LDA Revisited: Entropy, Prior and Convergence

Jianwei Zhang[1], Jia Zeng[1,2,3,*], Mingxuan Yuan[3,*], Weixiong Rao[4,*] and Jianfeng Yan[1,2,*]
[1]School of Computer Science and Technology, Soochow University, Suzhou 215006, China
[2]Collaborative Innovation Center of Novel Software Technology and Industrialization
[3]Huawei Noah's Ark Lab, Hong Kong
[4]School of Software Engineering, Tongji University, China
[*]Corresponding Authors: zeng.jia@acm.org, yuan.mingxuan@huawei.com, wxrao@tongji.edu.cn, yanjf@suda.edu.cn

ABSTRACT

Inference algorithms of latent Dirichlet allocation (LDA), either for small or big data, can be broadly categorized into expectation-maximization (EM), variational Bayes (VB) and collapsed Gibbs sampling (GS). Looking for a unified understanding of these different inference algorithms is currently an important open problem. In this paper, we revisit these three algorithms from the entropy perspective, and show that EM can achieve the best predictive perplexity (a standard performance metric for LDA accuracy) by minimizing directly the cross entropy between the observed word distribution and LDA's predictive distribution. Moreover, EM can change the entropy of LDA's predictive distribution through tuning priors of LDA, such as the Dirichlet hyperparameters and the number of topics, to minimize the cross entropy with the observed word distribution. Finally, we propose the adaptive EM (AEM) algorithm that converges faster and more accurate than the current state-of-the-art SparseLDA [20] and AliasLDA [12] from small to big data and LDA models. The core idea is that the number of active topics, measured by the residuals between E-steps at successive iterations, decreases significantly, leading to the amortized $\mathcal{O}(1)$ time complexity in terms of the number of topics. The open source code of AEM is available at GitHub.

Keywords

Latent Dirichlet allocation; entropy; adaptive EM algorithms; big data; prior; convergence

1. INTRODUCTION

Latent Dirichlet allocation (LDA) [4] is a three-layer hierarchical Bayesian model widely used for probabilistic topic modeling, computer vision and computational biology. The collections of documents can be represented as a document-word co-occurrence matrix, where each element is the number of word count in the specific document. Modeling each document as a mixture topics and each topic as a mixture of vocabulary words, LDA assigns thematic labels to explain non-zero elements in the document-word matrix, segmenting observed words into several thematic groups called topics. From the joint probability of latent labels and observed words, existing inference algorithms of LDA approximately infers the posterior probability of topic labels given observed words, and estimate multinomial parameters for document-topic distributions and topic-word distributions. In Bayesian view, LDA adds the Dirichlet prior constraints on its predecessor, probabilistic latent semantic analysis (PLSA) [10], and shows a better generalization ability for predicting the words in unseen corpus.

In the past decade, the batch/online/parallel inference algorithms of LDA for either small or big data mainly fall into three categories: 1) expectation-maximization (EM) [5], 2) variational Bayes (VB) [4], and 3) collapsed Gibbs sampling [8]. For example, EM for LDA has many variants such as batch EM [5, 3, 23], online EM [24], and parallel EM [13]. Similarly, VB also has batch [4], online [9] and parallel [25] variants. Among these inference algorithms, GS variants [8, 15, 20, 12, 1, 21, 19] have gained significantly more interests in both academia and industry because of its sampling efficiency (low space and time complexities). Unfortunately, there still lacks a unified understanding of these three types of inference schemes:

- Which algorithm can achieve the best predictive performance?
- What is the effect of prior information, such as the Dirichlet hyperparameters and the number of topics, on the predictive performance of these algorithms?
- Which algorithm converges the fastest to the local optimum of the LDA objective function?

Satisfactory answers to these questions may help researchers and engineers choose the proper inference algorithms for LDA or other probabilistic topic models in real-world applications [3, 16, 12].

In this paper, we address these questions within the entropy framework. The corpus provides the observed word distribution $\mathbf{x}$. LDA can use its multinomial parameters to reconstruct the predictive word distribution $\hat{\mathbf{x}}$. Inference algorithms aim to minimize the Kullback-Leibler (KL) divergence between $\mathbf{x}$ and $\hat{\mathbf{x}}$ by tuning LDA's multinomial parameters conditioned on the Dirichlet hyperparameters. First, we show that minimizing the KL divergence is equivalent to minimizing the cross entropy between $\mathbf{x}$ and $\hat{\mathbf{x}}$, which is the same with the definition of predictive perplexity [4, 3, 9, 23], a standard performance metric for different inference algorithms of LDA. Minimizing the cross entropy can be done directly by EM [5] rather than VB [4] and GS [8]. Therefore, EM can learn $\hat{\mathbf{x}}$ with better generalization ability on unseen corpus than both VB and GS. Second, as far as prior information is concerned [16], we show that tuning Dirichlet hyperparameters can minimize the entropy of $\hat{\mathbf{x}}$, which in turn can minimize the cross entropy for the best predictive performance. Nevertheless, when the number of training

CIKM'16 , October 24-28, 2016, Indianapolis, IN, USA

DOI: http://dx.doi.org/10.1145/2983323.2983794

Table 1: Definitions of Notation.

$1 \le d \le D$	Document index
$1 \le w \le W$	Word index in vocabulary
$1 \le k \le K$	Topic index
$\mathbf{x}_{W\times D} = \{x_{w,d}\}$	Observed word distribution
$\hat{\mathbf{x}}_{W\times D} = \{\hat{x}_{w,d}\}$	Predictive word distribution
NNZ	Number of nonzero elements
$N = \sum_{w,d} x_{w,d}$	Total number of word tokens
$\mathbf{z}_{W\times D} = \{z^k_{w,d}\}$	Topic labels for words
$\mathbf{r}_{K\times D} = \{r_{k,d}\}$	Residual matrix
$\boldsymbol{\theta}_{K\times D}$	Document-topic distribution
$\boldsymbol{\phi}_{K\times W}$	Topic-word distribution
$\boldsymbol{\mu}_{K\times NNZ}$	Responsibility matrix
α, β	Dirichlet hyperparameters

documents increases, the best hyperparameters approach to zeros. Under this situation, LDA reduces to PLSA without Dirichlet prior constraints. This makes sense according to the Bayesian theory, where the data-driven likelihood plays more important roles than prior information when data become big. Also, increasing the total number of topics K will decrease the entropy of $\hat{\mathbf{x}}$ for a lower cross entropy with $\mathbf{x}$. Finally, we propose the adaptive EM (AEM) algorithm for minimizing the cross entropy with fast convergence speed. The basic idea is that in each document AEM calculates a few number of active topics measured by residuals between E-steps at successive iterations. We find that the number of active topics in each document shrinks significantly after a few iterations, which is irrelevant to K provided by users. In this sense, AEM has an amortized $\mathcal{O}(1)$ time complexity in terms of K, similar to the current state-of-the-art AliasLDA [12] and LightLDA [21]. As a summary, we have the following contributions in this paper:

- We explain the major difference among three inference algorithms for LDA by the cross entropy. We show that EM can achieve the best predictive performance because it minimizes directly the cross entropy between the observed word distribution and the LDA's predictive word distribution.
- We discuss how prior information of LDA such as the Dirichlet hyperparameters and the number of topics K affect the entropy of LDA's predictive word distribution. When data (corpus) become bigger, the best Dirichlet hyperparameters become smaller, which makes LDA reduce to PLSA. Also, the larger K always leads to a better predictive performance on unseen data because it decreases steadily the entropy of LDA's predictive word distribution.
- We propose the AEM algorithm that converges significantly faster and more accurate than the state-of-the-art GS fast variants (SparseLDA [20] and AliasLDA [12]), VB fast variants (online VB [9]) and EM variants (fast EM [24]), as confirmed by extensive experiments from small to big datasets and LDA models with large K.

Section 2 reviews EM, VB and GS for LDA. Section 3 presents the entropy formulation of LDA's objective function, and discusses the influence of prior information on LDA's word prediction performance. We propose AEM based on residual learning with amortized $\mathcal{O}(1)$ time complexity in terms of K. Section 4 shows extensive experiments from small to big datasets and LDA models. Finally, Section 5 makes conclusions and envisions further work.

2. LDA INFERENCE ALGORITHMS

LDA allocates a set of thematic topic labels, $\mathbf{z} = \{z^k_{w,d}\}$, to explain nonzero elements in the document-word co-occurrence matrix $\mathbf{x}_{W\times D} = \{x_{w,d}\}$, where $1 \le w \le W$ denotes the word index in the vocabulary, $1 \le d \le D$ the document index in the corpus, and $1 \le k \le K$ the topic index. Usually, the number of topics K is a prior information provided by users. Some topic models like hierarchical Dirichlet process (HDP) [17] can learn automatically the suitable K from corpus, which is not considered in the context of this paper. The nonzero element $x_{w,d} \ne 0$ denotes the number of word counts at the index $\{w, d\}$. For each word token $x_{w,d,n} = \{0, 1\}, x_{w,d} = \sum_n x_{w,d,n}$, there is a topic label $z^k_{w,d,n} = \{0, 1\}, \sum_{k=1}^K z^k_{w,d,n} = 1, 1 \le n \le x_{w,d}$. Each nonzero element $x_{w,d} \ne 0$ is associated with a topic probability vector $\sum_k \mu_{w,d}(k) = 1$, which denotes the posterior probability of a topic label $z^k_{w,d} = 1$ given the observed word $\{w, d\}$. The objective of inference algorithms is to calculate the posterior probability from the full joint probability $p(\mathbf{x}, \mathbf{z}, \boldsymbol{\theta}, \boldsymbol{\phi}|\alpha, \beta)$ of LDA, where $\mathbf{z}$ is the topic labeling configuration, $\boldsymbol{\theta}_{K\times D}$ and $\boldsymbol{\phi}_{K\times W}$ are two non-negative matrices of multinomial parameters for document-topic and topic-word distributions, satisfying $\sum_k \theta_d(k) = 1$ and $\sum_w \phi_w(k) = 1$. Both multinomial matrices are generated by two Dirichlet distributions with hyperparameters α_k and β_w. Although learning asymmetric hyperparameters can lead to a better prediction performance [18], it needs complicated optimization steps. For simplicity, we consider the smoothed LDA with fixed symmetric hyperparameters [8], i.e., $\forall k\ \alpha_k = \alpha; \forall w\ \beta_w = \beta$. Table 1 summarizes the important notations in this paper.

E-step:

$$\mu_{w,d}(k) \propto \frac{[\hat{\theta}_d(k) + \alpha - 1][\hat{\phi}_w(k) + \beta - 1]}{\sum_w [\hat{\phi}_w(k) + \beta - 1]}, \quad (1)$$

M-step:

$$\hat{\theta}_d(k) = \sum_w x_{w,d}\mu_{w,d}(k), \quad (2)$$

$$\hat{\phi}_w(k) = \sum_d x_{w,d}\mu_{w,d}(k). \quad (3)$$

Hyperparameters:

$$\alpha^* = \alpha - 1, \quad \beta^* = \beta - 1. \quad (4)$$

Algorithm 1: E-step and M-step in EM.

2.1 EM, VB and GS

EM maybe the earliest *maximum a posteriori* (MAP) inference algorithm for LDA [5, 3]. It marginalizes out the latent topic variables $\mathbf{z}$ in the full joint probability of LDA $p(\mathbf{x}, \mathbf{z}, \boldsymbol{\theta}, \boldsymbol{\phi}|\alpha, \beta)$, and tunes the multinomial parameters $\{\boldsymbol{\theta}, \boldsymbol{\phi}\}$ conditioned on $\{\alpha, \beta\}$ to maximize the following posterior probability,

$$p(\boldsymbol{\theta}, \boldsymbol{\phi}|\mathbf{x}, \alpha, \beta) = \frac{p(\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}|\alpha, \beta)}{p(\mathbf{x}|\alpha, \beta)} \propto p(\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}|\alpha, \beta). \quad (5)$$

Maximizing this posterior finds the best parameters $\{\boldsymbol{\theta}^*, \boldsymbol{\phi}^*\}$ to explain the observed words $\mathbf{x}$, no matter what topic labeling configuration $\mathbf{z}$ is. EM is an iterative coordinate ascent algorithm composed of E-step and M-step in Algorithm 1. In the E-step, EM infers the responsibility $\mu_{w,d}(k)$ by fixing parameters $\{\hat{\theta}, \hat{\phi}\}$. In the M-step, EM updates parameters $\{\hat{\theta}, \hat{\phi}\}$ based on the inferred responsibility $\mu_{w,d}(k)$. Normalizing $\{\hat{\theta}, \hat{\phi}\}$ obtains the multinomial parameters $\{\boldsymbol{\theta}, \boldsymbol{\phi}\}$. For simplicity, we replace the hyperparameters by $\alpha^* = \alpha - 1$ and $\beta^* = \beta - 1$ as shown in Algorithm 1. When $\alpha^*, \beta^* \to 0$, the Dirichlet distribution becomes the uniform distribution without constraints on the generated multinomial variables. Under this condition, EM for LDA reduces to EM

Variational E-step:

$$\mu_{w,d}(k) \propto \frac{\exp[\Psi(\hat{\theta}_d(k)+\alpha)]\exp[\Psi(\hat{\phi}_w(k)+\beta)]}{\exp[\Psi(\sum_w[\hat{\phi}_w(k)+\beta])]}, \quad (6)$$

$$\hat{\theta}_d(k) = \sum_w x_{w,d}\mu_{w,d}(k). \quad (7)$$

Variational M-step:

$$\hat{\phi}_w(k) = \sum_d x_{w,d}\mu_{w,d}(k). \quad (8)$$

Hyperparameters:

$$\alpha^* = \alpha - 0.5, \quad \beta^* = \beta - 0.5. \quad (9)$$

Algorithm 2: Variational E-step and M-step in VB.

for PLSA [10]. According to [13, 24], EM for LDA has many variants with different names such as the collapsed variational Bayes having zero-order approximation (CVB0) [3] and the belief propagation (BP) [23, 22].

Unlike standard EM, VB [4] treats $\{\boldsymbol{\theta}, \mathbf{z}\}$ as latent variables, and tunes the multinomial parameter $\boldsymbol{\phi}$ to maximize the following posterior from the full joint probability,

$$p(\boldsymbol{\theta}, \mathbf{z}|\mathbf{x}, \boldsymbol{\phi}, \alpha, \beta) = \frac{p(\mathbf{x}, \mathbf{z}, \boldsymbol{\theta}, \boldsymbol{\phi}|\alpha, \beta)}{p(\mathbf{x}, \boldsymbol{\phi}|\alpha, \beta)}. \quad (10)$$

Maximizing this posterior optimizes the topic-word distribution $\boldsymbol{\phi}^*$ from training data to generate the best $\{\boldsymbol{\theta}, \mathbf{z}\}$ for unseen corpus, i.e., for the best generalization performance. Nevertheless, marginalizing out latent variables $\{\boldsymbol{\theta}, \mathbf{z}\}$ in full joint probability of LDA, $\int_{\boldsymbol{\theta},\mathbf{z}} p(\mathbf{x}, \mathbf{z}, \boldsymbol{\theta}, \boldsymbol{\phi}|\alpha, \beta)$, is intractable. As a result, VB maximizes an approximate variational posterior rather than (10) using the variational EM algorithm [14] in Algorithm 2. In the variational E-step, fixing the parameter $\hat{\phi}_w(k)$, we update $\mu_{w,d}(k)$ and $\hat{\theta}_d(k)$ until convergence, which makes the variational posterior approximate the true posterior $p(\boldsymbol{\theta}, \mathbf{z}|\mathbf{x}, \boldsymbol{\phi}, \alpha, \beta)$ by minimizing the KL divergence between them. In the variational M-step (8), we update $\hat{\phi}_w(k)$ to maximize the variational posterior having the same form with the M-step of EM (2). Normalizing $\{\hat{\theta}, \hat{\phi}\}$ yields the multinomial parameters $\{\boldsymbol{\theta}, \boldsymbol{\phi}\}$. According to [3], $\exp[\Psi(x)] \approx x - 0.5$ for $x > 1$, where Ψ is the digamma function. Since $\{\hat{\theta}, \hat{\phi}\}$ are unnormalized document-topic and topic-word distributions, we expect many elements to be greater than 1 when the number of topics K is small. So, the variational E-step (6) can be approximated by

$$\mu_{w,d}(k) \approx \frac{[\hat{\theta}_d(k)+\alpha-0.5][\hat{\phi}_w(k)+\beta-0.5]}{\sum_w[\hat{\phi}_w(k)+\beta]-0.5}. \quad (11)$$

Replacing $\alpha^* = \alpha - 0.5$ and $\beta^* = \beta - 0.5$ in (11) (shown in Algorithm 2) obtains a similar E-step with EM in (1) but with one main distinction: though the numerator in (11) is the same with that in (1), the denominator in (11) is always $0.5(W-1)$ constant smaller than that in (1). Moreover, when the number of topics $K \to \infty$, the accumulated $\{\hat{\theta}, \hat{\phi}\}$ become very sparse and many non-zero elements are close to zeros. In this case, the approximation $\exp[\Psi(x)] \approx x - 0.5$ for $x > 1$ cannot be made. Therefore, the conclusion made in [3] is incorrect that (11) can produce almost the same results with (1) by tuning hyperparameters properly. Our experimental results (in Section 4) also confirm that VB cannot yield comparable accuracy with EM by tuning only hyperparameters when K and W are large.

Finally, in contrast to EM and VB, GS [8] integrates out the multinomial parameters $\{\boldsymbol{\theta}, \boldsymbol{\phi}\}$ based on the Dirichlet conjugacy,

MCMC E-step:

$$\mu_{w,d,n}(k) \propto \frac{[\hat{\theta}_d^{-z^{k,old}_{w,d,n}}(k)+\alpha][\hat{\phi}_w^{-z^{k,old}_{w,d,n}}(k)+\beta]}{\sum_w[\hat{\phi}_w^{-z^{k,old}_{w,d,n}}(k)+\beta]}, \quad (12)$$

$$\text{Sampling } z^{k,new}_{w,d,n} = 1 \text{ from } \mu_{w,d,n}(k). \quad (13)$$

MCMC M-step:

$$\hat{\theta}_d(k) = \hat{\theta}_d^{-z^{k,old}_{w,d,n}}(k) + z^{k,new}_{w,d,n}, \quad (14)$$

$$\hat{\phi}_w(k) = \hat{\phi}_w^{-z^{k,old}_{w,d,n}}(k) + z^{k,new}_{w,d,n}. \quad (15)$$

Hyperparameters:

$$\alpha^* = \alpha, \quad \beta^* = \beta. \quad (16)$$

Algorithm 3: MCMC E-step and M-step in GS.

and tunes the latent topic variables $\mathbf{z}$ to maximize the following posterior from the full joint probability,

$$p(\mathbf{z}|\mathbf{x}, \alpha, \beta) = \frac{p(\mathbf{x}, \mathbf{z}|\alpha, \beta)}{p(\mathbf{x}|\alpha, \beta)} \propto p(\mathbf{x}, \mathbf{z}|\alpha, \beta). \quad (17)$$

Maximizing this posterior finds the best topic labeling configuration $\mathbf{z}^*$ to explain the observed word distribution $\mathbf{x}$. The total number of word tokens is $N = \sum_{w,d} x_{w,d}$. Because searching a total of K^N topic configurations for the best $\mathbf{z}^*$ is intractable, GS uses an approximate searching called Markov chain Monte Carlo (MCMC) EM [14] in Algorithm 3. In the MCMC E-step, GS computes per word token the topic posterior conditioned on the rest word topic labeling configuration, $\mu_{w,d,n}(k) = p(z^{k,new}_{w,d,n} = 1|\mathbf{z}^{k,old}_{w,d,-n}, \mathbf{x}, \alpha, \beta)$, and samples a new topic label $z^{k,new}_{w,d,n} = 1$ from this posterior. The notation $-z^{k,old}_{w,d,n}$ denotes subtracting the old topic label from the accumulated and unnormalized document-topic and topic-word matrices $\{\hat{\theta}, \hat{\phi}\}$. In the MCMC M-step, per word token, GS updates immediately $\{\hat{\theta}, \hat{\phi}\}$ by adding the new sampled topic label $z^{k,new}_{w,d,n} = 1$. In this sense, GS can be viewed as an incremental EM algorithm [13, 24] that estimates the best topic label $z^{k,new}_{w,d,n} = 1$ for each word token sequentially. As shown in Algorithm 3, the hyperparameters $\alpha^* = \alpha$ and $\beta^* = \beta$ for a fair comparison with both EM and VB. Thus, the MCMC E-step (12) is almost the same with (1) but with one additional sampling step for each word token $x_{w,d,n}$ rather than $x_{w,d}$. Since the sampling step determines the topic label for each word token, GS can be viewed as a *hard* version of EM.

2.2 Convergence for Big LDA Models

As reported by the Linguistic Data Consortium, there are millions of vocabulary words in English, Chinese, Spanish, and Arabic. Taking polysemy and synonyms into consideration, a rough estimate of the number of word senses is close to the same magnitude of vocabulary words—that is, around 10^5 or 10^6 topics for semantics of small correlated word sets. Extensive experiments on big search query data confirm that inferring at least $K = 10^5$ topics can achieve a significant improvement on industrial search engine and online advertising systems [19]. Therefore, fast algorithms with low time complexity in terms of K have attracted intensive research interests recently [15, 20, 12, 21, 24].

These fast algorithms use the fact that the accumulated matrices $\{\hat{\theta}_{K\times D}, \hat{\phi}_{W\times K}\}$ become very sparse when K is large. The time complexity of GS for one iteration is $\mathcal{O}(KN)$. FastLDA [15] introduces the upper bound on the normalization factor of the pos-

terior (12) using the Hölder's inequality. Without visiting all possible K topics, FastLDA is able to draw the target topic sample from the posterior probability normalized by the upper bound. By refining the upper bound to approximate the true normalization factor, FastLDA samples the topic label equal to that drawn from the true posterior probability as GS does. So, FastLDA yields exactly the same result as GS but with much less computation. Due to sparseness, SparseLDA [20] partitions the K-tuple posterior vector into three buckets from which a random topic label can be efficiently sampled, which avoids additional computations in GS during the sampling process. SparseLDA is often twice faster than FastLDA [20] because it does not need to update the upper bound of normalization factor. Both methods have time complexity $\mathcal{O}(K_d + K_w)$ for each word token, where $K_d \ll K$ and $K_w \ll K$ are the number of nonzero (NNZ) elements per document and word in sparse matrices $\{\hat{\theta}, \hat{\phi}\}$. Theoretically, they are lossless algorithms having the same accuracy with GS.

To speedup SparseLDA, AliasLDA [12] combines Metropolis-Hastings scheme and the use of the Walker's alias method for estimating $\hat{\phi}$ with amortized $\mathcal{O}(1)$ complexity. So, it reduces the overall time complexity for each word token to $\mathcal{O}(K_d)$. In practice, K_d is often bounded due to sparsity of matrix $\hat{\theta}$ when $K \to \infty$. Under this condition, AliasLDA's time complexity is independent of K. However, the alias table is an approximation to the slowly changing word-topic distribution, which loses accuracy when compared with GS. Our results also confirm that AliasLDA converges faster than SparseLDA but with slightly worse accuracy. Furthermore, based on AliasLDA, LightLDA [21] builds the alias tables for updating both $\{\hat{\theta}, \hat{\phi}\}$ with the overall amortized time complexity $\mathcal{O}(1)$ for each word token. Likewise, because of the alias approximation and delayed update, LightLDA cannot ensure the same accuracy with GS, though it converges significantly faster.

Both VB and EM have time complexity $\mathcal{O}(K \times NNZ)$. Because $NNZ \ll N$, VB and EM should be faster for scanning one iteration of corpus than GS. Moreover, due to MCMC sampling process, GS is well-known to sweep data more iterations than EM and VB for convergence. But, VB is often slower than GS for each iteration, because it computes K time-consuming digamma functions for each $x_{w,d} \neq 0$ [3, 23]. Till now, VB has no specific variants designed for large K. Online VB [9] partitions the data into small minibatches and performs the variational M-step for each minibatch sequentially, which converges using significantly less iterations than batch VB. As far as EM variant is concerned, fast EM (FEM) [24] is designed for large K. At its core is the idea that the responsibility vector $\mu_{w,d}(k)$ in (1) and (2) can be updated on a dynamically changing subset of active topics. Nevertheless, the size of this subset is fixed for all documents in FEM, causing more learning time on those fast convergent documents.

3. LDA REVISITED

We first reformulate LDA's objective function within the entropy framework, and explain why EM can achieve better predictive perplexity than both VB and GS. Prior information, such as Dirichlet hyperparameters $\{\alpha^*, \beta^*\}$ and the number of topics K, can change the entropy of LDA's predictive word distribution for a better performance. Finally, we propose AEM with fast convergence speed for big LDA models, having the amortized time complexity $\mathcal{O}(1)$ for each $x_{w,d} \neq 0$ in terms of K.

3.1 Entropy

The bag-of-word representation of a given corpus is denoted by a sparse matrix $\mathbf{x}_{W\times D}$. We normalize this matrix by the total number of word tokens, $x_{w,d} \leftarrow x_{w,d}/N$, so that each element becomes the observed probability $x_{w,d} = p(\{w,d\}|\text{corpus}), \sum_{w,d} x_{w,d} = 1$. In this way, the observed word distribution of a fixed corpus is represented by the normalized matrix $\mathbf{x}_{W\times D}$, from which we can randomly sample words to reconstruct the corpus. However, this observed word distribution does not provide latent thematic relations of co-occurred words, which requires topic models like PLSA and LDA to learn. LDA represents each document as a finite mixture of topics, and each topic as a finite mixture of vocabulary words. Through finding latent topic structures, LDA can reconstruct the predictive word distribution $\hat{\mathbf{x}}_{W\times D}$ by document-topic and topic-word distributions as follows,

$$\hat{\mathbf{x}}_{W\times D} = \boldsymbol{\phi}_{W\times K} \times \boldsymbol{\theta}_{K\times D}, \tag{18}$$

where $\boldsymbol{\theta} \sim \text{Dir}(\alpha) \propto \prod_d \prod_k [\theta_d(k)]^{\alpha-1}$ and $\boldsymbol{\phi} \sim \text{Dir}(\beta) \propto \prod_k \prod_w [\phi_w(k)]^{\beta-1}$. The inference algorithms of LDA aim to minimize the KL divergence between the observed word distribution and the predictive word distribution,

$$D_{KL}(\mathbf{x}||\hat{\mathbf{x}};\alpha,\beta) = H(\mathbf{x},\hat{\mathbf{x}}|\alpha,\beta) - H(\mathbf{x}), \tag{19}$$

where the entropy of the corpus $H(\mathbf{x})$ is a constant, so that minimizing the KL divergence is equivalent to maximizing the minus cross entropy, i.e., $-H(\mathbf{x},\hat{\mathbf{x}}|\alpha,\beta)$,

$$\begin{aligned} -H &\propto \sum_{w,d} x_{w,d} \log\Big(\hat{x}_{w,d} \prod_d \prod_k [\theta_d(k)]^{\alpha-1} \prod_k \prod_w [\phi_w(k)]^{\beta-1}\Big) \\ &\propto \sum_{w,d} x_{w,d}\Big[\log \sum_k \mu_{w,d}(k) \frac{\theta_d(k)\phi_w(k)}{\mu_{w,d}(k)}\Big] + \\ &\sum_{w,d} x_{w,d}\Big(\sum_d \sum_k \log[\theta_d(k)]^{\alpha-1} + \sum_k \sum_w \log[\phi_w(k)]^{\beta-1}\Big), \end{aligned} \tag{20}$$

where $\mu_{w,d}(k)$ is some topic distribution over the word index $\{w,d\}$ satisfying $\sum_k \mu_{w,d}(k) = 1, \mu_{w,d}(k) \geq 0$. Note that the third line in (20) can be re-written as follows,

$$\begin{aligned} &\sum_{w,d} x_{w,d}\Big(\sum_d \sum_k \log[\theta_d(k)]^{\alpha-1} + \sum_k \sum_w \log[\phi_w(k)]^{\beta-1}\Big) \\ &= \Big(\sum_d \sum_k \log[\theta_d(k)]^{\alpha-1} + \sum_k \sum_w \log[\phi_w(k)]^{\beta-1}\Big) \sum_{w,d} x_{w,d}, \end{aligned} \tag{21}$$

where $\sum_{w,d} x_{w,d} = 1$. According to the Jensen's inequality, we obtain the following lower bound to (20),

$$\begin{aligned} -H \geq &\sum_{w,d} \sum_k x_{w,d}\mu_{w,d}(k)\Big[\log \frac{\theta_d(k)\phi_w(k)}{\mu_{w,d}(k)}\Big] \\ &+ \sum_d \sum_k \log[\theta_d(k)]^{\alpha-1} + \sum_k \sum_w \log[\phi_w(k)]^{\beta-1}. \end{aligned} \tag{22}$$

The equality holds true if and only if

$$\mu_{w,d}(k) \propto \theta_d(k)\phi_w(k). \tag{23}$$

For this choice of $\mu_{w,d}(k)$, Eq. (22) gives a **tight lower bound** on the minus cross entropy (20) we are trying to maximize. Note that Eq. (23) is the same as the E-step (1) in EM. Since the hyperparameters $\{\alpha,\beta\}$ are fixed, without loss of generality, we derive the optimal update for the parameter $\theta_d(k)$. Because $\theta_d(k)$ is parameter of a multinomial distribution. there is an additional constraint that $\sum_k \theta_d(k) = 1$. To deal with this constraint, we construct the Lagrangian from (22) by grouping together only the terms that depend on $\theta_d(k)$,

$$\ell(\theta) = \sum_d \sum_k \Big[\sum_w x_{w,d}\mu_{w,d}(k) + \alpha - 1\Big] \log \theta_d(k) + \delta\Big(\sum_k \theta_d(k) - 1\Big), \tag{24}$$

where δ is the Lagrange multiplier. Taking derivatives, we find

$$\frac{\partial}{\partial_{\theta_d(k)}}\ell(\theta) = \frac{\sum_w x_{w,d}\mu_{w,d}(k) + \alpha - 1}{\theta_d(k)} + \delta. \quad (25)$$

Setting this to zero, we get

$$\theta_d(k) = \frac{\sum_w x_{w,d}\mu_{w,d}(k) + \alpha - 1}{-\delta}. \quad (26)$$

Using the constraint that $\sum_k \theta_d(k) = 1$, we easily find that $-\delta = \sum_k[\sum_w x_{w,d}\mu_{w,d}(k) + \alpha - 1]$. We therefore have our M-step update for the parameter $\theta_d(k)$ as

$$\theta_d(k) = \frac{\hat{\theta}_d(k) + \alpha - 1}{\sum_k[\hat{\theta}_d(k) + \alpha - 1]}. \quad (27)$$

where $\hat{\theta}_d(k) = \sum_w x_{w,d}\mu_{w,d}(k)$ is the expected sufficient statistics. Note that the denominator in (27) is a constant $\sum_w x_{w,d} + K\alpha - K$. The other multinomial parameter can be estimated by

$$\phi_w(k) = \frac{\hat{\phi}_w(k) + \beta - 1}{\sum_w[\hat{\phi}_w(k) + \beta - 1]}, \quad (28)$$

where $\hat{\phi}_w(k) = \sum_d x_{w,d}\mu_{w,d}(k)$ is the expected sufficient statistics. Till now, minimizing the KL divergence between the observed word distribution and the predictive word distribution (19) results in the same E-step and M-step in Algorithm 1.

Minimizing the KL divergence (19) bridges LDA's objective function with non-negative matrix factorization (NMF) [11] as follows,

$$\mathbf{x} \approx \boldsymbol{\phi}\boldsymbol{\theta}, \quad \boldsymbol{\theta} \sim \text{Dir}(\alpha), \boldsymbol{\phi} \sim \text{Dir}(\beta), \quad (29)$$

$$\min\Big(-\sum_{w,d} x_{w,d}\log(\boldsymbol{\phi}\boldsymbol{\theta})_{w,d}\Big), \forall x_{w,d} \neq 0, \quad (30)$$

where the observed matrix $\mathbf{x}$ is decomposed into two non-negative matrices $\{\boldsymbol{\theta}, \boldsymbol{\phi}\}$ in (29). When $\alpha = \beta = 1$ ($\alpha^* = \beta^* = 0$ in Algorithm 1), the Dirichlet constraint (29) can be discarded, so that NMF (29) and (30) become the standard PLSA model [7, 6]. We note that [2] studies the recoverability of the parameters of the LDA by NMF algorithms for topic modeling with provable guarantees.

Different inference algorithms for LDA can be fairly compared by the perplexity metric [4, 3, 9, 23],

$$\exp\Big\{ -\frac{\sum_{w,d} x_{w,d}\log(\boldsymbol{\phi}\boldsymbol{\theta})_{w,d}}{\sum_{w,d} x_{w,d}} \Big\} \propto -\sum_{w,d} x_{w,d}\log(\boldsymbol{\phi}\boldsymbol{\theta})_{w,d}. \quad (31)$$

which has been previously interpreted as the geometric mean of the likelihood in the probabilistic framework. Comparing Eqs. (31) with (30), we find that the perplexity metric can be also interpreted as the cross entropy between the observed word distribution $\mathbf{x}$ and the multiplication of two factorized matrices $\boldsymbol{\phi}\boldsymbol{\theta}$. Because EM can directly minimizes the KL divergence (19) or the cross entropy (30), it often has a much lower predictive perplexity on unseen test data than both VB and GS in Section 2 for predicting the words in unseen corpus. This theoretical analysis has also been supported by extensive experiments in Section 4, where the predictive perplexity on the unseen test data is calculated as follows: Step 1) We randomly partition the dataset into training and test sets in terms of documents. Step 2) We estimate $\hat{\phi}$ on the training set by a number of iterations until convergence. Step 3) We randomly partition each document into 80% and 20% subsets on the test set. Fixing $\hat{\phi}$, we estimate $\hat{\theta}^{80\%}$ on the 80% subset by a number of iterations until convergence, and then calculate the predictive perplexity on the rest 20% subset,

$$\exp\Big\{ -\frac{\sum_{w,d} x^{20\%}_{w,d}\log\big[\sum_k \theta^{80\%}_d(k)\phi_w(k)\big]}{\sum_{w,d} x^{20\%}_{w,d}} \Big\}. \quad (32)$$

3.2 Prior

In this subsection, we shall discuss the effect of prior information, such as the hyperparameters $\{\alpha^*, \beta^*\}$ and the number of topics K, on the predictive perplexity (32). According to (18), (27), (28) and Algorithm 1, we obtain the following equations,

$$\hat{\mathbf{x}} \sim \hat{x}_{w,d} = \sum_k \theta_d(k)\phi_w(k), \quad (33)$$

$$\theta_d(k) = \frac{\hat{\theta}_d(k) + \alpha^*}{\sum_k[\hat{\theta}_d(k) + \alpha^*]}, \phi_w(k) = \frac{\hat{\phi}_w(k) + \beta^*}{\sum_w[\hat{\phi}_w(k) + \beta^*]}, \quad (34)$$

where tuning hyperparameters $\{\alpha^*, \beta^*\}$ and the number of topics K will influence directly the entropy of predicted word distribution, i.e., $H(\hat{\mathbf{x}})$, which is the upper bound of the KL divergence $D_{KL}(\hat{\mathbf{x}}||\mathbf{x})$. The lower $H(\hat{\mathbf{x}})$ results in the lower $D_{KL}(\hat{\mathbf{x}}||\mathbf{x})$.

From (34), when $\{\alpha^*, \beta^*\} \to \infty$, we get $\theta_d(k) \to 1/K$ and $\phi_w(k) \to 1/W$ for uniform distributions, so that $H(\hat{\mathbf{x}})$ is maximized in (33), which also increases the cross entropy (20). On the contrary, the smaller hyperparameters $\{\alpha^*, \beta^*\}$ make the matrices $\{\boldsymbol{\theta}, \boldsymbol{\phi}\}$ in (34) more sparse. According to the theoretical analysis [16], both hyperparameters should be set small (e.g., $\alpha^* \approx 0.1, \beta^* = 0.01$) for better learning efficiency, because each document is associated with a few topics and the topics are known to be word-sparse. Nevertheless, the smallest $\{\alpha^*, \beta^*\} = 0$ does not result in the best predictive performance (32), because PLSA often generalizes worse than LDA on unseen test data [4]. If the training data are sufficiently big, PLSA will approximate LDA because the Bayesian Dirichlet prior plays a less important role in preventing overfitting. To summarize, we have the following hypotheses on hyperparameters:

- Changing $\{\alpha^*, \beta^*\}$ from zeros to infinity will first decreases and then increases $H(\hat{\mathbf{x}})$ as well as (32).
- When D increases, the best hyperparameters $\{\alpha^*, \beta^*\} \to 0$.

Observing (33), we also find that increasing K can decrease $H(\hat{\mathbf{x}})$ because more sparsity or expression power is added on matrices $\{\boldsymbol{\theta}, \boldsymbol{\phi}\}$. When $K \to \infty$, the decrease level of $H(\hat{\mathbf{x}})$ will become small because the minimum of $H(\hat{\mathbf{x}})$ exists. Hence, we summarize the following hypothesis on K for predictive performance:

- Increasing K will steadily decrease $H(\hat{\mathbf{x}})$ to the minimum.

However, our observation is inconsistent with the conclusion made in [16] that the user needs to avoid selecting overly large number of topics for LDA. We conjecture that the difference lies in the different performance metrics used for LDA. For the predictive perplexity metric (32), we find that more topics are beneficial, which is supported by the recent practice in learning big topic models [19] for search engine and online advertisement applications.

Due to similar algorithmic structures, the above hypotheses for EM are also applicable to VB in Algorithm 2 and GS in Algorithm 3 but with one exception for VB that has not been fully discussed in [3]. When K or W is very large, the approximation (11) does not hold true. Under this condition, VB cannot minimize the cross entropy (20) for the best predictive perplexity (32) even if we carefully tune the Dirichlet hyperparameters $\{\alpha^*, \beta^*\}$.

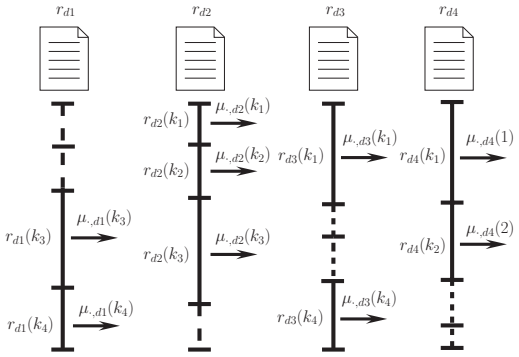

Figure 1: An example of adaptive EM (AEM).

3.3 Convergence

The core idea of the proposed AEM for fast convergence is that we do not need to update the responsibility vector $\mu_{w,d}(k), \forall k$ in both E-step and M-step due to sparsity of matrices $\{\hat{\theta}, \hat{\phi}\}$. Instead, we can update only a subset of active topics $\mathcal{K}_d$ for each document selected by ranking the residual,

$$r_d(k) = \sum_w x_{w,d} |\mu_{w,d}(k) - \mu^{old}_{w,d}(k)|, \tag{35}$$

where $\mu^{old}_{w,d}(k)$ stores the responsibility vector in the previous iterations. For the dth document, if $r_d(k_1) > r_d(k_2)$, the topic k_1 is more important or active than the topic k_2 for updating because k_1 has a larger change of topic assignment than k_2. When the convergence is achieved, it is expected that $r_d(k) \to 0, \forall k$ representing all topics become inactive. Because EM can converge to the local optimum of (20), $r_d(k^*)$ will always become smaller after $\mu_{\cdot,d}(k^*)$ has been selected and updated in both E-step and M-step. Hence, the topic k^* becomes more inactive, and may not be selected in the next iteration by dynamically ranking in descending order. The complexity of initial quick-sort ranking is at most $\mathcal{O}(D \times K \log K)$ for each iteration, and that of the subsequent insert-sort ranking is nearly independent of K. Both ranking operations will not add much time complexity when compared with the original EM complexity $\mathcal{O}(K \times NNZ)$. In this way, all important or active topics of each document can be selected to form an active topic subset $\mathcal{K}_d$ at different learning iterations. In practice, the cardinality of this subset $|\mathcal{K}_d| \ll K$ when K is very large. Specifically, Fast EM (FEM) [24] fixes $|\mathcal{K}_d| = c, \forall d$ at all iterations, where c is a constant, e.g., $c = 10$. Since EM has a time complexity $\mathcal{O}(K)$, FEM thus has an amortized time complexity $\mathcal{O}(1)$ in terms of K. However, $|\mathcal{K}_d| = c, \forall d$ in FEM cannot ensure the fastest convergence speed with two reasons. First, different documents may have different topic subset cardinalities. Second, documents may shrink the subset cardinality with more learning iterations. These weaknesses motivate the proposed AEM algorithm to find the adaptive subset cardinality.

Fig. 1 shows an example of the adaptive subset cardinality. All solid and dashed segments denote normalized responsibility vector $\mu_{w,d}(k)$ in proportion where $K = 4$. The solid segments and arrows denote those selected active topic values $\mu_{w,d}(k^*)$ for updating in both E-step and M-step, while the dashed segments denote those unchanged values. Suppose that there are four documents $d1$, $d2$, $d3$ and $d4$ having residual vectors $r_{d1}(k)$, $r_{d2}(k)$, $r_{d3}(k)$ and $r_{d4}(k)$. For the document $d1$, if two residuals $\{r_{d1}(k_3), r_{d1}(k_4)\}$ are ranked higher, the topics $\{k_3, k_4\}$ are activated in the subset $\mathcal{K}_{d1}$ so that the values $\{\mu_{\cdot,d1}(k_3), \mu_{\cdot,d1}(k_4)\}, \forall w \in d1$ are updated in both E-step and M-step. As far as the document $d2$ is concerned, if three residuals $\{r_{d2}(k_1), r_{d2}(k_2), r_{d2}(k_3)\}$ are ranked higher, then $\{k_1, k_2, k_3\}$ are activated in the subset $\mathcal{K}_{d2}$ for updating responsibility $\{\mu_{\cdot,d2}(k_1), \{\mu_{\cdot,d2}(k_2), \{\mu_{\cdot,d2}(k_3)\}, \forall w \in d2$. Similar rules are applied to documents $d3$ and $d4$. Note that documents may have different subset cardinalities, e.g., $|\mathcal{K}_{d1}| \neq |\mathcal{K}_{d2}|$. Moreover, the number of active topics for each document will become smaller and smaller when the number of learning iterations increases due to convergence, i.e., $|\mathcal{K}_d| \to 0$. Using the adaptive subset cardinality, AEM consumes less computation than FEM and EM in each loop. Also, AEM dynamically refines and sorts document residuals, where the active topics in the previous iterations may be inactive in the subsequent iterations. As a result, those unchanged or inactive responsibility values will be activated and updated as indicated by the dashed segments in Fig. 1. Because all topics can be updated but with different priorities, AEM retains almost the same topic modeling accuracy with EM.

```
1 for k in adaptive subset K_d do
2     θ̂_{-w,d}(k) ← θ̂_d(k) − x_{w,d} μ_{w,d}(k);
3     φ̂_{w,-d}(k) ← φ̂_w(k) − x_{w,d} μ_{w,d}(k);
4     μ_{w,d}(k) ← normalize([θ̂_{-w,d}(k) + α*][φ̂_{w,-d}(k) +
      β*]/Σ_w[φ̂_{w,-d}(k) + β*]);
5     θ̂_d(k) ← θ̂_{-w,d}(k) + x_{w,d} μ_{w,d}(k);
6     φ̂_w(k) ← φ̂_{w,-d}(k) + x_{w,d} μ_{w,d}(k);
7     r_d(k) ← Σ_w x_{w,d}|μ_{w,d}(k) − μ^{old}_{w,d}(k)|; μ^{old}_{w,d}(k) ← μ_{w,d}(k);
8     K_d ← 1 : K*, Σ_{k=1}^{K*} r_d(k) ≈ η Σ_{k=1}^{K} r_d(k), η ∈ (0, 1);
9 end
```

Figure 2: The adaptive EM (AEM) algorithm.

Fig. 2 shows the AEM algorithm based on incremental EM [13, 24] (lines from 2 to 6), which performs E-step and M-step for each $x_{w,d} \neq 0$ incrementally, similar to GS in Algorithm 3. Previous algorithms such as CVB0 [3] and asynchronous BP [23] are equivalent to incremental EM, which has a faster convergence speed than EM in Algorithm 1 confirmed by experiments in [3, 23]. Because we only update $\mu_{w,d}(k)$ for a subset of topics, we need to do local normalization in line 4 as follows:

$$\hat{\mu}^{new}_{w,d}(k) = \frac{\mu_{w,d}(k)}{\sum_k \mu_{w,d}(k)} \times \sum_k \mu^{old}_{w,d}(k), k \in \mathcal{K}_d, \tag{36}$$

where $\mu_{w,d}(k)$ is updated according to (1). Line 7 updates dynamically document residuals according to (35). For each $x_{w,d} \neq 0$, AEM calculates an adaptive subset of topics $\mathcal{K}_d$ in line 8 as follows,

$$\mathcal{K}_d \leftarrow 1 : K^*, \sum_{k=1}^{K^*} r_d(k) \approx \eta \sum_{k=1}^{K} r_d(k), \ \eta \in (0, 1), \tag{37}$$

where $r_d(k)$ is sorted in descending order in terms of k, and η is a tunable parameter to control the proportion. Eq. (37) shows that we adaptively select a subset with cardinality K^* that can at least occupy a controlled proportion of the overall document residual $\sum_{k=1}^{K} r_d(k)$ at the current iteration. Obviously, $\sum_{k=1}^{K} r_d(k)$ will shrink with more learning iterations due to convergence of the EM algorithm, so that K^* will also shrink according to (37) satisfying our analysis that different iterations will have different subset cardinalities. Moreover, some converged documents may have smaller K^* and some non-converged documents may have bigger K^*, leading to the expected adaptive subset cardinality. Because of similar algorithmic structures, the idea of AEM can be also applied to speeding up both VB and GS.

Similar to FEM, AEM also has the amortized $\mathcal{O}(1)$ time complexity for each $x_{w,d} \neq 0$. Nevertheless, the space complexity of batch AEM is $\mathcal{O}(K \times (NNZ + D + W))$ because the matrices

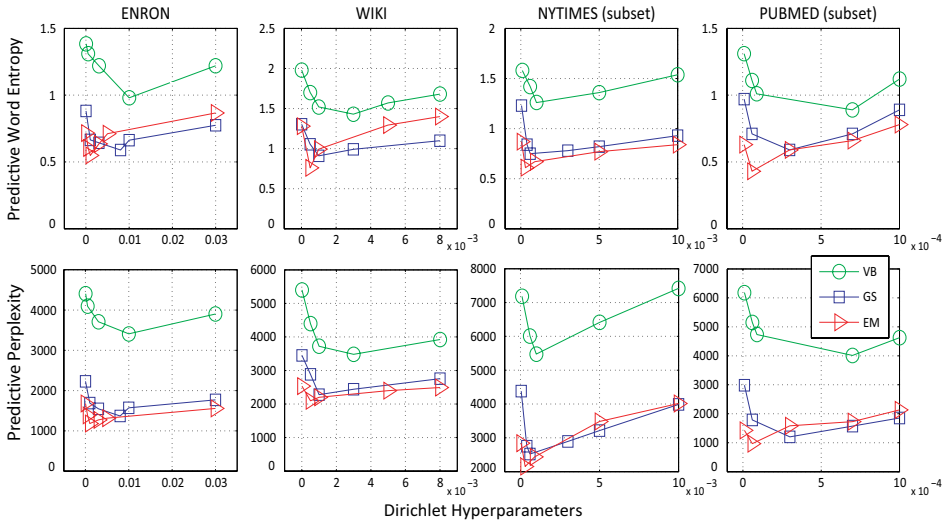

Figure 3: Changing hyperparameters will change the entropy of predictive word distribution and predictive perplexity.

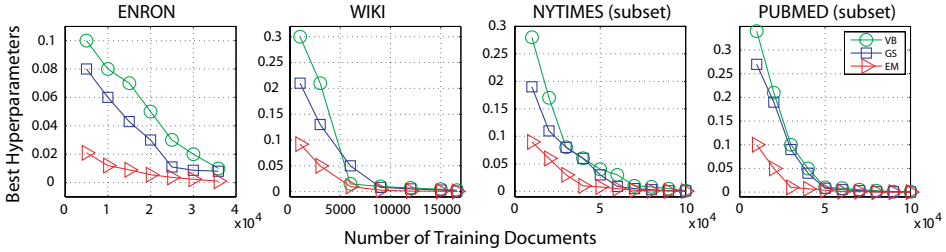

Figure 4: Big data make the best Dirichlet hyperparameters close to zeros.

Table 2: Statistics of Datasets.

Data	D_{tr}	D_{te}	W	NNZ	N	$H(\mathbf{x})$
KOS	3087	343	6906	353160	467714	2903.58
NIPS	1350	150	12419	746316	1932365	13472.34
ENRON	35874	3987	28102	3710420	6412172	21195.39
WIKI	16606	4152	83470	9272290	21051241	53601.87
NYTIMES	290000	10000	102660	0.7×10^8	1×10^8	2.05×10^6
PUBMED	8180000	20000	141043	4.7×10^8	5.6×10^8	2.16×10^6

$\{\boldsymbol{\mu}_{K\times NNZ}, \hat{\theta}_{K\times D}, r_{K\times D}, \hat{\phi}_{W\times K}\}$ are in memory, which are unaffordable for big data and models with large NNZ, D, and K. Indeed, AEM increases the space complexity to reduce the time complexity. Online algorithms [9, 24] partition D documents into $1 \le s \le S$ small mini-batches so that online AEM has space complexity $\mathcal{O}(K \times (NNZ_s + D_s + W))$, where $NNZ_s \ll NNZ$ and $D_s \ll D$ for each mini-batch. More details for online algorithm implementation can be found in [9, 24].

4. EXPERIMENTS

Table 2 shows 6 publicly available datasets [15] with varying D, W and document lengths NNZ/D or N/D. Also, we calculate the entropy $H(\mathbf{x})$ of each dataset, and find that $H(\mathbf{x})$ increases with W because the large W corresponds to the large uncertainty space for word prediction. Each dataset is randomly partitioned into training and test sets with D_{tr} and D_{te} documents, respectively. We also extract small subsets from the original NYTIMES and PUBMED with $D_{tr} = D_{te} = 10^5$. We evaluate different inference algorithms of LDA using predictive perplexity (32) on test sets. The lower perplexity means the better performance. All experiments are carried out on a server with two Intel Xeon X5690 3.47G processors and 140G memory. For a fair comparison, only one CPU core is activated and only 256KB L2 cache is available. All algorithms are implemented using C++ [9, 22, 12], and VB or EM uses single-precision floating-point computation [13].

4.1 Results on Prior

We study the effect of hyperparameters $\{\alpha^*, \beta^*\}$ and K on predictive performance of EM, VB and GS. We repeat experiments three times from three random initializations and report the average results. For each experiment, EM, VB and GS start from the same random initialization. For simplicity, we set $\alpha^* = \beta^*$ and use grid search with fine-grained step size 0.001 for the best values.

The first line of Fig. 3 (fixing $K = 300$) shows that the entropy of predictive word distribution $H(\hat{\mathbf{x}})$ first goes down and then goes up when the hyperparameters $\{\alpha^*, \beta^*\}$ increase from zeros with small steps on four datasets. The second line of Fig. 3 confirms that the perplexity changes similarly with $H(\hat{\mathbf{x}})$. The minimum perplexity of EM is much lower than those of VB and GS for much better predictive performance. Also, we find that $\alpha^* = \beta^* = 0$ does not produce the lowest perplexity. For example, on ENRON, EM has 1678 perplexity value when $\alpha^* = \beta^* = 0$, and achieves the minimum 1180 perplexity value when $\alpha^* = \beta^* = 0.001$. This result reconfirms that LDA indeed has a better generalization ability than PLSA by adding Dirichlet prior constraints. Moreover, EM has much smaller best hyperparameters than both VB and GS based on the grid search. For example, on ENRON, the best hyperparameters for EM, VB and GS are 0.001, 0.01 and 0.008, respectively. The major reason is that EM can fit the observed word distribu-

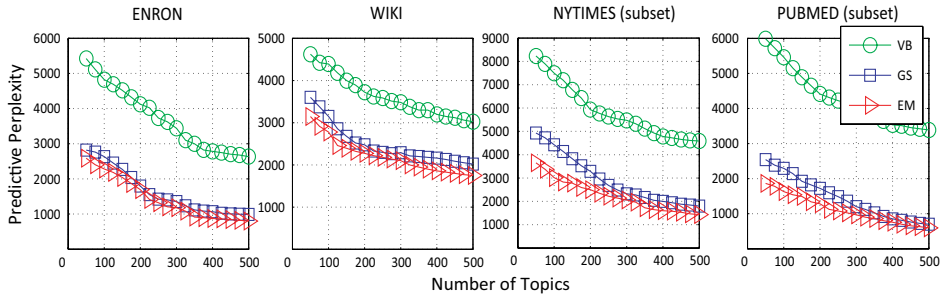

Figure 5: More topics lead to lower predictive perplexity.

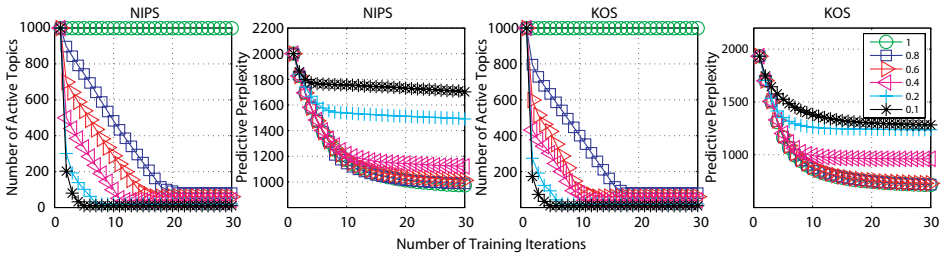

Figure 6: The number of active topics decreases significantly after a few training iterations.

tion $\mathbf{x}$ better (measured by the cross entropy) than VB and GS, so that EM does not need much Dirichlet constraints to improve the generalization ability. GS can be viewed as a hard version of incremental EM, so it needs slightly smaller hyperparameters than VB for the lowest predictive perplexity. When the hyperparameters increase to bigger values, GS may have even better predictive performance than EM, because bigger hyperparameters will oversmooth the multinomial parameters generated by EM. Since VB approximates EM when K and W are small, it needs slightly bigger hyperparameters than EM and GS for the best performance. Our conclusion is that different inference algorithms may have different hyperparameters to achieve their own best predictive performance. EM needs smaller hyperparameters than VB and GS, and always achieves the lowest predictive perplexity because it directly minimizes (32). All results are consistent on four different small datasets, and reconfirm our hypotheses in Section 3.2.

Fig. 4 (fixing $K = 300$) shows that if we increase steadily the number of training documents D_{tr}, the best hyperparameters values $\{\alpha^* = \beta^*\}$ decrease to almost zeros based on grid search for EM, VB and GS. For example, on PUBMED (subset), the best hyperparameters are $\{0.1, 0.34, 0.27\}$ for EM, GS and VB when $D_{tr} = 10^4$, and decrease to $\{0.00006, 0.0007, 0.0003\}$ when training data increase to $D_{tr} = 10^5$. In anticipation, this trend remains as $D_{tr} \to \infty$, the best $\{\alpha^*, \beta^*\} \to 0$. This finding reconfirms that big data will make LDA reduce to PLSA without Dirichlet prior constraints in Section 3.2. In Bayesian theory, the data-driven likelihood plays more important roles than the prior information when training data are sufficiently big.

Fig. 5 shows that increasing the number of topics K will always decrease the predictive perplexity for all three algorithms. We increase from $K = 50$ with step size 25 until $K = 500$. In each experiment, for a fair comparison, we fix different best hyperparameters (grid search when $K = 300$) for EM, VB and GS on different datasets. For example, on WIKI, we fix $\alpha^* = \beta^* = \{0.0005, 0.003, 0.001\}$ for EM, VB and GS, respectively. On NYTIMES (subset), we fix $\alpha^* = \beta^* = \{0.0004, 0.001, 0.0009\}$ for EM, VB and GS, respectively. Consistent with Fig. 3, we see that EM yields the lowest predictive perplexity, while VB has the highest predictive perplexity. For example, on NYTIMES (subset), EM has perplexity value 1434, VB 4589, and GS 1800 when $K = 500$, respectively. Likewise, on PUBMED (subset), EM has perplexity value 597, VB 3389, and GS 700 when $K = 500$, respectively. These results confirm that there is a big accuracy difference between VB and EM/GS, because VB cannot approximate EM when K and W are very large. Note that previous work [3, 23] shows that VB has comparable accuracy with both EM and GS just because the dataset has small $W \leq 3 \times 10^4$ and the experimental setting uses small $K \leq 100$. Similar to Fig. 3, increasing K can also decrease the entropy of predictive word distribution $H(\hat{\mathbf{x}})$ as well as the predictive perplexity values, reconfirming our hypothesis in Section 3.2.

4.2 Results on Convergence

All experiments on convergence are repeated three times with three different random initializations, and the average results are reported. For the same experiment, all algorithms start from the same random initialization. For a fair comparison, the hyperparameters $\{\alpha^* = \beta^*\}$ are carefully tuned based on grid search for the best predictive performance. If the difference of predictive perplexity < 10 between two successive iterations, we assume the algorithm achieves the convergence state.

Fig. 6 shows the average number of active topics K^* in (37) for all documents as a function of number of training iterations ($\alpha^* = \beta^* = 0.01, K = 1000$). We change the tunable parameter $\eta = \{0.1, 0.2, 0.4, 0.6, 0.8, 1\}$ and calculate the predictive perplexity curves on NIPS and KOS datasets. When $\eta = 1$, we get $K^* = K = 1000$ for scanning all topics in all documents. In this case, AEM has the same predictive perplexity as EM. When $\eta = \{0.8, 0.6\}$, we find that the average K^* drops significantly to 60 in less than 15 iterations, and the predictive perplexity keeps almost the same with EM at the convergence point. For example, AEM ($\eta = 0.6$) converges at $\{991, 717\}$, while EM converges at $\{971, 713\}$ on NIPS and KOS. However, if $\eta = 0.1$, though the average K^* drops to 10 in 5 iterations, the predictive perplexity has

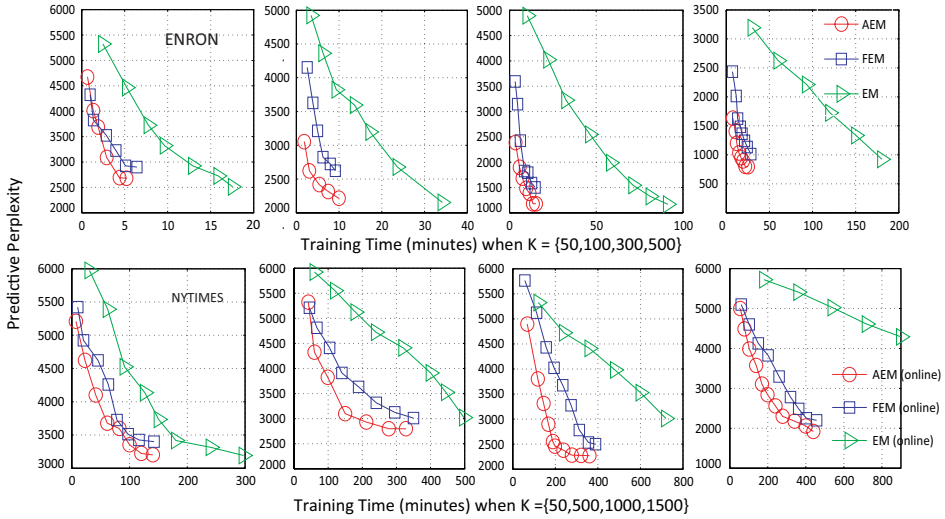

Figure 7: Batch/online AEM converges faster than both FEM and EM on small and big datasets/models.

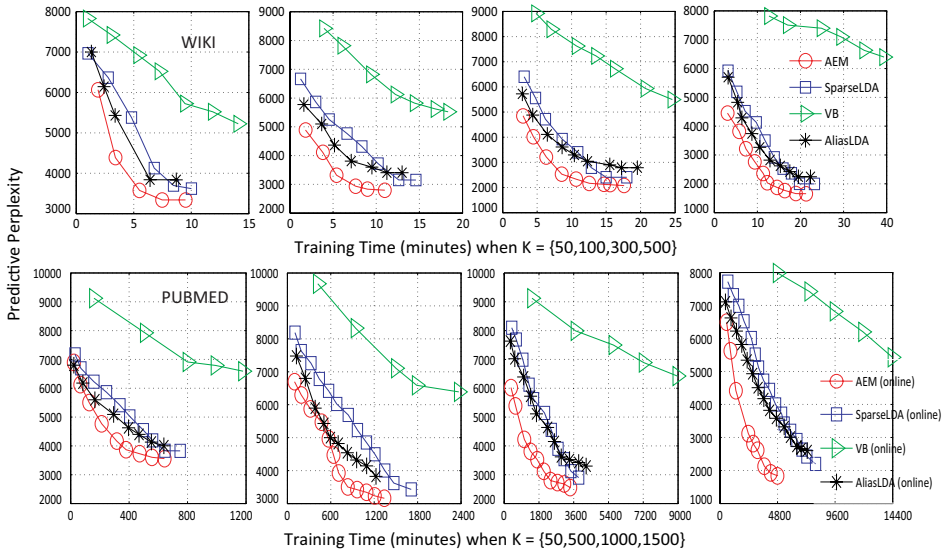

Figure 8: Batch/online AEM converges faster than VB, SparseLDA and AliasLDA on small and big datasets/models.

an obvious loss when compared with EM at the convergence point. For example, AEM ($\eta = 0.1$) converges at $\{1702, 1283\}$, far from those of EM. Therefore, we set the parameter of AEM $\eta = 0.6$ in the following experiments.

Fig. 7 shows the predictive perplexity as a function of training time (minutes) of batch/online AEM, FEM and EM on small (first line for batch algorithms on ENRON, $K = \{50, 100, 300, 500\}$, $\alpha^* = \beta^* = 0.001$) and big (second line for online algorithms on NYTIMES, $K = \{50, 500, 1000, 1500\}$, $\alpha^* = \beta^* = 0.0004$) datasets as well as models. The experimental setting of online algorithms for big data exactly follows that described in [24] with default parameters, including the mini-batch size $D_s = 5120$ and the stop condition that the delta training perplexity at two successive iterations < 10 for scanning each mini-batch. In the first line on ENRON, batch AEM converges around $30\% \sim 100\%$ faster than batch FEM, and around $6 \sim 10$ times faster than batch EM. When K increases to the maximum 500 in our experiment, batch AEM achieves the maximum 14 times speedup over batch EM. Note that the convergence time of AEM increases slowly with the increase of K. For example, from $K = 50$ to $K = 500$, the convergence time of AEM increases from 4.5 to 26 minutes, while the convergence time of EM increases from 27 to 356 minutes. In anticipation, when K increases to a very large value, AEM will have almost a constant convergence time. In the second line on NYTIMES, online AEM saves around 12% and 96% convergence time of online FEM and EM when $K = 1500$. All the perplexity curves of batch/online EM locate left-bottom to those of batch/online FEM and EM, showing significant convergence speedup. Generally, AEM converges at a slightly lower level than FEM, which implies that the adaptive subset cardinality in AEM can produce a higher topic modeling accuracy than the fixed subset cardinality in FEM.

Fig. 8 shows the predictive perplexity as a function of training time (minutes) of batch/online AEM, VB, SparseLDA (GS) and AliasLDA (GS) on small (first line for batch algorithms on WIKI, $K = \{50, 100, 300, 500\}$, $\alpha^* = \beta^* = 0.0005, 0.003, 0.001$ for batch AEM, VB and GS, respectively) and big (second line

for online algorithms on PUBMED, $K = \{50, 500, 1000, 1500\}$, $\alpha^* = \beta^* = 0.00006, 0.0007, 0.0003$ for online AEM, VB and GS, respectively) datasets as well as models. In the first line on WIKI, AliasLDA is around $10\% \sim 30\%$ faster than SparseLDA for different K, but it often converges at a higher perplexity value. The major reason is that SparseLDA is a lossless GS algorithm but AliasLDA is a lossy implementation due to approximate alias tables. For different K, AEM on average is around $10\% \sim 20\%$ faster than AliasLDA, and converges at a much lower perplexity value. First, for each iteration AEM scans NNZ but AliasLDA scans N elements, where $NNZ \ll N$ in Table 2. Second, AEM directly minimizes the cross entropy (20) for a lower perplexity value. VB performs the worst among these algorithms either on convergence speed or on predictive performance. In the second line on PUBMED, online AEM (5923 minutes) converges significantly faster than online AliasLDA (8531 minutes) when $K = 1500$. Also, online AEM reaches the predictive perplexity value 1840, while online AliasLDA converges at a much higher value 2600. Likewise, all the perplexity curves of batch/online AEM locate left-bottom to those of other algorithms, which confirms a salient convergence speedup.

5. CONCLUSIONS

In this paper, we revisit three major inference algorithms of LDA from the entropy view, and find answers to three important questions raised in Section 1:

- EM can achieve the best predictive performance because it directly minimizes the cross entropy between the observed word distribution and the predictive word distribution generated by LDA.
- Hyperparameters can change the entropy of the predictive word distribution produced by LDA, and thus improve the predictive performance for unseen corpus. When the training data become big, hyperparameters play an unimportant role in tuning the predictive performance. Also, more topics in LDA can yield better predictive performance in terms of the perplexity metric.
- Empirically, the proposed batch/online AEM algorithm converges the fastest to the local optimum of LDA objective function. The idea of updating only a varying subset of active topics may inspire more efficient topic modeling algorithms such as HDP [17] or efficient EM for other latent variable models [14]. Future work may parallelize AEM in multi-core [13] and multi-machine [19] systems for real-world content analysis applications.

Acknowledgements

This work was supported by NSFC (Grant No. 61373092, 61572365, 61272449 and 61572339). This work was partially supported by Collaborative Innovation Center of Novel Software Technology and Industrialization, and Science and Technology Commission of Shanghai Municipality (Grant No. 15ZR1443000).

6. REFERENCES

[1] A. Ahmed, M. Aly, J. Gonzalez, S. Narayanamurthy, and A. Smola. Scalable inference in latent variable models. In *WSDM*, pages 123–132, 2012.

[2] S. Arora, R. Ge, Y. Halpern, D. Mimno, A. Moitra, D. Sontag, Y. Wu, and M. Zhu. A practical algorithm for topic modeling with provable guarantees. In *ICML*, 2013.

[3] A. Asuncion, M. Welling, P. Smyth, and Y. W. Teh. On smoothing and inference for topic models. In *UAI*, pages 27–34, 2009.

[4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[5] N. de Freitas and K. Barnard. Bayesian latent semantic analysis of multimedia databases. Technical report, University of British Columbia, 2001.

[6] E. Gaussier and C. Goutte. Relation between PLSA and NMF and implications. In *SIGIR*, pages 15–19, 2005.

[7] M. Girolami and A. Kabán. On an equivalence between PLSI and LDA. In *SIGIR*, pages 433–434, 2003.

[8] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proc. Natl. Acad. Sci.*, 101:5228–5235, 2004.

[9] M. Hoffman, D. Blei, and F. Bach. Online learning for latent Dirichlet allocation. In *NIPS*, pages 856–864, 2010.

[10] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42:177–196, 2001.

[11] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

[12] A. Q. Li, A. Ahmed, S. Ravi, and A. J. Smola. Reducing the sampling complexity of topic models. In *KDD*, pages 891–900, 2014.

[13] X. Liu, J. Zeng, X. Yang, J. Yan, and Q. Yang. Scalable parallel EM algorithms for latent Dirichlet allocation in multi-core systems. In *WWW*, pages 669–679, 2015.

[14] K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[15] I. Porteous, D. Newman, A. Ihler, A. Asuncion, P. Smyth, and M. Welling. Fast collapsed Gibbs sampling for latent Dirichlet allocation. In *KDD*, pages 569–577, 2008.

[16] J. Tang, Z. Meng, X. Nguyen, Q. Mei, and M. Zhang. Understanding the limiting factors of topic modeling via posterior contraction analysis. In *ICML*, 2014.

[17] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(576):1566–1581, 2006.

[18] H. Wallach, D. Mimno, and A. McCallum. Rethinking LDA: Why priors matter. In *NIPS*, pages 1973–1981, 2009.

[19] Y. Wang, X. Zhao, Z. Sun, H. Yan, L. Wang, Z. Jin, L. Wang, Y. Gao, C. Law, and J. Zeng. Peacock: Learning long-tail topic features for industrial applications. *ACM Transactions on Intelligent Systems and Technology*, 6(4):47, 2015.

[20] L. Yao, D. Mimno, and A. McCallum. Efficient methods for topic model inference on streaming document collections. In *KDD*, pages 937–946, 2009.

[21] J. Yuan, F. Gao, Q. Ho, W. Dai, J. Wei, X. Zheng, E. P. Xing, T.-Y. Liu, and W.-Y. Ma. LightLDA: Big topic models on modest compute clusters. In *WWW*, pages 1351–1361, 2015.

[22] J. Zeng. A topic modeling toolbox using belief propagation. *J. Mach. Learn. Res.*, 13:2233–2236, 2012.

[23] J. Zeng, W. K. Cheung, and J. Liu. Learning topic models by belief propagation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(5):1121–1134, 2013.

[24] J. Zeng, Z.-Q. Liu, and X.-Q. Cao. Fast online EM for big topic modeling. *IEEE Trans. Knowl. Data Eng.*, page arXiv:1210.2179 [cs.LG], 2015.

[25] K. Zhai, J. Boyd-Graber, and N. Asadi. Mr. LDA: A flexible large scale topic modeling package using variational inference in MapReduce. In *WWW*, pages 879–888, 2012.