Access to this work was provided by the University of Maryland, Baltimore County (UMBC) ScholarWorks@UMBC digital repository on the Maryland Shared Open Access (MD-SOAR) platform.

Please provide feedback

Please support the ScholarWorks@UMBC repository by emailing scholarworks-group@umbc.edu and telling us what having access to this work means to you and why it’s important to you. Thank you.
Scalable Collapsed Inference for High-Dimensional Topic Models

Rashidul Islam and James Foulds
Department of Information Systems
University of Maryland, Baltimore County
{islam.rashidul, jfoulds}@umbc.edu

Abstract

The bigger the corpus, the more topics it can potentially support. To truly make full use of massive text corpora, a topic model inference algorithm must therefore scale efficiently in 1) documents and 2) topics, while 3) achieving accurate inference. Previous methods have achieved two out of three of these criteria simultaneously, but never all three at once. In this paper, we develop an online inference algorithm for topic models which leverages stochasticity to scale well in the number of documents, sparsity to scale well in the number of topics, and which operates in the collapsed representation of the topic model for improved accuracy and run-time performance. We use a Monte Carlo inner loop in the online setting to approximate the collapsed variational Bayes updates in a sparse and efficient way, which we accomplish via the Metropolis-Hastings Walker method. We showcase our algorithm on LDA and the recently proposed mixed membership skip-gram topic model. Our method requires only amortized $O(k_d)$ computation per word token instead of $O(K)$ operations, where the number of topics occurring for a particular document $k_d \ll$ the total number of topics in the corpus $K$, to converge to a high-quality solution.

1 Introduction

Topic models are powerful tools for analyzing today’s massive, constantly expanding digital text information by representing high-dimensional data in a low-dimensional subspace. We can recover the main themes of a corpus by using topic models such as latent Dirichlet allocation (LDA) to organize, understand, search, and explore the documents (Blei et al., 2003).

Traditional LDA inference techniques such as variational Bayes and collapsed Gibbs sampling do not readily scale to corpora containing millions of documents. To scale up inference, the main approaches are distributed algorithms (Newman et al., 2008) and stochastic algorithms (Hoffman et al., 2010, 2013). Stochastic algorithms, such as stochastic variational inference (SVI), operate in an online fashion, and hence do not need to see all of the documents before updating the topics, so they can be applied to corpora of any size, without expensive distributed hardware (Hoffman et al., 2010). The “collapsed” representation of topic models is also frequently important, as it leads to faster convergence, efficient updates, and lower variance in estimation (Griffiths and Steyvers, 2004). The stochastic collapsed variational Bayesian inference (SCVB0) algorithm, proposed by (Foulds et al., 2013), combines the benefits of stochastic and collapsed inference.

Larger corpora typically support more topics, which brings the additional efficiency challenge of training a larger model (Mimno et al., 2012). This challenge has been addressed by exploiting sparsity to perform updates in time sublinear in the number of topics. A sparse variant of the SVI algorithm for LDA, SSVI, proposed by (Mimno et al., 2012), is scalable to large numbers of topics, but does not fully exploit the collapsed representation of LDA, which is important for faster convergence and improved inference accuracy, due to a better variational bound (Teh et al., 2007). The Metropolis Hastings Walker (MHW) method (Li et al., 2014) scales well in the number of topics, and uses a collapsed inference algorithm, but it operates in the batch setting, so it is not scalable to large corpora. LightLDA (Yuan et al., 2015)
is a distributed approach to the MHW method which adopts a data-and-model-parallel strategy to maximize memory and CPU efficiency. However, it is not an online approach, and furthermore requires multiple expensive computer clusters to converge faster. Tensor methods are another approach to speeding up topic models (Anandkumar et al., 2014; Arora et al., 2012), which theoretically guarantee the recovery of the true parameters by overcoming the problem of local optima. These techniques use the method of moments instead of maximum likelihood estimation or Bayesian inference, which leads to lower data efficiency, and sometimes unreliable performance.

In this work, we propose a highly efficient and scalable inference algorithm for topic models. We develop an online algorithm which leverages stochasticity to scale well in the number of documents, sparsity to scale well in the number of topics, and which operates in the collapsed representation of topic models. We thereby combine the individual benefits of SVI, SSVI, SCVB0, and MHW into a single algorithm. Our approach is to develop a sparse version of SCVB0. Inspired by SSVI, we use a Monte Carlo inner loop to approximate the SCVB0 variational distribution updates in a sparse and efficient way, which we accomplish via MHW method. To show the generality of our algorithm, we explore the benefits of our inference method for LDA and another recently proposed topic model, MMSGTM, with experiments on both small and large-scale datasets.

2 Background

To build the foundation for our proposed method, in this section we provide the necessary background on LDA and MMSGTM topic models and their associated inference algorithms. This is followed by a description of the MHW sampler for reducing topic model sampling complexity.

2.1 Latent Dirichlet Allocation and SCVB0

Probabilistic topic models such as LDA (Blei et al., 2003) use latent variables to encode co-occurrence patterns between words in text corpora and other bag-of-words represented data. In LDA, we assume that the D documents in a corpus are each from mixture distributions of K individual topics φk, k ∈ {1, ..., K}, each of which are discrete distributions over words. For a document j of length Nj, the local (document-level) variables θj are a distribution over topics drawn from a Dirichlet prior with parameters αk and for each token, global variables (corpus-level) φk are drawn from a Dirichlet prior with parameters βw. Due to conjugacy, we can marginalize out topics Θ and distributions over topics Φ, and perform inference only on the topic assignments Z in the collapsed representation of LDA (Griffiths and Steyvers, 2004). For scalable and accurate inference, Foulds et al. (2013) proposed a stochastic collapsed variational inference algorithm, SCVB0. The SCVB0 approach computes a variational discrete distribution γijk over the K topic assignment probabilities for each word i in each document j, but does not maintain the γ variables that increase the memory requirement of original batch CVB0 algorithm (Asuncion et al., 2009). SCVB0 iteratively updates each γijk using

\[
\gamma_{ijk} := \frac{N_{w_{ij}k} + \beta_{w_{ij}}}{N^k + \sum_{w} \beta_w} (N_{jk}^{\Theta} + \alpha_k)
\]

for each topic k, with jth document’s jth word w_{ij}. The N^Z, N^{\Theta}, and N^{\Phi} are referred to as CVB0 statistics, where N^Z is the vector of expected number of words assigned to each topic, each entry j, k of matrix N^{\Theta}, and each entry w, k of matrix N^{\Phi} are the expected number of times document j, and word w are assigned to topic k, respectively, across the corpus. To do stochastic updates of these variables, one sequence of step-sizes ρ^{\Phi} for N^{\Phi} and N^Z and another sequence ρ^{\Theta} for N^{\Theta} are maintained. The update of N_{kj}^{\Theta} for every token i of document j with an online average of the current value and its expected value is

\[
N_{kj}^{\Theta} := (1 - \rho_t^{\Theta})N_{kj}^{\Theta} + \rho_t^{\Theta} C_j \gamma_{ij}
\]

where C_j is the document length. In practice, it is too expensive to update N^{\Phi} after every token. This leads to the use of minibatch updates with the average of the M per-token estimates of the form Y_{ij}, which is a W × K matrix with the w_{ij}th row being γ_{ij} and with zeros in the other entries:

\[
N^{\Phi} := (1 - \rho_t^{\Phi})N^{\Phi} + \rho_t^{\Phi} N^{\Phi}
\]
\[
N^Z := (1 - \rho_t^{\Theta})N^Z + \rho_t^{\Theta} N^Z
\]
where $\hat{N}^\Phi = \frac{C}{|M|} \sum_{ij \in M} Y_{ij}$, $\hat{N}^Z = \frac{C}{|M|} \sum_{ij \in M} \gamma_{ij}$, and $C$ is the number of words in the corpus. The SCVB0 algorithm outperforms stochastic VB (Hoffman et al., 2010) on large corpora by converging faster and often to a better solution (Foulds et al., 2013). However, the SCVB0 algorithm does not leverage sparsity, and hence requires $O(K)$ operations per word token.

2.2 MMSG Topic Model

To show the generality of our approach to topic models other than LDA, we will also apply our method to a recent model called the Mixed Membership Skip-gram Topic Model (MMSGTM) (Foulds, 2018), which combines ideas from topic models and word embeddings (cf. also (Das et al., 2015; Liu et al., 2015)). MMSGTM’s generative model for words and their surrounding context is:

- For each word $w_i$ in the corpus
  - Sample a topic $z_i \sim \text{Discrete}(\theta_{w_i})$
  - For each word $w_c \in \text{context}(i)$
    * Sample a context word $w_c \sim \text{Discrete}(\phi_{zi})$.

The inferred model can then be used to train embeddings for topics and words, although we do not consider this here. The MMSGTM admits a collapsed Gibbs sampler (CGS) which efficiently resolves the assignment. With Dirichlet priors on the parameters, the CGS update is

$$p(z_i = k | .) \propto N^{(\Phi)-i}_{w_i} + \alpha_k \times \frac{\prod_{c=1}^{\text{context}(i)} N^{(\Phi)-i}_{w_c} + \beta_{w_c} + N^{(i,c)}_{w_c}}{N^Z_{i} + \sum_w \beta_w + c - 1} ,$$

where $\alpha$ and $\beta$ are parameter vectors for Dirichlet priors over the topic and word distributions, $N^{\Phi}_{w_i}$ and $N^{\Phi}_{w_c}$ are input and output word-topic counts (excluding the current word), $N^Z$ is the total topic counts in output word-topic counts, and $N^{(i,c)}_{w_c}$ is the number of occurrences of word $w_c$ before the $c^{th}$ word in the $i^{th}$ context. MMSGTM exploits the MHW algorithm, which scales sub-linearly in $K$, but not in the number of training documents.

2.3 Metropolis-Hastings-Walker Sampler

The MHW method (Li et al., 2014), which is a key component of our approach, uses a data structure called an alias table which allows sampling from a discrete distribution in amortized $O(1)$ time. Assuming initial probabilities $p_0, p_1, ..., p_{l-1}$ of a distribution over $l$ outcomes and average of probabilities $a = \frac{1}{l}$, the alias table $A$ can be formed as follows (Marsiglia et al., 2004):

- Initialize: for $i$ from 0 to $l - 1$
  - $A_{\text{alias}}[i] = i$ and $A_{\text{prob}} = (i + 1)a$
- Do the following steps $n - 1$ times
  - Find smallest $p_i$ and largest $p_j$
  - Set $A_{\text{alias}}[i] = j$ and $A_{\text{prob}} = i \times a + p_i$
  - $p_j := p_j - (a - p_i)$ and $p_i := a$.

Then, to sample from $p$ using the alias table:

- Roll $l$-sided fair die to choose element $i$ of $A$
- If $\text{Rand}(1) < A_{\text{prob}}[i]$ return $i$, else return $A_{\text{alias}}[i]$.

Li et al. (2014) cache alias table samples, avoiding the need to store the table. Once the supply of samples is exhausted they compute a new alias table. They draw samples from the Gibbs sampling update, analogous to $\gamma_{ij}$ in Equation 1, in amortized $O(k_d)$ time by decomposing the update into

$$p(z_{ij} = k|.) \propto N^{\Phi}_{w_{ij}} + \beta_{w_{ij}} + \alpha_k N^{\Phi}_{w_{ij,k}} + \beta_{w_{ij}} N^{\Phi}_{w_{ij,k}} + \sum_w \beta_w ,$$

where the first term, sparse in $k_d$, is the number of occurrences of word $w_c$ before the $c^{th}$ word in the $i^{th}$ context. MMSGTM exploits the MHW algorithm, which scales sub-linearly in $K$, but not in the number of training documents.

3 Sparse Stochastic CVB0

In this section, we introduce our approach, a sparse version of SCVB0, which combines the individual benefits of the SVI, SSVI, SCVB0 and MHW algorithms, to scale well not only in the
number of documents but also in the number of topics, while gaining the benefits of collapsed inference. We refer to our method as the sparse stochastic collapsed variational Bayesian inference (SparseSCVB0) algorithm.

In SparseSCVB0, we obtain sparsity by substituting sparse Monte Carlo approximations \( \gamma^{(\text{pseudo})}_{ij} \) for the original SCVB0 variational distributions \( \gamma_{ij} \). The justification for this procedure, also used by (Mimno et al., 2012), is that the expected value of an average over one-hot samples from a distribution is equal to that distribution:

\[
E_{s_i \sim p(s)} \left[ \sum_{i=1}^{S} \delta_k(s_i) \right] = \frac{1}{S} \sum_{i=1}^{S} E_{s_i \sim p(s)} \left[ \delta_k(s_i) \right]
\]

\[
= \frac{1}{S} \sum_{i=1}^{S} \sum_{k'} \delta_k(s_i = k')p(s_i = k') = p(s = k).
\]

Thus, the overall procedure is still a valid stochastic optimization algorithm. We approximate the inner loop sampler in time sublinear in \( K \), constructing \( \gamma^{(\text{pseudo})}_{ij} \) by generating \( S \) samples from \( \gamma_{ij} \) using the MHW method. To describe the general form of our algorithm, we introduce SparseSCVB0 statistics: local (e.g. document-level) expected counts \( N^L \), global (corpus-level) expected counts \( N^G \), and total expected topic counts \( N^Z \). We approximate local sufficient statistics \( N^L \) for each token \( i \) in document \( j \) via:

\[
N^L_{ij} \approx C_j E_{s_i \sim p(s)} \left[ \sum_{k \in S} \delta_{i_z=k} \right],
\]

Since \( \gamma^{(\text{pseudo})}_{ij} \) is sparse, we can efficiently update these statistics using only its non-zero entries. This approach allows us to learn high-dimensional topic models efficiently on very large corpora. Before updating global parameters in a similar fashion, it may also be beneficial to perform a small number of burn-in passes to learn the local parameters \( N^L \) (Foulds et al., 2013). For large-scale datasets (e.g. Wikipedia), SparseSCVB0 operates in a “mini-epoch” approach where we process a large subset of the corpus (e.g. 5,000 documents) several (e.g. 3) times, before discarding and processing the next subset, and so on. This allows a “warm start” of \( N^L \) in repeating iterations, with a small memory overhead. Pseudo-code of SparseSCVB0 for a mini-epoch is provided in Algorithm 1, which we discuss more in the next two sections, including model-specific aspects.

<table>
<thead>
<tr>
<th>Algorithm 1 SparseSCVB0 for TM Inference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Randomly initialize ( N^G, N^L, N^Z := \sum_w N^G_w );</td>
</tr>
<tr>
<td>( \text{doSparse} = \text{true or } \text{false} )</td>
</tr>
<tr>
<td>for each minibatch ( M ) do</td>
</tr>
<tr>
<td>( \tilde{N}^G := 0; \tilde{N}^Z := 0 )</td>
</tr>
<tr>
<td>for each document ( j ) in ( M ) do</td>
</tr>
<tr>
<td>for each token ( i ) in ( j ) do</td>
</tr>
<tr>
<td>( \gamma_{ij}^{(\text{pseudo})} := 0 )</td>
</tr>
<tr>
<td>for each sample ( s ) in ( D ) do</td>
</tr>
<tr>
<td>draw ( z_{ij}^{(\text{new})} \sim q(k) )</td>
</tr>
<tr>
<td>(via efficient sampling or cached alias samples)</td>
</tr>
<tr>
<td>accept or reject ( z_{ij}^{(\text{new})} ) via Eq. 7</td>
</tr>
<tr>
<td>if (accept), ( z_{ij} := z_{ij}^{(\text{new})} ) via Eq. 7</td>
</tr>
<tr>
<td>( \gamma_{ij}^{(\text{pseudo})}[z_i] := \gamma_{ij}^{(\text{pseudo})}[z_i] + \frac{1}{S} )</td>
</tr>
<tr>
<td>end for</td>
</tr>
<tr>
<td>( N^L_{ij} := (1 - \rho^L_i) N^L_{ij} + \rho^L_i C_j \gamma_{ij}^{(\text{pseudo})} )</td>
</tr>
<tr>
<td>if (not burn-in pass),</td>
</tr>
<tr>
<td>update estimates for ( i ) or for each context word of ( i ):</td>
</tr>
<tr>
<td>( \tilde{N}^G_{w_i} := \tilde{N}^G_{w_i} + C_i \gamma_{ij}^{(\text{pseudo})} )</td>
</tr>
<tr>
<td>( \tilde{N}^Z := \tilde{N}^Z + \gamma_{ij}^{(\text{pseudo})} )</td>
</tr>
<tr>
<td>( \text{elseif(}\text{doSparse}), N^L_{ij}[k] &lt; \tau \rho^L_{E_{ij}} C_j := 0 )</td>
</tr>
<tr>
<td>end for</td>
</tr>
<tr>
<td>end for</td>
</tr>
<tr>
<td>update ( N^G := (1 - \rho^G_i) N^G + \rho^G_i \tilde{N}^G )</td>
</tr>
<tr>
<td>update ( N^Z := (1 - \rho^Z_i) N^Z + \rho^Z_i \tilde{N}^Z )</td>
</tr>
<tr>
<td>end for</td>
</tr>
</tbody>
</table>
gence in early iterations. We believe that this classification approximations actually improve convergence counter-intuitively, the Monte Carlo and sparsity.

Preliminary experiments, we found that, somewhat
stant for the last token of this document, and constant $0 < \tau \leq 1$ controls the sparsity. In our preliminary experiments, we found that, somewhat
counter-intuitively, the Monte Carlo and sparsification approximations actually improve convergence in early iterations. We believe that this is because they help SCVB0 escape the initial high-entropy regime, during which convergence of variational algorithms is poor (Salakhutdinov et al., 2003). This property makes the benefit of annealing insignificant, so we do not use simulated annealing for LDA inference, fixing $T_j = 1$.

An additional optimization of SparseSCVB0 for LDA inference can be performed by “clumping” (Teh et al., 2007; Foulds et al., 2013), where one update of the local parameters is performed for each distinct word type in each document. This is performed by fixing the variational distribution, and scaling the update by number of copies of the distinct word type in the document. If we observe the distinct word type $w_{aj}$, which occurs $m_{aj}$ times in document $j$, the update is

\[
N_j^L := (1 - \rho_j^L)^{m_{aj}} N_j^L + (1 - (1 - \rho_j^L)^{m_{aj}}) C_j \gamma_{aj}^{(pseudo)}.
\]

5 SparseSCVB0 for MMSGTM

The main contribution of our approach for the MMSGTM algorithm is to scale this algorithm in number of documents with online inference, as MMSGTM already scales sublinearly in $K$ using MHW. Foulds et al. (2018) use an MHW proposal which approximates the CGS update, interpreted as a product of experts (Hinton, 2002) in which each word in the context is an “expert” which weighs in multiplicatively on the update, with a mixture of experts. In the proposal, they draw a
context word $w_c$ uniformly from the context of the current word, $w_c \sim \text{Uniform}(\text{context}(w_i))$, and then sample a word based on the chosen context word’s contribution to the update:

$$q(k) \propto \frac{N_{w_i k}^G + \beta_{w_c}}{N_k^Z + \sum_w \beta_w}$$

(10)

where $N_{w_c}^G$ is analogous to the output context word-topic counts $N_{w_c}^\Phi$ of original MMSGTM model. The proposal samples via the alias method in amortized $O(1)$ time, instead of $O(k_d)$ time, since it does not involve the sparse term. We use this proposal to approximate the CVB0 update for the model, which is a deterministic version of Equation 5, neglecting to exclude the current assignment of $z_i$. We update $N_{w_i}^G$ (analogous to $N_{w_i}^\Phi$) for each current word $w_i$, locally, but update $N_{w_c}^G$ and $N^Z$ via minibatch counts $\tilde{N}_{w_c}^G$ and $\tilde{N}^Z$, respectively, for each context word $w_c$. Unlike for LDA, $C_j$ in the local updates represents the total number of input word $j$ in the corpus. As we draw multiple output words from each topic assignment, the effective temperature of the MMSGTM model is much lower than for standard LDA which may cause problems with mixing and leads it to get stuck in the initial regime. Following Foulds et al. (2018), we perform simulated annealing which varies the M-H acceptance ratio to improve mixing. We parameterize the temperature schedule as $T_j = T_0 + \lambda \kappa^j$, where $T_0$ is the target final temperature, $\kappa \leq 1$, constant $\mu$ controls the amount of temperature reduction after each document iteration $j$ and $\lambda$ controls the initial temperature.

6 Experiments

In this section we study the performance of our SparseSCVB0 ¹ algorithm, on small as well as large corpora to validate the proposed method for topic models such as LDA and MMSGTM, and to compare with other state-of-the-art algorithms.

6.1 Experimental Environment and Datasets

We compared SparseSCVB0 to SCVB0 and SVI. For a fair comparison, we implemented all of them in the fast high-level language Julia V0.6.2 (Bezanson et al., 2017). We conducted all experiments on a computer with 64GB memory and an Intel Xeon E5-2623 V4 processor with 2.60 GHz clock rate, 8 x 256KB L2 Cache and 10MB L3 Cache. As we only use one single thread for sampling across all experiments, only one CPU core is active throughout the experiment with only 256KB available L2 Cache.²

We used NIPS, Reuters-150, PubMed Central, and Wikipedia as representative very small, small, medium, and large-scale datasets, respectively. The NIPS corpus has 1740 scientific articles from years 1987-1999 with 2.3M tokens, due to Sam Roweis. The newswire corpus Reuters-150 contains 15, 500 articles with dictionary size of 8, 350 words. PubMed Central has 320M tokens across 165, 000 scientific articles and a vocabulary size of around 38, 500 words. The Wikipedia corpus contained 4.6 million articles from the online

¹Code implementing SparseSCVB0 can be found at https://github.com/dr97531/SparseSCVB0.

²Since SSVI relies on multiple complex implementation details, we were unable to develop a fair implementation, nor were we able to obtain source code for a previous implementation. We expect that its accuracy would be similar to SVI, with a speed-up at or below that bestowed by a MHW-based inner loop. (Mimno et al., 2012) apply it with only 200 topics for most of their experiments, and at most 1000 topics.

<table>
<thead>
<tr>
<th>Input word = learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top words in top 2 topics for the input word</td>
</tr>
<tr>
<td><strong>Table 4:</strong> Top words in the top 2 topics for an input word using original SCVB0 and SparseSCVB0 for MMSGTM.</td>
</tr>
</tbody>
</table>
Figure 2: Per-topic coherence for LDA when $K = 1,000$ on (a) NIPS, (b) PubMed, and (c) Wikipedia. SparseSCVB0 completely outperforms other models for large-scale corpus.

Table 5: Randomly selected topics from a 10,000-topic model trained using SparseSCVB0 on Wikipedia encyclopedia. We used the dictionary of 7,700 words which was extracted by Hoffman et al. (Hoffman et al., 2013). There were 811M tokens in the corpus.

6.2 Performance for LDA

We implement SparseSCVB0, original SCVB0 and SVI algorithms using the clumping optimization (Teh et al., 2007) technique. In all LDA experiments, each algorithm was trained using mini-batches of size 20 for the NIPS corpus and 100 for other corpora. For PubMed and Wikipedia, we chose mini-epoch subsets of size 5,000 documents and processed for 5 passes. We used a step-size schedule of $\frac{\text{scale}}{(t+1)^\beta}$ as in original SCVB0 for global parameters, where $t$ is the document iteration with $\text{scale} = 100.0$, $\eta = 1000.0$ and $\kappa = 0.9$. For document-level parameters, we used the $\text{scale} = 1.0$, $\eta = 10.0$ and $\kappa = 0.9$, with $t$ referring here to the word iteration of the current document. In case of PubMed corpus, we found out that original SCVB0 and SVI tend to stuck in the initial regime for document-level step-size parameter $\eta = 1.0$ which we later fixed by setting $\eta = 10.0$, while SparseSCVB0 didn’t suffer from this problem due to the extra randomness from the sparse sampled updates. Finally, we choose hyper-parameters $\alpha = 0.1$ and $\beta = 0.01$ and burn-in pass of 5 for each document in all LDA experiments. For SparseSCVB0, we used sample size $S = 5$ to approximate $\gamma^{(\text{pseudo})}$ and $\tau = 1/K$ for the sparsification heuristic on local parameters.

To study the acceleration benefits of our approach, we evaluated the runtime performance per
iteration on the number topics and the number of iterations. In Figure 1(a), SparseSCVB0 is compared to original SCVB0 in terms of the average runtime per document iteration as a function of the number topics. We see that original SCVB0 requires average linear runtime due to $O(K)$ operations to compute collapsed variational distribution, while the average runtime for SparseSCVB0 grows sublinearly in $K$, due to $O(k_d)$ operations instead of $O(K)$ operations. SparseSCVB0 starts with approximately $O(K)$ operations in its initial stage of iterations, but it starts getting a benefit from sparsification heuristic after burning in, as shown in Figure 1(b) for $K = 10,000$.

To evaluate the performance in terms of learned topic quality, we start by comparing all of the algorithms in qualitative experiments (see Table 1, Table 2, and Table 3) where we show randomly selected example topics, while all the models were trained on the NIPS, PubMed, and Wikipedia corpus for $K = 500$, $K = 1,000$, and 1,000, respectively. To get a quantitative insight we evaluated the topics using the per topic coherence metric, which measures the semantic quality of a topic based on the $W$ most probable words for the topics (Mimno et al., 2011), thereby approximating the user viewing experience. In Figure 2, we see that SparseSCVB0 generates better quality topics with higher coherence scores than the other two models for $K = 1000$ with $W = 10$ after running all the models on each corpus for the same amount of time. The coherence performance of SparseSCVB0 increases substantially in the case of the large-scale corpus (Figure 2(c)), since it gets the opportunity to use its runtime advantage and process more documents than the other algorithms.

To investigate model convergence, we measured the held-out log-probability versus wall-clock time for all the algorithms. For each experiment we held-out a set of documents (150 documents for NIPS, 3500 documents for Reuters, and 1000 documents for all other corpora) as test data.
Figure 4: Comparison of runtime per iteration for MMSGTM in terms of: (a) number of topics \( K \) and (b) number of iterations when \( K = 5,000 \). SparseSCVB0 runs in amortized \( O(1) \) time, while original SCVB0 is linear in \( K \).

Figure 5: Per-topic coherence for MMSGTM, \( K = 500 \), on (a) NIPS and (b) Wikipedia. SparseSCVB0 has higher coherence scores for both the small and large corpora.

and trained the model on the rest of the corpus. Then, we split each test document in half, estimated local parameters on first half and finally computed the log-likelihood of the remaining half of the document. Figure 3 shows the comparison of average log-likelihood versus wall-clock time for all four corpora. In terms of log-likelihood, SparseSCVB0 provides an approximately similar result to original SCVB0 for the small corpus, but it converged to a better solution than others in the case of large corpora like Wikipedia (see Figure 3(c)), likely due to its processing a larger number of documents.

SparseSCVB0 enables the large-scale computation needed to learn high-dimensional topic models that could not feasibly be trained using previous methods due to their runtime complexity in the number of documents and/or topics. We show randomly selected topics from the LDA model with \( K = 10,000 \) in Table 5. This big topic model was trained for 36 hours using SparseSCVB0 on Wikipedia. We performed a dense initialization, running original SCVB0 for the first 5 hours, which was found to help avoid local optima.

6.3 Performance for MMSGTM

We also conducted experiments to evaluate the performance of SparseSCVB0 for MMSGTM and compare with original SCVB0. In all MMSGTM experiments, we kept the same step size schedule for global parameters as \( \text{scale} = 1.0, \eta = 5.0 \) and \( \kappa = 0.9 \), but for local parameter updates we maintain a separate step-size schedule of \( \text{scale} (\gamma + t)^{-\iota} \) for each input word, with \( t \) referring to the number of times we processed this input word, while \( \eta \) and \( \kappa \) values remained the same. For simulated annealing of SparseSCVB0, we used \( T_0 = 0.00001, \kappa = 0.9, \mu = 5 \) and \( \lambda = |\text{context}| \) with a context size of 5. We kept the same number of document burn-in passes as we did for the LDA experiments.

In Figure 4, we show the runtime improvement of SparseSCVB0 over original SCVB0 for MMSGTM in a similar experiment to the one for LDA. For MMSGTM, SparseSCVB0 substantially outperforms original SCVB0 by processing each document in amortized \( O(1) \) time. We provide qualitative results in the case of MMSGTM model by showing several top words in the top 2 topics for an input word using original SCVB0 and SparseSCVB0 in Table 4 for \( K = 500 \). As for LDA, SparseSCVB0 allows us to generate topics with higher coherence scores compared to the original SCVB0 after running for the same amount of time (Figure 5) on both small and large corpora.

7 Conclusions

This paper introduced SparseSCVB0, a sparse version of the SCVB0 inference algorithm which performs fast, scalable high-dimensional topic model inference. SparseSCVB0 leverages stochasticity to scale well in both the corpus size and in the number of topics. It operates in the collapsed representation of topic models which leads to fast convergence while providing an improved variational bound. We show that SparseSCVB0 reduces the operational complexity for the variational Bayes update of online topic models from \( O(K) \) to \( O(k_d) \) time for LDA and amortized \( O(1) \) time for MMSGTM. We evaluated and compared the performance of our approach with state-of-the-art models such as original SCVB0 and SVI to demonstrate that SparseSCVB0 converges much more efficiently, while maintaining high quality topics with a better per-topic coherence score.
References


