fLDA: Matrix Factorization through Latent Dirichlet Allocation

Deepak Agarwal
Yahoo! Research
701 First Ave, Sunnyvale, CA
dagarwal@yahoo-inc.com

Bee-Chung Chen
Yahoo! Research
701 First Ave, Sunnyvale, CA
beechun@yahoo-inc.com

ABSTRACT

We propose fLDA, a novel matrix factorization method to predict *ratings* in recommender system applications where a "bag-of-words" representation for item meta-data is natural. Such scenarios are commonplace in web applications like content recommendation, ad targeting and web search where items are articles, ads and web pages respectively. Because of data sparseness, regularization is key to good predictive accuracy. Our method works by regularizing both user and item factors simultaneously through user features and the bag of words associated with each item. Specifically, each word in an item is associated with a discrete latent factor often referred to as the topic of the word; item topics are obtained by averaging topics across all words in an item. Then, user rating on an item is modeled as user's affinity to the item's topics where user affinity to topics (user factors) and topic assignments to words in items (item factors) are learned jointly in a supervised fashion. To avoid overfitting, user and item factors are regularized through Gaussian linear regression and Latent Dirichlet Allocation (LDA) priors respectively. We show our model is accurate, interpretable and handles both cold-start and warm-start scenarios seamlessly through a single model. The efficacy of our method is illustrated on benchmark datasets and a new dataset from Yahoo! Buzz where fLDA provides superior predictive accuracy in cold-start scenarios and is comparable to state-of-the-art methods in warm-start scenarios. As a by-product, fLDA also identifies interesting topics that explains user-item interactions. Our method also generalizes a recently proposed technique called supervised LDA (sLDA) to collaborative filtering applications. While sLDA estimates item topic vectors in a supervised fashion for a single regression, fLDA incorporates multiple regressions (one for each user) in estimating the item factors.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]: Information filtering

General Terms: Algorithms, Design, Experimentation

WSDM'10, February 4–6, 2010, New York City, New York, USA.

1. INTRODUCTION

Accurate prediction of response associated with dyadic data is an important task in several applications like recommender systems, web applications, online advertising and social networks. For instance, predicting movie ratings by users is an important input to a movie recommender system like Netflix; predicting user click-rates on articles enables a portal like Yahoo! to improve content recommendation; click-rate estimation for ads in the context of a given query by a user is a crucial component of almost all online advertising systems. These are difficult prediction problems that entails several challenges. Data incompleteness, sparseness and non-uniform distribution of available observations across user-item pairs makes it difficult to obtain good performance through simple methods. Other than data sparseness, the *cold-start* problem whereby predictions are required for new dyads adds to the complexity.

Several methods have been proposed to solve the aforementioned problems. Of these, factorization models have become popular and have provided good performance in several real-world applications [25, 5]. The main idea is to predict response y_{ij} (e.g., user i's rating on item j) for a dyad (i, j) through a multiplicative function $u_i'v_j$; u_i and v_j are unknown vectors associated with user i and item j respectively (often referred to as latent factors). Such factorization models provide a flexible class but often overfit even for moderate number of user and item factors; hence, it is imperative to impose constraints through appropriate regularization. Early work in this area [5, 25, 1, 22] regularized factors through a zero-mean Gaussian prior. Recent work [2, 28, 30] improves by incorporating more flexibility in the Gaussian priors through regressions on users and items factors. Such a prior also provides good performance in both cold-start and warm start scenarios. Specifically, ratings for new users and/or new items are predicted by the prior means that are functions of user and item features. However, all previous work assume the factors are vectors in an Euclidean space that does not provide a natural way to incorporate meta-data available as free form text.

In this paper, we propose a new factor model fLDA that is suited to the task of incorporating both rich "bag-of-words" type meta-data on items and user ratings simultaneously to enhance predictions. Such scenarios are commonplace in web applications like content recommendation, advertising and web search. We note that "word" in our context is a general term used to denote elements like *phrases*, *entitites*, *movie actors* in different applications. We show our model provides better accuracy compared to state-of-the-art

factor models when items have rich textual meta-data. As a by-product, interpretable item topics may help in explaining recommendations in some applications. We also show that when rich item meta-data is not available or noisy, our method is still comparable in accuracy to state-of-the-art factorization models.

The key idea of our method is to let the user factors (or profiles) take values in an Euclidean space as in existing factorization models, but assign item factors through a richer prior based on Latent Dirichlet Allocation (LDA) [8]. Specifically, we model the affinity between user i and item j as $s_i'\bar{z}_j$, where $\bar{z}_j$ is a multinomial probability vector representing the soft cluster membership score of item j to K different latent topics; s_i represents user i's affinity to those topics. The main idea in LDA is to attach a *discrete* latent factor to each word of an item that can take K different values (K topics) and produce item topics by averaging the per-word topics in the item. Thus, a news article where 80% of the words are assigned to politics and the rest to education can be thought of as being a political article but perhaps related to an issue in education. Since the number of latent factors in fLDA are large, regularization is key. In LDA, this is done by modeling the word-topic association and the item-topic association, and finally averaging the word topics for each item. In our case, we also include user ratings on items as an additional source of information when determining the item topics. In fact, ratings on items influence the estimation of the *global* word-topic association which in turn influence the *local* assignment of topics to words in each item. For instance, if political articles with mention of word "Obama" are highly rated by many users, they may form a separate topic in fLDA ; this may not happen in unsupervised LDA that is only influenced by word occurrences. In fact, one can think of ratings as providing additional information to attach importance scores to different words in an item; in unsupervised LDA, the scores are solely based on occurence frequencies. The key is the ability of fLDA to learn these scores automatically from the data. We also note that the latent profile of users who rate items play a crucial role in determining item topics and vice-versa. Thus, fLDA will appropriately weight ratings on Obama articles by politically savvy versus politically naive users in deciding topic attribution. This simultaneous estimation of both user profiles and topic attribution makes our method distinct from recent work called sLDA [7] that incorporates a response variable (like reviews on articles) in deciding LDA topics through a *global* regression (while our model performs per-user local regression). In fact, if we assume all users share the same profile, fLDA reduces to sLDA. However, such an assumption is not realistic in the applications we consider whereby user affinity to items play a significant role in obtaining better predictions on ratings.

The topic representation of items in fLDA also provide interpretability and may help in explaining recommendations to users in applications. For well understood topics, user factors can be thought of as providing an interest profile for topics. The LDA model is well known to provide such interpretation since the probability mass of topics tend to be concentrated on a small set of words. This interpretability is important for a number of reasons.

- **Explanation of recommendations:** In some applications, users expect explanations for items that are recommended to them. fLDA provides one such natural explanation in terms of users' affinity to topics, which can be presented, e.g., using word clouds.
- **Content programming:** Consider an application that recommends articles to users visiting a portal [3]. The articles in this application are programmed by human editors to ensure high quality. User interest profiles obtained from fLDA when coupled with visit patterns during different times of the day and week may provide valuable insights that may help editors *program* appropriate content more effectively; e.g., if many users are interested in sports at noon, editors can use this information to ensure enough sports articles before noon.
- **Ad targeting:** Another application where such profiles can be useful is in the context of display advertising where advertizers are targeting their ads to users based on profiles composed of both demographic and browse behavior [10]. A method that obtains interpretable user interest profiles to maximize click-rates on ads can help an ad-network produce better and finer user-targeting attributes which may eventually lead to better ROI for advertisers and increased revenue for the ad-network.

We also emphasize that fLDA does not compromise on predictive accuracy in warm-start scenarios where an item receives a large number of ratings due to its supervised nature. In fact, it is as accurate as the usual state-of-the-art factor models in applications without item meta-data. However, providing a topic interpretation in such cases may become difficult. Through the LDA model, it also provides a powerful method of incorporating meta-data in applications where items have some form of document-like representation and, similar to previous work, it provides a single unified model to tackle both warm-start and cold-start scenarios in a seamless way.

Contributions: Our contributions are as follows. We provide fLDA, a new factorization model based on Latent Dirichlet Allocation for predicting dyadic response that is both accurate and interpretable when items have a bag-of-words like representation. While LDA has been explored in the context of recommender systems [17, 21], using LDA to regularize factorization models based on bag-of-word item meta-data has not been previously studied (see Section 5 for details). Our method works by generalizing the supervised LDA model [7] to perform multiple regressions (one for each user) simultaneously on the item topics. Unlike previous supervised LDA research, we perform exact model fitting through a Monte Carlo EM algorithm and do not rely on variational approximations. We illustrate the performance of our method on benchmark datasets and a new dataset obtained from Yahoo! Buzz on which we obtain significant improvements in predictive performance relative to state-of-the-art factor models.

2. MODEL

In this section, we define the fLDA model. We begin with a high level overview of our model and point out the key differences from previous work. This is followed by a detailed mathematical specification. We then describe our fitting procedure based on Monte-Carlo EM (MCEM) algorithm in the next section.

2.1 Overview

We shall use (i, j) to denote a user-item dyad and the response y_{ij} will be referred to as rating user i gives to item j.

For instance, items may be articles in a content optimization problem, movies in movie recommender systems or ads in online advertising. Ratings may be explicit ratings or clicks. In particular, we are interested in scenarios where each item has a natural bag-of-words representation. Needless to say this is pervasive in web applications.

Our prediction method is based on fitting a two-stage hierarchical mixed-effects model to training data. In particular, we attach latent factors $(\alpha_i, s_i^{K\times 1})$ to user i and $(\beta_j, \bar{z}_j^{K\times 1})$ to item j. Item factors $\bar{z}_j$ are obtained by averaging $\{z_{jn}\}$,

$$\bar{z}_j = \sum_{n=1}^{W_j} \frac{z_{jn}}{W_j},$$

where z_{jn} is a discrete latent factor[1] (with K possible topics) that is attached to the nth word in item j, and W_j is the number of words in item j. As we will see, this is a key difference between fLDA and existing factor models like [2] that attach K continuous latent factors $v_j^{K\times 1}$ to each item.

The fLDA model specifies a generative process of ratings and words in two stages. The first stage specifies the relationship between ratings y_{ij} for known latent factors. In fact, the mean of y_{ij} (to be more precise, some monotone function of the mean) and latent factors are connected through an easily interpretable bilinear function of factors

$$\alpha_i + \beta_j + s_i' \bar{z}_j,$$

where α_i is the rating bias of user i, β_j is the item bias representing the global popularity of item j, vector $\bar{z}_j$ is the (empirical) probability distribution for item j over the K topics, and vector s_i quantifies user i's affinity to each of the K topics. As will be seen, these topics are interpretable.

The estimation of the *multiplicative* term $s_i' \bar{z}_j$ that capture user interaction with items is the main modeling challenge. In fact, given the data incompleteness in applications (typically, we have response available for 1% to 5% of all possible dyads), it is clear that latent factors cannot be estimated reliably even for small number of topics K. Hence a second stage that specifies constraints on factors through priors reduce the effective degrees of freedom and results in good performance. The crux of the problem is in specifying the prior, the first stage model is too flexible and would overfit if not regularized appropriately. Existing factor models assume both user and item factors take values in an K-dimensional Euclidean space but the values are moderated through an L_2 norm constraint or equivalently a zero-mean Gaussian prior. The regression based latent factor model (RLFM hereafter) recently proposed in [2] relax the prior to have a flexible mean that is obtained through a regression on user(item) features. Yu et al. [30] go further and regularize the factors through a non-linear kernel function. Other than providing better regularization, such a strategy helps in providing better prediction in cold-start scenarios through the *fallback* mechanism – for user (or items) with small number of ratings, we give more weight to the prior mean (predicted by user and/or item features) in estimating the factor but with increasing number of ratings we converge to a user (or item) specific factor estimate. The model transitions from global feature-based prediction to user (or item) specific ones in a smooth way after incorporating the sample size and correlations effects in the data appropriately.

Our fLDA model is also similar in spirit but, while we assume the user factors still take values in an Euclidean space, for item factors we deviate significantly from previous work and assume factors are discrete with K possible values (topics). Moreover, we attach a latent topic to each word in an item and assume the average of per-word topics to provide item topics. The granular topics at the word level are regularized through user ratings and by assuming the joint (word, topic, item) occurrence probabilities could be appropriately modeled through (word, topic) and (topic, item) interactions as done in the classical LDA model. More mathematically, while the prior on user factors is still based on a regression on user features as in RLFM , for the item topics the prior is now given by a latent Dirichlet allocation (LDA) model [8] that have been shown to provide excellent and interpretable soft clustering of documents into topics. However, unlike the LDA model, the posterior distribution of item factors depends on both the prior (unsupervised LDA) and user ratings on items. The supervised LDA (sLDA)[7] technique is also similar in spirit but does not allow for a per-user regression which is the main focus of fLDA . In fact, the sLDA model is a special case of fLDA when $s_i = s$ for each i, i.e., every user has the same factor which gets estimated from the data. Clearly, such a model cannot capture interactions in dyadic data applications, the main focus of this research. The use of LDA prior provides significant advantages. While it improves predictive accuracy in the presence of rich item meta-data, it may also provide interpretable topics as in an unsupervised LDA model. Once we get interpretable item topics, it is natural to interpret user factors as affinities to different topics. We also note that fLDA models the conditional distribution of ratings through a model on the joint distribution of ratings and word occurences in items. This is in contrast to other existing models where features are assumed non-stochastic, the additional smoothing on the word occurrences contributes to better regularization of item factors and may lead to better performance in practice.

2.2 Model Specification

We provide a detailed description of our model in this section. We begin by setting up notations.

Notations: To specify the model, we use the following notations. Let index i runs through users, index j runs through items, index k runs through item topics, and index n runs through words assuming a bag-of-words representation for an item. Let M, N, K and W denote the numbers of users, items, topics and distinct words in the item corpus respectively. We use W_j to denote item length, i.e., the number of words in item j. We abuse notation a little to let x_i, x_j and x_{ij} denote the feature vectors for user i, item j and the user-item dyad (i, j) respectively. For instance, x_i may include features like user's age, gender, location, browse behavior, x_j may include features like item's publisher, category and x_{ij} may include features like the number of times a user saw an item in earlier visits. In addition to x_j, items have a bag-of-words vector w_j, where w_{jn} denotes the nth word in item j $(n = 1, ..., W_j)$ that are modeled through the LDA prior.

First stage observation model: Our first stage observation model specifies the distribution of response condi-

[1]We abuse the notation a little bit by using z_{jn} to denote both a discrete variable with K possible values and a vector of length K, where there is exactly one element equal to one and all the others are zero.

tional on the latent factors and topics and is given as follows.

- For the Gaussian model, the rating that user i gives item j is given by $y_{ij} \sim \mathcal{N}(\mu_{ij}, \sigma^2)$, where
$$\mu_{ij} = x'_{ij}\, b + \alpha_i + \beta_j + s'_i\, \bar{z}_j.$$
- For the Logistic model, the binary rating is generated by $y_{ij} \sim \text{Bernoulli}(\mu_{ij})$, where
$$\log\left(\frac{\mu_{ij}}{1-\mu_{ij}}\right) = x'_{ij}\, b + \alpha_i + \beta_j + s'_i\, \bar{z}_j.$$

Note that b is the regression weight vector for dyadic features x_{ij}; and α_i, β_j, s_i and z_{jn} are the latent factors. Each word w_{jn} in item j has an underlying latent topic z_{jn}, and $\bar{z}_j = \sum_{n=1}^{W_j} \frac{z_{jn}}{W_j}$ denotes the empirical distribution of topics for item j averaged over the topic distribution of words in item j (z_{jn} is interpreted as a vector of zeros with length K except the kth position equals 1 if z_{jn} represents topic k). We also note that in general b is a global parameter whose dimension is small and hence does not require further regularization. We now discuss our second stage state model that specifies priors on the latent factors.

Second stage state model: Our goal here is to specify the prior distribution on latent factors $[\{\alpha_i\}, \{\beta_j\}, \{s_i\}, \{z_{jn}\}]$ conditional on the features $[\{x_i\}, \{x_j\}, \{w_{jn}\}]$. We assume the factor distributions are statistically independent, i.e.,

$$[\{\alpha_i\}, \{\beta_j\}, \{s_i\}, \{z_{jn}\}] = (\prod_i [\alpha_i] \prod_j [\beta_j] \prod_i [s_i]) \cdot [\{z_{jn}\}]$$

with priors given as follows.

1. User bias $\alpha_i = g'_0 x_i + \epsilon_i^\alpha$, where $\epsilon_i^\alpha \sim \mathcal{N}(0, a_\alpha)$, and g_0 is the regression weight vector on user features x_i.
2. User factor $s_i = H x_i + \epsilon_i^s$ is a $K \times 1$ vector of topic affinity scores, where $\epsilon_i^s \sim \mathcal{N}(\mathbf{0}, A_s)$, and H is the regression weight matrix on user features x_i.
3. Item popularity $\beta_j = d'_0 x_j + \epsilon_j^\beta$, where $\epsilon_j^\beta \sim \mathcal{N}(0, a_\beta)$ and d_0 is the regression weight vector on item features x_j.

The prior for $\{z_{jn}\}$ is given by the LDA model. For the sake of completeness and to setup notations, we now briefly describe the LDA prior below (see [16, 8] for more details.)

LDA prior: The Latent Dirichlet Allocation model is an unsupervised clustering method that works well when each element to be clustered has a bag-of-words representation. Thus, it clusters data that are categorical, high-dimensional but sparse; classical methods like K-means does not work well in such scenarios. It has found widespread use in text mining applications where it provides a soft clustering of each document into topics that are interpretable.

The LDA model works by assuming the occurrence probabilities in the three way (word, item, topic) contingency table can be modeled in terms of (word, topic) and (item, topic) interactions. More concretely, it assumes word vectors for items are generated in the following way. Associate with each topic k a multinomial distribution $\Phi_k^{1\times W}$ over the words in the entire corpus; i.e., $\Phi_{k\ell} = \Pr[\text{observe word } \ell \mid \text{topic } k]$. Also assume a multinomial distribution $\theta_j^{K\times 1}$ for item j over the K topics; i.e., $\theta_{jk} = \Pr[\text{the latent topic of a word is } k \mid \text{item } j]$. Now, the generative model for the corpus is modeled as $[\{w_{jn}\}, \{z_{jn}\} | \{\Phi_k\}, \{\theta_j\}] \propto [\{w_{jn}\} | \{z_{jn}\}, \{\Phi_k\}] \cdot [\{z_{jn}\} | \{\theta_j\}]$ where

Rating:	$y_{ij} \sim \mathcal{N}(\mu_{ij}, \sigma^2)$, or	(Gaussian)
	$y_{ij} \sim \text{Bernoulli}(\mu_{ij})$	(Logistic)
	$l(\mu_{ij}) = x'_{ij}\, b + \alpha_i + \beta_j + s'_i\, \bar{z}_j$	
User factors:	$\alpha_i = g'_0 x_i + \epsilon_i^\alpha$,	$\epsilon_i^\alpha \sim \mathcal{N}(0, a_\alpha)$
	$s_i = H x_i + \epsilon_i^s$,	$\epsilon_i^s \sim \mathcal{N}(\mathbf{0}, A_s)$
Item factors:	$\beta_j = d'_0 x_j + \epsilon_j^\beta$,	$\epsilon_j^\beta \sim \mathcal{N}(0, a_\beta)$
	$\bar{z}_j = \sum_n z_{jn} \;/\; W_j$	
Topic model:	$\theta_j \sim \text{Dirichlet}(\lambda)$	
	$\Phi_k \sim \text{Dirichlet}(\eta)$	
	$z_{jn} \sim \text{Multinom}(\theta_j)$	
	$w_{jn} \sim \text{Multinom}(\Phi_{z_{jn}})$	

Note: $l(\mu_{ij}) = \mu_{ij}$ for Gaussian; $l(\mu_{ij}) = \log \frac{\mu_{ij}}{1-\mu_{ij}}$ for Logistic. See the text for details.

Table 1: LDA-based Factorization Model

1. $z_{jn}|\theta_j \sim \text{Multinom}(\theta_j)$, i.e., we draw a latent topic for each word in item j from the document specific multinomial, and
2. $w_{jn}|z_{jn} \sim \text{Multinom}(\Phi_{z_{jn}})$, i.e., after drawing the latent topics for each word in the item, the words are drawn from the topic specific (document independent) multinomial distributions with topic $= z_{jn}$.

To regularize the multinomial probabilities associated with high dimensional simplices, we assume $\theta_j \sim \text{Dirichlet}(\lambda)$ and $\Phi_k \sim \text{Dirichlet}(\eta)$. Here, λ and η are hyperparameters of symmetric Dirichlet priors that indirectly control the entropy induced in the posterior distribution $[\bar{z}_j | \{w_{jn}\}]$. Large values of hyperparameters would lead to less concentration and higher entropy. The Dirichlet-multinomial conjugacy enables us to marginalize over $\{\Phi_k\}$ and $\{\theta_j\}$, so that one could work directly with $[\{w_{jn}\}, \{z_{jn}\} | \eta, \lambda]$ and draw samples from the posterior of latent topics efficiently through a collapsed Gibbs sampler for unsupervised LDA model fitting as proposed in [16]. In fLDA , we shall also work with the marginalized prior since the item factors are functions of latent topic variables $\{z_{jn}\}$ that do not depend on the multinomial probabilities $\{\Phi_k\}$ or $\{\theta_j\}$. However, the functional form of the collapsed Gibbs sampler for fLDA gets modified by an exponential tilt through contribution from the log-likelihood part of the first stage model that depend on the ratings as we shall see later in Section 3. For easy reference, we summarize our two-stage model succinctly in Table 1 and show the graphical representation in Figure 1.

We now briefly describe the RLFM model in [2] since it will be used as a baseline for comparison in our experiments.

RLFM Model: In this model, the item factors $\bar{z}_j$ are replaced by continuous factors v_j in the first stage observation equation, in the second stage state equation the item factor v_j are assumed to have a regression prior, i.e., $v_j = D x_j + \epsilon_j^v$, where D is the regreession weight matrix on item features x_j and $\epsilon_j^v \sim \mathcal{N}(0, A_v)$. All other details are same as in fLDA .

2.3 Discussion

We end this section with a brief discussion of why we chose $\bar{z}_j$ to capture interactions in the first stage model instead of θ_j, the item multinomial topic probability vector. We note that $\bar{z}_j$ being the empirical distribution of word latent factors in an item has more variability than θ_j, this helps

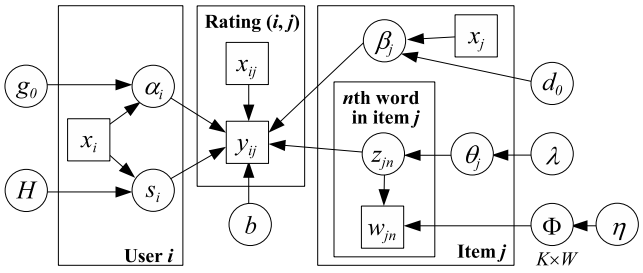


Figure 1: Graphical representation of fLDA . Variance components (σ^2, a_α, a_β, A_s) are omitted for succinctness.

in better behaved user level regressions and leads to faster convergence.

3. TRAINING AND PREDICTION

In this section, we first provide a detailed description of our model training procedure that is based on a Monte Carlo Expectation Maximization (MCEM) algorithm and then discuss the prediction procedure. We begin with a precise formulation of the optimization problem in the model training phase followed by a description of our EM fitting algorithm. For ease of exposition, we focus on the Gaussian first stage model and discuss the Logistic model in Section 3.1.3.

For ease of exposition, let $X_{ij} = [x_i, x_j, x_{ij}]$ denote the features, $\Delta_{ij} = [\alpha_i, \beta_j, s_i]$ denote the continuous latent factors, and $\Theta = [b, g_0, d_0, H, \sigma^2, a_\alpha, a_\beta, A_s, \lambda, \eta]$ denote the model parameters. We use the convention that $y = \{y_{ij}\}$, $X = \{X_{ij}\}$, $\Delta = \{\Delta_{ij}\}$, $z = \{z_{jn}\}$ and $w = \{w_{jn}\}$.

Following the empirical Bayes approach, given observed ratings y and words w, the goal of training is to find the parameter setting $\hat{\Theta}$ that maximizes the incomplete data likelihood (marginalizing over latent factors Δ and $\{z_{jn}\}$):

$$\hat{\Theta} = \arg\max_{\Theta} \Pr[y, w \,|\, \Theta, X],$$

Having obtained optimal value of Θ after optimizing the incomplete data likelihood, inference and predictions can be done through the posterior $[\Delta, \{z_{jn}\} \,|\, y, w, \hat{\Theta}, X]$.

3.1 Model Fitting

The EM algorithm[12] is well suited to fit factor models. The factors in this case form the missing data that are augmented to the observed data; complete data log-likelihood is then obtained as product of likelihoods from our observation (first-stage) and state (second-stage) models. The EM algorithm iterates between an E-step (taking expectation of complete data likelihood with respect to the posterior of missing data $(\Delta, \{z_{jn}\})$ conditional on observed data and current value of Θ) and an M-step (in which we maximize the expected complete data likelihood from the E-step to obtain updated values of Θ). At each iteration, the EM algorithm is guaranteed not to deteriorate the value of incomplete data log-likelihood. The major computational bottleneck is in the E-step because the posterior of factors are not available in closed form. Hence, we take recourse to Monte Carlo methods. We draw samples from this posterior and approximate the expectation in E-step by taking Monte Carlo mean. This is known as the Monte Carlo EM (MCEM) algorithm [9] in the literature. Alternatively, one can apply variational approximation to derive a closed-form formula for the expectation, or apply the iterative conditional mode (ICM) algorithm in which the expectation computation is replaced by "plugging-in" the mode of the conditional distributions. However,in our experience and other studies [24], sampling usually provides better performance in terms of predictive accuracy, while still being scalable. One reason for this behavior is the highly multi-modal nature of the posterior; sampling in our experience ensures we do not get stuck in a bad local mode region. In fact, we have found sampling to be highly resistant to over-fitting even with increasing number of factors; this is not the case with mode-finding approaches that may overfit (see [2] for an example). Thus, in this section, we focus on the Monte Carlo EM algorithm.

Let $LL(\Theta; \Delta, z, y, w, X) = \log(\Pr[\Delta, z, y, w \,|\, \Theta, X])$ denote the complete data log likelihood. Let $\hat{\Theta}^{(t)}$ denote our current estimate of Θ at the tth iteration. The EM algorithm iterates through the following two steps until convergence.

- **E-step:** Compute $E_{\Delta,z}[LL(\Theta; \Delta, z, y, w, X) \,|\, \hat{\Theta}^{(t)}]$ as a function of Θ, where the expectation is taken over the posterior distribution of $(\Delta, z \,|\, \hat{\Theta}^{(t)}, y, w, X)$.
- **M-step:** Find the Θ that maximizes the expectation computed in the E-step.

$$\hat{\Theta}^{(t+1)} = \arg\max_{\Theta} E_{\Delta,z}[LL(\Theta; \Delta, z, y, w, X) \,|\, \hat{\Theta}^{(t)}]$$

3.1.1 Monte-Carlo E-Step

Since $E_{\Delta,z}[LL(\Theta; \Delta, z, y, w, X) \,|\, \hat{\Theta}^{(t)}]$ is not available in closed form, we compute the Monte-Carlo expectation based on L samples generated by a Gibbs sampler[13]. The Gibbs sampler repeats the following procedure L times. In the following, we use $(\delta \,|\, \text{Rest})$, where δ can be one of α_i, β_j, s_i, and z_{jn}, to denote the conditional distribution of δ given all the others. Let $\mathcal{I}_j$ denote the set of users who rated item j, and $\mathcal{J}_i$ denote the set of items that user i rated.

1. For each user i, sample α_i from $(\alpha_i \,|\, \text{Rest})$, which is Gaussian.

$$\begin{aligned} \text{Let } o_{ij} &= y_{ij} - x'_{ij} b - \beta_j - s'_i \bar{z}_j \\ Var[\alpha_i|\text{Rest}] &= (\tfrac{1}{a_\alpha} + \textstyle\sum_{j \in \mathcal{J}_i} \tfrac{1}{\sigma^2})^{-1} \\ E[\alpha_i|\text{Rest}] &= Var[\alpha_i|\text{Rest}](\tfrac{g'_0 x_i}{a_\alpha} + \textstyle\sum_{j \in \mathcal{J}_i} \tfrac{o_{ij}}{\sigma^2}) \end{aligned}$$

2. For each item j, sample β_j from $(\beta_j \,|\, \text{Rest})$, which is Gaussian.

$$\begin{aligned} \text{Let } o_{ij} &= y_{ij} - x'_{ij} b - \alpha_i - s'_i \bar{z}_j \\ Var[\beta_j|\text{Rest}] &= (\tfrac{1}{a_\beta} + \textstyle\sum_{i \in \mathcal{I}_j} \tfrac{1}{\sigma^2})^{-1} \\ E[\beta_j|\text{Rest}] &= Var[\beta_j|\text{Rest}](\tfrac{d'_0 x_j}{a_\beta} + \textstyle\sum_{i \in \mathcal{I}_j} \tfrac{o_{ij}}{\sigma^2}) \end{aligned}$$

3. For each user i, sample s_i from $(s_i \,|\, \text{Rest})$, which is Gaussian, for all i.

$$\begin{aligned} \text{Let } o_{ij} &= y_{ij} - x'_{ij} b - \alpha_i - \beta_j \\ Var[s_i|\text{Rest}] &= (A_s^{-1} + \textstyle\sum_{j \in \mathcal{J}_i} \tfrac{\bar{z}_j \bar{z}'_j}{\sigma^2})^{-1} \\ E[s_i|\text{Rest}] &= Var[s_i|\text{Rest}](A_s^{-1} H x_i + \textstyle\sum_{j \in \mathcal{J}_i} \tfrac{o_{ij} \bar{z}_j}{\sigma^2}) \end{aligned}$$

4. For each item j and each word n in item j, sample z_{jn} from $(z_{jn} \,|\, \text{Rest})$, which is multinomial. Assume the word corresponding to z_{jn} is $w_{jn} = \ell$. Let $Z^{\neg jn}_{j'k\ell}$ denotes the number of times word ℓ belongs to topic k in item j' with z_{jn} removed; i.e.,

$$Z^{\neg jn}_{jk\ell} = \sum_{n' \neq n} \mathbf{1}\{z_{jn'} = k \text{ and } w_{jn'} = \ell\} \text{ and}$$
$$Z^{\neg jn}_{j'k\ell} = \sum_{n'} \mathbf{1}\{z_{j'n'} = k \text{ and } w_{j'n'} = \ell\}, \text{ for } j' \neq j.$$

Then, the multinomial probabilities are given by

$$\Pr[z_{jn} = k \,|\, \text{Rest}] \propto \frac{Z^{\neg jn}_{k\ell} + \eta}{Z^{\neg jn}_{k} + W\eta} \; (Z^{\neg jn}_{jk} + \lambda_k) \; g(y),$$

where $Z^{\neg jn}_{k\ell} = \sum_{j'} Z^{\neg jn}_{j'k\ell}$, $Z^{\neg jn}_{k} = \sum_{\ell} Z^{\neg jn}_{k\ell}$, $Z^{\neg jn}_{jk} = \sum_{\ell} Z^{\neg jn}_{jk\ell}$ and, letting $o_{ij} = y_{ij} - x'_{ij}\, b - \alpha_i - \beta_j$,

$$g(y) = \exp\left\{ \bar{z}'_j B_j - \tfrac{1}{2} \bar{z}'_j C_j \bar{z}_j \right\}$$
$$B_j = \sum_{i \in \mathcal{I}_j} \tfrac{o_{ij} s_i}{\sigma^2} \text{ and } C_j = \sum_{i \in \mathcal{I}_j} \tfrac{s_i s'_i}{\sigma^2}$$

Note that here $\bar{z}_j = \sum_{n'} z_{jn'} \;/\; W_j$ is the empirical topic distribution of item j with z_{jn} set to topic k. For detailed derivation, see the appendix.

3.1.2 M-Step

In the M-step, we want to find the parameter setting $\Theta = [b, g_0, d_0, H, \sigma^2, a_\alpha, a_\beta, A_s, \lambda, \eta]$ that maximizes the expected complete data likelihood computed in the E-step.

$$\hat{\Theta}^{(t+1)} = \arg\max_{\Theta} E_{\Delta,z}[LL(\Theta; \Delta, z, y, w, X) \,|\, \hat{\Theta}^{(t)}]$$

where $-LL(\Theta; \Delta, z, y, w, X) = \text{constant}$

$$\begin{aligned}
&+ \tfrac{1}{2} \textstyle\sum_{ij} \left(\tfrac{1}{\sigma^2} (y_{ij} - \alpha_i - \beta_j - x'_{ij} b - s'_i \bar{z}_j)^2 + \log \sigma^2 \right) \\
&+ \tfrac{1}{2a_\alpha} \textstyle\sum_i (\alpha_i - g'_0 x_i)^2 + \tfrac{M}{2} \log a_\alpha \\
&+ \tfrac{1}{2} \textstyle\sum_i (s_i - H x_i)' A_s^{-1} (s_i - H x_i) + \tfrac{M}{2} \log(\det A_s) \\
&+ \tfrac{1}{2a_\beta} \textstyle\sum_j (\beta_j - d'_0 x_j)^2 + \tfrac{N}{2} \log a_\beta \\
&+ N \left(K \log \Gamma(\lambda) - \log \Gamma(K\lambda) \right) \\
&+ \textstyle\sum_j \left(\log \Gamma\left(Z_j + K\lambda \right) - \sum_k \log \Gamma(Z_{jk} + \lambda) \right) \\
&+ K \left(W \log \Gamma(\eta) - \log \Gamma(W\eta) \right) \\
&+ \textstyle\sum_k \left(\log \Gamma(Z_k + W\eta) - \sum_\ell \log \Gamma(Z_{k\ell} + \eta) \right).
\end{aligned}$$

It can be easily seen from the above equation that (b, σ^2), (g_0, a_α), (d_0, a_β), (H, A_s), λ and η can be optimized separately. In particular, the first four can be optimized by solving four regression problems. The last two are single dimensional and can be solved easily by a grid search. In the rest of this section, we provide the details. We use $\tilde{E}[\cdot]$ and $\tilde{Var}[\cdot]$ to denote the Monte Carlo mean and variance.

Regression for (b, σ^2): Let $o_{ij} = \alpha_i + \beta_j + s'_i \bar{z}_j$. Here, we want to minimize

$$\tfrac{1}{\sigma^2} \textstyle\sum_{ij} \tilde{E}[(y_{ij} - x'_{ij} b - o_{ij})^2] + D \log(\sigma^2),$$

where D is the number of observed ratings. It can be seen that the optimal solution to b can be found by least squares regression using x_{ij} as features to predict $(y_{ij} - \tilde{E}[o_{ij}])$. Let RSS denote the residual sum of squares from this regression. Then, the optimal σ^2 is $(\sum_{ij} \tilde{Var}[o_{ij}] + \text{RSS})/D$, where RSS is the residual sum of squares of the regression.

Regression for (g_0, a_α): Similar to the previous case, the optimal g_0 can be found by solving a regression problem using x_i as features to predict $\tilde{E}[\alpha_i]$, and the optimal a_α is $(\sum_i \tilde{Var}[\alpha_i] + \text{RSS})/M$.

Regression for (d_0, a_β): The optimal d_0 can be found by solving a regression problem using x_j as features to predict $\tilde{E}[\beta_j]$, and the optimal $a_\beta = (\sum_j \tilde{Var}[\beta_j] + \text{RSS})/N$.

Regression for (H, A_s): For simplicity, we assume the variance-covariance matrix to be diagonal, i.e., $A_s = a_s I$. Let H_k denote the kth row of H and s_{ik} be the kth component in s_i. We find H_k by solving a regression problem using x_i as features to predict $\tilde{E}[s_{ik}]$, for each topic k. Let RSS_k denote the residual sum of squares of the kth regression. Then, $a_s = (\sum_{ik} \tilde{Var}[s_{ik}] + \sum_k \text{RSS}_k)/KM$.

Optimization over η: We find η that minimizes

$$\begin{aligned}
&K \left(W \log \Gamma(\eta) - \log \Gamma(W\eta) \right) \\
&+ \textstyle\sum_k \left(\tilde{E}[\log \Gamma(Z_k + W\eta)] - \sum_\ell \tilde{E}[\log \Gamma(Z_{k\ell} + \eta)] \right)
\end{aligned}$$

Since this optimization is just one dimensional and η is a nuisance parameter, we can simply try a number of fixed possible η values.

Optimization over λ: We find λ that minimizes

$$\begin{aligned}
&N \left(K \log \Gamma(\lambda) - \log \Gamma(K\lambda) \right) \\
&+ \textstyle\sum_j \left(\tilde{E}[\log \Gamma\left(Z_j + K\lambda \right)] - \sum_k \tilde{E}[\log \Gamma(Z_{jk} + \lambda)] \right)
\end{aligned}$$

Again, this optimization is single dimensional. We search through a number of fixed points to find the best λ value.

3.1.3 Remarks

Number of Gibbs samples: Replacing the precise E-step with a Monte Carlo average no longer guarantees an increase in the marginal likelihood at each step. If the Monte Carlo sampling error associated with Θ_{new} (an estimate of Θ^*_{new} that is obtained from true E-step) is large relative to $||\Theta_{curr} - \Theta^*_{new}||$, the Monte Carlo E-step is wasteful since it is swamped by the Monte Carlo error. There are no rigorous solutions to this problem in the literature (especially when samples are dependent) except for some practical guidelines [9]. For instance, it is better to use fewer Monte Carlo simulations during early iterations. We performed extensive experiments with various schemes and found 20 EM iterations with 100 samples (drawn after 10 burn-in samples) at each iteration performed adequately in our experiments. In fact, the performance was not too sensitive to the choice of number of samples, even small number of samples like 50 did not hurt performance by much. We also note that Gibbs sampler was chosen due to its simplicity; better sampling methods to make the sampler *mix* faster is an open research problem in the context of bilinear factor models.

Regularizing regressions: In the M-step, each regression is performed by using a t-prior on the coefficients to avoid overfitting.

Number of Topics: Based on several experiments in the past and simulation studies, we have found the MCEM algorithm to be resistant to over-fitting even when the number of factors are mis-specified to be large, we did not try several K values on the test data since that may inadvertently lead to over-fitting and undermine the validity of our experiments. Hence, in our experiments we run fLDA using a large number of factors $(20 - 25)$. In practice, one can also perform cross validation within the training data to find the best number of factors.

Scalability: Fixing the number of topics, the number of EM iterations and the number of Gibbs samples per itera-

tion, the MCEM algorithm is essentially *linear* in the number of (rating + word) observations. In our experience, we observe that the MCEM algorithm converges quickly after a fairly small number of EM iterations (usually around 10). We also note that the algorithm is highly parallelizable. In particular, when drawing a sample for user (or item) factors, one can partition the users (or items) and draw a sample for each partition independently. For parallel algorithms to draw LDA samples, see for example [29].

Fitting Logistic regression: This is done through a variational approximation that involves a weighted Gaussian regression after each EM iteration (see [2] for details).

3.2 Prediction

Given the observed ratings y and words w in the training data, our goal is to predict the rating y_{ij}^{new} of user i on item j. We can predict the rating by the posterior mean $E[y_{ij}^{\text{new}}|y, w, \hat{\Theta}, X]$. For computational efficiency, we approximate the posterior mean by

$$\begin{aligned} E[y_{ij}^{\text{new}} \,|\, y, w, \hat{\Theta}, X] &= x'_{ij}\,\hat{b} + \hat{\alpha}_i + \hat{\beta}_j + E[s'_i\,\bar{z}_j] \\ &\approx x'_{ij}\,\hat{b} + \hat{\alpha}_i + \hat{\beta}_j + \hat{s}'_i\,\hat{\bar{z}}_j, \end{aligned}$$

where $\hat{\delta} = E[\delta \,|\, y, w, \hat{\Theta}, X]$ (δ take values in $\{\alpha_i\}, \{\beta_j\}, \{s_i\}$) is estimated in the training phase. For estimating $\hat{\bar{z}}_j$ for a new item j in cold-start scenarios, we use Gibbs sampling to obtain topic distribution of words w_j in item j through unsupervised LDA sampling formula. However, note that the topic×word matrix Φ used during the sampling is the one obtained from fLDA ; hence the predictions are influenced by ratings even for new items.

4. EXPERIMENTS

We show the effectiveness of fLDA using three real-life datasets. Specifically, we use the widely studied MovieLens (movie rating) dataset to show that the predictive accuracy of fLDA is among the best compared to six popular collaborative filtering methods. Here, because each movie in the test set has enough ratings in the training set to estimate the movie factors, fLDA does not provide improved accuracy. Next, we report a case study of how fLDA can be used to provide interpretable, personalized recommendation on a social news service site using the Yahoo! Buzz dataset. We show that fLDA significantly outperforms the state-of-the-art method when there are many new items that do not appear in the training dataset, and it can also identify high quality topics from news stories. Finally, we present another case study using a book rating dataset. Again, we show that fLDA can provide better predictive performance than state-of-the-art methods. We note that we also ran fLDA with a few different random seeds and obtained essentially the same results.

4.1 MovieLens Data

In this section, we illustrate our method on the widely studied movie recommender problem. Since both user and item features are central to our method, we did not consider Netflix (no user features are available). Instead, we analyzed the MovieLens data that consists of $1M$ ratings on 6,040 users and 3,706 movies. User features used include age, gender, zipcode (we used the first digit only), and occupation; item features include movie genre (this was only used with RLFM). To construct bag-of-word features, we supplemented the movie features with actor, actress and director names, we also extracted words from movie titles and plots. We further split the data into training set and test set based on time; the first 75% ratings in chronological order are used as training while the rest is used as test. This data was analyzed in [2] and compared to several benchmark methods; we report root mean square errors (RMSE) here along with results from fLDA in Table 2.

Model	Test RMSE
fLDA	0.9381
RLFM	0.9363
unsup-LDA	0.9520
Factor-Only	0.9422
Feature-Only	1.0906
FilterBot	0.9517
MostPopular	0.9726
Constant	1.1190

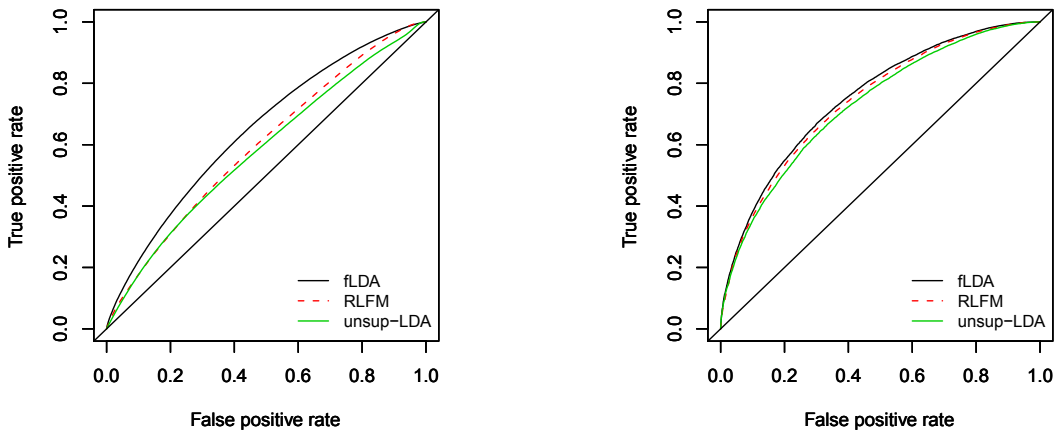
Table 2: Test-set RMSE on Movielens

Methods: The **Constant** model predicts the rating as training data average for all test cases; **Feature-Only** is a regression model trained on user and item features (genre) only, **Factor-Only** is the usual matrix factorization model with zero-mean priors, **Most-Popular** is a model based only on user and item bias that are regularized using features and **Filter-Bot** is a collaborative filtering method that is used to deal with warm-start and cold-start problems simultaneously (we do not report on other collaborative filtering methods like item-item similarity since they were all worse than Filter-Bot). **RLFM** is the regression-based latent factor model proposed in [2]. **unsup-LDA** is the counterpart of fLDA by using unsupervised LDA; i.e., it first applies unsupervised LDA to identify topics of each word in each item and then fit fLDA by fixing $\bar{z}_j$ to be the unsupervised topics.

For this dataset, a large fraction of ratings in the test set (almost 56%) involved new users, but most of them involved old items for which ratings are available in the training data. The key to have good accuracy is whether a model can handle cold-start for users, which is not the target application scenario for fLDA . Thus, we did not expect substantial improvements from fLDA relative to RLFM, which is the best among the methods we tried on this dataset. The main purpose of this analysis was to demonstrate that fLDA performs equally well in warm-start scenarios where we have ratings on most items seen in the test set. We note that it is not unreasonable to see RLFM slightly outperforms fLDA, because its item factors are initialized with possibly cleaner human labeled movie genres (while fLDA only uses possibly noisier bag-of-word features) and are trained using a large number of observations.

4.2 Yahoo! Buzz Application

Yahoo! Buzz (at buzz.yahoo.com) is a social news service site that recommends "buzzing" news stories to users; votes on articles ("buzz up" or "buzz down") is an important source of information used in deciding recommendations. Currently, it has a global ranked list of news stories and 12 category-specific ranked lists. Once a user is signed in, Yahoo! Buzz can also recommend articles buzzed by the user's friend. At present, the site does not use collaborative-filtering-style algorithms. In this section, we conduct an offline analysis to explore this possibility through fLDA.



(a) Yahoo! Buzz data

(b) BookCrossing data

Figure 2: ROC curves of different methods

We collected 620,883 votes that were generated by 4,026 users rating 10,468 articles over a three-month period. To minimize the effect of spamming votes, we only selected articles from trusted sources and users with reasonable numbers of votes (no more than 1000 votes). We treat "buzz up" votes as ratings with value 1, and "buzz down" votes as ratings with value -1. In this application, the majority of votes are "buzz up" since users usually "buzz down" a article only when they really dislike it; it is difficult to obtain explicit feedback on articles that are not of interest to them. Thus, for each user who had N votes, we randomly selected N articles not voted by the user and gave them a user rating of 0. Since users usually like a small number of articles, it is reasonable to assume that random articles would not interest them. Each user is associated with his/her age and gender. We converted age values into 10 age groups (each of which has a equal number of users). Each article is associated with its title, description and a set of categories determined by Yahoo! Buzz. We removed the stop words, ran an entity recognizer to identify named entities, and stem the terms by the Porter stemmer. Then, we created a training dataset by using the first two months and a test dataset using the last one month of data. Since the articles are related to news, most articles in the test dataset are new and do not appear in the training dataset.

We compare three models: fLDA with 25 supervised topics, RLFM (the state-of-the-art method) with 25 factors and unsup-LDA with 25 unsupervised topics. The ROC curves (plotting true positive rate as a function of false positive rate) are shown in Figure 2(a). To plot ROC curves, we treat ratings with value 1 as positives and ratings with value 0 or -1 as negatives. It can be clearly seen that fLDA significantly outperforms RLFM and unsup-LDA, and RLFM is slightly better than unsup-LDA.

In Table 3, we list interesting topics identified by fLDA. As can be seen, fLDA was able to identify important news topics in Yahoo! Buzz, e.g., CIA's harsh interrogation techniques in the Bush administration (topic 1), swine flu (topic 2), the gay marriage issue (topic 4), the economy downturn (topic 13), the North Korean issue (topic 14), the American International Group and General Motor issue (topic 25). We note that most of the topics are quite easy to interpret. However, we found 6 out of 25 topics are too general to be useful. We also note that unsupervised LDA also identified interpretable topics. Roughly half of the topics look similar to those identified by fLDA. However the predictive performance of unsup-LDA is much worse than fLDA.

To summarize our analysis of Buzz data, fLDA was able to identify interesting topics (more than 3/4 are easily interpretable) and provides superior predictive performance over the state-of-the-art method.

4.3 BookCrossing Dataset

BookCrossing (at www.bookcrossing.com) is an online book club. Uses can rate books on a 1-to-10 scale. In prior work, Ziegler et al. [31] collected book ratings from the site.[2] The data is quite noisy; there are invalid ISBN's, and some of the ISBN's in the rating file cannot be found in the book description file. We cleanup the dataset by only taking the books that have at least three ratings and have reviews from the product description pages of Amazon.com. We then took the book reviews from Amazon.com as the text description about the books. For each book, we take the top 70 terms in the review with high TF/IDF scores. We selected users with at least 6 ratings. Each user in the dataset is associated with his/her age and location. We converted age values into 10 age groups and only took countries with at least 50 users as the location feature. We removed all the "implicit" ratings since they are not real ratings and their meaning is unclear. To prevent test-set errors from being dominated by a small number of outliers, we removed ratings with values from 1 to 4 (which accounts for less than 5% of explicit ratings), and rescaled rating values from the range between 5 and 10 to the range between -2.5 and 2.5. At the end, we have 149,879 ratings for 25,137 books from 6,981 users. We then create training-test split by three-fold cross validation. For each user, we randomly put the books that he/she rated into three buckets. In the nth fold, ratings in the nth bucket is used as test data and the rest are used as the training data.

[2]The BookCrossing dataset can be downloaded from http://www.informatik.uni-freiburg.de/ cziegler/BX/

Topic	Terms (after stemming)
1	bush, tortur, interrog, terror, administr, cia, offici, suspect, releas, investig, georg, memo, al, prison, george_w._bush, guantanamo, us, secret, harsh, depart, attornei, detaine, justic, iraq, alleg, probe, case, said, secur, waterboard
3	mexico, flu, pirat, swine, drug, ship, somali, border, mexican, hostag, offici, somalia, captain, navi, health, us, attack, cartel, outbreak, coast, case, piraci, violenc, u.s., held, spread, pandem, kill
4	nfl, player, team, suleman, game, nadya, star, high, octuplet, nadya_suleman, michael, week, school, sport, fan, get, vick, leagu, coach, season, mother, run, footbal, end, dai, bowl, draft, basebal
6	court, gai, marriag, suprem, right, judg, rule, sex, pope, supreme_court, appeal, ban, legal, allow, state, stem, case, church, california, immigr, law, fridai, cell, decis, feder, hear, cathol, justic
8	palin, republican, parti, obama, limbaugh, sarah, rush, gop, presid, sarah_palin, sai, gov, alaska, steel, right, conserv, host, fox, democrat, rush_limbaugh, new, bristol, tea, senat, levi, stewart, polit, said
9	brown, chri, rihanna, chris_brown, onlin, star, richardson, natasha, actor, actress, natasha_richardson, sai, madonna, milei, singer, divorc, hospit, cyru, angel, wife, charg, adopt, lo, assault, di, ski, accid, year, famili, music
10	idol, american, night, star, look, michel, win, dress, susan, danc, judg, boyl, michelle_obama, susan_boyl, perform, ladi, fashion, hot, miss, leno, got, contest, photo, tv, talent, sing, wear, week, bachelor
11	nation, scienc, christian, monitor, new, obama, christian_science_monitor, com, time, american, us, world, america, climat, peopl, week, dai, michel, just, warm, ann, coulter, chang, state, public, hous, global
12	obama, presid, hous, budget, republican, tax, barack, democrat, barack_obama, parti, sai, senat, congress, tea, administr, palin, group, spend, white, lawmak, politico, offic, gop, right, american, stimulu, feder, anti, health
13	economi, recess, job, percent, econom, bank, expect, rate, jobless, year, unemploy, month, record, market, stock, financi, week, wall, street, new, number, sale, rise, fall, march, billion, februari, crisi, reserv, quarter
14	north, korea, china, north_korea, launch, nuclear, rocket, missil, south, said, russia, chines, iran, militari, weapon, countri, chavez, korean, defens, journalist, japan, secur, nkorea, us, council, u.n., leader, talk, summit, warn
20	com, studi, space, livesci, research, earth, scientist, new, like, year, ic, station, nasa, water, univers, diseas, planet, human, discov, ancient, rare, intern, risk, live, find, expert, red, size, centuri, million
22	israel, iran, said, isra, pakistan, kill, palestinian, presid, iraq, war, gaza, taliban, soldier, leader, attack, troop, milit, govern, afghanistan, countri, offici, peac, group, us, minist, mondai, bomb, militari, polic, iraqi
23	plane, citi, air, high, resid, volcano, mondai, peopl, crash, jet, flight, erupt, south, itali, forc, flood, mile, alaska, small, hit, near, pilot, dai, mount, island, storm, river, travel, crew, earthquak
25	bonus, american_international_group, bank, billion, gener, compani, million, madoff, motor, financi, insur, treasuri, govern, bailout, bankruptci, execut, chrysler, gm, corp, general_motor, monei, pai, auto, ceo, giant, group, automak

Table 3: Some Topics identified by fLDA from the Yahoo! Buzz data

Model	RMSE	MAE
fLDA	1.3088	1.0317
RLFM	1.3278	1.0553
unsup-LDA	1.3539	1.0835

Table 4: Test-set RMSE for Book-Crossing

The test-set RMSE's, MAE's of fLDA with 25 supervised topics, RLFM with 25 factors and unsup-LDA with 25 unsupervised topics are listed in Table 4. The ROC curves (by treating ratings above zero as positives and ratings below zero as negatives) are shown in Figure 2(b). On this dataset, fLDA outperforms RLFM, which outperforms unsup-LDA. However, the differences are small.

5. RELATED WORK

Recommender systems have a rich literature by now and have seen rapid progress in the past few years primarily triggered by the Netflix competition (a proper review is not possible here). Although several new methods have been proposed, those based on matrix factorization [25, 24, 5, 1] and neighborhood based methods [18, 6] have become popular and are widely used. Neighborhood based methods are popular due to their easy interpretation; a user gets an item recommendation if other users with similar tastes have liked the item. Factorization based methods are in general more accurate but the factors are hard to interpret. Recently, several papers have explored models to better regularize the latent factors, primarily inspired by the task of improving RMSE for the Netflix data[19]. Using unsupervised LDA topics as features in recommender systems was explored in [17]. In Section 4, we showed fLDA can significantly outperform unsupervised LDA.

Models to address both cold-start and warm-start that have been studied in collaborative filtering are also related to fLDA . Although studied extensively [4, 11, 15, 20, 26], only recently has this topic started getting attention in a factorization model framework [30, 2, 28]. Our fLDA model is an addition to this literature and works with discrete factors for items regularized through an LDA prior. Other than providing good predictive performance, it also helps interpretation that are useful in some applications.

Our work is related to supervised LDA [7], a variation of LDA that incorporates regression in estimating the topics. However, sLDA only fits a single global regression to item factors, we fit one per user. This work was also inspired by conjoint analysis that is often conducted in marketing [23]. The goal is to estimate individual partworths (user factors) to items. However, item characteristics in this analysis are known features, fLDA obtains item factors by converting item meta-data that are in bag-of-words form into concise topic vectors.

Relational learning [14] is also related. In a very general sense, fLDA is a relational model. However, joint modeling of ratings given by users to items and words in items has not been studied in the literature. In related work, Porteous et al. [21] apply LDA to cluster users and items into groups and model ratings using the group membership; Singh et al. [27] propose to jointly model user-to-item rating relation, user-to-feature relation and item-to-feature relation via joint factorization of multiple matrix.

6. CONCLUSION

We presented a new factorization model fLDA that provides significantly better performance in applications where items have a bag-of-words representation, commonplace in many web applications. The key idea that differentiates fLDA from previous work in the area is the use of an LDA prior to regularize item factors. As an additional benefit, fLDA may provide interpretable user factors as affinities to latent item topics. In fact, we illustrate both prediction accuracy and interpretability on a real world example related to content recommendation on Yahoo! Buzz.

Our work has opened up several avenues for future research. In section 1, we discussed the potential benefit of using output from fLDA for applications like content programming to help in *media-scheduling* problems and user targeting in display advertising. This requires more follow-up work. On the algorithmic front, we note that updating the posterior of s_i requires inverting a $K \times K$ matrix which can be computationally expensive for large K. One solution to this problem could be to express ${s_i}' \bar{z}_j$ as $S_i^{'} Q \bar{z}_j$ where $Q_{K_s \times K}$ is a global matrix estimated from data and $K_s < K$. Finally, although we have illustrated fLDA on a real world application and on benchmark datasets, we are currently scaling up computations in a map-reduce framework to work with massive datasets on several other applications in advertising and content recommendation.

7. REFERENCES

[1] KDD cup and workshop. 2007.

[2] D. Agarwal and B.-C. Chen. Regression-based latent factor models. In *KDD*, 2009.

[3] D. Agarwal and B.-C. Chen, et al. Online models for content optimization. In *NIPS*, 2008.

[4] M. Balabanovic and Y. Shoham. Fab: content-based, collaborative recommendation. *Comm. of the ACM*, 1997.

[5] R. Bell, Y. Koren, and C. Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *KDD*, 2007.

[6] R. M. Bell and Y. Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *ICDM*, 2007.

[7] D. Blei and J. McAuliffe. Supervised topic models. In *NIPS*, 2008.

[8] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *JMLR*, 3, 2003.

[9] J. Booth and J. Hobert. Maximizing generalized linear mixed model likelihoods with an automated monte carlo EM algorithm. *J. R. Statist. Soc. B*, 1999.

[10] Y. Chen, D. Pavlov, and J. F. Canny. Large-scale behavioral targeting. In *KDD*, 2009.

[11] M. Claypool and A. Gokhale, et al. Combining content-based and collaborative filters in an online newspaper. In *Recommender Systems Workshop*, 1999.

[12] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Statist. Soc. B*, 1977.

[13] A. E. Gelfand. Gibbs sampling. *JASA*, 1995.

[14] L. Getoor and B. Taskar. *Introduction to Statistical Relational Learning*. MIT Press, 2007.

[15] N. Good and J. B. Schafer, et al. Combining collaborative filtering with personal agents for better recommendations. In *AAAI*, 1999.

[16] T. L. Griffiths and M. Steyvers. Finding scientific topics. In *Proc. of National Academy of Sciences*, 2004.

[17] X. Jin, Y. Zhou, and B. Mobasher. A maximum entropy web recommendation system: Combining collaborative and content features. In *KDD*, 2005.

[18] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD*, 2008.

[19] N. Lawrence and R. Urtasun. Non-linear matrix factorization with gaussian processes. In *ICML*, 2009.

[20] S.-T. Park and D. Pennock, et al. Naïve filterbots for robust cold-start recommendations. In *KDD*, 2006.

[21] I. Porteous, E. Bart, and M. Welling. Multi-hdp: A non parametric bayesian model for tensor factorization. In *AAAI*, 2008.

[22] J. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *ICML*, 2005.

[23] P. E. Rossi, G. Allenby, and R. P. McCulloch. *Bayesian Statistics and Marketing*. John Wiley, 2005.

[24] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *ICML'08*, 2008.

[25] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *NIPS*, 2008.

[26] A. I. Schein and R. Popescul, et al. Methods and metrics for cold-start recommendations. In *SIGIR*, 2002.

[27] A. P. Singh and G. J. Gordon. Relational learning via collective matrix factorization. In *KDD*, 2008.

[28] D. H. Stern, R. Herbrich, and T. Graepel. Matchbox: large scale online bayesian recommendations. In *WWW*, 2009.

[29] Y. Wang and H. Bai, et al. Plda: Parallel latent dirichlet allocation for large-scale applications. In *AAIM*, 2009.

[30] K. Yu, J. Lafferty, and S. Zhu. Large-scale collaborative prediction using a nonparametric random effects model. In *ICML*, 2009.

[31] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *WWW*, 2005.

APPENDIX

In the following, we derive the formula for $\Pr[z_{jn} = k \,|\, \text{Rest}]$. Let $z_{\neg jn}$ denote z with z_{jn} removed, and $w_{jn} = \ell$. We have

$$\Pr[z_{jn} = k \,|\, \text{Rest}] \propto \Pr[z_{jn} = k, y \,|\, z_{\neg jn}, \Delta, \hat{\Theta}^{(t)}, w, X]$$
$$\propto \Pr[z_{jn} = k \,|\, w, z_{\neg jn}, \hat{\Theta}^{(t)}] \prod_{i \in I_j} \Pr[y_{ij} \,|\, z_{jn} = k, z_{\neg jn}, \Delta, \hat{\Theta}^{(t)}, X].$$

$$\Pr[z_{jn} = k \,|\, w, z_{\neg jn}, \hat{\Theta}^{(t)}]$$
$$\propto \Pr[z_{jn} = k, w_{jn} = \ell \,|\, w_{\neg jn}, z_{\neg jn}, \hat{\Theta}^{(t)}]$$
$$= \Pr[w_{jn} = \ell \,|\, w_{\neg jn}, z_{jn} = k, z_{\neg jn}, \eta] \;\Pr[z_{jn} = k \,|\, z_{\neg jn}, \lambda]$$
$$= E[\Phi_{k\ell} \,|\, w_{\neg jn}, z_{\neg jn}, \eta] \;E[\theta_{jk} \,|\, z_{\neg jn}, \lambda]$$
$$= \frac{Z_{k\ell}^{\neg jn} + \eta}{Z_k^{\neg jn} + W\eta} \;\frac{Z_{jk}^{\neg jn} + \lambda_k}{Z_j^{\neg jn} + \sum_k \lambda_k}$$

Note that the denominator of the second term $(Z_j^{\neg jn} + \sum_k \lambda_k)$ is independent of k. Thus, we obtain

$$\Pr[z_{jn} = k \,|\, \text{Rest}] \propto \frac{Z_{k\ell}^{\neg jn} + \eta}{Z_k^{\neg jn} + W\eta} \;(Z_{jk}^{\neg jn} + \lambda_k) \prod_{i \in \mathcal{I}_j} f_{ij}(y_{ij})$$

where $f_{ij}(y_{ij})$ is the probability density at y_{ij}, which is Gaussian with mean $x_{ij}' b + \alpha_i + \beta_j + s_i' \bar{z}_j$ and variance σ^2, and $\bar{z}_j$ is computed by setting $z_{jn} = k$. Let $o_{ij} = y_{ij} - x_{ij}' b - \alpha_i - \beta_j$.

$$\prod_{i \in \mathcal{I}_j} f_{ij}(y_{ij}) \propto \exp\left\{ -\frac{1}{2} \sum_{i \in \mathcal{I}_j} \frac{(o_{ij} - s_i' \bar{z}_j)^2}{\sigma^2} \right\}$$
$$\propto \exp\left\{ \bar{z}_j' B_j - \frac{1}{2} \bar{z}_j' C_j \bar{z}_j \right\}$$
$$\text{where } B_j = \sum_{i \in \mathcal{I}_j} \frac{o_{ij} s_i}{\sigma^2} \text{ and } C_j = \sum_{i \in \mathcal{I}_j} \frac{s_i s_i'}{\sigma^2}$$