, which means that the occurred failure exposures increase, the variance is reduced and gradually approaches the expectation because the number of occurred events raises. As such, the NB is reduced to the Poisson distribution when $r_{ui} \to \infty$. By contrast, when we have not observed any occurred failure exposure yet (i.e., $r_{ui} \approx 0$), we cannot estimate the expected number of success exposures certainly, thus leading to large variance, which may represent the fact that a user always plays the same songs on her own initiative. Hence, compared with Poisson, where the mean equals the variance, NB can reasonably model data with large variance.

NB is equivalent to a Poisson-gamma mixture (Lawless 1987; Gardner, Mulvey, and Shaw 1995) denoted by

$$d_{ui} \sim \mathrm{Ga}(r_{ui}, r_{ui}), \quad x_{ui} \sim \mathrm{Poi}(d_{ui}\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i),$$

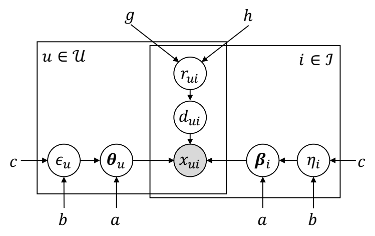
where latent variable d_{ui} denotes the dispersion of variable x_{ui}. Since the variance of d_{ui} is r_{ui}^{-1}, we can control the variability of d_{ui} via tuning r_{ui}. When r_{ui} is too small, d_{ui} will correlate with $\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$ freely, thus leading to overfitting to nonzero entries. In contrast, if r_{ui} is too large, the model will not well estimate overdispersed data since d_{ui} can only vary slightly around its mean (i.e., $\mathbb{E}[d_{ui}] = r_{ui}/r_{ui} = 1$).

In light of this, we introduce a hierarchical Bayesian structure, where r_{ui} is modeled by a latent variable rather than a constant prior. As such, r_{ui} can be estimated precisely according to d_{ui} and its prior. Thus, the proposed model is more feasible and robust to overdispersed data.

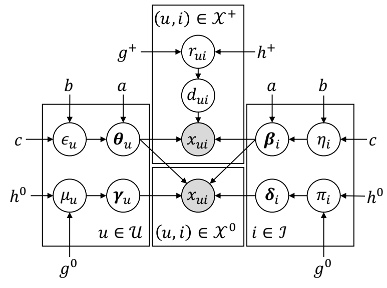
Hierarchical Negative Binomial Factorization

The generative process of HNBF consists of two parts: HPF (Gopalan, Hofman, and Blei 2015) as the *inference model* to estimate the expectation of an entry value; a hierarchical *dispersion model* to estimate its dispersion, as shown in Table 1. Figure 1 (a) shows the Bayesian graphical model of HNBF. Entry value x_{ui} can be defined by a Poisson distribution parameterized in terms of inference $\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$ and dispersion d_{ui}.

In the dispersion model, for each entry (u, i), we introduce a dispersion variable d_{ui} and its prior r_{ui} to estimate the dispersion of x_{ui}. According to the Poisson-gamma mixture, dispersion d_{ui} is a gamma variable with shape and rate parameter that shares the same latent variables, r_{ui}, which represents FEC. When r_{ui} is low, the failure exposure event is seldom observed, which causes high uncertainty, thus raising dispersion d_{ui}. When r_{ui} is high, the failure exposure event has happened many times, which leads to low uncertainty, thus reducing dispersion d_{ui}. Since the success probability $p_{ui} = \frac{\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i}{r_{ui}+\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i}$ of NB is beta distributed, which can be parameterized by two gamma variables $\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$ and r_{ui} according to conjugate prior, we assume variable r_{ui} to be gamma distributed.



(a) HNBF



(b) FastHNBF

Figure 1: The Bayesian graph models of HNBF.

In HNBF, we estimate the dispersion of Poisson variable x_{ui} by introducing a dispersion model, where d_{ui} can be updated precisely by the conditional $p(d_{ui}|x_{ui}, \boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i, r_{ui})$. Though the setting enhances the model expressiveness, HNBF is not practical for large datasets owing to the computational cost $\mathcal{O}(MN)$ caused by the introduced dispersion model, which estimates d_{ui} for each entry (u, i). Thus, to improve the efficiency of model updating, reducing the number of variables is necessary.

FastHNBF

From the observation of Basbug et al. (Basbug and Engelhardt 2016, 2017), the variance of entry value drastically varies when the probability of non-missingness (i.e., the probability of a zero entry being nonzero) equals zero, whereas the nonzero entries (i.e., data with the probability of non-missingness 1) in test datasets often have lower variance. We can explain that the meaning of zero entries is ambiguous because zero entry (u, i) represents two possibilities: 1) u does not like i; 2) u has not seen i yet. Consequently, we develop a *nonzero dispersion model* and a *zero dispersion model* for the dispersion of nonzero entries and zero ones, respectively, because of their different behaviors.

In MF for recommender systems, latent factors are updated primarily according to the ground truth values in nonzero/observed entries. Hence, the precision of the estimation in nonzero entries determines the performance of model inference. In light of this, we model the dispersion of a Poisson variable for each nonzero entry in the *nonzero dispersion model*. By estimating the dispersion individually, one can obtain precise dispersion d_{ui}, thus updating $\boldsymbol{\theta}_u$ and $\boldsymbol{\beta}_i$ precisely by the conditionals $p(\boldsymbol{\theta}_u|x_{ui}, d_{ui}, \boldsymbol{\beta}_i, \epsilon_u, a)$ and $p(\boldsymbol{\beta}_i|x_{ui}, d_{ui}, \boldsymbol{\theta}_u, \eta_i, a)$, respectively.

HNBF	FastHNBF
Dispersion Model	**Dispersion Model**
• $\forall(u,i)$, draw FEC $r_{ui} \sim \mathrm{Ga}(g, \frac{g}{h})$	• $\forall(u,i) \in \mathcal{X}^+$, draw FEC $r_{ui} \sim \mathrm{Ga}(g, \frac{g^+}{h^+})$
• $\forall(u,i)$, draw dispersion $d_{ui} \sim \mathrm{Ga}(r_{ui}, r_{ui})$	• $\forall(u,i) \in \mathcal{X}^+$, draw dispersion $d_{ui} \sim \mathrm{Ga}(r_{ui}, r_{ui})$
	• $\forall u$, draw exposure factor $\mu_u \sim \mathrm{Ga}(g^0, \frac{g^0}{h^0})$
	• $\forall u$, draw dispersion factor $\boldsymbol{\gamma}_u \sim \mathrm{Ga}(\mu_u, \mu_u)$
	• $\forall i$, draw exposure factor $\pi_i \sim \mathrm{Ga}(g^0, \frac{g^0}{h^0})$
	• $\forall i$, draw dispersion factor $\boldsymbol{\delta}_i \sim \mathrm{Ga}(\pi_i, \pi_i)$
Inference Model	**Inference Model**
• $\forall u$, draw user activity $\epsilon_u \sim \mathrm{Ga}(b, \frac{b}{c})$	• $\forall u$, draw user activity $\epsilon_u \sim \mathrm{Ga}(b, \frac{b}{c})$
• $\forall u$, draw user preference $\boldsymbol{\theta}_u \sim \mathrm{Ga}(a, \epsilon_u)$	• $\forall u$, draw user preference $\boldsymbol{\theta}_u \sim \mathrm{Ga}(a, \epsilon_u)$
• $\forall i$, draw item popularity $\eta_i \sim \mathrm{Ga}(b, \frac{b}{c})$	• $\forall i$, draw item popularity $\eta_i \sim \mathrm{Ga}(b, \frac{b}{c})$
• $\forall i$, draw item attribute $\boldsymbol{\beta}_i \sim \mathrm{Ga}(a, \eta_i)$	• $\forall i$, draw item attribute $\boldsymbol{\beta}_i \sim \mathrm{Ga}(a, \eta_i)$
• $\forall(u,i)$, draw score $x_{ui} \sim \mathrm{Poi}(d_{ui}\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i)$	• $\forall(u,i) \in \mathcal{X}^+$, draw score $x_{ui} \sim \mathrm{Poi}(d_{ui}\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i)$
	• $\forall(u,i) \in \mathcal{X}^0$, draw score $x_{ui} \sim \mathrm{Poi}(\frac{1}{K}\boldsymbol{\gamma}_u^\top \boldsymbol{\delta}_i \boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i)$

Table 1. The generative processes of the proposed model, where FEC denotes failure exposure count.

On the other hand, since zero entries are unobserved and most of them are undesirable, precise estimation of dispersion for zero entries is unnecessary. When $\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$ of zero entry (u,i) is large, we have $d_{ui} \ll 1$ since $d_{ui}\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$ should approach its observation $x_{ui} = 0$ to maximize likelihood $p(x_{ui}|d_{ui}\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i)$. As such, considering $d_{ui} \sim \mathrm{Ga}(r_{ui}, r_{ui})$ with variance r_{ui}^{-1} and expectation 1, FEC r_{ui} should be small to make d_{ui} be far from its expectation, i.e., $d_{ui} \ll 1$. Hence, latent variables d_{ui} and r_{ui} are affected by factorized model inference $\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$. Since the inference model is based on MF, we utilize MF to approximate the dispersion of zero entries as well. Although the concept is also applied in prior works (Basbug and Engelhardt 2016, 2017; Liang et al. 2016), none of them utilize a hierarchical structure for the estimation of dispersion. Hence, in the *zero dispersion model*, we factorize the dispersion of zero entries via a hierarchical structure, which comprises *dispersion factors* $\boldsymbol{\gamma} = [\gamma_{uk}] \in \mathbb{R}^{M \times K}$ and $\boldsymbol{\delta} = [\delta_{ik}] \in \mathbb{R}^{N \times K}$, where each entry is gamma distributed, i.e., $\gamma_{uk} \sim \mathrm{Ga}(\mu_u, \mu_u)$ and $\delta_{ik} \sim \mathrm{Ga}(\pi_i, \pi_i)$, where μ_u and π_i denote the factor of FEC with respect to u and i, respectively. We call $\boldsymbol{\mu} = [\mu_u] \in \mathbb{R}^M$ and $\boldsymbol{\pi} = [\pi_i] \in \mathbb{R}^N$ *failure exposure factors*, which differ from FEC r_{ui}.

FastHNBF comprises three models, namely, *inference model*, *nonzero dispersion model* and *zero dispersion model*, each of which is based on the favorable Poisson-gamma structure, as shown in Figure 1 (b), where $\mathcal{X}^+ = \{(u,i)|x_{ui} > 0\}$ denotes the nonzero entry set and $\mathcal{X}^0 = \{(u,i)|x_{ui} = 0\}$ denotes the zero entry set. The generative process is shown in Table 1. By utilizing a factorized dispersion model for zero entries, we can reduce the computational cost from $\mathcal{O}(MN)$ to $\mathcal{O}(|\mathcal{X}^+|)$. Hence, FastHNBF is feasible to large datasets since $|\mathcal{X}^+| \ll MN$ in real-world data.

Variables Estimation

We use variational inference (VI) to estimate the posterior probability. VI is a method for approximating posterior distributions by maximizing the evidence lower bound (ELBO), which is equivalent to minimizing the KL divergence from the true posteriors to the approximate ones (Jordan et al. 1999; Hoffman et al. 2013).

To factorized an entry, let $\mathbf{x}_{ui} = [x_{ui1}, ..., x_{uiK}] \in \mathbb{R}^K$ be a K-dimensional vector of counts in entry (u,i), namely, $x_{ui} = \sum_k x_{uik}$, which can be modeled as a multinomial distribution. The complete conditional for this vector is

$$\boldsymbol{\varphi}_{ui}|x_{ui}, \boldsymbol{\theta}_u, \boldsymbol{\beta}_i \sim \mathrm{Mult}(x_{ui}, \frac{\theta_{uk}\beta_{ik}}{\sum_k \theta_{uk}\beta_{ik}}). \quad (1)$$

Recall that dispersion factors γ_{uk} and δ_{ik}, and dispersion d_{ui} are assumed to be the gamma distribution, of which the rate and the shape parameters share the same variable, which does not follow the conjugate prior relationship. Even so, the variational parameters can be estimated by the definition of ELBO, and thus the traditional coordinate ascent algorithm for fitting the variational parameters is still applicable. Next, we will show how to update this kind of gamma variables. Without loss of generality, we assume that a gamma posterior, $\prod_{i=1}^n \mathrm{Ga}(d_i; r, r)\mathrm{Ga}(r; g, \frac{g}{h})$, is given. In this assumption, both the shape and rate parameters of gamma variable d_i share the same gamma variable, denoted by r, representing the unconjugated structure in HNBF. According to the ELBO, we have the conditional

$$\mathcal{L}(r) = \sum_i^n \mathbb{E}_q[\log \mathrm{Ga}(d_i; r, r)] + \mathbb{E}_q[\log \mathrm{Ga}(r; g, \frac{g}{h})] - \mathbb{E}_q[\log \mathrm{Ga}(r; r^{shp}, r^{rte})] + \text{const}.$$

To derive the coordinate ascent update, we take the gradient

$$\nabla_r \mathcal{L} \propto \mathbb{E}_q[n \log(r) - n\Psi(r) + \frac{g-1}{r} - \frac{r^{shp}-1}{r}] + n + \sum_{i=1}^n \log(d_i) - \sum_{i=1}^n d - \frac{g}{h} + r^{rte},$$

where q is only related to r since q is the variational probability of r. Thus, we split the gradient of $\mathcal{L}$ into two parts: terms with expectation $\mathbb{E}_q[\cdot]$ and the others, each of which

Latent Variable	Type	Complete Conditional	Variational Parameter
ϵ_u	Ga	$b + Ka,\ \frac{b}{c} + \sum_k \theta_{uk}$	$\tilde{\epsilon}_u^{shp}, \tilde{\epsilon}_u^{rte}$
η_i	Ga	$b + Kd,\ \frac{b}{c} + \sum_k \beta_{ik}$	$\tilde{\eta}_i^{shp}, \tilde{\eta}_i^{rte}$
θ_{uk}	Ga	$a + \sum_{i=1}^{N} x_{ui}\varphi_{uik},\ \epsilon_u + \sum_{i\in\mathcal{I}_u^+} \beta_{ik} d_{ui} + \frac{1}{K}\sum_{i\in\mathcal{I}_u^0} \beta_{ik} \sum_{k'=1}^{K} \gamma_{uk'}\delta_{ik'}$	$\tilde{\theta}_{uk}^{shp}, \tilde{\theta}_{uk}^{rte}$
β_{ik}	Ga	$a + \sum_{u=1}^{M} x_{ui}\varphi_{uik},\ \eta_i + \sum_{u\in\mathcal{U}_i^+} \theta_{uk} d_{ui} + \frac{1}{K}\sum_{u\in\mathcal{U}_i^0} \theta_{uk} \sum_{k'=1}^{K} \gamma_{uk'}\delta_{ik'}$	$\tilde{\beta}_{ik}^{shp}, \tilde{\beta}_{ik}^{rte}$
μ_u	Ga	$g^0 + K\mu_u(\log\mu_u - \Psi(\mu_u)),\ \frac{g^0}{h^0} + \sum_{k=1}^{K}\gamma_{uk} - \sum_{k=1}^{K}\log\gamma_{uk} - K$	$\tilde{\mu}_u^{shp}, \tilde{\mu}_u^{rte}$
π_i	Ga	$g^0 + K\pi_i(\log\pi_i - \Psi(\pi_i)),\ \frac{g^0}{h^0} + \sum_{k=1}^{K}\delta_{ik} - \sum_{k=1}^{K}\log\delta_{ik} - K$	$\tilde{\pi}_i^{shp}, \tilde{\pi}_i^{rte}$
γ_{uk}	Ga	$\mu_u,\ \mu_u + \frac{1}{K}\sum_{i\in\mathcal{I}_u^0}\delta_{ik}\sum_{k'=1}^{K}\theta_{uk'}\beta_{ik'}$	$\tilde{\gamma}_{uk}^{shp}, \tilde{\gamma}_{uk}^{rte}$
δ_{ik}	Ga	$\pi_i,\ \pi_i + \frac{1}{K}\sum_{u\in\mathcal{U}_i^0}\gamma_{uk}\sum_{k'=1}^{K}\theta_{uk'}\beta_{ik'}$	$\tilde{\delta}_{ik}^{shp}, \tilde{\delta}_{ik}^{rte}$
d_{ui}	Ga	$r_{ui} + x_{ui},\ r_{ui} + \sum_{k=1}^{K}\theta_{uk}\beta_{ik}$	$d_{ui}^{shp}, d_{ui}^{rte}$
r_{ui}	Ga	$g^+ + r_{ui}(\log r_{ui} - \Psi(r_{ui})),\ \frac{g^+}{h^+} + d_{ui} - \log d_{ui} - 1$	$r_{ui}^{shp}, r_{ui}^{rte}$
$\mathbf{x}_{ui}$	Mult	$\log\theta_{uk} + \log\beta_{ik}$	$\boldsymbol{\varphi}_{ui}$

Table 2. Latent variables, variational parameters and their complete conditionals of FastHNBF.

Algorithm 1 Updating of FastHNBF

```
1:  Input: Utility matrix $\mathbf{X}$, number of latent factors $K$,
2:  Output: Trained latent factors $\boldsymbol{\theta}$ and $\boldsymbol{\beta}$
3:  Initialize variables $(\boldsymbol{\theta}, \boldsymbol{\beta}, \boldsymbol{\epsilon}, \boldsymbol{\eta}, \mathbf{d}, \mathbf{r}, \boldsymbol{\gamma}, \boldsymbol{\delta}, \boldsymbol{\mu}, \boldsymbol{\pi})$;
4:  while not converge do
5:      For each $(u,i) \in \mathcal{X}^+$, update
        $\boldsymbol{\varphi}_{ui} \propto \exp\{\Psi(\theta_{uk}) - \log\theta_{uk} + \Psi(\beta_{ik}) - \log\beta_{ik}\}$;
6:      For each $(u,k)$, update $(\tilde{\theta}_{uk}^{shp}, \tilde{\theta}_{uk}^{rte}, \theta_{uk})$, and
        for each $(i,k)$, update $(\tilde{\beta}_{ik}^{shp}, \tilde{\beta}_{ik}^{rte}, \beta_{ik})$;
7:      For each user $u$, update $(\tilde{\epsilon}_u^{shp}, \tilde{\epsilon}_u^{rte}, \epsilon_u)$, and
        for each item $i$, update $(\tilde{\eta}_i^{shp}, \tilde{\eta}_i^{rte}, \eta_i)$;
8:      For each $(u,i) \in \mathcal{X}^+$, update $d_{ui}$ and $r_{ui}$;
9:      For each $(u,k)$, update $(\tilde{\gamma}_{uk}^{shp}, \tilde{\gamma}_{uk}^{rte}, \gamma_{uk})$, and
        for each $(i,k)$, update $(\tilde{\delta}_{ik}^{shp}, \tilde{\delta}_{ik}^{rte}, \delta ik)$;
10:     For each user $u$, update $(\tilde{\mu}_u^{shp}, \tilde{\mu}_u^{rte}, \mu_u)$, and
        for each item $i$, update $(\tilde{\pi}_i^{shp}, \tilde{\pi}_i^{rte}, \pi_i)$;
11: end while
```

includes a variational parameter respectively. We treat them individually and let the gradient be zero by setting

$$r^{shp} = g + nr\left(\log(r) - \Psi(r)\right), \tag{2}$$

$$r^{rte} = \frac{g}{h} + \sum_{i=1}^{n} d_i - \sum_{i=1}^{n}\log(d_i) - n. \tag{3}$$

The latent variables, their variational parameters and complete conditionals are shown in Table 2, where $\mathcal{I}_u^+ = \{i|(u,i) \in \mathcal{X}^+\}$ denotes the items consumed by u, $\mathcal{I}_u^0 = \{i|(u,i) \in \mathcal{X}^0\}$ denotes the items that has not consumed by u yet, $\mathcal{U}_i^+ = \{u|(u,i) \in \mathcal{X}^+\}$, and $\mathcal{U}_i^0 = \{u|(u,i) \in \mathcal{X}^0\}$.

Updating Algorithm and Complexity Analysis

The updating algorithm is shown in Algorithm 1. In Line 5, we compute the K-factor log expectation of θ_{uk} and β_{ik} for each nonzero entry, which costs $\mathcal{O}(|\mathcal{X}^+|K)$. In Line 6, it costs $\mathcal{O}(|\mathcal{X}^+|K)$ to update variables $\tilde{\theta}_{uk}^{shp}$ and $\tilde{\beta}_{ik}^{shp}$ since $\boldsymbol{\varphi}_{ui}$ of each nonzero entry is computed twice (i.e., for $\tilde{\theta}_{uk}^{shp}$ and $\tilde{\beta}_{ik}^{shp}$). To update parameters $\tilde{\boldsymbol{\theta}}^{rte} = [\tilde{\theta}_{uk}^{rte}] \in \mathbb{R}^{M\times K}$ and $\tilde{\boldsymbol{\beta}}^{rte} = [\tilde{\beta}_{ik}^{rte}] \in \mathbb{R}^{N\times K}$, the corresponding complete conditionals in Table 2 is written in the form of matrix as

$$\tilde{\boldsymbol{\theta}}^{rte} = \boldsymbol{\epsilon}\mathbf{1}^\top + (\mathbf{D} - \overline{\mathbf{D}})\boldsymbol{\beta} + \boldsymbol{\gamma}(\boldsymbol{\delta}^\top\boldsymbol{\beta}), \tag{4}$$

$$\tilde{\boldsymbol{\beta}}^{rte} = \boldsymbol{\eta}\mathbf{1}^\top + (\mathbf{D} - \overline{\mathbf{D}})^\top\boldsymbol{\theta} + \boldsymbol{\delta}(\boldsymbol{\gamma}^\top\boldsymbol{\theta}), \tag{5}$$

where $\mathbf{1} \in \mathbb{R}^K$ denotes a K-dimensional vector of ones, and M-by-N sparse matrices $\mathbf{D} = [d_{ui}]$ and $\overline{\mathbf{D}} = [\boldsymbol{\gamma}_u^\top\boldsymbol{\delta}_i]$ only record the true dispersion and estimated dispersion of nonzero entries $(u,i) \in \mathcal{R}^+$ respectively. Hence, when updating $\tilde{\boldsymbol{\theta}}^{rte}$ and $\tilde{\boldsymbol{\beta}}^{rte}$, we only consider the nonzero entries to compute the second term, which costs $\mathcal{O}(|\mathcal{X}^+|K)$. Computing the third term costs $\mathcal{O}(MK^2 + NK^2)$. In Line 8, it takes $\mathcal{O}(|\mathcal{X}^+|)$ to update d_{ui} and r_{ui} for each nonzero entry. In Line 9, from Table 2, variational parameters $\tilde{\boldsymbol{\gamma}}^{rte} = [\tilde{\gamma}_{uk}^{rte}] \in \mathbb{R}^{M\times K}$ and $\tilde{\boldsymbol{\delta}}^{rte} = [\tilde{\delta}_{ik}^{rte}] \in \mathbb{R}^{N\times K}$ is updated by

$$\tilde{\boldsymbol{\gamma}}^{rte} = \boldsymbol{\mu}\mathbf{1}^\top - \overline{\mathbf{X}}\boldsymbol{\delta} + \boldsymbol{\theta}(\boldsymbol{\beta}^\top\boldsymbol{\delta}), \tag{6}$$

$$\tilde{\boldsymbol{\delta}}^{rte} = \boldsymbol{\pi}\mathbf{1}^\top - \overline{\mathbf{X}}^\top\boldsymbol{\gamma} + \boldsymbol{\beta}(\boldsymbol{\theta}^\top\boldsymbol{\gamma}), \tag{7}$$

where M-by-N sparse matrix $\overline{\mathbf{X}} = [\boldsymbol{\theta}_u^\top\boldsymbol{\beta}_i]$ only records the inference score of nonzero entries. Thus, it costs $\mathcal{O}(|\mathcal{X}^+|K + MK^2 + NK^2)$ for updating. In Line 7 and 10, it costs $\mathcal{O}(MK + NK)$. Since $|\mathcal{X}^+| \gg M, N$ and K is constant, the time complexity of FastHNBF is $\mathcal{O}(|\mathcal{X}^+|)$.

Experimental Results

In this section, we conduct the experiments to compare the performance and speed gap between Gaussian-based and Poisson-based models and discuss their adaptability.

Experimental Settings

Datasets and data partition. The statistics are shown in Table 3. The first three datasets are implicit count data. Last.fm1K has long-tailed item popularity, where some users consume more than 5,000 items, whereas most users consume 50 items in Last.fm2K. In explicit rating data, except for well-known MovieLens datasets, we use Jester2 and EachMovie to test the performance on tall-and-skinny (i.e.,

Dataset	Users	Items	$\lvert\mathcal{X}^+\rvert$	Density	Value Range
Last.fm1K	992	174K	898K	0.520%	0-26K
Last.fm2K	1892	17.6K	93K	0.278%	0-353K
Last.fm360K	359K	293K	18M	0.017%	0-419K
MovieLens100K	943	1682	100K	6.304%	0-5
MovieLens1M	6040	3900	1M	4.245%	0-5
MovieLens20M	138K	25.8K	20M	0.529%	0-5
Jester2	55.6K	140	1.17M	15.03%	0-11
EachMovie	61.2K	1623	2.81M	2.828%	0-5

Table 3. The statistics of the datasets.

$M \gg N$) utility matrices, which is more close to the real-world since movies/artists (items) are much less than the audience (users). We conduct experiments on large datasets, Last.fm360K and MovieLens20M, to show the scalability. We follow the works (Gopalan et al. 2014; Basbug and Engelhardt 2016, 2017) to randomly select 20% of nonzero entries for each dataset to be used as a test set, and randomly select 1% of the nonzeros in each dataset as a validation set.

Competing methods. We include 2 Gaussian-based methods as baselines: WMF (Hu, Koren, and Volinsky 2008) and ExpoMF (Liang et al. 2016). Additionally, 5 Poisson-based baselines are included: HPF (Gopalan, Hofman, and Blei 2015), BHPF (i.e., HPF on binarized data, where nonzeros are set to 1 and zeros are set to 0), PRPF (Kuo, Chou, and Chen 2018), CCPF (Basbug and Engelhardt 2017), and NBF (Gouvert, Oberlin, and Févotte 2018). We compare the difference of performance and updating time between the two groups on various datasets. Since the paper's scope is limited to the recommendation from implicit data, methods for explicit ratings, e.g., PMF (Mnih and Salakhutdinov 2008), are inapplicable to the baselines. Since FastHNBF performs as well as HNBF, we only use FastHNBF in the experiments.

Initialization and stopping criterion. The variational parameters are initialized based on the prior on the corresponding latent variables and include small uniform noises as random offsets. Empirically, the uniform noise covers $[0, 1]$. Since $\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$ is used to infer zero entries x_{ui}, we calculate $p(x_{ui}|\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i)$ per 10 epochs on the validation set by using *validation-based early stopping*.

Results and Discussion

Hyperparameters. According to the proposed model, hyperparameters c, h^+ and h^0 control the expectation of the corresponding variables $\{\epsilon_u, \eta_i\}$, r_{ui} and $\{\mu_u, \pi_i\}$, respectively. In the inference model, prior parameter (a, b, c) is set to $(3, 1, 0.1)$ on implicit count and $(0.3, 0.1, 1)$ on explicit ratings, respectively. Since h priorly represents the expectation of failure exposure, we set (g^+, h^+, g^0, h^0) to $(100, 50, 10, 10^8)$ on implicit count and $(1, 1, 10, 10^6)$ on explicit rating. We empirically set $h^+ = \min(\frac{\mathbb{E}_{\mathcal{X}^+}[x_{ui}]^2}{\mathrm{Var}_{\mathcal{X}^+}[\boldsymbol{\theta}_u \boldsymbol{\beta}_i]}, \frac{\mathbb{E}_{\mathcal{X}^+}[x_{ui}]}{3})$ and $g^+ = h^+$ per iteration during the training phase.

Evaluation criteria. For each method, we select Z unconsumed items with the highest predictive score $\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$ for each user u as the top-Z recommendations during validation and testing. To evaluate the top-Z recommendations, we compute precision@Z, recall@Z, and nDCG@Z individually, followed by computing the mean of the metrics for each user. We calculate the mean and the standard deviation by executing each method for ten runs separately.

Inference speed. HPF updates the fastest since HPF does not consider data dispersion. FastHNBF updates faster than NBF even though the former uses a hierarchical dispersion model because FastHNBF estimates factorized dispersion rather than entry-wise dispersion. Although FastHNBF takes more iterations for convergence, it updates much more efficiently per epoch. Hence, FastHNBF can not only inherit the fast inference of HPF but be robust to overdispersed data, thus performing the best in Poisson-based methods. On the other hand, Gaussian-based methods modeling user exposure (i.e., WMF and ExpoMF) outperform Poisson-based ones with relatively large computing time per epoch. What is worse, they may not be feasible for large datasets due to colossal memory usage. On Last.fm360K and MovieLens20M, they fail to train due to out of memory.

Results on implicit count data. The upper row of Table 4 shows the experimental results[2]. Owing to the limitation of memory and time cost, we only compare four efficient methods with the updating time per epoch linear to the number of nonzero entries on Last.fm360K. FastHNBF outperforms the other Poisson-based methods on the three datasets. NBF performs better than HPF because its dispersion model contributes to the robustness of entry value estimation. Nevertheless, the failure exposure count in NBF is given as a prior, thus limiting the model's capability. In contrast, FastHNBF utilizes a hierarchical Bayesian structure to enhance the capability for estimating dispersion, thus improving the inference performance. Data usually are too overdispersed to be fitted by the models with limited data variance assumption on implicit count, so that BHPF outperforms HPF.

Results on explicit rating data. The results are shown in the lower row of Table 4. Only three methods can run on MovieLens20M owing to the limitation of memory. Since HPF performs better than BHPF, which factorizes a binarized utility matrix, value rather than exposure of nonzero entries is vital to performance on explicit rating. Notice that HNBF outperforms HPF even though data are not overdispersed because the well-estimated dispersion can also be viewed as a correction term of the model inference. The estimation for data dispersion can smooth the updating of latent variables, thus helping the inference model optimize. Hence, HNBF outperforms the Poisson-based models since the hierarchical structure estimates data dispersion more accurately.

Effect of long-tailed item popularity. The performance gap between Gaussian-based models and Poisson-based ones reduces when $M \gg N$, as shown in Table 5. In EachMovie, FastHNBF outperforms ExpoMF in terms of all the criteria except for Prec@5. Poisson-based models are prone to slightly underestimate item popularity (i.e., how many users consumed an item) and user activity (i.e., how many

[2]The experiment is conducted on PC with Quad-Core Intel Core i5 CPU @ 1.4GHz and 16GB main memory.

	Last.fm1K				Last.fm2K				Last.fm360K			
	Prec@5	Rec@5	nDCG@5	time	Prec@5	Rec@5	nDCG@5	time	Prec@5	Rec@5	nDCG@5	time
WMF	**57.8±0.2**	**2.40±0.0**	**59.8±0.3**	68.9	22.3±0.2	10.3±0.1	24.0±0.2	11.56				
ExpoMF	55.3±0.2	2.20±0.0	57.3±0.2	85.1	**24.1±0.2**	**11.2±0.1**	**26.9±0.2**	11.15				
HPF	34.8±1.1	1.39±0.1	35.5±1.2	0.78	14.8±0.5	6.84±0.2	16.0±0.6	0.07	9.17±0.2	4.27±0.1	9.96±0.2	16.21
BHPF	45.4±1.0	1.70±0.1	46.9±1.0	0.81	20.1±0.3	9.29±0.1	22.6±0.3	0.07	10.0±0.2	4.70±0.1	11.0±0.2	16.24
PRPF	46.7±1.1	1.86±0.1	47.6±1.1	1257	20.6±0.5	9.55±0.2	23.0±0.5	14.37	9.95±0.2	4.64±0.1	10.9±0.1	16330
CCPF	33.5±1.6	1.37±0.1	34.3±1.6	111.9	14.1±0.8	6.52±0.4	15.1±0.9	16.42				
NBF	36.4±0.7	1.51±0.0	37.5±0.8	5.96	15.7±0.8	7.29±0.4	17.0±1.0	0.82				
FastHNBF	**50.8±1.0**	**2.00±0.1**	**52.3±0.9**	1.39	**21.0±0.3**	**9.77±0.1**	**23.3±0.2**	0.12	**10.2±0.1**	**4.76±0.0**	**11.1±0.2**	31.06

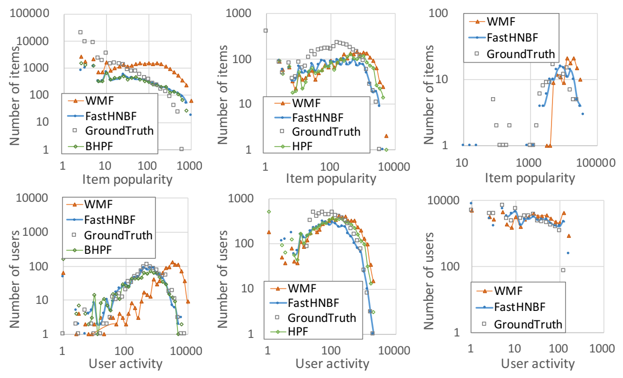
	MovieLens100K				MovieLens1M				MovieLens20M			
	Prec@5	Rec@5	nDCG@5	time	Prec@5	Rec@5	nDCG@5	time	Prec@5	Rec@5	nDCG@5	time
WMF	**45.7±0.2**	**14.7±0.1**	**42.7±0.1**	0.67	**40.8±0.1**	**9.17±0.0**	**38.1±0.0**	10.31				
ExpoMF	44.6±0.2	14.1±0.1	42.1±0.3	0.71	40.4±0.0	9.01±0.0	38.1±0.1	10.23				
HPF	40.2±0.8	12.3±0.4	37.5±0.9	0.06	36.1±0.4	7.46±0.2	33.4±0.3	0.65	32.1±0.3	9.02±0.1	30.0±0.3	17.88
BHPF	39.6±0.9	12.4±0.3	35.5±0.8	0.06	35.9±0.4	7.49±0.2	32.2±0.3	0.65	31.9±0.3	9.11±0.1	29.0±0.3	17.94
PRPF	40.7±0.7	12.5±0.3	37.1±0.7	10.49	35.5±0.5	7.39±0.2	32.5±0.4	441.46				
CCPF	39.0±1.0	11.6±0.3	36.2±0.7	0.74	35.9±0.4	7.45±0.1	33.1±0.4	13.55				
NBF	40.0±0.7	12.2±0.4	37.3±0.8	0.16	36.1±0.3	7.46±0.2	33.4±0.3	1.63				
FastHNBF	**41.2±0.6**	**12.7±0.4**	**38.2±0.6**	0.10	**36.4±0.3**	**7.54±0.2**	**33.6±0.3**	1.20	**32.3±0.3**	**9.13±0.1**	**30.0±0.3**	37.17

Table 4. The result on implicit count and explicit rating data represented in percentage with dimensionality $K = 20$.

	Jester2			
	Prec@5	Rec@5	nDCG@5	time/epoch
WMF	25.7±0.3	24.7±1.0	28.8±0.7	9.29
ExpoMF	**26.0±0.6**	**27.9±1.9**	**30.6±1.5**	5.03
HPF	30.9±0.5	41.2±0.7	40.8±0.7	0.84
BHPF	31.3±0.5	**43.0±0.7**	40.6±0.6	0.85
PRPF	30.9±0.3	41.8±0.8	40.3±0.8	188.85
CCPF	30.7±0.5	41.3±1.3	40.7±0.7	5.28
NBF	30.9±0.3	41.3±1.0	41.1±1.0	1.45
FastHNBF	**31.7±0.4**	42.8±0.9	**41.9±0.7**	1.80

	EachMovie			
	Prec@5	Rec@5	nDCG@5	time/epoch
WMF	**40.0±0.3**	**34.1±1.0**	**50.7±0.7**	45.21
ExpoMF	37.7±0.1	31.0±0.3	47.3±0.2	51.13
HPF	36.8±0.4	32.3±0.2	46.9±0.5	2.18
BHPF	37.0±0.3	32.5±0.1	47.6±0.2	2.16
PRPF	36.7±0.3	32.1±0.3	46.4±0.4	626.81
CCPF	36.4±0.2	32.0±0.2	46.2±0.2	70.60
NBF	36.7±0.3	32.1±0.3	46.7±0.4	5.39
FastHNBF	**37.6±0.4**	**32.5±0.4**	**47.8±0.6**	4.13

Table 5. The result on explicit rating data, where $M \gg N$, represented in percentage with dimensionality $K = 20$.



(a) Last.fm1K (b) MovieLens1M (c) Jester2

Figure 2: Posterior predictive checks of the distributions of item popularity and user activity. All the predicted values larger than 0.8 are set to 1 and the rest are set to 0.

items a user consumed). Gaussian-based methods that consider user exposure often overestimate them, as shown in Figure 2. In Jester2, the item popularity distribution centers at 6,300 and does not appear in long-tail, making WMF badly overestimate item popularity. Thus, all the Poisson-based models perform better on Jester2. By contrast, on a dataset with the long-tail of items, it is difficult for Poisson-based models to fit the entries with values close to 0. Hence, the long-tailed item popularity decreases the performance.

Conclusions

We firstly address that modeling user exposure requires a hierarchical dispersion model due to the overdispersion caused by different failure exposure counts. In light of this, we propose HNBF and its accelerated version, FastHNBF, which outperforms Poisson-based methods with a slight loss of inference speed. In the experiments, we discuss the performance and speed gap between Gaussian-based and Poisson-based models and show that Poisson-based methods perform better on datasets with non-long-tailed item popularity.

References

Basbug, M. E.; and Engelhardt, B. E. 2016. Hierarchical compound Poisson factorization. *arXiv preprint arXiv:1604.03853* .

Basbug, M. E.; and Engelhardt, B. E. 2017. Coupled Compound Poisson Factorization. *arXiv preprint arXiv:1701.02058* .

Cemgil, A. T. 2009. Bayesian inference for nonnegative matrix factorisation models. *Computational intelligence and neuroscience* 2009.

Charlin, L.; Ranganath, R.; McInerney, J.; and Blei, D. M. 2015. Dynamic poisson factorization. In *Proceedings of the 9th ACM Conference on Recommender Systems*, 155–162. ACM.

da Silva, E. d. S.; Langseth, H.; and Ramampiaro, H. 2017. Content-based social recommendation with poisson matrix factorization. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 530–546. Springer.

Gardner, W.; Mulvey, E. P.; and Shaw, E. C. 1995. Regression analyses of counts and rates: Poisson, overdispersed Poisson, and negative binomial models. *Psychological bulletin* 118(3): 392.

Gopalan, P.; Hofman, J. M.; and Blei, D. M. 2015. Scalable Recommendation with Hierarchical Poisson Factorization. In *UAI*, 326–335.

Gopalan, P.; Ruiz, F. J.; Ranganath, R.; and Blei, D. 2014. Bayesian nonparametric poisson factorization for recommendation systems. In *Artificial Intelligence and Statistics*, 275–283.

Gouvert, O.; Oberlin, T.; and Févotte, C. 2018. Negative Binomial Matrix Factorization for Recommender Systems. *arXiv preprint arXiv:1801.01708* .

Gouvert, O.; Oberlin, T.; and Févotte, C. 2019. Recommendation from Raw Data with Adaptive Compound Poisson Factorization. *arXiv preprint arXiv:1905.13128* .

Hoffman, M. D.; Blei, D. M.; Wang, C.; and Paisley, J. 2013. Stochastic variational inference. *The Journal of Machine Learning Research* 14(1): 1303–1347.

Hosseini, S.; Khodadadi, A.; Alizadeh, K.; Arabzadeh, A.; Farajtabar, M.; Zha, H.; and Rabiee, H. R. 2018. Recurrent poisson factorization for temporal recommendation. *IEEE Transactions on Knowledge and Data Engineering* .

Hu, Y.; Koren, Y.; and Volinsky, C. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, 263–272. Ieee.

Johnson, C. C. 2014. Logistic matrix factorization for implicit feedback data. *Advances in Neural Information Processing Systems* 27.

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.

Kuo, L.-Y.; Chou, C.-K.; and Chen, M.-S. 2018. Personalized Ranking on Poisson Factorization. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, 720–728. SIAM.

Lawless, J. F. 1987. Negative binomial and mixed Poisson regression. *Canadian Journal of Statistics* 15(3): 209–225.

Lee, D. D.; and Seung, H. S. 1999. Learning the parts of objects by non-negative matrix factorization. *Nature* 401(6755): 788.

Lee, D. D.; and Seung, H. S. 2001. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, 556–562.

Liang, D.; Charlin, L.; McInerney, J.; and Blei, D. M. 2016. Modeling user exposure in recommendation. In *Proceedings of the 25th International Conference on World Wide Web*, 951–961. International World Wide Web Conferences Steering Committee.

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.

Schein, A.; Paisley, J.; Blei, D. M.; and Wallach, H. 2015. Bayesian poisson tensor factorization for inferring multilateral relations from sparse dyadic event counts. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1045–1054. ACM.

Schmidt, M. N.; Winther, O.; and Hansen, L. K. 2009. Bayesian non-negative matrix factorization. In *International Conference on Independent Component Analysis and Signal Separation*, 540–547. Springer.

Zhou, M. 2018. Nonparametric Bayesian negative binomial factor analysis. *Bayesian Analysis* 13(4): 1061–1089.