Variational Low Rank Multinomials for Collaborative Filtering with Side-Information

Ehtsham Elahi
Netflix Inc.
Los Gatos, CA, USA
eelahi@netflix.com

Wei Wang
Netflix Inc.
Los Gatos, CA, USA
wwang@netflix.com

Dave Ray
Netflix Inc.
Los Gatos, CA, USA
dray@netflix.com

Aish Fenton
Netflix Inc.
Los Gatos, CA, USA
afenton@netflix.com

Tony Jebara
Netflix Inc.
Los Gatos, CA, USA
tjebara@netflix.com

ABSTRACT

We are interested in Bayesian models for collaborative filtering that incorporate side-information or metadata about items in addition to user-item interaction data. We present a simple and flexible framework to build models for this task that exploit the low-rank structure in user-item interaction datasets. Although the resulting models are non-conjugate, we develop an efficient technique for approximating posteriors over model parameters using variational inference. We borrow the "re-parameterization trick" from Bayesian deep learning literature to enable variational inference in our models. The resulting approximate Bayesian inference algorithm is scalable and can handle large scale datasets. We demonstrate our ideas on three real world datasets where we show competitive performance against widely used baselines.

CCS CONCEPTS

• **Mathematics of computing → Variational methods**; *Bayesian computation*; • **Information systems → Collaborative filtering**; **Personalization**.

KEYWORDS

collaborative filtering, variational inference, side-information

ACM Reference Format:
Ehtsham Elahi, Wei Wang, Dave Ray, Aish Fenton, and Tony Jebara. 2019. Variational Low Rank Multinomials for Collaborative Filtering with Side-Information. In *Thirteenth ACM Conference on Recommender Systems (RecSys '19), September 16–20, 2019, Copenhagen, Denmark.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3298689.3347036

1 INTRODUCTION

Let's imagine a data scientist working on a movie recommendation problem. He is provided a dataset of users and movies and their binary interactions (clicks, likes or plays). The data scientist is asked to develop a model for movie recommendations that makes use of the user-movie interaction data. He wants his model to perform well for new movies as well which have little-to-no user interaction (cold start). Hence, he would like to use the metadata of movies like genre, year of release, cast members etc. to help in the cold start of new movies. Before building a model, he notices that users in his dataset have different activity level; some users have played more movies than others. This difference implies that there is more information (or less uncertainty) about the behavior of some users than others. The data scientist would like his model to automatically adjust for both kinds of users; allowing for representing higher uncertainty for users with less data and vice-versa.

Our data scientist decides to model user-movie binary interactions as a categorical dataset. He realizes that it is a classic collaborative filtering setting. Being well versed in machine learning and recommendation algorithms literature, he knows that low rank assumption has been exploited with great success for such problems [10] and he is interested in models that let him discover this structure in his dataset too. He wants these models to be flexible i-e he should be able to extend them easily to incorporate metadata of movies. To handle uncertainty induced by the sparse activity of users, he decides to make use of Bayesian inference.

For categorical datasets, Latent Dirichlet Allocation (LDA) [1] is an attractive choice. LDA has been widely applied to text datasets to recover low rank representation (topics) in documents. For the collaborative filtering task, the users can be viewed as documents and the movies played by them as words. LDA is a fully Bayesian model too. Although exact Bayesian inference is intractable in LDA but techniques like Gibbs sampling or Variational Inference work very well for approximate Bayesian inference. Despite all these benefits, there is one downside to LDA that makes it less suitable for the task at hand of our data scientist; LDA is very hard to extend. LDA has a special model structure that is commonly known as conditionally conjugate. This model structure is very challenging to maintain while extending the model; to make use of metadata of movies for example. Such extensions of LDA are not trivial and generally break the conjugate structure (for example [13] and [14]) and require custom derivations of the inference algorithms using techniques like data augmentation with auxiliary variables [23].

The data scientist then turns his attention to an alternative class of models which are inspired by PCA [20] or Latent Semantic Indexing [17]. He notices that he can encode relationships between any number of variables (like users, movies, metadata of movies) using latent factors associated with each variable and then pass the encoded relationships through an appropriate link function like softmax for converting the encoding into a categorical distribution appropriate for categorical data. He starts calling these models as Low Rank Multinomial Models for his use of low rank latent factors to parameterize multinomial distributions. This modeling approach appears very flexible to him compared to LDA but at the expense of loss of conjugacy. Unlike LDA, approximating posteriors using Gibbs sampling or Mean-field variational inference are not possible for the reason that complete conditionals of latent factors in such models don't have density functions that can be normalized analytically. It seems there is no easy way out for him. Fortunately, he comes across approximate Bayesian inference techniques which have been recently popularized in the Bayesian deep learning community and are very easily applicable to non-conjugate models like his low rank multinomial models.

There have been some very exciting developments in the field of Bayesian deep learning where approximate posterior learning using variational inference has been made possible [8] and [18]. Most of these developments rely on the so-called "re-parameterization trick" which re-writes the gradient over variational parameters as an expectation over simpler distributions. The re-parameterization trick allows the use of low-variance, unbiased Monte Carlo estimators and thus gradient computations become more efficient. The same technique can be used for various low rank multinomial models structures. The optimization work-horse is still stochastic gradient descent or its variants like Adam [7]; note that Monte Carlo estimation of gradients adds another source of stochasticity in addition to the random sub-sampling of data in the SGD algorithm. After learning about this, he finally finds himself at peace at the possibility of accomplishing all of his modeling goals.

To summarize:

- We take a class of bi-linear latent factor models equipped with the softmax link function as a suitable model for learning low rank structure underneath categorical datasets for collaborative filtering and extend it in different ways to allow modeling of additional information like metadata.
- Our model variants, despite being non-conjugate, admit straightforward Bayesian inference and our algorithm for Bayesian inference is highly scalable.
- We show that our various extensions of the model follow a straight forward recipe and do not require custom derivation of approximate Bayesian inference.
- In our experiments on three large scale real world datasets, our models show competitive performance compared to various popular baselines. To show the benefit of Bayesian inference, we show that explicit modeling of uncertainty creates models that can handle data sparsity much more elegantly compare to their point-estimate counterparts.

2 RELATED WORK

There is an extensive literature on bilinear latent factor models. Latent Dirichlet Allocation (LDA) [1] approaches bilinear modeling using mean-field variational inference to represent the uncertainty in document level latent factors while learning the word level latent factors (the topic matrix) via regularized maximum likelihood estimation. Subsequent works [14], [13] and [23] etc. described ways to extend LDA to admit multiple datasets under a single learning task. One of the applications that we showcase is very similar to [2] and [22]. Some generalizations of latent semantic indexing (LSI) or matrix factorization (MF) for similar problems are described in [21]. Bayesian treatment of Gaussian Matrix Factorization is described in [19]. They deal with a conditionally conjugate model structure which allows the use of Gibbs Sampling for approximating posteriors. Another relevant paper which includes item features in a factorization setting is [9] where they also have a similar non-conjugate model structure with the use of Variational inference for posterior approximation. They deal with non-conjugate likelihood (logistic instead of softmax) by replacing it with a lower bound to get an approximate conjugate model whereas our use of re-parameterization gradients makes such approximations unnecessary. [8] describes the re-parameterization trick and its use in variational auto-encoding models.

3 MODEL DETAILS

For describing our models, we consider a setup where we have U users and D items. User u's observation vector $\boldsymbol{r_u}$ is the list of items they have interacted with (liked/purchased). Then each user u in the dataset can be described using a simple model as:

$$\boldsymbol{\eta}_u = \boldsymbol{B}^T \boldsymbol{z}_u,\ \Pr(\boldsymbol{r}_u|\ \boldsymbol{\eta}_u) = \prod_{r_{ud}} \Pr(r_{ud}|\boldsymbol{\eta}_u) \tag{1}$$

where $\boldsymbol{Z} = [\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_U], \boldsymbol{z}_. \sim \mathcal{N}(.|0, \lambda_U^{-1} I_k)$, $\boldsymbol{B} = [\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, ..., \boldsymbol{\beta}_D]$, $\boldsymbol{\beta}_. \sim \mathcal{N}(.|0, \lambda_D^{-1} I_k)$ and $\Pr(r_{ud}|\ \boldsymbol{\eta}_u)$ is the softmax likelihood, $\Pr(r_{ud}|\ \boldsymbol{\eta}_u) = \frac{\exp(\eta_{u r_{ud}})}{\sum_v \exp(\eta_{uv})}$.

In the above model, $\boldsymbol{B}$ is the matrix of item latent factors (sometimes called as factor loading matrix in principal component analysis (PCA) literature) and for each users u we assume a normally distributed latent factor $\boldsymbol{z}_u$ of dimensionality k. $\boldsymbol{\eta}_u$ encodes the relationship between all D items and user u which is then passed through a softmax link function to generate an appropriate likelihood for categorical dataset. We call this a low rank multinomial model to emphasize the low rank structure parameterizing the multinomial distribution through a softmax. We would like to highlight that the main assumption in the above model is the low rank structure underneath the data generating process that many other people have also considered [1] and [6] but we don't go any further to use any special model structure to achieve conditional conjugacy for the purpose of posterior inference. Despite being a non-conjugate model, we'll still be able to learn approximate posteriors distributions.

Being able to extend our model to capture relationships between multiple categorical variables with ease was another important goal of our work. To demonstrate this, we consider a learning task that has been considered in [2], [9] and [22]. The setup now involves interaction of three entities: U users, D documents (or items) and V words. For each document, we are given its meta-data using the vocabulary of V words. Each document's meta-data can provide useful information about the document like year, genre or other qualitative description. We refer to this meta-data as the tags of the documents and represent it as matrix $\mathcal{M}$ of size D x V. Like the first model, each user's observation vector $\boldsymbol{r}_u$ contains the list of documents they have read or liked. We are interested in making use of the extra information about the documents in the generative model of $\boldsymbol{r}_u$. We posit the following extension of the above model for each document d and user u

$$\eta_u = (\boldsymbol{G}\mathcal{M}^T + \boldsymbol{B})^T z_u, \tag{2}$$

where $\boldsymbol{G} = [\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, ..., \boldsymbol{\gamma}_V]$, $\boldsymbol{\gamma}_. \sim \mathcal{N}(.|0, \lambda_V^{-1} I_k)$. $\boldsymbol{Z}, \boldsymbol{B}$ are defined as before and $\boldsymbol{\eta}_u$ parameterizes $\Pr(r_{ud}|\boldsymbol{\eta}_u)$ through a softmax like above. In this model, $\boldsymbol{G}$ is a matrix of latent factors for tags and the matrix-matrix product $\boldsymbol{G}\mathcal{M}^T$ embeds each document in the latent space using its tags information whereas $\boldsymbol{B}$ is included to learn additional structure present in user-document co-occurrence. In essence, we extended the first model by encoding additional bilinear interactions between new categorical variables, combined them in log-additive fashion to introduce coupling between the two model components and then passed the encodings through a softmax function to construct the likelihood appropriate for the categorical dataset. As opposed to the approach of positing another generative model for tags of the document followed in [2] and [22], we *condition* upon the tags of documents to describe a single generative model for only user-level observations $\boldsymbol{r}_u$. We can include user level tags as well in a similar fashion. These are just two examples that demonstrate the flexibility with which we can construct low rank multinomial models for different learning tasks; one of the primary goals that we stated at the beginning.

4 APPROXIMATE POSTERIOR LEARNING

In this section, we discuss the approximate posterior learning for our low-rank multinomial models. We focus on the second variant of our model which deals with the interaction of the three entities; U users, D items/documents and V words. The inference algorithm for the first variant can be easily recovered by ignoring terms related to the document-word interaction. The approximate posterior learning technique presented below is general and can be applied to all latent factors involved in the model. For simplicity, we focus on Bayesian treatment of user-level latent factors $\boldsymbol{Z}$ as for each user we typically observe a few data points and because of this sparsity, our algorithms will benefit the most from uncertainty modeling. We therefore learn the other latent factors $\boldsymbol{B}$ and $\boldsymbol{G}$ by ℓ_2 regularized maximum likelihood. Let $\mathcal{X}$ denote the observed data of user-level observations, then a lower-bound on the log-likelihood $L(\mathcal{X}|\boldsymbol{G}, \boldsymbol{Z}, \mathcal{M})$ can be derived as

$$\begin{aligned}
&= \log\left(\int_Z \prod_u \Pr(z_u)\Pr(\mathbf{r}_u|\eta(z_u))dZ\right) \\
&= \log\left(\int_Z \frac{q(Z)}{q(Z)} \prod_u \Pr(z_u)\Pr(\boldsymbol{r}_u|\eta(z_u))dZ\right) \\
&\geq \mathbb{E}_{q(Z)} \log \frac{\prod_u \Pr(z_u)}{q(Z)} + \mathbb{E}_{q(Z)} L(\mathcal{X}|\boldsymbol{B}, \boldsymbol{G}, \mathcal{M}, Z)
\end{aligned}$$

where $q(\mathrm{Z}) = \prod_u q(z_u) = \prod_u N(z_u|\boldsymbol{\mu}_u, \sigma_u^2 I_k)$. Moreover, we use $\eta(z_u)$ instead of symbol η_u for notational ease of the following mathematical derivation.

The above lower bound is typically referred to as Evidence Lower Bound (ELBO). We have invoked the mean-field assumption to factorize $q(\mathrm{Z})$ and then we choose an isotropic Gaussian distribution as our variational approximation to posterior distribution of z_u. More expressive variational approximations like a diagonal Gaussian are also possible but we take the simplest possible step to enable uncertainty learning. With these two assumptions, we have essentially introduced a scalar parameter σ_u for each z_u to capture uncertainty. We aim to optimize this with respect to the variational parameters $\boldsymbol{\mu}_., \sigma_.$ and the fixed parameters $\boldsymbol{B}$ and $\boldsymbol{G}$ using some gradient based approach. With the stated assumptions on our variational representation, we can further simplify the lower bound as

$$-\sum_u \mathrm{KL}(q(z_u)||\Pr(z_u)) + \sum_u \mathbb{E}_{q(z_u)} \log \Pr(\boldsymbol{r}_u|\eta(z_u)).$$

In the above equation, the KL divergence is between two Gaussian distributions and thus has an analytical form and the third term is simply the log-likelihood of the document level observation vector. Both these terms are straightforward to take gradients of. However, the expectation of log-softmax under the Gaussian variational distribution q is intractable. We now use the re-parameterization trick [8] to first decouple the source of randomness from the variational parameters and then replace the expectations with their Monte Carlo estimate. For each u:

$$\begin{aligned}
\mathbb{E}_{q(z_u)} \log \Pr(\mathbf{r}_u|\eta(z_u)) &= \mathbb{E}_{\varepsilon_u} \log \Pr(\mathbf{r}_u|\eta(\boldsymbol{\mu}_u + \sigma_u \boldsymbol{\varepsilon}_u)) \\
&\approx \frac{1}{S}\sum_{i=1}^{S} \log \Pr(\mathbf{r}_u|\eta(\boldsymbol{\mu}_u + \sigma_u \boldsymbol{\varepsilon}_{u,i}))
\end{aligned}$$

where $\boldsymbol{\varepsilon}_u \sim \mathcal{N}(.|0, I_k)$ and $\boldsymbol{\varepsilon}_{u,i}$ denotes sample number i used in the Monte Carlo estimate of the expectation. With the expectations replaced with their re-parameterized Monte Carlo estimates, the gradients of variational parameters $\boldsymbol{\mu}_.$ and $\sigma_.$ and fixed parameters $\boldsymbol{B}, \boldsymbol{G}$ are straightforward to compute.

4.1 Training Task

We assume that we will be provided with a training dataset which contains interactions between a subset of users, items and words. On the training set, we use Adam to optimize the stochastic objective function that approximates ELBO. Typically we draw one sample of $\boldsymbol{\varepsilon}_u$ for each z_u in every iteration of Adam algorithm to construct the noisy gradient estimates. This, combined with traditional random sub-sampling of data, enables the above derived Bayesian inference algorithm to easily handle large scale datasets. One major benefit of our inference algorithm is that it can be easily implemented in a modern auto-grad tool like TensorFlow and and we can take advantage of all the research and engineering innovation in scalable stochastic optimization. The accompanying code for this paper is available on GitHub[1].

4.2 Prediction Task

Information retrieval is an example of prediction tasks where these low-rank multinomial models can be useful i-e for a user construct a ranking of items in order of relevance. For the second model variant, for example, we can take advantage of user's behavioral data as well as item's tag features to create such ranking. A function

[1]https://github.com/ehtsham/recsys19vlm

that can provide scores for all D items for any user u and can be used for sorting items is

$$\text{score}_u = \mathbb{E}_{q(z_u)}[(G\mathcal{M}^T + B)^T z_u] = (G\mathcal{M}^T + B)^T \mu_u$$

In practice, such models are trained on a subset of the entire dataset and then tested on users which were not part of the training set. To keep things simple, we assume that our training set was large enough so that observing a new user would not change our estimates of B and G as well as variational parameters of $q(Z)$ for users in the training set. Therefore during prediction time, we only need to optimize ELBO with respect to variational parameters of the new/out-of-matrix user and then use the above scoring function to sort items for ranking. Although this can be done using the same stochastic optimization technique described above using Adam. However, through experiments we found that using a conjugate gradient optimizer results in higher quality estimates of these variational parameters instead of Adam. Keeping rest of the model parameters fixed and only optimizing the variational parameter for a new user only requires optimizing $k + 1$ parameters (variational mean and variance), which is straightforward with standard conjugate gradient optimizer since k is generally not too large (100 in our experiments). Since the standard conjugate gradient optimizer does not work with stochastic objective functions, we estimate gradients using a fixed set of samples realized at the beginning of optimization for solving user-level variational parameters. We show the out-of-matrix predictive performance as function of number of samples used for gradient computation in out-of-matrix prediction task in the experiments section.

5 EXPERIMENTS

We use three datasets in our experiments. MovieLens-20M [4], Netflix Prize dataset [2] and Million Song dataset [12]. We pre-process these datasets exactly using the open source code published along with [11]. To summarize the three datasets,

- **MovieLens-20M (ML-20M):** In ML-20M dataset, user-movie interactions are described by numerical 1-5 ratings. To simplify, we interpret any ratings of 4 or higher as a categorical outcome of liking/watching the movie. We only keep users who have watched at least five movies. Also, movies have tags like documents have words which we use to form the metadata matrix $\mathcal{M}$. These tags provide qualitative description of the movie like its genre, year of release etc.
- **Netflix Prize dataset (Netflix):** Similar to ML-20M dataset, user-movie interactions are described by numerical ratings of 1-5 which we binarize and filter similarly.
- **Million Song dataset (MSD):** This dataset describes interaction of users with songs in terms of play count. We binarize these play counts and only keep users with at least 20 songs in their listening history and songs that are listened to by at least 200 users.

In the terminology we have used above, users interact with movies (or songs) to generate $r_.$ (user level observation vector). In the first task, we only capture the relationship between users and watched/liked movies and compute predictions for unseen movies for each test user. This prediction task is typically referred to as collaborative filtering. In the second task, we demonstrate the ability to model the interaction between users, movies and tags using ML-20M dataset. An important benefit of modeling such interactions is to take advantage of metadata of movies when the user-movie interaction data is sparse or the movie is new ("item cold-start"). We still compute predictions for unseen movies for each test user but now our model additionally takes advantage of tags of movies. Table 1 summarizes various attributes of the three datasets. As mentioned earlier, we treat the prediction task as an information retrieval task and use normalized discounted cumulative gain at rank 100 (NDCG@100) for model selection and test metrics. Additionally, we report recall at ranks 20 and 50 (Recall@20 and Recall@50) on test set as well.

[2]http://www.netflixprize.com

Table 1: Description of Datasets

Attribute	ML-20M	Netflix	MSD
# of users	136,677	463,435	571,355
# of items	20,108	17,769	41,140
# of tags	137	-	-
# of user-item interactions	10 M	56.9 M	33.6 M
# of item-tags interactions	61,129	-	-
# of validation users	10 K	40 K	50 K
# of test users	10 K	40 K	50 K

5.1 Collaborative Filtering task

In the collaborative filtering task, we compare the following methods:

- **Variational Low-Rank Multinomial (VLM-1):** This is the model proposed in equation 1. We use a model with 100 latent dimensions (a relatively small number for faster experimentation). Variance for prior distribution for latent factors Z was set to 1.0 and ℓ_2 regularization for B was set to $1e^{-9}$. The learning rate for Adam was set to $4e^{-3}$. For out-of-matrix predictions, number of samples for computing re-parametrization gradient in the conjugate gradient optimizer were set to 100.
- **Low-Rank Multinomial (LM):** This is the identical model as equation 1 but without the Bayesian treatment. All model parameters, including user-level latent factors Z, are learned by regularized maximum likelihood. All hyper-parameters for LM are set to the same values as for VLM-1.
- **Latent Dirichlet Allocation (LDA):** We performed a grid-search on $25 - 500$ (in steps of 50) and $[1e^{-4}, 1e^{-2}, 0.1, 1.0]$) to select the number of topics and document-level concentration parameter whereas the concentration parameter for latent topics' Dirichlet distribution was held fixed to a small value of $1e^{-8}$. We experimented with both stochastic variational inference [5] and collapsed Gibbs sampling [3] and found collapsed Gibbs sampling to give superior results and therefore we use that for comparison. We use a fast multi-threaded implementation of collapsed Gibbs sampling that

lets us sweep number of topics to as large as 500. For all datasets, we found the best results at 500 number of topics.

- **Weighted Matrix Factorization (WMF):** We train WMF [6] with alternating least squares. The training data matrix is treated as a binary matrix with 1's for each user-item interaction and 0's else where. The weight on 0's is held fixed to 1 while we tune the number of latent factors and weights on 1âĂŹs (by grid search on [50, 60, 100, 150, 200] and [2, 5, 10, 30, 50, 100] respectively).
- **SLIM:** SLIM [16] is also a popular model used for collaborative filtering tasks. The model selection was done by a grid-search over the regularization parameters in the model over 0.1, 0.5, 1, 5.

Table 2 summarizes the comparison. For all datasets, the comparison between VLM-1 and LM clearly shows the benefit of learning posteriors over point-estimates. In all but one dataset, the variational low rank multinomial model variant for collaborative filtering task outperformed all baselines significantly. For MSD, SLIM is unable to complete in a reasonable amount of time hence we don't report its results on this dataset. We would like to mention that [11] also used the same data set and got better results (around 3.44% improvement in NDCG@100) on the collaborative filtering task but their model structure is more complex and does not admit straight forward variational inference (see section on annealing in their paper).

With gradients being estimated using the Monte Carlo method detailed above during training and prediction tasks, we would like to see the convergence performance of our algorithms. For model training, the noisy gradients are constructed by a single sample for each user u in each iteration of Adam. Hence the interesting question is to see the convergence with number of iterations. Figure 1 (Above) shows that model converges in a few hundred iterations on ML-20M dataset as accessed by computing Recall@20 of a held-out data point in each training user's observation vector. For the out-of-matrix prediction task, we use a standard conjugate gradient optimizer. Since standard conjugate gradient does not work with stochastic objective functions, we estimate the gradients by a fixed number of samples and we would like to see the out-of-matrix predictive performance as a function of number of samples needed to estimate gradient. Figure 1 (Below) shows the results again on ML-20M dataset. We advise that the optimal number of samples for computing re-parameterization gradient be chosen through cross-validation as it is dependent on the underlying dimensionality of latent space and affects the run-time.

5.2 Collaborative filtering task with additional tag information

In this section, we compare performance of models which make use of additional tag information of items in the collaborative filtering task. As mentioned earlier, we use ML-20M dataset for experiments in this task to make use of metadata of movies. Details of the competing methods are

- **Variational Low-Rank Multinomial with Tags (VLM-2):** This is the model proposed in equation 2. We use the same hyper-parameters as before. The new parameter G also has ℓ_2 regularization set to $1e^{-9}$.

Table 2: Comparison of VLM-1 with LDA, WMF, SLIM and LM on the collaborative filtering task. The standard error is between 0.002 and 0.003 for ML-20M and around 0.001 for Netflix and MSD.

(a) ML-20M

Algo.	Recall@20	Recall@50	NDCG@100
VLM-1	**0.377**	**0.513**	**0.406**
SLIM	0.370	0.495	0.401
WMF	0.360	0.495	0.389
LDA	0.343	0.473	0.374
LM	0.315	0.446	0.346

(b) Netflix

Algo.	Recall@20	Recall@50	NDCG@100
VLM-1	0.331	0.424	0.367
SLIM	**0.347**	**0.428**	**0.379**
WMF	0.316	0.404	0.351
LDA	0.311	0.399	0.346
LM	0.315	0.446	0.346

(c) MSD

Algo.	Recall@20	Recall@50	NDCG@100
VLM-1	**0.254**	**0.350**	**0.304**
SLIM	-	-	-
WMF	0.211	0.312	0.257
LDA	0.163	0.233	0.206
LM	0.217	0.305	0.261

- **MetaLDA (M-LDA):** MetaLDA [23] is an extension of LDA developed to handle document and word level features. MetaLDA was found to be superior to many models (for example models in [13] and [14]) and we would like to see its comparison to our model on the same task. We use MetaLDA's capability to handle word level features to model tags of the items. Number of topics are set to 150 (chosen by grid search on [50, 100, 150, 200, 300]).
- **Collaborative topic Poisson factorization (CTPF):** CTPF [2] is another technique that can jointly model the three datasets. Number of latent factors are set to 125 (chosen by grid search on [50, 60, 75, 100, 125]).

In contrast to CTPF, VLM-2 and M-LDA *condition* on the tags dataset instead of positing a generative model for it. Moreovere, VLM-2 uses a non-conjugate structure compared to a conjugate structure of CTPF. M-LDA uses a locally conjugate structure through auxiliary variable technique.

To ensure that the comparison with CTPF is not confounded because of our more "discriminative" approach compared to a full generative approach of CTPF, we also train a full generative version of variational low rank multinomial model with tags data. Let $\boldsymbol{m}_d$ be the document level vector of tags (d-th row in $\mathcal{M}$). Then a full

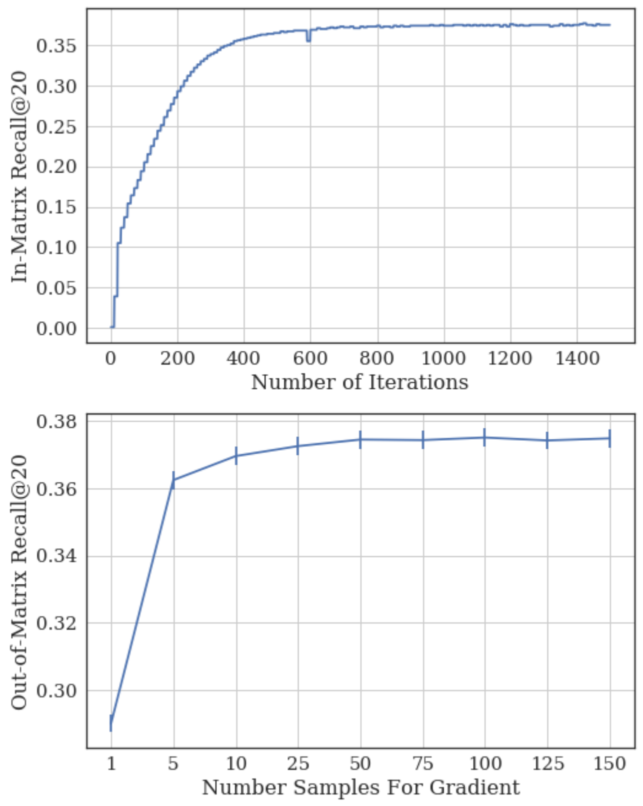

Figure 1: Above: Convergence of SGD with number of training iterations. Below: Out-of-matrix predictive performance with number of samples used to compute re-parameterization gradient

generative model for $\boldsymbol{m}_d$ for each document d and $\boldsymbol{r}_u$ for each user u is

$$\boldsymbol{\epsilon}_d = \boldsymbol{G}^T \hat{\boldsymbol{\beta}}_d, \ \Pr(\boldsymbol{m}_d | \ \boldsymbol{\epsilon}_d) = \prod_{m_{dv}} \Pr(m_{dv} | \boldsymbol{\epsilon}_d) \tag{3}$$

$$\boldsymbol{\eta}_u = (\boldsymbol{B} + \hat{\boldsymbol{B}})^T z_u, \ \Pr(\mathbf{r}_u | \ \boldsymbol{\eta}_u) = \prod_{r_{ud}} \Pr(r_{ud} | \boldsymbol{\eta}_u) \tag{4}$$

where $\hat{\boldsymbol{B}} = [\hat{\boldsymbol{\beta}}_1, \hat{\boldsymbol{\beta}}_2, ..., \hat{\boldsymbol{\beta}}_D]$, $\hat{\boldsymbol{\beta}}_. \sim \mathcal{N}(.|0, \hat{\lambda}_D^{-1} I_k)$. $\Pr(m_{dv}|\boldsymbol{\epsilon}_d)$ and $\Pr(r_{ud}|\boldsymbol{\eta}_u)$ are defined by softmax link as shown earlier. $\boldsymbol{Z}, \boldsymbol{B}$ and $\boldsymbol{G}$ are defined identically as in VLM-2. $\hat{\boldsymbol{B}}$ is another matrix of document/item latent factors and is the shared parameter between the two components of the model and thus acts to share statistical strength between the three entities whereas $\boldsymbol{B}$ is included to learn additional structure present in user-document dataset. This model is defined very similarly to CTPF, the only difference being our use of non-conjugate model structure. We refer to this model as VLM-G. We train this model again with variational approximation for $\boldsymbol{Z}$ and ℓ_2 regularized maximum likelihood estimates for the remaining latent factors.All hyperparameters are set to the same values as for VLM-2 and $\hat{\boldsymbol{B}}$ is given an ℓ_2 regularization of 0.1.

Table 3 details the comparison between all four approaches. CTPF does not perform very well on this dataset and both VLM-2 and VLM-G outperform it significantly. M-LDA also lags behind both VLM-2 and VLM-G significantly.

However, when compared to the metrics of VLM-1 in Table 2, the additional Tags dataset does not seem to help the predictive performance. We believe the additional tags dataset helps predictive performance of items completely new to the model (out-of-matrix items). To validate that, we "censor" out 5% of randomly selected movies (around 1000 movies) from the training data set and then we evaluate model performance on both in-matrix and the "censored" out-of-matrix items. We plot the average ranks of the items by both VLM-1 and VLM-2 in Figure 2 (lower is better). VLM-1 and VLM-2 perform very similarly on in-matrix items whereas VLM-2 outperforms VLM-1 by a big margin on out-of-matrix items. We further analyze this result by slicing the items according to their popularity in the original (uncensored) training data. We assign each item a popularity level where the popularity levels are [0, 25, 50, 75, 100] percentiles of item popularities. An item with popularity at 0 percentile has level 1 (least popular), 0-25 percentile is level 2, 25-50 percentile is level 3, 50-75 percentile is level 4 and 75-100 percentile is level 5 (most popular). For in-matrix items, both VLM-1 and VLM-2 perform similarly at all popularity levels whereas for out-of-matrix items, VLM-2 outperforms VLM-1 at all popularity levels (although popularity does not matter for the out-of-matrix censored items). Moreover, by comparing the average ranks of out-of-matrix items with in-matrix items at different popularity levels, VLM-2 seems to treat out-of-matrix items similar to in-matrix items with popularity level 2.

Table 3: Comparison of Variational LRM with Tags, M-LDA and CTPF using ML-20M dataset. The standard error The standard error is between 0.002 **and** 0.003**.**

Algo.	Recall@20	Recall@50	NDCG@100
VLM-2	**0.377**	**0.515**	**0.406**
VLM-G	0.375	0.510	0.404
M-LDA	0.329	0.453	0.360
CTPF	0.279	0.405	0.314

5.3 Data sparsity and variational distributions

We view Bayesian inference as a powerful computational tool to handle the challenge of data sparsity. Comparison of LM, WMF and VLM-1 in Table 2 is a very strong result suggesting Bayesian approach generalizes much better to heldout data. To further understand the variational approximation, we analyze the variational standard deviation and 2-norm of variational mean as a function of data sparsity. We expect the variational standard deviation to be higher and 2-norm of variational mean to be lower for sparser observation vectors and vice versa for denser observation vectors. To show that, we plot the length of the user-level observation vector against the variational standard deviation and 2-norm of the variational mean in Figure 4. We believe this property is an important reason that our proposed Bayesian model outperforms LM and

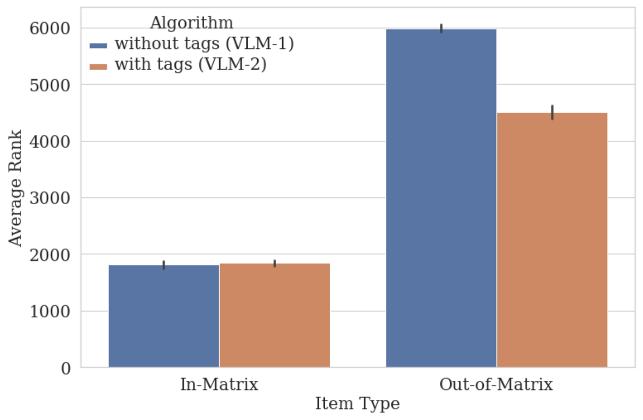

Figure 2: Comparison of average ranks of in-matrix and out-of-matrix items when predicted using VLM-1 and VLM-2. (Lower is better)

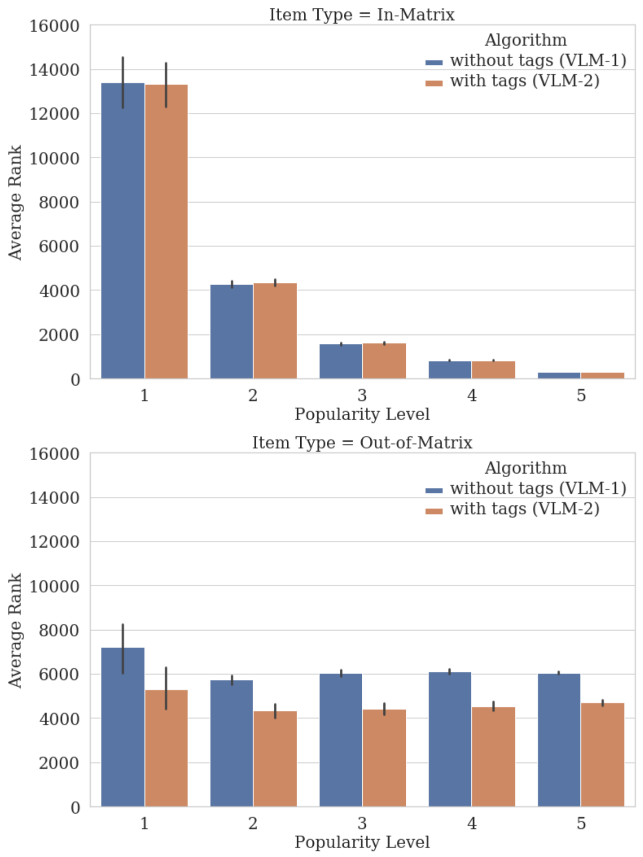

Figure 3: Comparison of average ranks of in-matrix and out-of-matrix items of different popularity levels when predicted using VLM-1 and VLM-2. (Lower is better)

.

WMF that are very similar models but are using point-estimates as it better enables it to avoid over-fitting to sparse data. Compared to LDA which also used a Bayesian inference technique, the latent space of low rank multinomial model is not constrained to a probability simplex and hence is more expressive even with a smaller number of latent factors (compare 100 latent factors vs 500 latent topics).

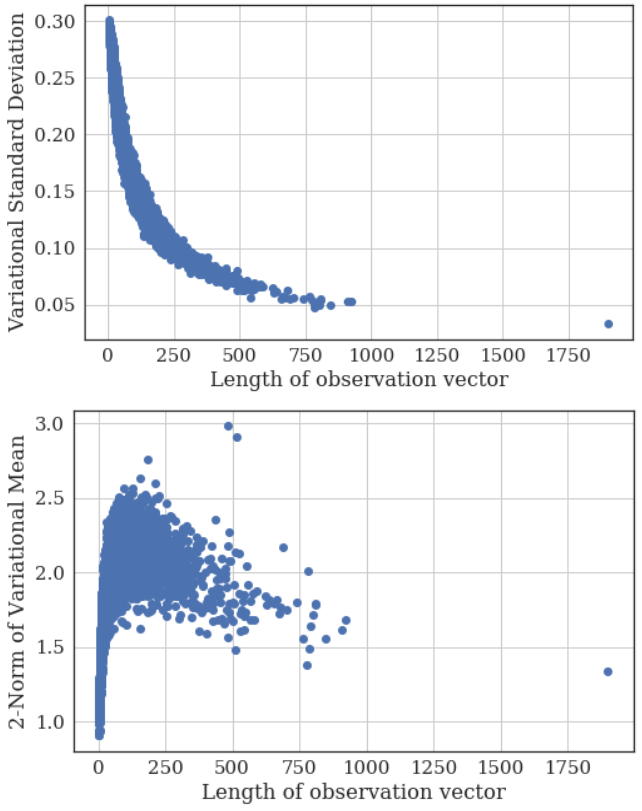

Figure 4: Relationship of mean and standard deviation of the approximate variational distribution with number of observations

5.4 Model Interpretability

With increasingly complex models being used to improve predictive performance, it is becoming more difficult to interpret such models. Model interpretability is not only important to understand the inner workings of the model but also for reasons of transparency and fairness in machine learning which is an increasingly important area of research in the era of Big Data. On that front, LDA has a significant advantage as its recovered latent topics are highly interpretable although its predictive performance is inferior compared to our proposed model. We would like to be able to interpret our proposed model as well. The easiest way to do that is to project the latent representations learned by our model on two-dimensions using some off-the-shelf tool like PCA. The resulting visualization of item representations $\boldsymbol{B}$ learned by the model used in collaborative filtering task is shown in Figure5 where we show around top 200 most popular movies (in training set) in three genres. Although this is a representation recovered by modeling the user behavior, the latent representation is well separated according to genres as well and we can interpret the latent factors of our model as a representation of the "taste" of users in movies.

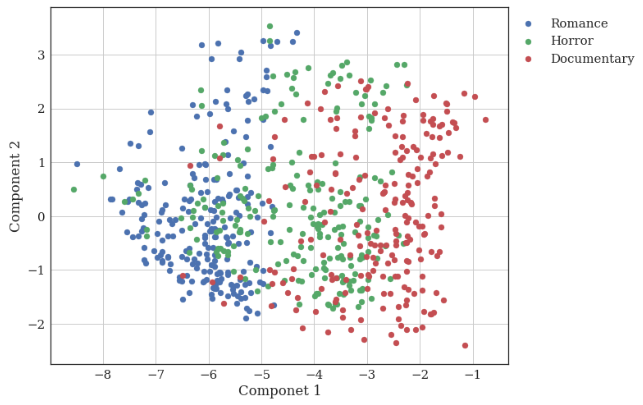

Figure 5: Projection of most popular movies in Horror, Romance and Documentaries genres.

6 CONCLUSION

We presented a flexible framework for constructing latent factor models for categorical datasets with multiple variables. We directly leveraged the widely-acknowledged intuition that low-dimensional latent factors are desirable by directly parameterizing the multinomial distribution with low-rank. Although the resulting models are non-conjugate, we leveraged recent computationally efficient approximations to achieve Bayesian learning of posteriors over the model parameters. We developed a scalable Bayesian inference algorithm and applied that on large scale real world datasets. We compared against widely used techniques and showed competitive results on two different modeling tasks. We attribute the success of the models to the Bayesian treatment as it enables our models to better handle sparsity. In future work, we plan to explore the use of low-discrepancy sampling schemes [15] of multivariate normals for high quality Monte Carlo estimation of gradients with a small number of samples.

REFERENCES

[1] David Blei, Andrew Ng, and Michael Jordan. 2001. Latent Dirichlet allocation. *JMLR* (January 2001), 993–1022.
[2] Prem Gopalan, Laurent Charlin, and David Blei. 2014. Content-based recommendations with Poisson factorization. *NIPS* (2014).
[3] Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *PNAS, 101(Suppl 1)* (2004), 5228–5235.
[4] F. Maxwell Harper and Joseph Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19* (December 2015). https://doi.org/10.1145/2827872
[5] Matthew D. Hoffman, David Blei, Chong Wang, and John Paisley. 2013. Stochastic Variational Inference. *JMLR* (May 2013), 1303–1347.
[6] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. *Eighth IEEE International Conference on Data Mining* (2008), 263–272.
[7] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. *ICLR* (2015).
[8] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. *ICLR* (2014).
[9] Noam Koenigstein and Ulrich Paquet. 2013. Xbox Movies Recommendations: Variational Bayes Matrix Factorization with Embedded Feature Selection. In *Proceedings of the 7th ACM Conference on Recommender Systems (RecSys '13)*. ACM, New York, NY, USA, 129–136. https://doi.org/10.1145/2507157.2507168
[10] Yehuda Koren. 2009. The BellKor Solution to the Netflix Grand Prize. (2009). https://www.netflixprize.com/assets/GrandPrize2009_BPC_BellKor.pdf
[11] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. 2018. Variational autoencoders for collaborative filtering. *WWW* (2018).
[12] Thierry B. Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. 2011. The Million Song Dataset. *ISMIR, Vol.2.10* (2011).
[13] David Mimno and Andrew McCallum. 2008. Topic models conditioned on arbitrary features with Dirichlet-multinomial regression. *UAI* (2008).
[14] Dat Q. Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. 2015. Improving topic models with latent feature word representations. *TACL* (2015).
[15] Harald Niederreiter. 1992. Random Number Generation and Quasi-Monte Carlo Methods. *Society for Industrial and Applied Mathematics* (1992).
[16] Xia Ning and George Karypis. 2011. Slim: Sparse linear methods for top-n recommender systems. *IEEE 11th International Conference on Data Mining (ICDM)* (2011).
[17] Barbara Rosario. 2000. Latent Semantic Indexing: An overview. (2000). https://www.cse.msu.edu/~cse960/Papers/LSI/LSI.pdf
[18] Francisco J. R. Ruiz, Michalis K. Titsias, and David Blei. 2016. The Generalized Reparameterization Gradient. *NIPS* (2016).
[19] Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian Probabilistic Matrix Factorization using Markov Chain Monte Carlo. *ICML* (2008).
[20] Jonathan Shlens. 2005. A Tutorial on Principal Component Analysis. (2005). https://www.cs.cmu.edu/~elaw/papers/pca.pdf
[21] Ajit P. Singh and Geoffrey J. Gordon. 2008. Relational Learning via Collective Matrix Factorization. *KDD* (2008).
[22] Chong Wang and David Blei. 2011. Collaborative Topic Modeling for Recommending Scientific Articles. *KDD* (2011).
[23] He Zhao, Lan Du, Wray Buntine, and Gang Liu. 2017. MetaLDA: a Topic Model that Efficiently Incorporates Meta information. *ICDM* (2017).