Zero-Truncated Poisson Tensor Factorization for Massive Binary Tensors

Changwei Hu[*]
ECE Department
Duke University
Durham, NC 27708

Piyush Rai[*]
ECE Department
Duke University
Durham, NC 27708

Lawrence Carin
ECE Department
Duke University
Durham, NC 27708

Abstract

We present a scalable Bayesian model for low-rank factorization of massive tensors with binary observations. The proposed model has the following key properties: (1) in contrast to the models based on the logistic or probit likelihood, using a zero-truncated Poisson likelihood for binary data allows our model to scale up in the number of *ones* in the tensor, which is especially appealing for massive but sparse binary tensors; (2) side-information in form of binary pairwise relationships (e.g., an adjacency network) between objects in any tensor mode can also be leveraged, which can be especially useful in "cold-start" settings; and (3) the model admits simple Bayesian inference via batch, as well as *online* MCMC; the latter allows scaling up even for *dense* binary data (i.e., when the number of ones in the tensor/network is also massive). In addition, non-negative factor matrices in our model provide easy interpretability, and the tensor rank can be inferred from the data. We evaluate our model on several large-scale real-world binary tensors, achieving excellent computational scalability, and also demonstrate its usefulness in leveraging side-information provided in form of mode-network(s).

1 INTRODUCTION

With the recent surge in multiway, multirelational, or "tensor" data sets (Nickel et al., 2011; Kang et al., 2012), learning algorithms that can extract useful knowledge from such data are becoming increasingly important. Tensor decomposition methods (Kolda and Bader, 2009) offer an attractive way to accomplish this task. Among tensor data, of particular interest are real-world *binary* tensors, which are now ubiquitous in problems involving social networks, recommender systems, and knowledge bases, etc. For instance, in a knowledge base, predicate relations defined over the tuples (subjects, objects,verbs) can be represented in form of a binary three-way tensor (Kang et al., 2012).

Usually, real-world binary tensors are massive (each dimension can be very large) but extremely sparse (very few ones in the tensor). For example, in a recommender system, each positive example (e.g., an item selected a set) implicitly creates several negative examples (items *not* chosen). Likewise, in a knowledge base, the validity of one relation automatically implies invalidity of several other relations. In all these settings, the number of negative examples greatly overwhelms the number of positive examples.

Unfortunately, binary tensor factorization methods (Nickel et al., 2011; Xu et al., 2013; Rai et al., 2014), based on probit or logistic likelihood, scale poorly for massive binary tensors because these require evaluating the likelihood/loss-function on *both* ones as well as zeros in the tensor. One possibility is to use heuristics such as *undersampling* the zeros, but such heuristics usually result in less accurate solutions. Another alternative is to use the *squared loss* (Hidasi and Tikk, 2012; Nickel et al., 2012) as the model-fit criterion, which facilitates linear scalability in the number of ones in the tensor. However, such an approach can often lead to suboptimal results (Ermis and Bouchard, 2014) in practice.

It is therefore desirable to have methods that can perform efficient tensor decomposition for such data, ideally with a computational-complexity that depends only on the number of nonzeros (i.e., the ones) in the tensor, rather than the "volume" of the tensor. Motivated by this problem, we present a scalable Bayesian model for the Canonical PARAFAC (CP) tensor decomposition (Kolda and Bader, 2009), with an inference-complexity that scales linearly in the number of ones in the tensor. Our model uses a zero-truncated Poisson likelihood for each binary observation in the tensor; this obviates the evaluation of the likelihoods for the zero entries. At the same time, the significant speed-up is not at the cost of sacrificing on the quality of the solution. As our experimental results show, the proposed like-

[*]Equal contribution

lihood model yields comparable or better results to logistic likelihood based models, while being an order of magnitude faster in its running-time on real-world binary tensors. Note that replacing the zero-truncated Poisson by the standard Poisson makes our model also readily applicable for count-valued tensors (Chi and Kolda, 2012); although, in this exposition, we will focus exclusively on binary tensors.

Often, side-information (Acar et al., 2011; Beutel et al., 2014), e.g., pairwise relationships (partially/fully observed), may also be available for objects in some of the tensor dimensions. For example, in addition to a binary tensor representing AUTHORS $\times$ WORDS $\times$ VENUES associations, the AUTHOR $\times$ AUTHOR co-authorship network may be available (at least for some pairs of authors). Such a network may be especially useful in "cold-start" settings where there is no data for some of the entities of a mode in the tensor (e.g., for some authors, there is no data in the tensor), but a network between entities in that mode may be available (See Fig 1 for an illustration). Our model allows leveraging such network(s), without a significant computational overhead, using the zero-truncated Poisson likelihood *also* to model these binary pairwise relationships.

To facilitate efficient fully Bayesian inference, we develop easy-to-implement batch as well as *online* MCMC inference; the latter is especially appealing for handling *dense* binary data, i.e., when the number of ones in the tensor and/or the network is also massive. Another appealing aspect about the model is its interpretability; a Dirichlet prior on the columns of each factor matrix naturally imposes non-negativity. In addition, the rank of decomposition can be inferred from the data.

2 CANONICAL PARAFAC (CP) TENSOR DECOMPOSITION

The Canonical PARAFAC (CP) decomposition (Kolda and Bader, 2009) offers a way to express a tensor as a sum of rank-1 tensors. Each rank-1 tensor corresponds to a specific "factor" in the data. More specifically, the goal in CP decomposition is to decompose a tensor $\mathcal{Y}$ of size $n_1 \times n_2 \times \cdots \times n_K$, with n_k denoting the size of $\mathcal{Y}$ along the k^{th} mode (or "way") of the tensor, into a set of K factor matrices $\mathbf{U}^{(1)}, \ldots, \mathbf{U}^{(K)}$ where $\mathbf{U}^{(k)} = [\boldsymbol{u}_1^{(k)}, \ldots, \boldsymbol{u}_R^{(k)}],\ k = \{1, \ldots, K\}$, denotes the $n_k \times R$ factor matrix associated with mode k.

In its most general form, CP decomposition expresses the tensor $\mathcal{Y}$ via a weighted sum of R rank-1 tensors as

$$\mathcal{Y} \sim f(\sum_{r=1}^{R} \lambda_r.\boldsymbol{u}_r^{(1)} \odot \ldots \odot \boldsymbol{u}_r^{(K)}) \tag{1}$$

In the above, the form of the link-function f depends on the type of data being modeled (e.g., f can be Gaussian for real-valued, Bernoulli-logistic for binary-valued, Poisson for count-valued tensors). Here λ_r is the weight associated with the r^{th} rank-1 component, the $n_k \times 1$ column vector $\boldsymbol{u}_r^{(k)}$ represents the r^{th} latent factor of mode k, and $\odot$ denotes vector outer product.

We use subscript $\boldsymbol{i} = \{i_1, \ldots, i_K\}$ to denote the K-dimensional index of the $\boldsymbol{i}$-th entry in the tensor $\mathcal{Y}$. Using this notation, the $\boldsymbol{i}$-th entry of the tensor $\mathcal{Y}$ can be written as $y_{\boldsymbol{i}} \sim f(\sum_{r=1}^{R} \lambda_r \prod_{k=1}^{K} u_{i_k r}^{(k)})$.

3 TRUNCATED POISSON TENSOR DECOMPOSITION FOR BINARY DATA

Our focus in this paper is on developing a probabilistic, fully Bayesian method for scalable low-rank decomposition of massive *binary* tensors. As opposed to tensor decomposition models based on the logistic likelihood for binary data (Xu et al., 2013; Rai et al., 2014), which require evaluation of the likelihood for both ones as well as zeros in the tensor, and thus can be computationally infeasible to run on massive binary tensors, our proposed model only requires the likelihood evaluations on the *nonzero* (i.e., the ones) entries in the tensor, and can therefore easily scale to massive binary tensors. Our model is applicable to tensors of any order $K \geq 2$ (the case $K = 2$ being a binary matrix).

Our model is based on a decomposition of the form given in Eq. 1; however, instead of using a Bernoulli-logistic link f to generate each binary observation $y_{\boldsymbol{i}}$ in $\mathcal{Y}$, we assume an additional layer (Eq. 2) which takes a *latent* count-valued $y_{\boldsymbol{i}}$ in $\mathcal{Y}$ and thresholds this latent count at one to generate the actual *binary*-valued entry $b_{\boldsymbol{i}}$ in the observed binary tensor, which we will denote by $\mathcal{B}$:

$$\begin{aligned} b_{\boldsymbol{i}} &= \mathbf{1}(y_{\boldsymbol{i}} \geq 1) && (2)\\ \mathcal{Y} &\sim \text{Pois}(\sum_{r=1}^{R} \lambda_r.\boldsymbol{u}_r^{(1)} \odot \ldots \odot \boldsymbol{u}_r^{(K)}) && (3)\\ \boldsymbol{u}_r^{(k)} &\sim \text{Dir}(a^{(k)}, \ldots, a^{(k)}) && (4)\\ \lambda_r &\sim \text{Gamma}(g_r, \frac{p_r}{1-p_r}) && (5)\\ p_r &\sim \text{Beta}(c\epsilon, c(1-\epsilon)) && (6) \end{aligned}$$

Marginalizing out $y_{\boldsymbol{i}}$ from Eq. 2 leads to the following (equivalent) likelihood model

$$b_{\boldsymbol{i}} \sim \text{Bernoulli}(1 - \exp(-\sum_{r=1}^{R} \lambda_r \prod_{k=1}^{K} u_{i_k r}^{(k)})) \tag{7}$$

Note that the thresholding in (2) looks similar to a probit model for binary data (which however thresholds a *normal* at *zero*); however, the probit model (just like the logistic model) also needs to evaluate the likelihood at the zeros, and can therefore be slow on massive binary data with lots of zeros. Likelihood models of the form (Eq. 7) have previously also been considered in work on statistical models of undirected networks (Morup et al., 2011; Zhou, 2015).

Interestingly, the form of the likelihood in 7 also resembles the complementary log-log function Collett (2002); Piegorsch (1992), which is known to be a better model for imbalanced binary data than the logistic or probit likelihood, making it ideal for handling sparse binary tensors.

The conditional posterior of the latent count $y_{\boldsymbol{i}}$ is given by

$$y_{\boldsymbol{i}}|b_{\boldsymbol{i}}, \boldsymbol{\lambda}, \{u^{(k)}_{i_k r}\}_{k=1}^K \sim b_{\boldsymbol{i}} \cdot \text{Pois}_+(\sum_{r=1}^R \lambda_r \prod_{k=1}^K u^{(k)}_{i_k r}) \quad (8)$$

where $\text{Pois}_+(\cdot)$ is zero truncated Poisson distribution. Eq. (8) suggests that if $b_{\boldsymbol{i}} = 0$, then $y_{\boldsymbol{i}} = 0$ almost surely with probability one, which can lead to significant computational savings, if the tensor has a large number of zeros. In addition, our model also enables leveraging a reparameterization (Section 3.2) of the Poisson distribution in terms of a multinomial, which allows us to obtain very simple Gibbs-sampling updates for the model parameters.

Note that the Dirichlet prior on the latent factors $\boldsymbol{u}_r^{(k)}$ naturally imposes non-negativity constraints (Chi and Kolda, 2012) on the factor matrices $\mathbf{U}^{(1)}, \ldots, \mathbf{U}^{(K)}$. Moreover, since the columns $\boldsymbol{u}_r^{(k)}$ of these factor matrices sums to 1, each $\boldsymbol{u}_r^{(k)}$ can also be interpreted as a *distribution* (e.g., a "topic") over the n_k entities in mode k. Furthermore, the gamma-beta hierarchical construction (Zhou et al., 2012) of λ_r (Eq 5 and 6) allows inferring the rank of the tensor by setting an upper bound R on the number of factors and inferring the appropriate number of factors by shrinking the coefficients λ_r's to close to zero for the irrelevant factors. These aspects make our model interpretable as well as provide it the ability to do model selection (i.e., inferring the rank), in addition to being computationally efficient by focusing the computations only on the nonzero entries in the tensor $\mathcal{B}$.

3.1 LEVERAGING MODE NETWORKS

Often, in addition to the binary tensor $\mathcal{B}$, *pairwise* relationships between entities in one or more tensor modes may be available in form of a symmetric binary network or an undirected graph. Leveraging such forms of side-information can be beneficial for tensor decomposition, especially if the amount of missing data in the main tensor $\mathcal{B}$ is very high (Acar et al., 2011; Beutel et al., 2014; Rai et al., 2015), and, even more importantly, in "cold-start" settings, where there is no data in the tensor for entities along some of the tensor mode(s), as shown in Fig 1. In the absence of any side-information, the posterior distribution of the latent factors $\boldsymbol{u}_r^{(k)}$ of such entities in that tensor mode would be the same as the prior (i.e., just a random draw). Leveraging the side-information (e.g., a network) helps avoid this.

For entities of the k-th mode of tensor $\mathcal{B}$, we assume a symmetric binary network $\mathbf{A}^{(k)} \in \{0,1\}^{n_k \times n_k}$, where $A^{(k)}_{i_k j_k}$

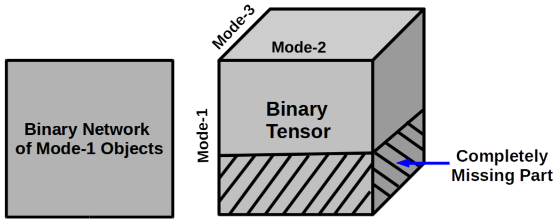

Figure 1: Binary tensor with an associated binary network between objects in mode-1 of the tensor (in general, network for other modes may also be available). In the "cold-start"setting as shown above, data along some of the tensor dimensions will be completely missing

denotes the relationship between mode-k entities i_k and j_k.

Just like our tensor decomposition model, we model the mode-k network $\mathbf{A}^{(k)}$ as a weighted sum of rank-1 symmetric matrices, with a similar likelihood model as we use for the tensor observations. In particular, we assume a latent count $X^{(k)}_{i_k j_k}$ for each binary entry $A^{(k)}_{i_k j_k}$, and threshold it at one to generate $A^{(k)}_{i_k j_k}$

$$A^{(k)}_{i_k j_k} = \mathbf{1}(X^{(k)}_{i_k j_k} \geq 1) \quad (9)$$

$$\mathbf{X}^{(k)} \sim \text{Pois}(\sum_{r=1}^R \beta_r . \boldsymbol{u}_r^{(k)} \odot \boldsymbol{u}_r^{(k)}) \quad (10)$$

$$\beta_r \sim \text{Gamma}(f_r, \frac{h_r}{1-h_r}) \quad (11)$$

$$h_r \sim \text{Beta}(d\alpha, d(1-\alpha)) \quad (12)$$

Note that since $\mathbf{A}^{(k)}$ is symmetric, only the upper (or lower) triangular portion needs to be considered, and moreover, just like in the case of the tensor $\mathcal{B}$, due to the truncated Poisson construction, the likelihood at only the nonzero entries needs to be evaluated for this part as well.

3.2 REPARAMETERIZED POISSON DRAWS

To simplify posterior inference (Section 4), we make use of two re-parameterizations of a Poisson draw (Zhou et al., 2012). The first parameterization is to express each latent count variable $y_{\boldsymbol{i}}$ and $X^{(k)}_{i_k j_k}$ as a sum of another set of R latent counts $\{\tilde{y}_{\boldsymbol{i}r}\}_{r=1}^R$ and $\{\tilde{X}^{(k)}_{i_k j_k r}\}_{r=1}^R$, respectively

$$y_{\boldsymbol{i}} = \sum_{r=1}^R \tilde{y}_{\boldsymbol{i}r}, \quad \tilde{y}_{\boldsymbol{i}r} \sim \text{Pois}(\lambda_r \prod_{k=1}^K u^{(k)}_{i_k r}) \quad (13)$$

$$X^{(k)}_{i_k j_k} = \sum_{r=1}^R \tilde{X}^{(k)}_{i_k j_k r}, \quad \tilde{X}^{(k)}_{i_k j_k r} \sim \text{Pois}(\beta_r u^{(k)}_{i_k r} u^{(k)}_{j_k r}) \quad (14)$$

The second parameterization assumes that the latent counts $\{\tilde{y}_{\boldsymbol{i}r}\}$ and $\tilde{X}^{(k)}_{i_k j_k r}$ are drawn from a multinomial

$$\tilde{y}_{\boldsymbol{i}1}, \ldots, \tilde{y}_{\boldsymbol{i}R} \sim \text{Mult}(y_{\boldsymbol{i}}; \zeta_{\boldsymbol{i}1}, \ldots, \zeta_{\boldsymbol{i}R})$$
$$\zeta_{\boldsymbol{i}r} = \frac{\lambda_r \prod_{k=1}^K u^{(k)}_{i_k r}}{\sum_{r=1}^R \lambda_r \prod_{k=1}^K u^{(k)}_{i_k r}} \quad (15)$$

$$\tilde{X}^{(k)}_{i_k j_k 1}, \dots, \tilde{X}^{(k)}_{i_k j_k R} \sim \text{Mult}(X^{(k)}_{i_k j_k}; \kappa^{(k)}_{i_k j_k 1}, \dots, \kappa^{(k)}_{i_k j_k R})$$
$$\kappa^{(k)}_{i_k j_k r} = \frac{\beta_r u^{(k)}_{i_k r} u^{(k)}_{j_k r}}{\sum_{r=1}^{R} \beta_r u^{(k)}_{i_k r} u^{(k)}_{j_k r}} \tag{16}$$

As we show in Section 4, these parameterizations enable us to exploit the gamma-Poisson as well as the Dirichlet-multinomial conjugacy to derive simple, closed-form Gibbs sampling updates for the model parameters.

4 INFERENCE

Exact inference in the model is intractable and we resort to Markov Chain Monte Carlo (MCMC) (Andrieu et al., 2003) inference. In particular, the reparameterization discussed in Section 3.2 allows us to derive simple Gibbs sampling updates for all the latent variables, except for the latent counts $y_{\boldsymbol{i}}$, which are drawn from a truncated Poisson distribution via rejection sampling. As discussed earlier, the computational-cost for our inference method scales linearly w.r.t. the number of ones in the tensor (plus the number of nonzeros in the network, if side-information is used). This makes our method an order of magnitude faster than models based on logistic or probit likelihood for binary data (Rai et al., 2014; Xu et al., 2013), without sacrificing on the quality of the results. The relative speed-up depends on the ratio of total volume of the tensor to the number of ones, which is given by $(\prod_{k=1}^{K} n_k)/\text{nnz}(\mathcal{B})$; here nnz$(\mathcal{B})$ denotes the number of nonzeros in the tensor.

In this section, we present both batch MCMC (Section 4.1) as well as an online MCMC (Section 4.2) method for inference in our model. The online MCMC algorithm is based on the idea of Bayesian Conditional Density Filtering (CDF) (Guhaniyogi et al., 2014), and can lead to further speed-ups over the batch MCMC if the number of nonzeros in the tensor is also massive. The CDF algorithm provides an efficient way to perform online MCMC sampling using surrogate conditional sufficient statistics (Guhaniyogi et al., 2014).

For both batch MCMC and CDF based online MCMC, we provide the update equations, with and without the side-information, i.e., the mode network(s). For what follows, we define four quantities: $s^{(k)}_{j,r} = \sum_{\boldsymbol{i}: i_k = j} \tilde{y}_{\boldsymbol{i}r}$, $s_r = \sum_{\boldsymbol{i}} \tilde{y}_{\boldsymbol{i},r}$, $v_{i_k,r} = \sum_{j_k}^{n_k} \tilde{X}^{(k)}_{i_k j_k r}$ and $v_r = \sum_{i_k}^{n_k} \sum_{j_k}^{n_k} \tilde{X}^{(k)}_{i_k j_k r}$, which denote aggregates computed using the latent counts $\tilde{y}_{\boldsymbol{i}r}$ and $\tilde{X}^{(k)}_{i_k j_k r}$. These quantities will be used at various places in the description of the inference algorithms that we present here.

4.1 BATCH MCMC INFERENCE

4.1.1 Tensor without Mode Network(s)

Sampling $y_{\boldsymbol{i}}$: For each observation $b_{\boldsymbol{i}}$ in the tensor, the latent count $y_{\boldsymbol{i}}$ is sampled as

$$y_{\boldsymbol{i}} \sim b_{\boldsymbol{i}} \cdot \text{Pois}_+(\sum_{r=1}^{R} \lambda_r \prod_{k=1}^{K} u^{(k)}_{i_k r}) \tag{17}$$

where Pois$_+(\cdot)$ is zero truncated Poisson distribution. Eq. (17) suggests that if $b_{\boldsymbol{i}} = 0$, then $y_{\boldsymbol{i}} = 0$ almost surely; and if $b_{\boldsymbol{i}} = 1$, then $y_{\boldsymbol{i}} \sim \text{Pois}_+(\sum_{r=1}^{R} \lambda_r \prod_{k=1}^{K} u^{(k)}_{i_k r})$. Therefore the $y_{\boldsymbol{i}}$'s only need to be sampled for the nonzero $b_{\boldsymbol{i}}$'s.

Sampling $\tilde{y}_{\boldsymbol{i}r}$: The latent counts $\{\tilde{y}_{\boldsymbol{i}r}\}$ are sampled from a multinomial as Eq. (15). Note that this also needs to be done only for the nonzero $b_{\boldsymbol{i}}$'s.

Sampling $\boldsymbol{u}^{(k)}_r$: The columns of each factor matrix have a Dirichlet posterior, and are sampled as

$$\boldsymbol{u}^{(k)}_r \sim \text{Dir}(a^{(k)} + s^{(k)}_{1,r}, a^{(k)} + s^{(k)}_{2,r}, \dots, a^{(k)} + s^{(k)}_{n_k,r}) \tag{18}$$

Sampling p_r: Using the fact that $s_r = \sum_{\boldsymbol{i}} \tilde{y}_{\boldsymbol{i},r}$ and marginalizing over the $u^{(k)}_{i_k r}$'s in (13), we have $s_r \sim \text{Pois}(\lambda_r)$. Using this, along with (5), we can express s_r using a negative binomial distribution, i.e., $s_r \sim \text{NB}(g_r, p_r)$. Due to the conjugacy between negative binomial and beta, we can then sample p_r as

$$p_r \sim \text{Beta}(c\epsilon + s_r, c(1-\epsilon) + g_r) \tag{19}$$

Sampling λ_r: Again using the fact that $s_r \sim \text{Pois}(\lambda_r)$ and (5), we have

$$\lambda_r \sim \text{Gamma}(g_r + s_r, p_r) \tag{20}$$

As can be observed, when updating $\boldsymbol{u}^{(k)}_r$, p_r and λ_r, the latent counts $y_{\boldsymbol{i}}$'s and $\tilde{y}_{\boldsymbol{i}r}$ corresponding to zero entries in $\mathcal{B}$ are all equal to zero, and have no contribution to sufficient statistics $s^{(k)}_{j,r}$ and s_r. Therefore, only the nonzero entries in tensor need to be considered in the computations.

4.1.2 Tensor with Mode Network(s)

In the presence of mode network(s), the update equations for the latent variables p_r, λ_r, $\tilde{y}_{\boldsymbol{i}r}$ and $y_{\boldsymbol{i}}$, that are associated solely with the binary tensor $\mathcal{B}$, remain unchanged, and can be sampled as described in Section 4.1.1. We however need to sample the additional latent variables associated with mode-k network $\mathbf{A}^{(k)}$, and the latent factors $\boldsymbol{u}^{(k)}_r$ of mode-k that are shared by the binary tensor $\mathcal{B}$ as well as the mode-k network.

Sampling $X^{(k)}_{i_k j_k}$: The latent counts $X^{(k)}_{i_k j_k}$ are sampled as

$$X^{(k)}_{i_k j_k} \sim A^{(k)}_{i_k j_k} \cdot \text{Pois}_+(\sum_{r=1}^{R} \beta_r u^{(k)}_{i_k r} u^{(k)}_{j_k r}) \tag{21}$$

This only needs to be done for the nonzero entries in $\mathbf{A}^{(k)}$.

Sampling $\tilde{X}_{i_k j_k r}$: The latent counts $\tilde{X}_{i_k j_k r}$ are sampled from a multinomial as equation (16). This also only needs to be done for the nonzero entries in $\mathbf{A}^{(k)}$.

Sampling $\boldsymbol{u}_r^{(k)}$: The columns of each factor matrix have a Dirichlet posterior, and are sampled as

$$\boldsymbol{u}_r^{(k)} \sim \text{Dir}(a^{(k)} + s_{1,r}^{(k)} + v_{1,r}, \dots, a^{(k)} + s_{n_k,r}^{(k)} + v_{n_k,r}) \tag{22}$$

Note that in the absence of the mode-k network, the terms $v_{\cdot,r}$ go away and Eq. 22 simply reduces to Eq. 18.

Sampling h_r: $h_r \sim \text{Beta}(d\alpha + v_r, d(1-\alpha) + f_r)$.

Sampling β_r: $\beta_r \sim \text{Gamma}(f_r + v_r, h_r)$.

4.1.3 Per-iteration time-complexity

For the binary tensor $\mathcal{B}$, computing each $\zeta_{\boldsymbol{i}r}$ (Eq. 15) takes $\mathcal{O}(K)$ time and therefore computing all the $\{\zeta_{\boldsymbol{i}r}\}$ takes $\mathcal{O}(\text{nnz}(\mathcal{B})RK)$ time. Likewise, for the binary mode-k network $\mathbf{A}^{(k)}$, computing all the $\{\kappa_{i_k j_k r}^{(k)}\}$ (Eq. 16) takes $\mathcal{O}(\text{nnz}(\mathbf{A}^{(k)})R)$ time. These are the most dominant computations in each iteration of our MCMC procedure; updating each $\boldsymbol{u}_r^{(k)}$ takes $\mathcal{O}(n_k)$ time and updating $\{p_r, h_r\}_{r=1}^R$ and $\{\lambda_r, \beta_r\}_{r=1}^R$ takes $\mathcal{O}(R)$ time each. Therefore, the per-iteration time-complexity of our batch MCMC method is $\mathcal{O}(\text{nnz}(\mathcal{B})RK + \text{nnz}(\mathbf{A}^{(k)})R)$. The linear dependence on $\text{nnz}(\mathcal{B}), \text{nnz}(\mathbf{A}^{(k)}), R$ and K suggests that even massive, sparse binary tensors and mode network(s) can be handled easily even by our simple batch MCMC implementation. Also note that our model scales linearly even w.r.t. R, unlike most other methods (Ermis and Bouchard, 2014; Rai et al., 2014) that have *quadratic* dependence on R.

The above computations can be further accelerated using a distributed/multi-core setting; we leave this for future work. In Section 4.2, however, we present an *online* MCMC method based on the idea of Bayesian Conditional Density Filtering (Guhaniyogi et al., 2014), which leads to further speed-ups, even in single-machine settings.

4.2 ONLINE MCMC INFERENCE

We develop an efficient online MCMC sampler for the model, leveraging ideas from the Conditional Density Filtering (CDF) Guhaniyogi et al. (2014). The CDF algorithm for our model selects a minibatch of the tensor (and mode network, if the side-information is available) entries at each iteration, samples the model parameters from the posterior, and updates the sufficient statistics $s_{j,r}^{(k)}$, s_r, $v_{i_k,r}$ and v_r using the data from the current minibatch.

4.2.1 Tensor without Mode Network(s)

We first provide the update equations for the case when there is no side-information (mode network). Denote I_t as indices of entries of tensor $\mathcal{B}$ from the minibatch selected at iteration t. The CDF algorithm at iteration t proceeds as:

Sampling $y_{\boldsymbol{i}}$: For all $\boldsymbol{i} \in I_t$, sample $y_{\boldsymbol{i}}$ according to equation (17); like in the batch MCMC case, the sampling only needs to be done for the nonzero $b_{\boldsymbol{i}}$'s.

Sampling $\tilde{y}_{\boldsymbol{i}r}$: For all $\boldsymbol{i} \in I_t$, sample the latent counts $\tilde{y}_{\boldsymbol{i}r(\boldsymbol{i}\in I_t)}$ using (15), again only for the nonzero $b_{\boldsymbol{i}}$'s.

Updating the conditional sufficient statistics: Update the conditional sufficient statistics $s_{j,r}^{(k)}$ as $s_{j,r}^{(k,t)} = s_{j,r}^{(k,t-1)} + \sum_{\boldsymbol{i}\in I_t: i_k=j} \tilde{y}_{\boldsymbol{i}r}$ and update s_r as $s_r^{(t)} = s_r^{(t-1)} + \sum_{\boldsymbol{i}\in I_t} \tilde{y}_{\boldsymbol{i},r}$. These updates basically add to the old sufficient statistics, the contributions from the data in the current minibatch. In practice, we also *reweight* these sufficient statistics by the ratio of the total number of ones in $\mathcal{B}$ and the minibatch size, so that they represent the average statistics over the entire tensor. This reweighting is akin to the way average gradients are computed in stochastic variational inference methods (Hoffman et al., 2013).

Updating $\boldsymbol{u}_r^{(k)}, p_r, \lambda_r$: Using the following conditionals, draw M samples $\{\boldsymbol{u}_r^{(k,m)}, p_r^{(m)}, \lambda_r^{(m)}\}_{m=1}^M$

$$\begin{aligned}
\boldsymbol{u}_r^{(k)} &\sim \text{Dir}(a^{(k)} + s_{1,r}^{(k,t)}, \dots, a^{(k)} + s_{n_k,r}^{(k,t)}) &(23)\\
p_r &\sim \text{Beta}(c\epsilon + s_r^{(t)}, c(1-\epsilon) + g_r) &(24)\\
\lambda_r &\sim \text{Gamma}(g_r + s_r^{(t)}, p_r) &(25)
\end{aligned}$$

and either store the sample averages of $\boldsymbol{u}_r^{(k)}, p_r$, and λ_r, or their analytic means to use for the next CDF iteration (Guhaniyogi et al., 2014).

4.2.2 Tensor with Mode Network(s)

For all the latent variables associated solely with the tensor $\mathcal{B}$, the sampling equations for the CDF algorithm in the presence of mode network(s) remain unchanged as the previous case with no network. In the presence of the mode network, the additional latent variables include the sufficient statistics $v_{i_k,r}$ and v_r, and these need to be updated in each CDF iteration.

Denote J_t as indices of entries selected from the mode-k network $\mathbf{A}^{(k)}$ in iteration t. The update equations for the latent variables that depend on $\mathbf{A}^{(k)}$ are as follows:

Sampling $X_{i_k j_k}$: For $(i_k, j_k) \in J_t$, latent count $X_{i_k j_k}$ is sampled using Eq. (21).

Sampling $\tilde{X}_{i_k j_k r}$: For $(i_k, j_k) \in J_t$, latent counts $\tilde{X}_{i_k j_k r}$ are sampled from a multinomial using Eq. (16).

Updating the conditional sufficient statistics: Update the sufficient statistics associated with the mode-k network as $v_{i_k,r}^{(t)} = v_{i_k,r}^{(t-1)} + \sum_{j_k,(i_k,j_k)\in J_t}^{n_k} \tilde{X}_{i_k j_k r}$ and $v_r^{(t)} = v_r^{(t-1)} + \sum_{i_k}^{n_k} \sum_{j_k,(i_k,j_k)\in J_t}^{n_k} \tilde{X}_{i_k j_k r}$. Just like the way we update the tensor sufficient statistics $s_{j,r}^{(k)}$ and s_r, we reweight these mode-k sufficient statistics by the ratio of the total number of ones in $\mathbf{A}^{(k)}$ and the minibatch size, so that they represent the average statistics over the entire mode-k network.

Updating $\boldsymbol{u}_r^{(k)}$, h_r, β_r: Using the following conditionals, draw M samples $\{\boldsymbol{u}_r^{(k,m)}, h_r^{(m)}, \beta_r^{(m)}\}_{m=1}^{M}$. We draw $\boldsymbol{u}_r^{(k)} \sim \text{Dir}(a^{(k)} + s_{1,r}^{(k,t)} + v_{1,r}^{(t)}, \ldots, a^{(k)} + s_{n_k,r}^{(k,t)} + v_{n_k,r}^{(t)})$, and h_r and β_r as

$$\begin{aligned} h_r &\sim \text{Beta}(d\alpha + v_r^{(t)}, d(1-\alpha) + f_r) \\ \beta_r &\sim \text{Gamma}(f_r + v_r^{(t)}, h_r) \end{aligned} \tag{26}$$

and either store the sample averages of $\boldsymbol{u}_r^{(k)}$, h_r, β_r, or their analytic means to use for the next CDF iteration.

4.2.3 Per-iteration time-complexity

The per-iteration time-complexity of the CDF based online MCMC is linear in the number of nonzeros in each minibatch (as opposed to the batch MCMC where it depends on the number of nonzeros in the *entire* tensor and network). Therefore the online MCMC is attractive for *dense* binary data, where the number of nonzeros in the tensor/network is also massive; using a big-enough minibatch size (that fits in the main memory and/or can be processed in each iteration in a reasonable amount of time), the online MCMC inference allows applying our model on such dense binary data as well, which may potentially have several billions of nonzero entries.

5 RELATED WORK

With the increasing prevalence of structured databases, social networks, and (multi)relational data, tensor decomposition methods are becoming increasingly popular for extracting knowledge and doing predictive analytics on such data (Bordes et al., 2011; Nickel et al., 2012; Kang et al., 2012). As the size of these data sets continues to grow, there has been a pressing need to design tensor factorization methods that can scale to massive tensor data.

For low-rank factorization of *binary* tensors, methods based on logistic and probit likelihood for the binary data have been proposed (Jenatton et al., 2012; London et al., 2013; Rai et al., 2014; Xu et al., 2013). However, these methods are not suited for massive binary tensors where the number of observations (which mostly consist of zeros, if the tensor is also sparse) could easily be millions or even billions (Inah et al., 2015). As a heuristic, these methods rely on subsampling (Rai et al., 2014) or partitioning the tensor (Zhe et al., 2015), to select a manageable number entries before performing the tensor decomposition, or alternatively going for a distributed setting (Zhe et al., 2013).

In the context of tensor factorization, to the best of our knowledge, the only method (and one that is closest in spirit to our work) that scales linearly w.r.t. the number of ones in the tensor is (Ermis and Bouchard, 2014). Their work explored quadratic loss (and its variations) as a surrogate to the logistic loss and proposed a method (`Quad-Approx`) with a per-iteration complexity $\mathcal{O}(\text{nnz}(\mathcal{B})R + R^2 \sum_{k=1}^{K} n_k)$. Note that its dependence on R is quadratic as opposed to our method which is also linear in R. They also proposed variations based on piecewise quadratic approximations; however, as reported in their experiments (Ermis and Bouchard, 2014), these variations were found to be about twice as slow than their basic `Quad-Approx` method (Ermis and Bouchard, 2014). Moreover, their methods (and the various other methods discussed in this section) have several other key differences from our proposed model: (1) our model naturally imposes non-negativity on the factor matrices; (2) R can be inferred from data; (3) our method provides a fully Bayesian treatment; (4) in contrast to their method, which operates in a batch setting, the online MCMC inference allows our model to scale to even bigger problems, where the number of nonzeros could also be massive; and (5) our model also allows incorporating (fully or partially observed) mode-networks as a rich source of side-information.

In another recent work (Zhou, 2015), a similar zero-truncated Poisson construction, as ours, was proposed for *edge-partioning* based network clustering, allowing the proposed model to scale in terms of the number of edges in the network. Our model, on the other hand, is more general and can be applied to multiway binary tensor data, with an optionally available binary network as a potential source of side-information. Moreover, the Dirichlet prior on the factor matrices, its reparametrizations (Section 3.2), and the online MCMC inference lead to a highly scalable framework for tensor decomposition with side-information.

Another line of work on scaling up tensor factorization methods involves developing distributed and parallel methods (Kang et al., 2012; Inah et al., 2015; Papalexakis et al., 2012; Beutel et al., 2014). Most of these methods, however, have one or more of the following limitations: (1) these methods lack a proper generative model of the data, which is simply assumed to be real-valued and the optimization objective is based on minimizing the Frobenius norm of the tensor reconstruction error, which may not be suitable for binary data; (2) these methods usually assume a parallel or distributed setting, and therefore are not feasible to run on a single machine; (3) missing data cannot be easily handled/predicted; and (4) the rank of the decomposition needs to be chosen via cross-validation.

Leveraging sources of side-information for tensor factorization has also been gaining a lot of attention recently. However, most of these methods cannot scale easily to massive tensors (Acar et al., 2011; Rai et al., 2015), or have to rely on parallel or distributed computing infrastructures (Beutel et al., 2014). In contrast, our model, by the virtue of its scalability that only depends on the number of nonzero entries in the tensor and/or the mode network, easily allows it to scale to massive binary tensors, with or without mode-network based side-information.

6 EXPERIMENTS

We report experimental results for our model on a wide range of real-world binary tensors (with and without mode-network based side-information), and compare it with several baselines for binary tensor factorization. We use the following data sets for our experiments:

- **Kinship:** This is a binary tensor of size $104 \times 104 \times 26$, representing 26 types of relationships between 104 members of a tribe (Nickel et al., 2011). The tensor has about 3.8% nonzeros.
- **UMLS:** This is a binary tensor of size $135 \times 135 \times 49$ representing 56 types of verb relations between 135 high-level concepts (Nickel et al., 2011). The tensor has about 0.8% nonzeros.
- **Movielens:** This is a binary *matrix* (two-way tensor) of size 943×1682 representing the binary ratings (thumbs-up or thumbs-down) by 943 users on 1682 movies [1]. This data set has a total of 100,000 ones.
- **DBLP:** This is a binary tensor of size $10,000 \times 200 \times 10,000$ representing (author-conference-keyword) relations (Zhe et al., 2015). This tensor has only about 0.001% nonzeros, and is an ideal example of a massive but sparse binary tensor.
- **Scholars:** This is a binary tensor of size $2370 \times 8663 \times 4066$, constructed from a database of research paper abstracts published by researchers at Duke University; the three tensor modes correspond to authors, words, and publication venues, respectively. Just like the DBLP data, this tensor is also massive but extremely sparse with only about 0.002% nonzeros. In addition, the co-authorship network (i.e., who has written papers with whom) is also available, which we use as a source of side-information, and use this network to experiment with the *cold-start* setting (i.e., when the main tensor has no information about some authors).
- **Facebook:** The Facebook data is a binary tensor of size $63731 \times 63730 \times 1837$ with the three modes representing wall-owner, poster, and days (Papalexakis et al., 2013). This tensor has only 737498 nonzeros. In addition to the binary tensor, the social network (friendship-links) between users is also given in form of a symmetric binary matrix of size 63731×63731, which has 1634180 nonzeros. We use the network to experiment with the cold-start setting.

We use all the 6 data sets for the tensor completion experiments (Section 6.1). We also use the Scholars and Faceboook data in the cold-start setting, where we experiment on the tensor completion task, leveraging the mode-network based side-information (Section 6.4).

[1]`http://grouplens.org/datasets/movielens/`

The set of experiments we perform includes: (1) binary tensor completion (Section 6.1) using only the tensor data; (2) scalability behavior of our model (both batch as well as online MCMC) in terms of tensor completion accuracy vs run-time (Section 6.2); we compare our model with Bayesian CP based on logistic-likelihood (Rai et al., 2014); (3) a qualitative analysis of our results using a *multiway* topic modeling experiment (Section 6.3) on the Scholars data, with the entities being authors, words, and publication venues; and (4) leveraging the mode network for tensor completion in the cold-start setting (Section 6.4); for this experiment, we also demonstrate how leveraging the network leads to improved qualitative results in the multi-way topic modeling problem.

In the experiments, we refer to our model as **ZTP-CP** (Zero-Truncated Poisson based CP decomposition). We compare **ZTP-CP** (using both batch MCMC as well as online MCMC inference) with the following baselines: (1) the quadratic loss minimization (**Quad-App**) proposed in (Ermis and Bouchard, 2014); (2) the refined piecewise quadratic approximation algorithm (**PW-QuadApp**) (Ermis and Bouchard, 2014); and (3) **Bayesian CP** decomposition based on logistic likelihood for binary data (Rai et al., 2014).

Experimental settings: All experiments are done on a standard desktop computer with Intel i7 3.4GHz processor and 24GB RAM. Unless specified otherwise, the MCMC inference was run for 1000 iterations with 500 burn-in iterations. The online MCMC algorithm was also run for the same number of iterations, with minibatch size equal to one-tenth of the number of nonzeros in the training data. For all the data sets, except Scholars and Facebook, we use $R = 20$ (also note that our model has the ability to prune the unnecessary factors by shrinking the corresponding λ_r to zero). For Scholars and Facebook data, we set $R = 100$. The hyperparameters g_r, f_r were set to 0.1, and ϵ and α are set to $1/R$, which worked well in our experiments.

6.1 TENSOR COMPLETION

In Table 1, we report the results on the tensor completion task (in terms of the AUC-ROC - the area under the ROC curve). For this experiment, although available, we do not use the mode network for the Scholars and the Facebook data; only the binary tensor is used (the results when also using the network are reported in Section 6.4). For each data set, we randomly select 90% of the tensor observations as the training data and evaluate each model on the remaining 10% observations used as the held-out data.

Since the code for **Quad-App** and **PW-QuadApp** baselines (both proposed in (Ermis and Bouchard, 2014)) is not publicly available, we are only able to report the results for the Kinship, UMLS, and MovieLens data set (using the results reported in (Ermis and Bouchard, 2014)). For Bayesian CP (Rai et al., 2014), we use the code provided

Table 1: Tensor completion accuracies in terms of AUC-ROC scores. Results are averaged over 10 splits of training and test data. Note: (1) Bayesian CP was infeasible to run on the Scholars and Facebook data; (2) Due to the lack of publicly available code for **Quad-App** and **PQ-QuadApp**, we only report its results on Kinship, UMLS, and MovieLens data (results taken from (Ermis and Bouchard, 2014)).

	Kinship	**UMLS**	**Movielens**	**DBLP**	**Scholars**	**Facebook**
Quad-App (Ermis and Bouchard, 2014)	0.8193	0.8205	0.8511	-	-	-
PW-QuadApp (Ermis and Bouchard, 2014)	0.9213	0.9387	0.9490	-	-	-
Bayesian-Logistic-CP (Rai et al., 2014)	**0.9865**	**0.9965**	0.9799	0.9307	-	-
ZTP-CP (Batch MCMC)	0.9674	0.9938	**0.9895**	**0.9759**	**0.9959**	0.9830
ZTP-CP (Online MCMC)	0.9628	0.9936	0.9841	0.9743	0.9958	**0.9844**

by the authors. Moreover, the Bayesian CP baseline was found infeasible to run on the Scholars and Facebook data (both of which are massive tensors), so we are unable to report those results. For fairness, on Kinship, UMLS, and MovieLens data, we use the same experimental settings for all the methods as used by (Ermis and Bouchard, 2014).

As shown in Table 1, our model outperforms **Quad-App** and **PW-QuadApp** in terms of the tensor-completion accuracies, and performs comparably or better than **Bayesian CP**, while being an order of magnitude faster (Section 6.2 shows the results on running times).

6.2 SCALABILITY

We next compare our model with Bayesian CP (Rai et al., 2014) in terms of the running times vs tensor completion accuracy on Kinship and UMLS data sets. As shown in Fig. 2 (top-row), our model (batch as well as online MCMC) runs/converges an order of magnitude faster than Bayesian CP in terms of running time. On Scholars and Facebook, since Bayesian CP was infeasible to run, we are only able to show the results (Fig. 2, bottom-row) for our model, with batch MCMC and online MCMC inference. On all the data sets, the online MCMC runs/converges faster than the batch MCMC.

We would like to note that, although the model proposed in (Ermis and Bouchard, 2014) also scales linearly [2] in the number of ones in the tensor, the per-iteration time-complexity of our model, which is linear in *both* $\mathrm{nnz}(\mathcal{B})$ as well as rank R, is better than the model proposed in (Ermis and Bouchard, 2014) (which has *quadratic* dependence on R). Moreover, the tensor completion results of our model (shown in Table 1) on these data sets are better than the ones reported in (Ermis and Bouchard, 2014).

6.3 MULTIWAY TOPIC MODELING

We also apply our model for a *multiway* topic modeling task on the Scholars data. The binary tensor represents AUTHORS $\times$ WORDS $\times$ VENUES relationships. We apply our model (with batch MCMC) and examine the latent factors of each of the three dimensions. Since each factor is drawn from a Dirichlet, it is non-negative and naturally corresponds to a "topic". In Table 2, after examining the words factor matrix, we show the top-10 words for four of the factors (topics) inferred by our model; these factors seem to represent topics Evolutionary Biology, Medical Imaging, Machine Learning/Signal Processing, and Oncology. For the Machine Learning/Signal Processing topic, we also examine the corresponding topic in the venues factor matrix and show the top-10 venues in that topic (based on their factor scores in that factor). In Fig. 3, we also show the histograms of authors' department affiliations for each of the four topics and the results make intuitive sense. The results in Table 2 and Fig. 3 demonstrate the usefulness of our model for scalable topic modeling of such multiway data.

[2]Although (Ermis and Bouchard, 2014) reported run times on Kinship and UMLS data sets, those number are not directly comparable with our run times reported here (due to possibly different machine configuration, which they do not specify in the paper).

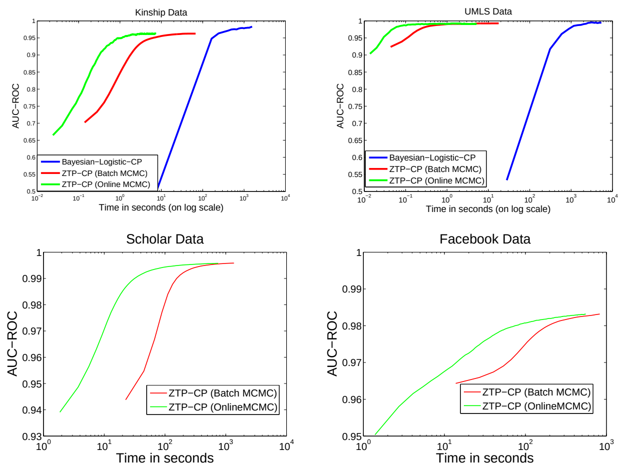

Figure 2: Running time (log-scale) comparison of various methods on Kinship (top left), UMLS (top right), Scholars (bottom left), and Facebook (bottom right) datasets.

6.4 LEVERAGING THE MODE NETWORK

Finally, to investigate the usefulness of leveraging the mode network, we experiment with using both the tensor *and* the mode network on Scholars and Facebook data sets. For each data set, we report the AUC-ROC (area under the ROC curve) and AUC-PR (area under the precision-recall curve) on the tensor completion task, with and without network. For both data sets, we experiment with the more challenging cold-start setting. In particular, for the Facebook data, we hold out all the entries of the tensor slices after the first 50,000 wall-owners and predict those entries(using only the rest of the tensor, and using the rest of the tensor as well as the friendship network). We run the experiment with $R = 20$ and minibatch size of 50,000 for the online

Table 2: For the Scholars data, the most probable words in topics related to evolutionary biology (Evo Bio), medical imaging (Med Imag), machine learning/signal processing(ML/SP) and oncology, and top ranked venues in ML/SP

Evo Bio	Med Imag	ML/SP	Oncology	Top Venues in ML/SP
species	imaging	bayesian	radiation	ICASSP
selection	contrast	algorithm	radiotherapy	JASA
genetic	computed	sampling	stage	ICML
evolution	resonance	features	tumor	IEEE trans img proc
populations	dose	process	survival	NIPS
evolutionary	tomography	sparse	lung	Compu stat data analy
gene	magnetic	nonparametric	chemotherapy	Biometrics
variation	image	gibbs	treated	Bayesian analysis
plants	quality	parameters	toxicity	JMLR
natural	diagnostic	inference	oncology	IEEE trans. inf. theory

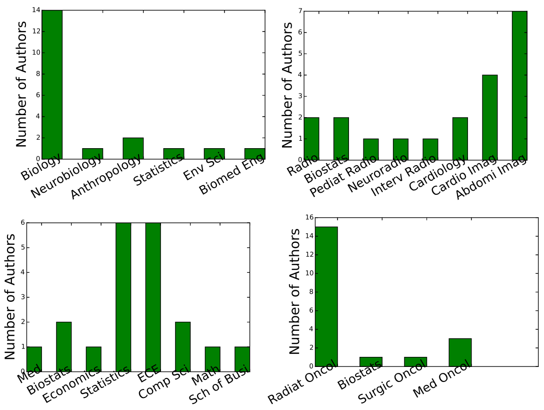

Figure 3: Histogram of the department-affiliations for the top 20 authors in factors related to evolutionary biology (top left), medical imaging (top right), machine learning/signal processing(bottom left) and oncology (bottom right).

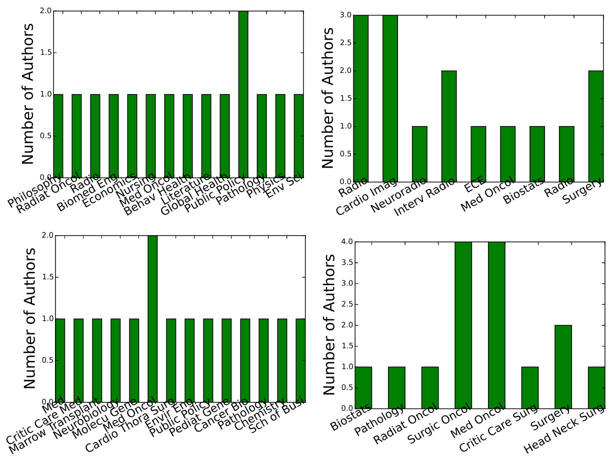

Figure 4: Histogram of the department-affiliations of the top 15 *held-out* authors associated with the factors of medical imaging (top) and oncology (bottom). The left column is obtained using no co-authorship information, and the right column is obtained using co-authorship information.

MCMC. The results in Table 3 show that using the network leads to better tensor completion accuracies.

We also perform a similar experiment on the Scholars data where we hold out all the entries in tensor slices after the first 1000 authors and predict those entries (using only the rest of the tensor, and using the rest of the tensor as well as the co-authorship network). We run the experiment with $R = 100$ and minibatch size of 50,000 for the online MCMC. The results shown in Table 3 again demonstrate the benefit of using the network.

Table 3: Cold-start setting

	Facebook		Scholars	
	AUC-ROC	**AUC-PR**	**AUC-ROC**	**AUC-PR**
Without network	0.8897	0.6076	0.8051	0.5763
With network	**0.9075**	**0.7255**	**0.8124**	**0.6450**

In Fig. 4, we show another result demonstrating the benefit of using the co-authorship network for the Scholars data. Note that in the cold-start setting, there is no information in the tensor for the *held-out* authors. Therefore the topics associated with such authors are expected to be roughly *uniformly random*. As shown in Fig. 4 (left column), the set of held-out authors assigned to the topics medical imaging and oncology seem very random and *arbitrary* (we only show the aggregate department-affiliations). Using side-information (in form of the co-authorship network), however, the model sensibly assigns authors who are indeed related to these topics, as shown in right column of Fig. 4.

7 CONCLUSION

We have presented a scalable Bayesian model for binary tensor factorization. In contrast to the models based on probit or logistic likelihood for binary tensor decomposition, the time-complexity of our model depends only in the number of ones in the tensor. This aspect of our model allows it to easily scale up to massive binary tensors. The simplicity of our model also leads to simple batch as well as online MCMC inference; the latter allows our model to scale up even when the number of ones could be massive. Our experimental results demonstrate that the model leads to speed-ups of an order of magnitude when compared to binary tensor factorization models based on the logistic likelihood, and also outperforms various other baselines. Our model also gives interpretable results which helps qualitative analysis of results. In addition, the ability to leverage mode networks (fully or partially observed) leads to improved tensor decomposition in cold-start problems.

Acknowledgments

The research reported here was supported in part by ARO, DARPA, DOE, NGA and ONR.

References

Acar, E., Kolda, T. G., and Dunlavy, D. M. (2011). All-at-once optimization for coupled matrix and tensor factorizations. *arXiv preprint arXiv:1105.3422*.

Andrieu, C., De Freitas, N., Doucet, A., and Jordan, M. I. (2003). An introduction to mcmc for machine learning. *Machine learning*, 50(1-2):5–43.

Beutel, A., Kumar, A., Papalexakis, E. E., Talukdar, P. P., Faloutsos, C., and Xing, E. P. (2014). Flexifact: Scalable flexible factorization of coupled tensors on hadoop. In *SDM*.

Bordes, A., Weston, J., Collobert, R., and Bengio, Y. (2011). Learning structured embeddings of knowledge bases. In *AAAI*.

Chi, E. C. and Kolda, T. G. (2012). On tensors, sparsity, and nonnegative factorizations. *SIAM Journal on Matrix Analysis and Applications*, 33(4):1272–1299.

Collett, D. (2002). *Modelling binary data*. CRC press.

Ermis, B. and Bouchard, G. (2014). Iterative splits of quadratic bounds for scalable binary tensor factorization. In *UAI*.

Guhaniyogi, R., Qamar, S., and Dunson, D. B. (2014). Bayesian conditional density filtering. *arXiv preprint arXiv:1401.3632*.

Hidasi, B. and Tikk, D. (2012). Fast ALS-based tensor factorization for context-aware recommendation from implicit feedback. In *ECML PKDD*.

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

Inah, J., Papalexakis, E., Kang, U., and Faloutsos, C. (2015). Haten2: Billion-scale tensor decompositions. In *ICDE*. IEEE.

Jenatton, R., Le Roux, N., Bordes, A., and Obozinski, G. (2012). A latent factor model for highly multi-relational data. In *NIPS*.

Kang, U., Papalexakis, E., Harpale, A., and Faloutsos, C. (2012). Gigatensor: scaling tensor analysis up by 100 times-algorithms and discoveries. In *KDD*.

Kolda, T. G. and Bader, B. W. (2009). Tensor decompositions and applications. *SIAM review*, 51(3):455–500.

London, B., Rekatsinas, T., Huang, B., and Getoor, L. (2013). Multi-relational learning using weighted tensor decomposition with modular loss. *arXiv preprint arXiv:1303.1733*.

Morup, M., Schmidt, M. N., and Hansen, L. K. (2011). Infinite multiple membership relational modeling for complex networks. In *MLSP*. IEEE.

Nickel, M., Tresp, V., and Kriegel, H. (2011). A three-way model for collective learning on multi-relational data. In *ICML*.

Nickel, M., Tresp, V., and Kriegel, H.-P. (2012). Factorizing YAGO: scalable machine learning for linked data. In *WWW*.

Papalexakis, E., Faloutsos, C., and Sidiropoulos, N. (2012). Parcube: Sparse parallelizable tensor decompositions. In *Machine Learning and Knowledge Discovery in Databases*, pages 521–536. Springer.

Papalexakis, E. E., Mitchell, T. M., Sidiropoulos, N. D., Faloutsos, C., Talukdar, P. P., and Murphy, B. (2013). Scoup-smt: Scalable coupled sparse matrix-tensor factorization. *arXiv preprint arXiv:1302.7043*.

Piegorsch, W. W. (1992). Complementary log regression for generalized linear models. *The American Statistician*.

Rai, P., Wang, Y., and Carin, L. (2015). Leveraging features and networks for probabilistic tensor decomposition. In *AAAI*.

Rai, P., Wang, Y., Guo, S., Chen, G., Dunson, D., and Carin, L. (2014). Scalable Bayesian low-rank decomposition of incomplete multiway tensors. In *ICML*.

Xu, Z., Yan, F., and Qi, Y. (2013). Bayesian nonparametric models for multiway data analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Zhe, S., Qi, Y., Park, Y., Molloy, I., and Chari, S. (2013). Dintucker: Scaling up gaussian process models on multidimensional arrays with billions of elements. *arXiv preprint arXiv:1311.2663*.

Zhe, S., Xu, Z., Chu, X., Qi, Y., and Park, Y. (2015). Scalable nonparametric multiway data analysis. In *AISTATS*.

Zhou, M. (2015). Infinite edge partition models for overlapping community detection and link prediction. In *AISTATS*.

Zhou, M., Hannah, L. A., Dunson, D., and Carin, L. (2012). Beta-negative binomial process and poisson factor analysis. In *AISTATS*.