Stochastic Inference for Scalable Probabilistic Modeling of Binary Matrices

José Miguel Hernández-Lobato[1] JMH233@CAM.AC.UK
Neil Houlsby[1] NMTH2@CAM.AC.UK
Zoubin Ghahramani ZOUBIN@ENG.CAM.AC.UK
University of Cambridge, Department of Engineering, Cambridge CB2 1PZ, UK

Abstract

Fully observed large binary matrices appear in a wide variety of contexts. To model them, probabilistic matrix factorization (PMF) methods are an attractive solution. However, current batch algorithms for PMF can be inefficient because they need to analyze the entire data matrix before producing any parameter updates. We derive an efficient stochastic inference algorithm for PMF models of fully observed binary matrices. Our method exhibits faster convergence rates than more expensive batch approaches and has better predictive performance than scalable alternatives. The proposed method includes new data subsampling strategies which produce large gains over standard uniform subsampling. We also address the task of automatically selecting the size of the minibatches of data used by our method. For this, we derive an algorithm that adjusts this hyper-parameter online.

1. Introduction

Many machine learning methods have been developed for modeling matrices with a large number of missing entries. Amongst these, matrix factorization (MF) approaches (Srebro et al., 2005; Koren, 2008) are probably the most successful because of their simplicity and often superior predictive performance. These methods assume that the partially observed data matrix $\mathbf{X}$ is well approximated by a low rank matrix $\mathbf{U}\mathbf{V}^{\mathrm{T}}$. The objective is then to find the two matrices $\mathbf{U}$ and $\mathbf{V}$ given $\mathbf{X}$. There is extensive literature on MF methods, so in this paper we focus on probabilistic approaches. Probabilistic methods are an attractive solution to the MF problem because i) they are robust to overfitting (Salakhutdinov & Mnih, 2008), ii) they can account for non-continuous matrix entries such as ordinal values (Stern et al., 2009; Paquet et al., 2012) and iii) they can produce estimates of uncertainty in their predictions. Fast approximate inference is usually implemented using variational Bayes (Lim & Teh, 2007; Raiko et al., 2007; Nakajima et al., 2010). These variational algorithms are computationally efficient because their cost depends only on the number of entries observed in $\mathbf{X}$, which is usually low, and not on the size of $\mathbf{X}$ which can be large.

Many real-world datasets are binary, that is, the entries of $\mathbf{X}$ take values in $\{0, 1\}$. Some examples include market basket data, click-stream data, network data or file data in complex software systems. In these cases $\mathbf{X}$ is fully observed and the aforementioned probabilistic approaches for solving the MF problem are infeasible in practice. This is because these methods are based on batch variational algorithms that require processing all the entries in $\mathbf{X}$ before producing even a single update to the variational parameters. An alternative is to use a likelihood function for continuous data instead of one for binary data (Nakajima et al., 2010). In this case, an analytic solution exists which scales with the number of ones in $\mathbf{X}$. However, this solution is restricted to zero-mean spherical priors on $\mathbf{U}$ and $\mathbf{V}$ and homoscedatic Gaussian likelihood functions for $\mathbf{X}$. These restrictions lead to poor predictions when $\mathbf{X}$ is binary.

We address the problem of scalable learning with probabilistic MF models that are accurate enough to produce state-of-the-art predictions on large binary matrices. To meet this challenge we propose a novel stochastic inference algorithm. Stochastic methods have the advantage that, with large datasets, they can make reasonably good predictions before a batch algorithm has generated a single parameter update. Our approach is based on stochastic variational inference (SVI) (Hoffman et al., 2013). Exist-

Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32. Copyright 2014 by the author(s).

[1]Equal contributors. NMTH and JMH are grateful for support from the Google European Doctoral Fellowship scheme and Infosys Labs, Infosys Limited, respectively.

ing implementations of SVI do not directly extend to MF models, which present specific challenges that are not encountered in models currently addressed by this inference algorithm, such as topic models. This is because in MF models we subsample individual matrix entries instead of complete data instances, e.g. an entire document in a topic model. In standard SVI all the variational parameters are updated each time a data instance is subsampled. With matrices, we have different parameters for each row and column in $\mathbf{X}$ and each time we subsample a matrix entry, we update only the variational parameters associated with the corresponding row and column. This makes the data sub-sampling strategy important because it determines which parameters are updated and how often. Therefore we develop novel data subsampling strategies with different sampling probabilities across the rows and columns of $\mathbf{X}$. These methods outperform standard uniform subsampling. Furthermore, parameter estimates in MF models often exhibit heavy-tailed empirical distributions (Lakshminarayanan et al., 2011). These heavy-tails can significantly reduce the convergence rate of stochastic algorithms. A solution is to use minibatches to reduce the effect of outliers in the noisy estimates of the gradients. However, the best minibatch size S can be dataset-dependent. To avoid having to hand-tune S to each dataset, which is common practice (Orr & Müller, 1998), we propose a method that adjusts the value of S online.

We scale probabilistic MF methods to large binary matrices whilst maintaining strong empirical performance. We validate our method experimentally, demonstrating faster convergence than batch alternatives (Raiko et al., 2007) and yielding more accurate solutions than existing scalable variational methods (Nakajima et al., 2010; Seeger & Bouchard, 2012; Paquet & Koenigstein, 2013). While we focus on improving the state-of-the-art in probabilistic MF methods, we also compare to one of the best non-probabilistic techniques for MF (Rendle et al., 2009). Although we do not attempt to beat every possible solution, our method also performs favorably. In summary, our algorithm has the following advantages:

1. We handle fully observed matrices and learn by subsampling individual matrix entries.
2. We use likelihood functions for binary data and not for continuous data.
3. Flexible priors and additional bias parameters can be easily incorporated with our method.
4. We use improved subsampling strategies and propose a rule to automatically select the minibatch size.

2. A Probabilistic Model for Binary Matrices

We describe a probabilistic model for an $L \times M$ sparse binary matrix $\mathbf{X}$. A common approach in matrix modeling is to assume a matrix factorization model (Salakhutdinov & Mnih, 2008). Let $\mathbf{U} \in \mathbb{R}^{L\times D}$ and $\mathbf{V} \in \mathbb{R}^{M\times D}$ be two low-rank matrices, where $D \ll \min(L, M)$. Then $\mathbf{X} = \Theta[\mathbf{U}\mathbf{V}^{\mathrm{T}} + z + \mathbf{E}]$, where $\Theta[\cdot]$ applies the Heaviside step function to each entry of a matrix, $z \in \mathbb{R}$ is a global bias parameter and $\mathbf{E}$ is an $L \times M$ additive noise matrix whose entries e_{ij} are i.i.d. with c.d.f. given by the logistic function $\sigma(x) = 1/[1 + \exp(-x)]$. The likelihood function is then

$$p(\mathbf{X}|\mathbf{U},\mathbf{V},z) = \prod_{i=1}^{L}\prod_{j=1}^{M} p(x_{i,j}|\mathbf{u}_i,\mathbf{v}_j,z)$$

$$= \prod_{i=1}^{L}\prod_{j=1}^{M} \left[\sigma(\mathbf{u}_i\mathbf{v}_j^{\mathrm{T}} + z)^{x_{i,j}} \cdot \sigma(-\mathbf{u}_i\mathbf{v}_j^{\mathrm{T}} - z)^{1-x_{i,j}}\right], \quad (1)$$

where $\mathbf{u}_i$ and $\mathbf{v}_j$ are the i-th and j-th rows of $\mathbf{U}$ and $\mathbf{V}$. We use fully factorized Gaussian priors for $\mathbf{U}$, $\mathbf{V}$ and z:

$$p(\mathbf{U}) = \prod_{i=1}^{L}\prod_{d=1}^{D} \mathcal{N}(u_{i,d}|\bar{u}^0_{i,d}, \tilde{u}^0_{i,d}),$$

$$p(\mathbf{V}) = \prod_{j=1}^{M}\prod_{d=1}^{D} \mathcal{N}(v_{j,d}|\bar{v}^0_{j,d}, \tilde{v}^0_{j,d})$$

and $p(z) = \mathcal{N}(z|\bar{z}^0, \tilde{z}^0)$. $\mathcal{N}(\cdot|m, v)$ denotes a Gaussian density with mean m and variance v. In our experiments we used priors with zero-mean and unit variance. We also incorporate a local bias to each row and column by fixing one column in each of $\mathbf{U}$ and $\mathbf{V}$ to a vector of ones. The posterior distribution for $\mathbf{U}$, $\mathbf{V}$ and z is

$$p(\mathbf{U},\mathbf{V},z|\mathbf{X}) = \frac{p(\mathbf{X}|\mathbf{U},\mathbf{V},z)p(\mathbf{U})p(\mathbf{V})p(z)}{p(\mathbf{X})}. \quad (2)$$

We can make predictions about the possible value $x^\star_{i,j}$ that an entry $x_{i,j}$ in $\mathbf{X}$ could have taken during the generation of $\mathbf{X}$ from $\mathbf{U}$, $\mathbf{V}$, b and $\mathbf{E}$. For this, we use

$$p(x^\star_{i,j}|\mathbf{X}) = \int \left[\sigma(\mathbf{u}_i\mathbf{v}_j^{\mathrm{T}} + z)^{x^\star_{i,j}} \cdot \sigma(-\mathbf{u}_i\mathbf{v}_j^{\mathrm{T}} - z)^{1-x^\star_{i,j}}\right] p(\mathbf{U},\mathbf{V},z|\mathbf{X})\, d\mathbf{U}d\mathbf{V}dz\,. \quad (3)$$

Equations (2) and (3) are intractable and approximations must be used. We now show how to use variational Bayes (Jordan et al., 1998) for computing approximations to (2) and (3).

2.1. Variational Bayes for Binary Matrices

Variational Bayes approximates the exact posterior (2) with a simpler, tractable distribution $q(\mathbf{U},\mathbf{V},z)$. We choose $q(\mathbf{U},\mathbf{V},z)$ to be a fully factorized Gaussian,

$$q(\mathbf{U},\mathbf{V},z) = \left[\prod_{i=1}^{L}\prod_{d=1}^{D} \mathcal{N}(u_{i,d}|\bar{u}_{i,d}, \tilde{u}_{i,d})\right] \times \left[\prod_{j=1}^{M}\prod_{d=1}^{D} \mathcal{N}(v_{j,d}|\bar{v}_{j,d}, \tilde{v}_{j,d})\right] \mathcal{N}(z|\bar{z}, \tilde{z}), \quad (4)$$

where $\Phi = \{\{\{\bar{u}_{i,d}, \tilde{u}_{i,d},\}_{i=1}^{L}, \{\bar{v}_{j,d}, \tilde{v}_{j,d}\}_{j=1}^{M}\}_{d=1}^{D}, \bar{z}, \tilde{z}\}$ are variational parameters that are adjusted so that

$q(\mathbf{U},\mathbf{V},z)$ is as similar as possible to $p(\mathbf{U},\mathbf{V},z|\mathbf{X})$ by minimizing the KL divergence between (4) and (2). Once $q(\mathbf{U},\mathbf{V},z)$ has been optimized, we approximate (3) by first approximating the posterior distribution of $\mathbf{u}_i\mathbf{v}_j^\text{T} + z$ with a Gaussian with mean $\mu_{i,j} = \sum_d \bar{u}_{i,d}\bar{v}_{j,d} + \bar{z}$, and variance $s^2_{i,j} = \sum_d \bar{u}^2_{i,d}\tilde{v}_{j,d} + \tilde{u}_{i,d}\bar{v}^2_{j,d} + \tilde{u}^2_{i,d}\tilde{v}_{j,d} + \tilde{z}$. Then we approximate the logistic function with a rescaled probit function that has the same slope at the origin as the logistic function $\sigma(\cdot)$ (MacKay, 1992). This yields

$$p(x^\star_{i,j}|\mathbf{X}) \approx \int \sigma[(2x^\star_{i,j}-1)a]\mathcal{N}(a|\mu_{i,j},s^2_{i,j})\,da \approx \sigma[\varphi(s^2_{i,j})\mu_{i,j}(2x^\star_{i,j}-1)]\,, \quad (5)$$

where $\varphi(x) = (1+\pi x/8)^{-1/2}$.

Equivalently, in variational Bayes $q(\mathbf{U},\mathbf{V},z)$ is optimized by maximizing the evidence lower bound or ELBO, $\mathcal{L}(\Phi) = \mathbb{E}_q\left[\log p(\mathbf{U},\mathbf{V},z,\mathbf{X})\right] - \mathbb{E}_q\left[\log q(\mathbf{U},\mathbf{V},z)\right]$. However, $\mathbb{E}_q\left[\log p(\mathbf{U},\mathbf{V},z,\mathbf{X})\right]$ is analytically intractable. To address this we use the Gaussian lower bound on the logistic function described in (Jaakkola & Jordan, 1997). We choose this approximation because it yields Gaussian complete conditional distributions. A complete conditional is the conditional distribution of a variable given all of the other variables and observations. Exponential family complete conditionals will allow us to use stochastic inference methods based on natural gradients, which improves convergence rates (Hoffman et al., 2013). We lower bound $\sigma(a)^{x_{i,j}}\cdot\sigma(-a)^{1-x_{i,j}}$ in (1) with

$$\tau(a,\xi) = e^{ax_{i,j}}\sigma(\xi)e^{-\frac{a+\xi}{2}+\lambda(\xi)(a^2-\xi^2)}\,, \quad (6)$$

where $\lambda(\xi) = (0.5-\sigma(\xi))/(2\xi)$ and ξ is adjusted to make the lower bound tight at $a = \pm\xi$. When we replace each $p(x_{i,j}|\mathbf{u}_i,\mathbf{v}_j,z)$ in (1) with an instantiation of (6) that includes its own parameter $\xi_{i,j}$, we obtain a new lower bound

$$\mathcal{L}'(\Phi,\Xi) = \sum_{i=1}^L\sum_{j=1}^M \alpha_{i,j} + \sum_{i=1}^L\sum_{d=1}^D \beta_{i,d} + \sum_{j=1}^M\sum_{d=1}^D \gamma_{j,d} + \kappa\,, \quad (7)$$

where $\Xi = \{\{\{\xi_{i,j}\}_{i=1}^L\}_{j=1}^M\}$,

$$\begin{aligned}\alpha_{i,j} &= \log\sigma(\xi_{i,j}) - \frac{\mu_{i,j}(1-2x_{i,j})+\xi_{i,j}}{2} + \\ &\quad \lambda(\xi_{i,j})(\mu^2_{i,j}+s^2_{i,j}-\xi^2_{i,j})\,,\\ \beta_{i,d} &= \rho(\tilde{u}_{i,d},\tilde{u}^0_{i,d},\bar{u}_{i,d},\bar{u}^0_{i,d}),\\ \gamma_{j,d} &= \rho(\tilde{v}_{j,d},\tilde{v}^0_{j,d},\bar{v}_{j,d},\bar{v}^0_{j,d}),\\ \kappa &= \rho(\tilde{z},\tilde{z},\bar{z}^0,\bar{z}^0)\,.\end{aligned}$$

and $\rho(a,b,c,d) = -0.5 - 0.5\log a/b + [(c-d)^2+a][2b]^{-1}$. One could tune q by the alternative maximization of $\mathcal{L}'$ with respect to Φ and Ξ. Given Φ, Ξ is optimized by setting

$$\xi_{i,j} = [\mu^2_{i,j}+s^2_{i,j}]^{0.5}\,. \quad (8)$$

Given Ξ, Φ can be optimized by doing an iteration of gradient descent (Raiko et al., 2007). This work contains a state-of-the-art batch algorithm for the optimization of the ELBO in MF models with Gaussian likelihood. Although effective with small datasets, the resulting batch algorithm is infeasible when $\mathbf{X}$ is very large because each iteration requires the examination of all of the entries in $\mathbf{X}$ before updating any parameters. For massive matrices, we propose to use stochastic optimization (Robbins & Monro, 1951). These techniques can produce parameter updates after examining only a reduced fraction of the data. The following section describes a stochastic method for optimizing $\mathcal{L}'$ in (7) based on stochastic variational inference (SVI) (Hoffman et al., 2013).

2.2. SVI for Binary Matrices

Stochastic optimization methods follow noisy estimates of the gradient of the target function. This function is often constructed by summing over a large number of terms. Noise in the gradient arises because the target function is approximated by a cheaper, noisy estimate which is obtained by summing over a reduced set of randomly subsampled terms. To optimize the correct objective function the subsampled terms must be re-scaled so that the expectation of the gradient of the noisy estimate is equal to the gradient of the original target function.

We apply stochastic optimization to $\mathcal{L}'(\Phi) = \max_\Xi \mathcal{L}'(\Phi,\Xi)$. For this, we iterate over the following steps. Firstly, we randomly select indexes $i \in \{1,\ldots,L\}$ and $j \in \{1,\ldots,M\}$ with probability $p(i,j)$. Secondly, we optimize $\xi_{i,j}$ by setting $\xi_{i,j} = [\mu^2_{i,j}+s^2_{i,j}]^{0.5}$ as in (8). Thirdly, we compute a noisy estimate of $\mathcal{L}'(\Phi)$:

$$\mathcal{L}'_\text{noisy}(\Phi_{i,j}) = [c^\alpha_{i,j}]^{-1}\alpha_{i,j} + \sum_{d=1}^D \beta_{i,d} + \sum_{d=1}^D \gamma_{j,d} + \kappa\,, \quad (9)$$

where $c^\alpha_{i,j}$ is a re-scaling constant. Finally, we update $\Phi_{i,j} = \{\{\bar{u}_{i,d},\tilde{u}_{i,d},\bar{v}_{j,d},\tilde{v}_{j,d}\}_{d=1}^D,\{\bar{z},\tilde{z}\}\}$ by making a small step in the direction of the gradient of (9). Intuitively, (9) is an appropriately re-scaled version of (7) that includes only those terms which have the same indexes i and j as the subsampled matrix entry $x_{i,j}$. Importantly, the constant $c^\alpha_{i,j}$ is chosen to guarantee that the expectation under the data-sampling strategy $p(i,j)$ of the gradient of (9) with respect to the elements of $\Phi_{i,j}$ is the same as the gradient of $\mathcal{L}'(\Phi)$. That is, when we update $\bar{u}_{i,d}$ or $\tilde{u}_{i,d}$ we set $c^\alpha_{i,j} = p(j|i)$. For $\bar{v}_{j,d}$ or $\tilde{v}_{j,d}$ we set $c^\alpha_{i,j} = p(i|j)$ and for $\bar{z}$ or $\tilde{z}$ we set $c^\alpha_{i,j} = p(i,j)$.

Instead of standard gradients, one can achieve much faster convergence using natural gradients (Amari, 1998). For this, we work with the natural parameters of (4):

$$\dot{u}_{i,d} = \bar{u}_{i,d}/\tilde{u}_{i,d}\,, \qquad \ddot{u}_{i,d} = 1/\tilde{u}_{i,d}\,,$$

and similarly for $\dot{v}_{j,d}$, $\ddot{v}_{j,d}$, $\dot{z}$ and $\ddot{z}$. Let $\mathring{\mathbf{u}}_{i,d} = (\dot{u}_{i,d},\ddot{u}_{i,d})$ and let $\nabla\mathcal{L}'(\mathring{\mathbf{u}}_{i,d})$ denote the natural gradient of (9) with

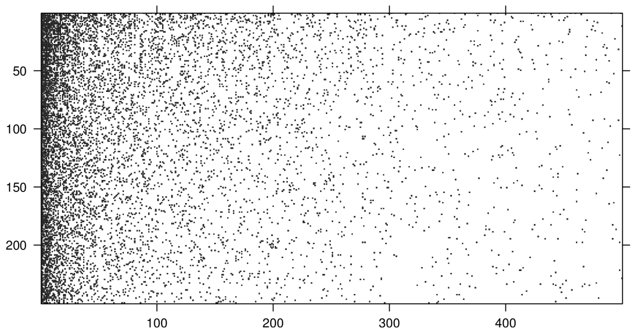

Figure 1. Binary matrix obtained by selecting randomly 250 rows with at least 10 ones and the 500 columns with the most ones from the BMS-POS dataset. This matrix is very sparse and has different frequencies of ones across rows and columns.

respect to $\mathring{\mathbf{u}}_{i,d}$. When the model has exponential family complete conditionals (as provided by the Gaussian approximation in (6)) then $\nabla\mathcal{L}'(\mathring{\mathbf{u}}_{i,d}) = \mathring{\mathbf{u}}^\star_{i,d} - \mathring{\mathbf{u}}_{i,d}$, where $\mathring{\mathbf{u}}^\star_{i,d} = (\dot{u}^\star_{i,d}, \ddot{u}^\star_{i,d})$ is the value of $\mathring{\mathbf{u}}_{i,d}$ that maximizes (9) when all the other natural parameters are fixed at their current values. Note that $\mathring{\mathbf{u}}^\star_{i,d}$ is a noisy estimate of the maximizer of the exact ELBO (7) with respect to $\mathring{\mathbf{u}}_{i,d}$. The resulting stochastic update for $\mathring{\mathbf{u}}_{i,d}$ is

$$\begin{aligned}\mathring{\mathbf{u}}^{\text{new}}_{i,d} &= \mathring{\mathbf{u}}^{\text{old}}_{i,d} + \rho^u_i \nabla\mathcal{L}'(\mathring{\mathbf{u}}_{i,d}) \\ &= (1-\rho^u_i)\mathring{\mathbf{u}}^{\text{old}}_{i,d} + \rho^u_i \mathring{\mathbf{u}}^\star_{i,d}, \qquad (10)\end{aligned}$$

where ρ^u_i is the size of the step taken in the direction of the natural gradient. The corresponding updates for $\mathring{\mathbf{v}}_{j,d}$ and $\mathring{z}$ are similar, see the supplementary material for details.

The resulting Stochastic Inference method for Binary Matrices (SIBM) iterates over the following two steps: i) randomly subsample an entry $x_{i,j}$ from $\mathbf{X}$ with probability $p(i,j)$ and ii) make a small update to the variational parameters that approximate the posterior distribution of the i-th row of $\mathbf{U}$, the j-th row of $\mathbf{V}$ and the global bias z. In practice, each time we sample the indices i and j, we first update $\mathring{\mathbf{z}}$, then all the $\mathring{\mathbf{v}}_{j,d}$ and finally all the $\mathring{\mathbf{u}}_{i,d}$. Each of these operations is performed using the updated parameter values produced by the previous operations. We also recompute the optimal value for $\xi_{i,j}$ whenever any of the natural parameters change.

2.3. The Sampling Distribution

We investigate the performance of different choices of $p(i,j)$, the probability distribution used to subsample the entries of $\mathbf{X}$. A common objective for binary matrix factorization is to predict the location of entries in $\mathbf{X}$ that would have taken value one but were flipped to value zero by the additive noise matrix $\mathbf{E}$. Real-world binary matrices are usually sparse, as illustrated in Figure 1. This means that when the sampling strategy $p(i,j)$ is uniform (denoted S-Uniform), $p(i,j) = 1/(LM)$, most of the sampled entries $x_{i,j}$ will take value zero. As a result, SIBM may take many iterations to converge to a good solution. We propose smarter strategies that subsample the more useful entries of $\mathbf{X}$ so that the model converges rapidly. This resembles 'active learning' (Settles, 2010). However, unlike in active learning, we must eliminate the bias introduced by our specific choice of $p(i,j)$, that is, we must select $c^\alpha_{i,j}$ in (9) so that the expected gradient of (9) is the same as the gradient of (7). Therefore, we propose two simple strategies for which we can compute the appropriate rescaling $c^\alpha_{i,j}$.

To ensure that we see enough ones, a better alternative (S-Balanced) is to sample zeros and ones with equal probability, regardless of the empirical frequencies in $\mathbf{X}$,

$$p(i,j) = 1/(2\sum_{a=1}^{L}\sum_{b=1}^{M}\mathbf{I}[x_{i,j} = x_{a,b}]),$$

where $\mathbf{I}[\cdot]$ is the indicator function. Now, each time that an entry is sampled we obtain a zero or a one with equal probability. However, another characteristic of real-world binary matrices is that the frequency of ones and zeros can vary considerably across the rows and columns. For example, the matrix in Figure 1 presents a few columns with a large number of ones and many with very few ones. A similar pattern is observed in the rows, although in this matrix the effect is smaller. In practice, it takes SIBM longer to model accurately the ones located in rows or columns with many zeros. Any entry sampled from these rows/columns will usually take value zero which is unlikely to be useful because SIBM can learn quickly that these rows/columns are very sparse. Therefore, we propose a strategy (S-Biased) to account for this by biasing S-Balanced so that the probability of sampling a one at location (i,j) is proportional to i) the number of zeros found in the i-th row and ii) the number of zeros found in the j-th column. An equivalent bias is introduced for the zeros. The result is the sampling distribution

$$p(i,j) = \frac{r_i^{(1-x_{i,j})} c_j^{(1-x_{i,j})}}{2\sum_{a=1}^{L}\sum_{b=1}^{M}\mathbf{I}[x_{i,j} = x_{a,b}] r_a^{(1-x_{a,b})} c_b^{(1-x_{a,b})}},$$

where $r_i^{(0)}$ and $r_i^{(1)}$ are the number of zeros and ones in the i-th row of $\mathbf{X}$ and $c_j^{(0)}$ and $c_j^{(1)}$ are the number of zeros and ones in the j-th column. These counts are thresholded to take a minimum value of 1 so that $p(i,j) \neq 0$.

2.4. Minibatches, Learning Rates and Minibatch Size

Stochastic methods often use minibatches to reduce the variance of the noisy estimates of the gradient and help the algorithm converge faster. Instead of updating the variational parameters after subsampling a single matrix entry, the updates are averaged over a minibatch of data. When using a minibatch of size S, we first subsample S entries from $\mathbf{X}$. Then for each subsampled entry $x_{i,j}$, we store the parameter values $\mathring{\mathbf{u}}^\star_{i,d}$ and $\mathring{\mathbf{v}}^\star_{j,d}$ that would have been produced during the execution of SIBM without minibatches. After subsampling S entries, we update each $\mathring{\mathbf{u}}_{i,d}$ if at least one of the entries in the minibatch belongs to the i-th row of $\mathbf{X}$. The minibatch update rule follows from (10),

$$\mathring{\mathbf{u}}^{\text{new}}_{i,d} = (1-\rho^u_i)\mathring{\mathbf{u}}^{\text{old}}_{i,d} + \rho^u_i \mathring{\mathbf{u}}^{\star,\text{avg}}_{i,d}\,, \tag{11}$$

$$\text{where} \quad \mathring{\mathbf{u}}^{\star,\text{avg}}_{i,d} = \frac{1}{n(i)}\sum_{s=1}^{n(i)} \mathring{\mathbf{u}}^{\star,s}_{i,d}\,. \tag{12}$$

$n(i)$ is the number of entries in the i-th row found in the minibatch and $\mathring{\mathbf{u}}^{\star,s}_{i,d}$ is the value of $\mathring{\mathbf{u}}^\star_{i,d}$ produced when the s-th of those entries appearing in the i-th row is subsampled. The minibatch update rule for $\mathring{\mathbf{v}}_{j,d}$ is similar.

An important question is how to choose the minibatch size S. The value of S is particularly important when working with matrix factorization models where parameter distributions are often heavy tailed (Lakshminarayanan et al., 2011). In our stochastic method this results in heavy tailed noisy estimates of the natural gradients. The choice of S governs a trade-off between reducing these heavy tails and slow convergence due to excessively large minibatches. To avoid having to hand-tune S to each dataset or run expensive cross validation searches, we propose an algorithm that selects S appropriately to the statistics of the data during learning. In particular, we choose S so that we bound the magnitude of the error in the noisy estimate of the optimum of the ELBO. Let $\mathring{\mathbf{u}}^{\star,\star}_{i,d}$ be the value of $\mathring{\mathbf{u}}_{i,d}$ that maximizes the exact ELBO (7). We obtain a probabilistic bound on the relative error of $\mathring{\mathbf{u}}^{\star,\text{avg}}_{i,d}$ in (11) with respect to the global maximizer of the ELBO, $\mathring{\mathbf{u}}^{\star,\star}_{i,d}$, using Markov's inequality,

$$\delta = p\left[\frac{\|\mathring{\mathbf{u}}^{\star,\text{avg}}_{i,d} - \mathring{\mathbf{u}}^{\star,\star}_{i,d}\|_2^2}{\|\mathring{\mathbf{u}}^{\star,\star}_{i,d}\|_2^2} \geq \theta\right] \leq \frac{\mathbb{E}[\|\mathring{\mathbf{u}}^{\star,\text{avg}}_{i,d} - \mathring{\mathbf{u}}^{\star,\star}_{i,d}\|_2^2]}{\theta\|\mathring{\mathbf{u}}^{\star,\star}_{i,d}\|_2^2} = \frac{\|\text{Var}[\mathring{\mathbf{u}}^\star_{i,d}]\|_1}{\theta\|\mathbb{E}[\mathring{\mathbf{u}}^\star_{i,d}]\|_2^2}\mathbb{E}\left[\frac{1}{n(i)}\right] \approx \frac{\|\text{Var}[\mathring{\mathbf{u}}^\star_{i,d}]\|_1}{\theta S p(i)\|\mathbb{E}[\mathring{\mathbf{u}}^\star_{i,d}]\|_2^2}\,,$$

where $\text{Var}[\mathring{\mathbf{u}}^\star_{i,d}]$ is a vector with the variances of the entries in $\mathring{\mathbf{u}}^\star_{i,d}$, $p(i)$ is the probability of sampling an element from the i-th row of $\mathbf{X}$, that is, $p(i) = \sum_j p(i,j)$. We have approximated $\mathbb{E}\left[1/n(i)\right]$ by $1/[p(i)S]$ and we have used the fact that $\mathring{\mathbf{u}}^{\star,\star}_{i,d} = \mathbb{E}[\mathring{\mathbf{u}}^{\star,\text{avg}}_{i,d}]$. We now solve for S and obtain a minibatch size that approximately limits the probability that the relative error of $\mathring{\mathbf{u}}^{\star,\text{avg}}_{i,d}$ is larger than θ:

$$S = \|\text{Var}[\mathring{\mathbf{u}}^\star_{i,d}]\|_1 \left[\theta\delta p(i)\|\mathbb{E}[\mathring{\mathbf{u}}^\star_{i,d}]\|_2^2\right]^{-1}\,. \tag{13}$$

Intuitively, the minibatch size increases with the inverse of the signal to noise ratio (SNR) in the estimate $\mathring{\mathbf{u}}^\star_{i,d}$ of the global maximizer of the exact ELBO in (7). If the SNR decreases, (13) chooses large minibatches to mitigate the large relative errors.

This approach requires choosing a single dataset-independent parameter, the product of θ and δ, as opposed to hand-tuning S to each dataset. By making $\theta\delta$ small we limit the expected deviation of $\mathring{\mathbf{u}}^{\star,\text{avg}}_{i,d}$ from $\mathring{\mathbf{u}}^{\star,\star}_{i,d}$. Note that (13) requires knowing $\mathbb{E}[\mathring{\mathbf{u}}^\star_{i,d}]$ and $\text{Var}[\mathring{\mathbf{u}}^\star_{i,d}]$. We estimate these quantities online using exponentially weighted moving averages. Equation (13) provides a different minibatch size for each of the $\mathring{\mathbf{u}}_{i,d}$ and the rule for each of the $\mathring{\mathbf{v}}_{j,d}$ is similar. Therefore we select S to be the mean minibatch size selected for each $\mathring{\mathbf{u}}_{i,d}$ and $\mathring{\mathbf{v}}_{i,d}$, details are in the supplementary material. The contribution of $\mathring{\mathbf{z}}$ to the minibatch size S is very small and so is ignored in practice.

The step sizes ρ^u_i, ρ^v_j and ρ_z should be reduced each time $\mathring{\mathbf{u}}_{i,d}$, $\mathring{\mathbf{v}}_{j,d}$ and $\mathring{\mathbf{z}}$ are updated. For this, we use a simple Robbins-Monro schedule (Robbins & Monro, 1951). The full SIBM routine is summarized in Algorithm 1.

Algorithm 1 Stochastic Inference for Binary Matrices

```
Input: matrix $\mathbf{X}$, initial parameters $\Phi$, # samples $T$
for $t = 1$ to $T$ do
  select minibatch size $S$ (see Section 2.4)
  for $s = 1$ to $S$ do
    sample row and column indices $(i,j) \sim p(i,j)$
    save $\mathring{\mathbf{u}}_{i,1},\ldots,\mathring{\mathbf{u}}_{i,D}$, $\mathring{\mathbf{v}}_{j,1},\ldots,\mathring{\mathbf{v}}_{j,D}$ and $\mathring{\mathbf{z}}$
    update $\xi_{i,j}$ using (8)
    compute $\mathring{\mathbf{z}}^\star$ and update $\mathring{\mathbf{z}}$ using (10)
    update $\mathring{\mathbf{z}}^{\star,\text{avg}}$ using (12)
    for $d = 1$ to $D$ do
      update $\xi_{i,j}$ using (8)
      compute $\mathring{\mathbf{v}}^\star_{j,d}$ and update $\mathring{\mathbf{v}}_{j,d}$ using (10)
      update $\mathring{\mathbf{v}}^{\star,\text{avg}}_{j,d}$ using (12)
    end for
    similarly, update $\mathring{\mathbf{u}}^\star_{i,1},\ldots,\mathring{\mathbf{u}}^\star_{i,D}$
    restore $\mathring{\mathbf{u}}_{i,1},\ldots,\mathring{\mathbf{u}}_{i,D}$, $\mathring{\mathbf{v}}_{j,1},\ldots,\mathring{\mathbf{v}}_{j,D}$ and $\mathring{\mathbf{z}}$
  end for
  for any row $i$ sampled in the last minibatch do
    compute step size $\rho^u_i$ using Robbins-Monro
    update $\mathring{\mathbf{u}}_{i,1},\ldots,\mathring{\mathbf{u}}_{i,D}$ using (11)
  end for
  similarly, update $\mathring{\mathbf{v}}_{j,1},\ldots,\mathring{\mathbf{v}}_{j,D}$
  compute step size $\rho^z$ using Robbins-Monro
  update $\mathring{\mathbf{z}}$ using (11)
end for
Output: $\{\mathring{\mathbf{u}}_{i,1},\ldots,\mathring{\mathbf{u}}_{i,D}\}_{i=1}^L$, $\{\mathring{\mathbf{v}}_{j,1},\ldots,\mathring{\mathbf{v}}_{j,D}\}_{j=1}^M$ and $\mathring{\mathbf{z}}$
```

3. Related Work

SVI has been applied to other probabilistic models (Hoffman et al., 2010; Wang et al., 2011; Bryant & Sudderth, 2012). In these cases there is a clear distinction between *local* and *global* variational parameters. Local parameters are updated only when a particular data point is subsampled. Global parameters are updated whenever any data point is subsampled. In MF models, we have variational parameters that are *partially global*, that is, they are only updated when elements in the corresponding row or column are subsampled. This makes the data sub-sampling strategy more important because it determines which parameters are updated and how often. Other non-uniform sampling strategies have been proposed for networks, an instance of binary matrices, (Gopalan & Blei, 2013). The stochastic blockmodel for networks used here differs to our general binary MF model.

An alternative stochastic MF algorithm just subsamples the zeros (Paquet & Koenigstein, 2013). However, unlike SIBM, this method does not correct for the bias introduced by the subsampling process and hence yields poorer solutions. Instead of stochastic schemes one could use the analytic solution described in (Nakajima et al., 2010). However, this is only applicable when i) the likelihood (1) is Gaussian with equal variance across matrix entries, ii) $\mathbf{U}$ and $\mathbf{V}$ have zero-mean isotropic priors, and iii) there are no bias parameters. These constraints have a large negative effect in predictive performance. An iterative scheme has been proposed to extend this approach to logistic likelihoods (Seeger & Bouchard, 2012) at the cost of making crude approximations to the logistic likelihood function. In practice, this method tends to produce only small gains with respect to (Nakajima et al., 2010) with binary matrices. Very recently, an MF model with a Poisson likelihood has been proposed (Gopalan et al., 2014). This algorithm is applicable to binary matrices and scales with the number of ones, although the Poisson likelihood is less specific than the logistic function for binary data.

A large number of alternative non-probabilistic algorithms have been proposed for MF. One of the best performing is Bayesian Personalized Ranking (BPR) which optimizes a ranking loss function. BPR has shown state-of-the-art results on item recommendation tasks (Rendle et al., 2009; Dror et al., 2012).

Our minibatch size S selection algorithm is similar to one in (Byrd et al., 2012). However, their method selects S to be inversely proportional to the SNR of the noisy estimate of the gradient. Since the gradient tends to zero at convergence, the value of S selected by their method quickly diverges. We do not have this problem because we use the SNR of the noisy estimate of the global maximizer of (7), which does not converge to zero.

4. Experiments

SIBM is evaluated in experiments with synthetic and real-world binary matrices. All the code and data used is publicly available[1]. We consider six datasets that include a synthetic dataset generated by sampling $\mathbf{X}$ from the generative model assumed by SIBM. We fix $D = 10$ and generate $\mathbf{U}$ and $\mathbf{V}$ by sampling all the $u_{i,d}$ and $v_{j,d}$ independently from $\mathcal{N}(0, 1)$. The global bias is fixed to $z = -7.5$, yielding binary matrices with about 98% sparsity. We consider two real-world datasets from the FIMI repository: purchase data from a retail store (retail) (Brijs et al., 1999) and click data from an online news portal (Kosarak). We include two datasets from the 2000 KDD Cup (Kohavi et al., 2000; Zheng et al., 2001), point of sale data from a retailer (POS, originally BMS-POS) and click data from an e-commerce website (WebView, originally BMS-WebView-2). Finally, we include the Netflix data, treating 4-5 star ratings as ones. We pre-process the original datasets so that we can compare to the computationally expensive batch approach. We keep the 1000 columns with the highest number of ones and discard rows with fewer than 10 ones. We consider *small* and *large* versions of each dataset. We subsample 2000 rows for the small and 40,000 rows for the large datasets, except in retail and WebView, where we use approximately the maximum number of rows for the large datasets, 10,000 and 5000, respectively.

[1] http://jmhl.org

Each matrix is randomly split into a training matrix and a set of test entries with value one. The training matrix is generated by randomly removing an entry with value one from each row in the original matrix and adding it to the test set. Predictive performance is evaluated using recall at N, a popular metric for recommendation tasks (Gunawardana & Shani, 2009) (equivalent to precision when a single one is held out from each row). For this, we iterate over the rows, using (3) to compute the probability of each zero entry actually taking value one. We select the top N zero entries with highest probability in that row. Recall is computed as the average number of times that the test entry appears in this list. We use $N = 10$ and repeat the experiment 25 times on each small dataset, and 10 times on each large one.

4.1. Sampling Strategies and Automatic Minibatch

The top of Figure 2 shows results for SIBM when using the sampling strategies S-Uniform, S-Balanced and S-Biased on the small Netflix dataset. To eliminate the dependence of these strategies on the minibatch size, we select the value of S for each strategy using cross-validation. We find that S-Biased performs best, and both S-Biased and S-Balanced improve over S-Uniform. Similar results are obtained on the other datasets, see the supplementary material.

The plot in the bottom of Figure 2 shows the evolution of the minibatch size S on each small dataset. We fixed $\theta\delta = 2$ in (13) in all of our experiments. We fixed the minimum value for S to $\max(L, M) = 2000$. This value is selected with the retail dataset. Similar results are obtained with the large datasets. This plot shows that the optimal value of S varies greatly across datasets. Interestingly, for some datasets S grows as learning progresses, but for others it shrinks.

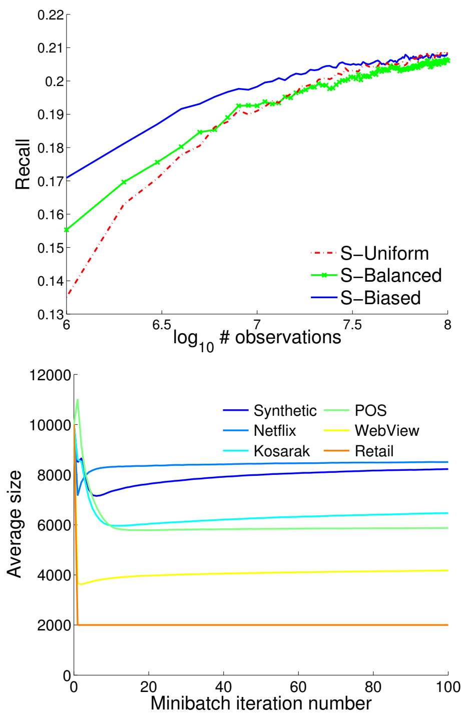

Figure 2. Top: average recall obtained for different sampling strategies with the small Netflix dataset. Bottom: evolution of the average minibatch size S selected in each small dataset.

4.2. Comparison with Batch and Alternative Methods

We compare the full SIBM algorithm that selects the minibatch size S automatically (SIBM-auto) to a version in which S is selected via cross-validation to maximize recall on a validation set (SIBM-recall). We also compare to a version of SIBM-recall that finds the *Maximum a Posteriori* (MAP) solution using stochastic gradient ascent (MAP-recall), see the supplementary material for details. On the small datasets we compare to the batch algorithm (batch) that maximizes (7) (Raiko et al., 2007). This method is too expensive with the large datasets. In these cases, we run it by subsampling zeros, keeping only 20 times as many zeros as ones.

We compare our method to the analytic solution with a Gaussian likelihood (Nakajima et al., 2010) (Nak10) and the extension of this method to binary matrices (Seeger & Bouchard, 2012) (See12). We also evaluate the scheme described in (Paquet & Koenigstein, 2013) (Paq13). Finally, we compare to one of the best performing non-variational Bayesian algorithms, BPR (Rendle et al., 2009).

4.3. Results

Figure 3 shows the average recall obtained by each method versus the number of entries from $\mathbf{X}$ that are observed (sampled). Other than the analytic solutions (Nak10 and See12), all algorithms have linear cost in the number of observations. It is hard to quantify the number of entries observed by Nak10 and See12 which are based on iterative calls to an SVD subroutine. Therefore, we assume that they run instantaneously and their performance is presented as a constant line[2]. Tables 1 and 2 show the average recall and average negative ELBO (7) (cost) after taking 10^7 samples with the small datasets and WebView and Retail large datasets, and 10^8 on the others. With the large datasets computing the ELBO is too expensive so we do not report cost. Bold typeface indicates the best results (and those statistically indistinguishable), underlining denotes the second best. Tables 1, 2 show that in terms of recall, the best method is SIBM-recall, with SIBM-auto coming close. Regarding the ELBO, SIBM-auto yields the best results.

Figure 3 shows that SIBM converges faster than batch and sometimes to better solutions, such as with the WebView dataset. SIBM-auto produces the greatest improvements during the first iterations of learning. These first iterations are the most relevant iterations for large scale learning. With massive data, only a few passes over the available data are possible. It is in these cases that stochastic methods are most useful. The results of SIBM-auto are very close to those of the gold-standard, SIBM-recall, and MAP-recall performs worse in general than the variational methods SIBM-auto and SIBM-recall. MAP-recall seems to overfit since its performance sometimes deteriorates during later iterations. The analytic algorithms (Nak10, See12) obtain poor results due to the simplistic modelling assumptions that they make. Paq13 can perform poorly because this method subsamples the zeros and does not account for the bias introduced by the subsampling process. As a result, it converges to suboptimal solutions. BPR converges to worse solutions than SIBM and batch. On the large datasets the relative performances are similar, see the supplementary material.

Figure 4 shows recall versus wall-clock time for the small Kosarak dataset. Plots for the other datasets are in the supplementary material. This plot is implementation-dependent (all methods are coded in C) and so is less objective than Figure 3 which uses the number of entries observed. Nevertheless, the results for recall vs. time and recall vs. number of observed entries are similar. The main difference is that SIBM-recall, MAP-recall and BPR are penalized due to the additional time required to run cross validation searches for selecting the minibatch size (SIBM-recall and MAP-recall) and regularization parameters (BPR).

[2]This is a generous assumption for See12, see wall-clock times in Figure 4 and in the supplementary material.

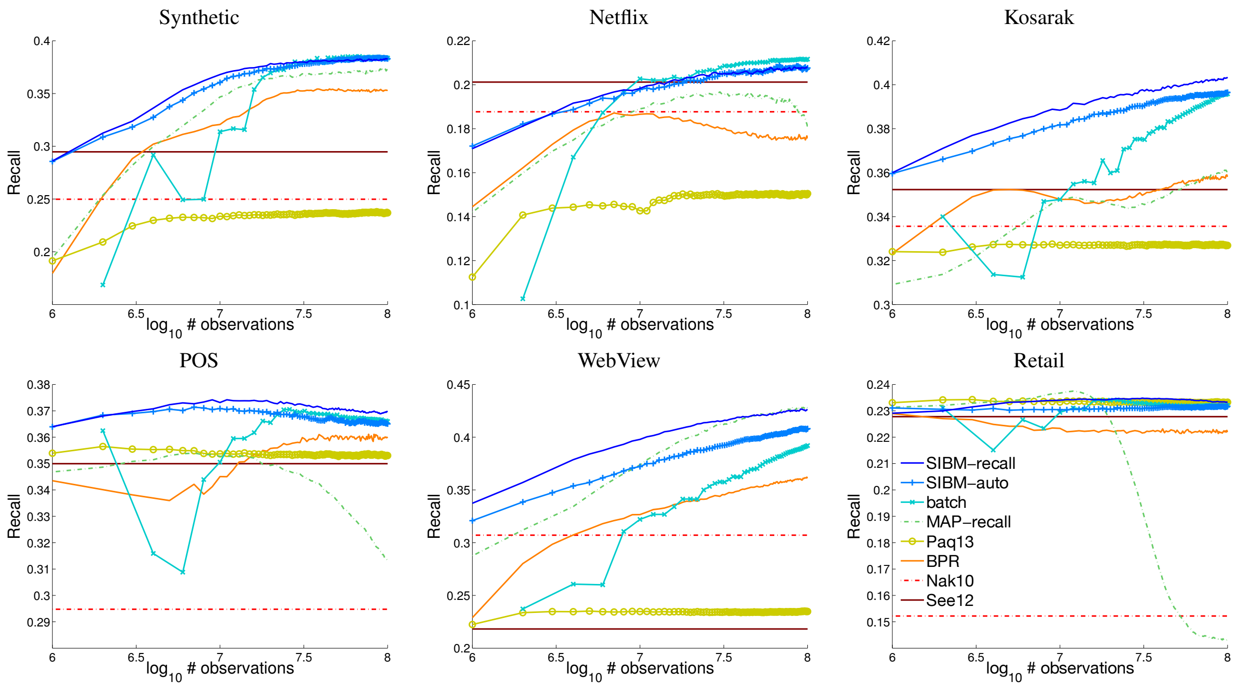

Figure 3. Average recall for each method on each small dataset versus number of samples drawn from $\mathbf{X}$.

	recall								cost$\times 10^{-5}$			
Dataset	SIBM recall	SIBM auto	batch	MAP recall	Paq13	BPR	Nak10	See12	SIBM recall	SIBM auto	batch	See12
Synthetic	**0.368**	0.360	0.314	0.347	0.234	0.321	0.250	0.295	**1.804**	**1.803**	1.821	4.313
Netflix	0.198	0.198	**0.203**	0.189	0.143	0.187	0.188	**0.201**	4.555	4.550	**4.383**	6.807
Kosarak	**0.388**	0.382	0.348	0.348	0.327	0.348	0.336	0.352	2.124	**1.963**	1.994	3.607
POS	**0.373**	**0.371**	0.351	0.353	0.354	0.345	0.295	0.350	**1.413**	**1.415**	1.437	2.674
WebView	**0.398**	0.372	0.322	0.374	0.235	0.327	0.307	0.218	1.672	**1.573**	1.630	2.886
Retail	0.234	0.230	0.229	**0.237**	0.233	0.223	0.152	0.228	1.557	**1.490**	1.511	2.430

Table 1. Small datasets, recall and cost after observing 10^7 samples.

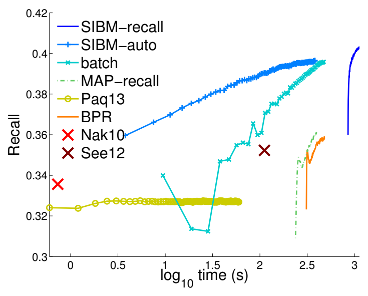

Figure 4. Average recall for each method on the small Kosarak dataset versus time.

Table 2. Large datasets, recall after observing 10^7 samples from WebView, Retail and 10^8 from others.

Dataset	SIBM recall	SIBM auto	batch	MAP recall	Paq13	BPR	Nak10	See12
Synthetic	**0.387**	0.367	0.324	0.368	0.249	0.374	0.262	0.266
Netflix	**0.203**	0.193	0.190	0.192	0.146	0.190	0.190	0.199
Kosarak	**0.391**	0.372	0.346	0.368	0.327	0.370	0.319	0.341
POS	0.373	0.368	0.348	0.352	0.352	**0.374**	0.289	0.347
WebView	**0.390**	0.343	0.359	0.360	0.235	0.326	0.303	0.213
Retail	**0.235**	0.230	0.233	**0.239**	0.235	**0.237**	0.149	0.228

5. Conclusions

We have proposed a new stochastic inference method for efficient factorization of large binary matrices. Our approach extends stochastic variational inference (SVI) to matrix factorization models, a class of models not previously addressed by SVI. The proposed method has the following advantages with respect to existing probabilistic solutions for binary matrix factorization: i) we can handle fully observed matrices, ii) learning occurs by subsampling the matrix entries, iii) we use likelihood functions for binary data instead of for continuous data, iv) flexible priors and additional bias parameters can be incorporated into the method easily. The resulting technique achieves faster convergence than an alternative batch approach and has better predictive performance than other state-of-the-art scalable solutions or analytic methods based on the SVD decomposition. Good performance in this domain requires smart data subsampling mechanisms and the use of minibatches. Therefore we have provided a novel non-uniform data subsampling strategy and a technique to adjust the minibatch size adaptively to the data. Our minibatch selection algorithm could be used more generally with other SVI algorithms.

References

Amari, Shun-Ichi. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

Brijs, Tom, Swinnen, Gilbert, Vanhoof, Koen, and Wets, Geert. Using association rules for product assortment decisions: a case study. In *KDD*, pp. 254–260, 1999.

Bryant, Michael and Sudderth, Erik. Truly nonparametric online variational inference for hierarchical Dirichlet processes. In *NIPS*, pp. 2708–2716, 2012.

Byrd, Richard H, Chin, Gillian M, Nocedal, Jorge, and Wu, Yuchen. Sample size selection in optimization methods for machine learning. *Mathematical programming*, 134(1):127–155, 2012.

Dror, Gideon, Koenigstein, Noam, Koren, Yehuda, and Weimer, Markus. The yahoo! music dataset and kdd-cup'11. *Journal of Machine Learning Research-Proceedings Track*, 18:8–18, 2012.

Gopalan, Prem, Ruiz, Francisco JR, Ranganath, Rajesh, and Blei, David M. Bayesian nonparametric poisson factorization for recommendation systems. In *AISTATS*, pp. 275–283, 2014.

Gopalan, Prem K and Blei, David M. Efficient discovery of overlapping communities in massive networks. *Proceedings of the National Academy of Sciences*, 110(36):14534–14539, 2013.

Gunawardana, Asela and Shani, Guy. A survey of accuracy evaluation metrics of recommendation tasks. *The Journal of Machine Learning Research*, 10:2935–2962, 2009.

Hoffman, Matthew, Blei, David M, and Bach, Francis. Online learning for latent dirichlet allocation. *NIPS*, 23:856–864, 2010.

Hoffman, Matthew D., Blei, David M., Wang, Chong, and Paisley, John. Stochastic variational inference. *Journal of Machine Learning Research*, 14:1303–1347, 2013.

Jaakkola, T and Jordan, M. A variational approach to Bayesian logistic regression models and their extensions. In *Sixth International Workshop on Artificial Intelligence and Statistics*, 1997.

Jordan, Michael I., Ghahramani, Zoubin, Jaakkola, Tommi S., and Saul, Lawrence K. An introduction to variational methods for graphical models. In *Learning in Graphical Models*, volume 89, pp. 105–161. Springer Netherlands, 1998.

Kohavi, Ron, Brodley, Carla E., Frasca, Brian, Mason, Llew, and Zheng, Zijian. KDD-Cup 2000 organizers' report: peeling the onion. *SIGKDD Explorations Newsletter*, 2:86–93, 2000.

Koren, Yehuda. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD*, KDD '08, pp. 426–434, New York, NY, USA, 2008. ACM.

Lakshminarayanan, Balaji, Bouchard, Guillaume, and Archambeau, Cedric. Robust Bayesian matrix factorisation. In *International Conference on Artificial Intelligence and Statistics*, pp. 425–433, 2011.

Lim, Yew Jin and Teh, Yee Whye. Variational Bayesian approach to movie rating prediction. In *Proceedings of KDD Cup and Workshop*, pp. 15–21, 2007.

MacKay, David JC. The evidence framework applied to classification networks. *Neural computation*, 4(5):720–736, 1992.

Nakajima, Shinichi, Sugiyama, Masashi, and Tomioka, Ryota. Global analytic solution for variational Bayesian matrix factorization. *NIPS*, 23:1759–1767, 2010.

Orr, Genevieve B. and Müller, Klaus-Robert (eds.). *Neural Networks: Tricks of the Trade, this book is an outgrowth of a 1996 NIPS workshop*, London, UK, UK, 1998. Springer-Verlag. ISBN 3-540-65311-2.

Paquet, Ulrich and Koenigstein, Noam. One-class collaborative filtering with random graphs. *WWW*, pp. 999–1008, 2013.

Paquet, Ulrich, Thomson, Blaise, and Winther, Ole. A hierarchical model for ordinal matrix factorization. *Statistics and Computing*, 22(4):945–957, 2012.

Raiko, T., Ilin, A., and Juha, K. Principal component analysis for large scale problems with lots of missing values. In *Machine Learning: ECML 2007*, volume 4701 of *Lecture Notes in Computer Science*, pp. 691–698. Springer Berlin / Heidelberg, 2007.

Rendle, Steffen, Freudenthaler, Christoph, Gantner, Zeno, and Schmidt-Thieme, Lars. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 452–461. AUAI Press, 2009.

Robbins, Herbert and Monro, Sutton. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

Salakhutdinov, Ruslan and Mnih, Andriy. Probabilistic matrix factorization. In *NIPS*, volume 20, 2008.

Seeger, Matthias and Bouchard, Guillaume. Fast variational Bayesian inference for non-conjugate matrix factorization models. *Journal of Machine Learning Research - Proceedings Track*, 22:1012–1018, 2012.

Settles, Burr. Active learning literature survey. *University of Wisconsin, Madison*, 2010.

Srebro, Nathan, Rennie, Jason DM, and Jaakkola, Tommi. Maximum-margin matrix factorization. In *NIPS*, pp. 1329–1336, 2005.

Stern, David H, Herbrich, Ralf, and Graepel, Thore. Matchbox: large scale online Bayesian recommendations. In *WWW*, pp. 111–120. ACM, 2009.

Wang, Chong, Paisley, John, and Blei, David M. Online variational inference for the hierarchical Dirichlet process. In *AISTATS*, pp. 752–760, 2011.

Zheng, Zijian, Kohavi, Ron, and Mason, Llew. Real world performance of association rule algorithms. In *SIGKDD*, pp. 401–406, 2001.