A Practical Algorithm for Topic Modeling with Provable Guarantees

Sanjeev Arora ARORA@CS.PRINCETON.EDU
Rong Ge RONGGE@CS.PRINCETON.EDU
Yoni Halpern HALPERN@CS.NYU.EDU
David Mimno MIMNO@CS.PRINCETON.EDU
Ankur Moitra MOITRA@IAS.EDU
David Sontag DSONTAG@CS.NYU.EDU
Yichen Wu YICHENWU@PRINCETON.EDU
Michael Zhu MHZHU@PRINCETON.EDU

Abstract

Topic models provide a useful method for dimensionality reduction and exploratory data analysis in large text corpora. Most approaches to topic model learning have been based on a maximum likelihood objective. Efficient algorithms exist that attempt to approximate this objective, but they have no provable guarantees. Recently, algorithms have been introduced that provide provable bounds, but these algorithms are not practical because they are inefficient and not robust to violations of model assumptions. In this paper we present an algorithm for learning topic models that is both provable and practical. The algorithm produces results comparable to the best MCMC implementations while running orders of magnitude faster.

1. Introduction

Topic modeling is a popular method that learns thematic structure from large document collections without human supervision. The model is simple: documents are mixtures of topics, which are modeled as distributions over a vocabulary (Blei, 2012). Each word token is generated by selecting a topic from a document-specific distribution, and then selecting a specific word from that topic-specific distribution. Posterior inference over document-topic and topic-word distributions is intractable — in the worst case it is NP-hard even for just two topics (Arora et al., 2012b). As a result, researchers have used approximate inference techniques such as singular value decomposition (Deerwester et al., 1990), variational inference (Blei et al., 2003), and MCMC (Griffiths & Steyvers, 2004).

Recent work in theoretical computer science focuses on designing *provably* polynomial-time algorithms for topic modeling. These treat the topic modeling problem as one of *statistical recovery*: assuming the data was generated *perfectly* from the hypothesized model using an unknown set of parameter values, the goal is to recover the model parameters in polynomial time given a reasonable number of samples.

Arora et al. (2012b) present an algorithm that provably recovers the parameters of topic models provided that every topic contains at least one *anchor word* that has non-zero probability only in that topic. If a document contains this anchor word, then it is guaranteed that the corresponding topic is among the set of topics used to generate the document. The input for the algorithm is the second-order moment matrix of word-word co-occurrences. The algorithm proceeds in two steps: a selection step that finds anchor words for each topic, and a recovery step that reconstructs topic distributions given those anchor words.

Anandkumar et al. (2012) present a provable algorithm based on third-order moments that does not require separability, but, unlike the algorithm of Arora et al., assumes that topics are not correlated. Although standard topic models like LDA (Blei et al., 2003) assume that topic proportions in a document are uncorrelated, there is strong evidence that topics are dependent (Blei & Lafferty, 2007; Li & McCallum, 2007): economics and politics are more likely to co-occur than economics and cooking.

Both algorithms run in polynomial time, but the bounds that have been proven on their sample complexity are weak and their empirical runtime

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28. Copyright 2013 by the author(s).

performance is slow. It is also unclear how they perform if the data does not satisfy the modeling assumptions. Subsequent work improves efficiency: Bittorf et al. (2012) and Gillis (2012) reduce the number of linear programs to be solved in the anchor word selection step. Gillis & Vavasis (2012) use linear projection ideas to avoid linear programs entirely.

The contributions of the current paper are three-fold. First, we present a combinatorial anchor selection algorithm that does not require solving linear programs. So long as the separability assumption holds, we prove that this algorithm is stable in the presence of noise and thus has polynomial sample complexity, running in seconds on very large data sets. As the anchor selection step is not a bottleneck, we do not empirically compare similar algorithms that have appeared in independent work by other authors (though not with a similar proof of correctness, see Section 4 for details). Second, we focus on the recovery step of Arora et al. (2012b), which computes topic-word distributions using matrix inversions. This step, which *does* noticeably affect performance, is sensitive to noise (whether from sampling or from model lack-of-fit) and so has high sample complexity and poor results even on synthetic data. We present a simple probabilistic interpretation of the recovery step that replaces matrix inversion with gradient-based inference. Third, we present an empirical comparison between recovery-based algorithms and existing likelihood-based algorithms with respect to empirical sample complexity on synthetic distributions and performance of the algorithms on real-world document corpora. Our algorithm performs as well as collapsed Gibbs sampling on a variety of metrics, and runs at least an order of magnitude faster, and as much as fifty times faster on large datasets, allowing real-time analysis of large data streams.

Our algorithm inherits the provable guarantees of Arora et al. (2012a;b) and results in simple, practical implementations. We view this work as combining the best of two approaches to machine learning: the tractability of statistical recovery with the robustness of maximum likelihood estimation.

2. Background

We consider the learning problem for a class of admixture distributions that are frequently used for probabilistic topic models. Examples of such distributions include latent Dirichlet allocation (Blei et al., 2003), correlated topic models (Blei & Lafferty, 2007), and Pachinko allocation (Li & McCallum, 2007). We denote the number of words in the vocabulary by V and the number of topics by K. Associated with each topic k is a multinomial distribution over the words in the vocabulary, which we will denote as the column vector A_k of length V. Each of these topic models postulates a particular prior distribution τ over the topic distribution of a document. For example, in latent Dirichlet allocation (LDA) τ is a Dirichlet distribution, and for the correlated topic model τ is a logistic Normal distribution. The generative process for a document d begins by drawing the document's topic distribution $W_d \sim \tau$. Then, for each position i we sample a topic assignment $z_i \sim W_d$, and finally a word $w_i \sim A_{z_i}$.

We can combine the column vectors A_k for each of the K topics to obtain the word-topic matrix A of dimension $V \times K$. We can similarly combine the column vectors W_d for M documents to obtain the topic-document matrix W of dimension $K \times M$. We emphasize that W is unknown and stochastically generated: we can never expect to be able to recover it. The learning task that we consider is to find the word-topic matrix A. For the case when τ is Dirichlet (LDA), we also show how to learn hyperparameters of τ.

Maximum likelihood estimation of the word-topic distributions is NP-hard even for two topics (Arora et al., 2012b), and as a result researchers typically use approximate methods. The most popular approaches are variational expectation maximization (Blei et al., 2003), which optimizes a lower bound on the likelihood, and Markov chain Monte Carlo (McCallum, 2002), which asymptotically samples from the posterior distribution.

Arora et al. (2012b) present an algorithm that provably learns the parameters of a topic model given samples from the model, provided that the word-topic distributions are *separable* (Donoho & Stodden, 2003):

Definition 2.1. The word-topic matrix A is p-separable for $p > 0$ if for each topic k, there is some word i such that $A_{i,k} \geq p$ and $A_{i,k'} = 0$ for $k' \neq k$.

Such a word is called an *anchor word*: when it occurs in a document, it is a perfect indicator that the document is at least partially about the corresponding topic, since there is no other topic that could have generated the word. Suppose that each document is of length $D \geq 2$, and let $R = \mathbb{E}_\tau[WW^T]$ be the $K \times K$ topic-topic covariance matrix. Let α_k be the expected proportion of topic k in a document generated according to τ. The main result of Arora et al. (2012b) is:

Theorem 2.2. *There is a polynomial time algorithm that learns the parameters of a topic model if the number of documents is at least*

$$M = \max\left\{ O\left(\frac{\log V \cdot a^4 K^6}{\epsilon^2 p^6 \gamma^2 D}\right), O\left(\frac{\log K \cdot a^2 K^4}{\gamma^2}\right) \right\},$$

Algorithm 1. High Level Algorithm

```
Input: Textual corpus $\mathcal{D}$, Number of topics $K$, Tolerance parameters $\epsilon_a, \epsilon_b > 0$.
Output: Word-topic matrix $A$, topic-topic matrix $R$
  $Q \leftarrow$ Word Co-occurences($\mathcal{D}$)
  Form $\{\bar{Q}_1, \bar{Q}_2, ...\bar{Q}_V\}$, the normalized rows of $Q$.
  $\mathbf{S} \leftarrow$ FastAnchorWords($\{\bar{Q}_1, \bar{Q}_2, ...\bar{Q}_V\}$, $K$, $\epsilon_a$) (Algorithm 4)
  $A, R \leftarrow$ RecoverKL($Q, \mathbf{S}, \epsilon_b$) (Algorithm 3)
  return $A, R$
```

where p is defined above, γ is the condition number of R, and $a = \max_{k,k'} \alpha_k/\alpha_{k'}$. The algorithm learns the word-topic matrix A and the topic-topic covariance matrix R up to additive error ϵ.

The Arora et al. (2012b) algorithm is not practical—it must solve V linear programs to identify anchor words, which is inefficient, and uses matrix inversion to recover A and the parameters of τ, which is unstable and sensitive to noise—but it forms the basis of our improved method, Algorithm 1. Both anchor selection and recovery take as input the $V \times V$ matrix of word-word co-occurrence counts Q, whose construction is described in the supplementary material. Q is normalized so that the sum of all entries is 1.

Algorithm 2. Original Recover (Arora et al., 2012b)

```
Input: Matrix $Q$, Set of anchor words $\mathbf{S}$
Output: Matrices $A$,$R$
  Permute rows and columns of $Q$
  Compute $\vec{p}_{\mathbf{S}} = Q_{\mathbf{S}}\vec{1}$          (equals $DR\vec{1}$)
  Solve for $\vec{z}$: $Q_{\mathbf{S},\mathbf{S}}\vec{z} = \vec{p}_{\mathbf{S}}$          (Diag($\vec{z}$) equals $D^{-1}$)
  Solve for $A^T = (Q_{\mathbf{S},\mathbf{S}}\text{Diag}(\vec{z}))^{-1}Q_{\mathbf{S}}^T$
  Solve for $R = \text{Diag}(\vec{z})Q_{\mathbf{S},\mathbf{S}}\text{Diag}(\vec{z})$
  return $A, R$
```

Algorithm 3. RecoverKL

```
Input: Matrix $Q$, Set of anchor words $\mathbf{S}$, tolerance parameter $\epsilon$.
Output: Matrices $A$,$R$
  Normalize the rows of $Q$ to form $\bar{Q}$.
  Store the normalization constants $\vec{p}_w = Q\vec{1}$.
  $\bar{Q}_{s_k}$ is the row of $\bar{Q}$ for the $k^{th}$ anchor word.
  for $i = 1, ..., V$ do
      Solve $C_{i\cdot} = \text{argmin}_{\vec{C}_i} D_{KL}(\bar{Q}_i || \sum_{k\in\mathbf{S}} C_{i,k}\bar{Q}_{s_k})$
      Subject to: $\sum_k C_{i,k} = 1$ and $C_{i,k} \geq 0$
      With tolerance: $\epsilon$
  end for
  $A' = \text{diag}(\vec{p}_w)C$
  Normalize the columns of $A'$ to form $A$.
  $R = A^{\dagger}QA^{\dagger T}$
  return $A, R$
```

3. Topic Recovery via Bayes' Rule

We introduce a new recovery method based on a probabilistic framework. The original recovery procedure (which we call "Recover") from Arora et al. (2012b) is as follows. First, it permutes the Q matrix so that the first K rows and columns correspond to the anchor words. We will use the notation $Q_{\mathbf{S}}$ to refer to the first K rows, and $Q_{\mathbf{S},\mathbf{S}}$ for the first K rows and just the first K columns. If constructed from infinitely many documents, Q would be the second-order moment matrix $Q = \mathbb{E}[AWW^TA^T] = A\mathbb{E}[WW^T]A^T = ARA^T$, with the following block structure:

$$Q = ARA^T = \begin{pmatrix} D \\ U \end{pmatrix} R \begin{pmatrix} D & U^T \end{pmatrix} = \begin{pmatrix} DRD & DRU^T \\ URD & URU^T \end{pmatrix}$$

where D is a diagonal matrix of size $K \times K$ whose rows correspond to anchor words. Next, it solves for A and R using the algebraic manipulations outlined in Algorithm 2.

For finite data, the matrix inversion in Algorithm 2 results in substantial imprecision. The returned A and R matrices can even contain small negative values, requiring a subsequent projection onto the simplex. As shown in Section 5, the original Recover algorithm performs poorly relative to a likelihood-based algorithm. One problem is that Recover uses only K rows of the matrix Q (the rows for the anchor words), whereas Q is of dimension $V \times V$. Although ignoring the remaining $V - K$ rows is sufficient for theoretical analysis, where the remaining rows contain no additional information, it is not sufficient for real data. Small sample sizes may also make estimates of co-occurrences between a word and the anchors inaccurate.

Here we adopt a new approach based on Bayes' rule. Consider any two words in a document w_1 and w_2, and let z_1 and z_2 refer to their topic assignments. We will use $A_{i,k}$ to index the matrix of word-topic distributions, i.e. $A_{i,k} = p(w_1 = i|z_1 = k) = p(w_2 = i|z_2 = k)$. Given infinite data, the elements of the Q matrix can be interpreted as $Q_{i,j} = p(w_1 = i, w_2 = j)$. The row-normalized Q matrix, denoted $\bar{Q}$, which plays a role in both finding the anchor words and the recovery step, can be interpreted as a conditional probability $\bar{Q}_{i,j} = p(w_2 = j|w_1 = i)$.

Denoting the indices of the anchor words as $\mathbf{S} = \{s_1, s_2, ..., s_K\}$, the rows indexed by elements of $\mathbf{S}$ are special in that every other row of $\bar{Q}$ lies in the convex hull of the rows indexed by the anchor words.

To see this, first note that for an anchor word s_k,

$$\bar{Q}_{s_k,j} = \sum_{k'} p(z_1 = k'|w_1 = s_k)p(w_2 = j|z_1 = k') \quad (1)$$
$$= p(w_2 = j|z_1 = k), \quad (2)$$

where (1) uses the fact that in an admixture model $w_2 \perp w_1 \mid z_1$, and (2) is because $p(z_1 = k|w_1 = s_k) = 1$. For any other word i, we have

$$\bar{Q}_{i,j} = \sum_k p(z_1 = k|w_1 = i)p(w_2 = j|z_1 = k).$$

Denoting the probability $p(z_1 = k|w_1 = i)$ as $C_{i,k}$, we have $\bar{Q}_{i,j} = \sum_k C_{i,k}\bar{Q}_{s_k,j}$. Since C is non-negative and $\sum_k C_{i,k} = 1$, we have that any row of $\bar{Q}$ lies in the convex hull of the rows corresponding to the anchor words. The mixing weights give us $p(z_1|w_1 = i)$. Using this together with $p(w_1 = i)$, we can recover the A matrix simply by using Bayes' rule:

$$p(w_1 = i|z_1 = k) = \frac{p(z_1 = k|w_1 = i)p(w_1 = i)}{\sum_{i'} p(z_1 = k|w_1 = i')p(w_1 = i')}.$$

Finally, we observe that $p(w_1 = i)$ is easy to solve for since $\sum_j Q_{i,j} = \sum_j p(w_1 = i, w_2 = j) = p(w_1 = i)$.

Our new algorithm finds, for each row of the empirical row normalized co-occurrence matrix, $\bar{Q}_i$, the vector of *non-negative* coefficients $p(z_1|w_1 = i)$ that best reconstruct it as a convex combination of the rows that correspond to anchor words. Here "best" will be measured using an objective function that allows this step to be solved quickly and in parallel (independently) for each word using the exponentiated gradient algorithm. Once we have $p(z_1|w_1)$, we recover the A matrix using Bayes' rule. The full algorithm using KL divergence as an objective is found in Algorithm 3. Further details of the exponentiated gradient algorithm are given in the supplementary material.

We use KL divergence as the measure of reconstruction error because it lets the recovery procedure be understood as maximum likelihood estimation. In particular, the recovery procedure can be shown to find the parameters $p(w_1)$, $p(z_1|w_1)$, $p(w_2|z_1)$ that maximize the likelihood of the word co-occurence counts (*not* the documents). However, the optimization problem does not explicitly constrain the parameters to correspond to an admixture model.

We define a similar algorithm using quadratic loss, which we call RecoverL2. Remarkably, the running time of this algorithm can be made independent of V. The objective can be re-written in kernelized form as

$$||\bar{Q}_i - C_i^T\bar{Q}_{\mathbf{S}}||^2 = ||\bar{Q}_i||^2 - 2C_i(\bar{Q}_{\mathbf{S}}\bar{Q}_i^T) + C_i^T(\bar{Q}_{\mathbf{S}}\bar{Q}_{\mathbf{S}}^T)C_i,$$

where $\bar{Q}_{\mathbf{S}}\bar{Q}_{\mathbf{S}}^T$ is $K \times K$ and can be computed once and used for all words, and $\bar{Q}_{\mathbf{S}}\bar{Q}_i^T$ is $K \times 1$ and can be computed once prior to running the exponentiated gradient algorithm for word i.

To recover the R matrix for an admixture model, recall that $Q = ARA^T$. We can find a least-squares approximation to R by pre- and post-multiplying Q by the pseudo-inverse $A^\dagger$. For the special case of LDA we can learn the Dirichlet hyperparameters. Recall that in applying Bayes' rule we calculated $p(z_1) = \sum_{i'} p(z_1|w_1 = i')p(w_1 = i')$. These values for $p(z)$ specify the Dirichlet hyperparameters up to a constant scaling. This constant could be recovered from the R matrix (Arora et al., 2012b), but in practice we find it is better to choose it using a grid search to maximize the likelihood of the training data.

We will see in Section 5 that our non-negative recovery algorithm performs much better on a wide range of performance metrics than the original Recover. In the supplementary material we show that the non-negative recovery algorithm also inherits the theoretical guarantees of Arora et al. (2012b): given polynomially many documents, it returns an estimate $\hat{A}$ at most ϵ from the true word-topic matrix A.

4. Efficiently Finding Anchor Words

We next consider the *anchor selection* step of the algorithm, in which our goal is to find the anchor words. In the *infinite data* case where we have infinitely many documents, the convex hull of the rows in $\bar{Q}$ will be a simplex where the vertices of this simplex correspond to the anchor words. Given a set of V points $a_1, a_2, ...a_V$ whose convex hull P is a simplex, we wish to find the vertices of P.[1] When we have a finite number of documents, the rows of $\bar{Q}$ are only an approximation to their expectation: we are given a set of points $d_1, d_2, ...d_V$ that are each a perturbation of $a_1, a_2, ...a_V$ whose convex hull P defines a simplex. We would like to find an approximation to the vertices of P. The anchor word selection algorithm from Arora et al. (2012a) tests whether or not each of the V points is a vertex of the convex hull by solving a linear program for each word. In this section we describe a purely combinatorial algorithm for this task that avoids linear programming altogether. The new "FastAnchorWords" algorithm is given in Algorithm 4.

Our approach is to iteratively find the farthest point from the subspace spanned by the anchor words found so far. The main technical step is to show that if

[1]See Arora et al. (2012a) and Arora et al. (2012b) for more details.

Algorithm 4. FastAnchorWords

Input: V points $\{d_1, d_2, ...d_V\}$ in V dimensions, almost in a simplex with K vertices and $\epsilon > 0$
Output: K points that are close to the vertices of the simplex.

```
Project the points $d_i$ to a randomly chosen $4 \log V/\epsilon^2$ dimensional subspace
$S \leftarrow \{d_i\}$ s.t. $d_i$ is the farthest point from the origin.
for $i = 1$ TO $K - 1$ do
    Let $d_j$ be the point in $\{d_1, \ldots, d_V\}$ that has the largest distance to span($S$).
    $S \leftarrow S \cup \{d_j\}$.
end for
$S = \{v'_1, v'_2, ...v'_K\}$.
for $i = 1$ TO $K$ do
    Let $d_j$ be the point that has the largest distance to span($\{v'_1, v'_2, ..., v'_K\} \backslash \{v'_i\}$)
    Update $v'_i$ to $d_j$
end for
Return $\{v'_1, v'_2, ..., v'_K\}$.
```

Notation: span(S) denotes the subspace spanned by the points in the set S. We compute the distance from a point x to the subspace span(S) by computing the norm of the projection of x onto the orthogonal complement of span(S).

one has already found r points S that are close to r (distinct) anchor words, then the point that is farthest from span(S) will also be close to a (new) anchor word. Thus the procedure terminates with one point close to each anchor word, even in the presence of noise. A practical advantage of this procedure is that when faced with many choices for a next anchor word to find, our algorithm tends to find the one that is *most different* from the ones we have found so far. The algorithm terminates when it has found K anchors. K is a tunable parameter of the overall algorithm that determines how many topics are fitted to the dataset.

To precisely state the guarantees of our algorithm, we use the following definition of robustness, which tries to formalize the intuition that topics should be fairly "distinct," meaning that none lies close to the affine hull of the rest.

Definition 4.1. A simplex P is γ-robust if for every vertex v of P, the ℓ_2 distance between v and the affine hull of the rest of the vertices is at least γ.

Remark: In the overall picture, the robustness of the polytope P spanned by the given points is related to the (unknown) parameters of the topic model. For example, in LDA, this γ is related to the largest ratio of any pair of hyper-parameters in the model.

Our goal is to find a set of points that are close to the vertices of the simplex, defined as follows:

Definition 4.2. Let $a_1, a_2, ...a_V$ be a set of points whose convex hull P is a simplex with vertices $v_1, v_2, ...v_K$. Then we say a_i *ϵ-covers* v_j if, whenever a_i is written as a convex combination of the vertices, $a_i = \sum_j c_j v_j$, then $c_j \geq 1 - \epsilon$. Furthermore we say that a set of K points ϵ-covers the vertices if each vertex is ϵ covered by some point in the set.

Suppose there is a set of points $\mathcal{A} = a_1, a_2, ...a_V$ whose convex hull P is γ-robust and has vertices $v_1, v_2, ...v_K$ (which appear in $\mathcal{A}$) and that we are given a perturbation $d_1, d_2, ...d_V$ of the points so that for each i, $\|a_i - d_i\| \leq \epsilon$.

Theorem 4.3. *Under the conditions stated in the previous paragraph, and if $\epsilon < \gamma^3/20K$, then there is a combinatorial algorithm that given $\{d_1, \ldots, d_V\}$ runs in time $\tilde{O}(V^2 + VK/\epsilon^2)$[2] and outputs a subset of $\{d_1, \ldots, d_V\}$ of size K that $O(\epsilon/\gamma)$-covers the vertices.*

We note that variants of this algorithm have appeared in other contexts. Our analysis rests on the following lemmas, whose proof we defer to the supplementary material. Suppose the algorithm has found a set S of k points that are each δ-close to distinct vertices in $\{v_1, v_2, ..., v_K\}$ and that $\delta < \gamma/20K$.

Lemma 4.4. *The point d_j found by the algorithm must be $\delta = O(\epsilon/\gamma^2)$ close to some vertex v_i. In particular, the corresponding a_j $O(\epsilon/\gamma^2)$-covers v_i.*

This lemma is used to show that the error does not accumulate too badly in our algorithm, since δ only depends on ϵ, γ (not on the δ used in the previous step of the algorithm). This prevents the error from accumulating exponentially in the dimension of the problem, which would be catastrophic for our proof.

After running the first phase of our algorithm, we run a cleanup phase (the second loop in Alg. 4) that can reduce the error in our algorithm.

Lemma 4.5. *Suppose $|S| = K-1$ and each point in S is $\delta = O(\epsilon/\gamma^2) < \gamma/20K$ close to distinct vertices v_i's, the farthest point found by the algorithm is d_j, then the corresponding a_j $O(\epsilon/\gamma)$-covers the remaining vertex.*

This algorithm can be seen as a greedy approach to maximizing the volume of the simplex. This view suggests other potential approaches for finding anchor

[2]To achieve this running time, we need to use (Ailon & Chazelle, 2009) to project the points into $\tilde{O}(1/\epsilon^2)$ dimensional subspace. In practice we find setting dimension to 1000 works well. The running time is then $O(V^2 + 1000VK)$.

words (where the goal is better solution quality rather than run time), but that study is left as an open question for future work.

Related work. The separability assumption has been previously studied under the name "pure pixel assumption" in the context of hyperspectral unmixing. A number of algorithms have been proposed that overlap with ours including VCA (Nascimento & Dias, 2004), which differs in that there is no clean-up phase, and N-FINDR (Gomez et al., 2007) which attempts to greedily maximize the volume of a simplex whose vertices are data points by changing one vertex at a time. However these algorithms have only been proven to work in the infinite data case, and for our algorithm we are able to give provable guarantees even when the data points are perturbed (e.g., as the result of sampling noise). Recent work by Thurau et al. (2010), Kumar et al. (2012) and Gillis & Vavasis (2012) designs algorithms for non-negative matrix factorization under the separability assumption. As mentioned above, we do not focus empirical comparisons on anchor-word selection variants as this step is not a bottleneck in our overall algorithm (see Section 5.2).

5. Experimental Results

We compare three parameter recovery methods, Recover (Arora et al., 2012b), RecoverKL, and RecoverL2, with a Gibbs sampling implementation (McCallum, 2002).[3] Linear programming-based anchor word finding is too slow to be comparable, so we use FastAnchorWords for all three recovery algorithms. Using Gibbs sampling we obtain the word-topic distributions by averaging over 10 saved states, each separated by 100 iterations, after 1000 burn-in iterations.

5.1. Methodology

We train models on two synthetic data sets to evaluate performance when model assumptions are correct, and on real documents to evaluate real-world performance. To ensure that synthetic documents resemble the dimensionality and sparsity characteristics of real data, we generate *semi-synthetic* corpora. For each real corpus, we train a model using MCMC and then generate new documents using the parameters of that model (these parameters are *not* guaranteed to be separable; we found that about 80% of topics fitted by MCMC had anchor words).

[3]We were not able to obtain Anandkumar et al. (2012)'s implementation of their algorithm, and our own implementation is too slow to be practical. We found that MALLET's Gibbs sampling is faster than lda-c, a popular variational method.

We use two real-world data sets, a large corpus of **New York Times** articles (295k documents, vocabulary size 15k, mean document length 298) and a small corpus of **NIPS** abstracts (1100 documents, vocabulary size 2500, mean length 68). Vocabularies were pruned with document frequency cutoffs. We generate semi-synthetic corpora of various sizes from models trained with $K = 100$ from NY Times and NIPS, with document lengths set to 300 and 70, respectively, and with document-topic distributions drawn from a Dirichlet with symmetric hyperparameters 0.03.

We use a variety of metrics to evaluate the learned models. For the semi-synthetic corpora, we compute the **reconstruction error** between the true word-topic distributions and the learned distributions. In particular, given a learned matrix $\hat{A}$ and the true matrix A, we use bipartite matching to align topics, and then evaluate the ℓ_1 distance between each pair of topics. When true parameters are not available, a standard evaluation for topic models is to compute **held-out probability**, the probability of previously unseen documents under the learned model. This computation is intractable in general, but there are reliable approximations (Wallach et al., 2009; Buntine, 2009).

Topic models are useful because they provide interpretable latent dimensions. We can evaluate the **semantic quality** of individual topics using a metric called *Coherence* (Mimno et al., 2011). This metric has been shown to correlate well with human judgments of topic quality. If we perfectly reconstruct topics, all the high-probability words in a topic should co-occur frequently, otherwise, the model may be mixing unrelated concepts. Given a set of words $\mathcal{W}$, coherence is

$$Coherence(\mathcal{W}) = \sum_{w_1, w_2 \in \mathcal{W}} \log \frac{D(w_1, w_2) + \epsilon}{D(w_2)}, \quad (3)$$

where $D(w)$ and $D(w_1, w_2)$ are the number of documents with at least one instance of w, and of w_1 and w_2, respectively. We set $\epsilon = 0.01$ to avoid taking the log of zero for words that never co-occur (Stevens et al., 2012). Coherence measures the quality of individual topics, but does not measure redundancy, so we measure **inter-topic similarity**. For each topic, we gather the set of the N most probable words. We then count how many of those words do not appear in any other topic's set of N most probable words. Some overlap is expected due to semantic ambiguity, but lower numbers of unique words indicate less useful models.

5.2. Efficiency

The Recover algorithms, in Python, are faster than a heavily optimized Java Gibbs sampling implementa-

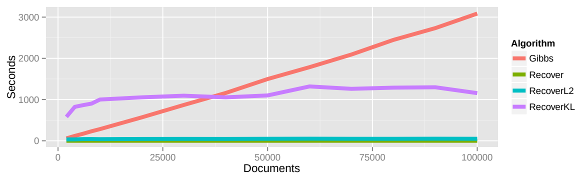

Figure 1. Training time on synthetic NIPS documents.

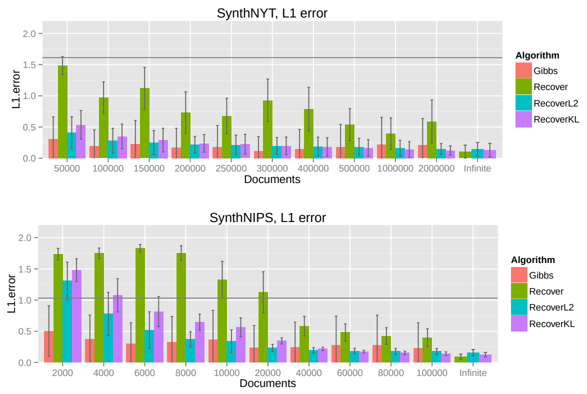

Figure 2. ℓ_1 error for learning semi-synthetic LDA models with $K = 100$ topics (**top**: based on NY Times, **bottom**: based on NIPS abstracts). The horizontal lines indicate the ℓ_1 error of K uniform distributions.

tion (Yao et al., 2009). Fig. 1 shows the time to train models on synthetic corpora on a single machine. Gibbs sampling is linear in the corpus size. RecoverL2 is also linear ($\rho = 0.79$), but only varies from 33 to 50 seconds. Estimating Q is linear, but takes only 7 seconds for the largest corpus. FastAnchorWords takes less than 6 seconds for all corpora.

5.3. Semi-synthetic documents

The new algorithms have good ℓ_1 reconstruction error on semi-synthetic documents, especially for larger corpora. Results for semi-synthetic corpora drawn from topics trained on NY Times articles are shown in Fig. 2 (top) for corpus sizes ranging from 50k to 2M synthetic documents. In addition, we report results for the three Recover algorithms on "infinite data," that is, the true Q matrix from the model used to generate the documents. Error bars show variation between topics. Recover performs poorly in all but the noiseless, infinite data setting. Gibbs sampling has the lowest ℓ_1 on smaller corpora. However, for the larger corpora the new RecoverL2 and RecoverKL algorithms have the lowest ℓ_1 error and smaller variance (running sampling longer may reduce MCMC error further). Results for semi-synthetic corpora drawn from NIPS topics are shown in Fig. 2 (bottom), and are similar.

Effect of separability. Notice that in Fig. 2, Recover does not achieve zero ℓ_1 error even with noiseless "infinite" data. Here we show that this is due to lack of separability, and that the new recovery algorithms are more robust to violations of the separability assumption. In our semi-synthetic corpora, documents are generated from an LDA model, but the topic-word distributions are learned from data and may not satisfy the anchor words assumption. We now add a synthetic anchor word to each topic that is, by construction, unique to that topic. We assign the synthetic anchor word a probability equal to the most probable word in the original topic. This causes the distribution to sum to greater than 1.0, so we renormalize. Results are shown in Fig. 3. The ℓ_1 error goes to zero for Recover, and close to zero for RecoverKL and RecoverL2 (not zero because we do not solve to perfect optimality).

Effect of correlation. The theoretical guarantees of the new algorithms apply even if topics are correlated. To test the empirical performance in the presence of correlation, we generated new synthetic corpora from the same $K = 100$ model trained on NY Times articles. Instead of a symmetric Dirichlet distribution, we use a logistic Normal distribution with a block-structured covariance matrix. We partition topics into 10 groups. For each pair of topics in a group, we add a non-zero off-diagonal element (ρ) to the covariance matrix. This block structure is not necessarily realistic, but shows the effect of correlation. Results for $\rho = 0.05$ and 0.1 are shown in Fig. 4. Recover performs much worse with correlated topics than with LDA-generated corpora (c.f. Fig. 2). The other three algorithms, especially Gibbs sampling, are more robust to correlation. Performance consistently degrades as correlation increases. For the recovery algorithms this is due to a decrease in γ, the condition number of the R matrix. With infinite data, ℓ_1 error is equal to the ℓ_1 error in the uncorrelated synthetic corpus (non-zero because of violations of the separability assumption).

5.4. Real documents

The new algorithms produce comparable quantitative and qualitative results on real data. Fig. 5 shows three metrics for both corpora. Error bars show the distribution of log probabilities across held-out *documents* (top

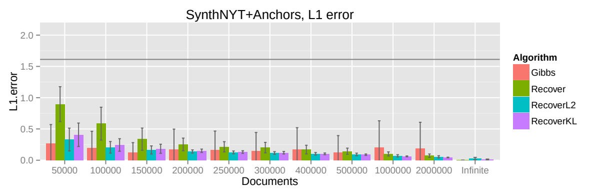

Figure 3. When we add artificial anchor words before generating synthetic documents, ℓ_1 error goes to zero for Recover and close to zero for RecoverKL and RecoverL2.

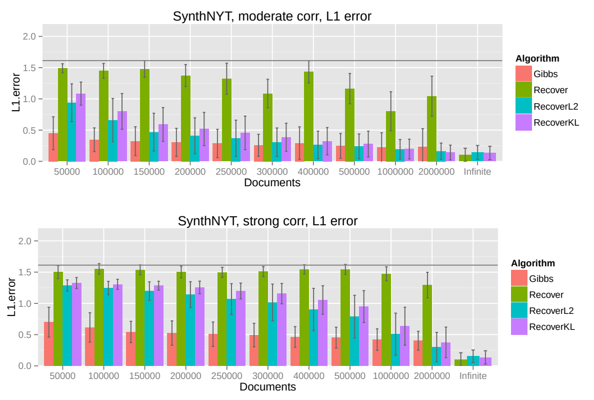

Figure 4. ℓ_1 error increases as we increase topic correlation (**top**: $\rho = 0.05$, **bottom**: $\rho = 0.1$). Based on the NY Times semi-synthetic model with 100 topics.

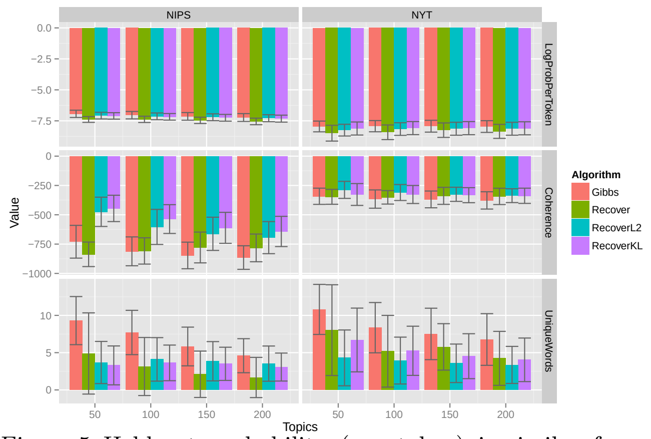

Figure 5. Held-out probability (per token) is similar for RecoverKL, RecoverL2, and Gibbs sampling. RecoverKL and RecoverL2 have better coherence, but fewer unique words than Gibbs. (Up is better for all three metrics.)

panel) and coherence and unique words across *topics* (center and bottom panels). Held-out sets are 230 documents for NIPS and 59k for NY Times. For the small NIPS corpus we average over 5 non-overlapping train/test splits. The matrix inversion step in Recover fails for the NIPS corpus so we modify the procedure to use pseudoinverse. This modification is described in the supplementary materials. In both corpora, Recover produces noticeably worse held-out log probability per token than the other algorithms. Gibbs sampling produces the best average held-out probability ($p < 0.0001$ under a paired t-test), but the difference is within the range of variability between documents. We tried several methods for estimating hyperparameters, but the observed differences did not change the relative performance of algorithms. Gibbs sampling has worse coherence than the Recover algorithms, but produces more unique words per topic. These patterns are consistent with semi-synthetic results for similarly sized corpora (details are in supplementary material).

For each NY Times topic learned by RecoverL2 we find the closest Gibbs topic by ℓ_1 distance. The closest, median, and farthest topic pairs are shown in Table 1. We observe that when there is a difference, recover-based topics tend to have more specific words (*Anaheim Angels* vs. *pitch*).

Table 1. Example topic pairs from NY Times (closest ℓ_1), anchor words in bold. The UCI NY Times corpus includes named-entity annotations, indicated by the *zzz* prefix. All 100 topics are shown in the supplementary material.

RecoverL2	run inning game hit season zzz_anaheim_angel
Gibbs	run inning hit game ball pitch
RecoverL2	father family **zzz_elian** boy court zzz_miami
Gibbs	zzz_cuba zzz_miami cuban zzz_elian boy protest
RecoverL2	**file** sport read internet email zzz_los_angeles
Gibbs	web site com www mail zzz_internet

6. Conclusions

We present new algorithms for topic modeling, inspired by Arora et al. (2012b), which are efficient and simple to implement yet maintain provable guarantees. The running time of these algorithms is effectively independent of the size of the corpus. Empirical results suggest that the sample complexity of these algorithms is somewhat greater than MCMC, but, particularly for the ℓ_2 variant, they provide comparable results in a fraction of the time. We have tried to use the output of our algorithms as initialization for further optimization (e.g. using MCMC) but have not yet found a hybrid that out-performs either method by itself. Finally, although we defer parallel implementations to future work, these algorithms are parallelizable, potentially supporting web-scale topic inference.

Acknowledgments

We thank Percy Liang for suggesting the use of semi-synthetic corpora, and David Blei for helpful comments. This work is partially supported by NSF grants CCF-0832797, CCF-1117309, and DMS-0835373, a Google Faculty Research Award, CIMIT award 12-1262, and grant UL1 TR000038 from NCATS, NIH. David Mimno and Ankur Moitra were supported by NSF Computing Innovation Fellowships, and Yoni Halpern by an NSERC Postgraduate Scholarship.

References

Ailon, N. and Chazelle, B. Approximate nearest neighbors and the fast johnson-lindenstrauss transform. *SICOMP*, pp. 302–322, 2009. 2

Anandkumar, A., Foster, D., Hsu, D., Kakade, S., and Liu, Y. Two svds suffice: Spectral decompositions for probabilistic topic modeling and latent dirichlet allocation. In *NIPS*, 2012. 1, 3

Arora, S., Ge, R., Kannan, R., and Moitra, A. Computing a nonnegative matrix factorization – provably. In *STOC*, pp. 145–162, 2012a. 1, 4, 1

Arora, S., Ge, R., and Moitra, A. Learning topic models – going beyond svd. In *FOCS*, 2012b. 1, 2, 2, 2, 3, 2, 3, 1, 5, 6

Bittorf, V., Recht, B., Re, C., and Tropp, J. Factoring nonnegative matrices with linear programs. In *NIPS*, 2012. 1

Blei, D. Introduction to probabilistic topic models. *Communications of the ACM*, pp. 77–84, 2012. 1

Blei, D. and Lafferty, J. A correlated topic model of science. *Annals of Applied Statistics*, pp. 17–35, 2007. 1, 2

Blei, D., Ng, A., and Jordan, M. Latent dirichlet allocation. *Journal of Machine Learning Research*, pp. 993–1022, 2003. Preliminary version in *NIPS* 2001. 1, 2

Buntine, Wray L. Estimating likelihoods for topic models. In *Asian Conference on Machine Learning*, 2009. 5.1

Deerwester, S., Dumais, S., Landauer, T., Furnas, G., and Harshman, R. Indexing by latent semantic analysis. *JASIS*, pp. 391–407, 1990. 1

Donoho, D. and Stodden, V. When does non-negative matrix factorization give the correct decomposition into parts? In *NIPS*, 2003. 2

Gillis, N. Robustness analysis of hotttopixx, a linear programming model for factoring nonnegative matrices, 2012. http://arxiv.org/abs/1211.6687. 1

Gillis, N. and Vavasis, S. Fast and robust recursive algorithms for separable nonnegative matrix factorization, 2012. http://arxiv.org/abs/1208.1237. 1, 4

Gomez, C., Borgne, H. Le, Allemand, P., Delacourt, C., and Ledru, P. N-findr method versus independent component analysis for lithological identification in hyperspectral imagery. *Int. J. Remote Sens.*, 28(23), January 2007. 4

Griffiths, T. L. and Steyvers, M. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101: 5228–5235, 2004. 1

Kumar, A., Sindhwani, V., and Kambadur, P. Fast conical hull algorithms for near-separable non-negative matrix factorization. 2012. http://arxiv.org/abs/1210.1190v1. 4

Li, W. and McCallum, A. Pachinko allocation: Dag-structured mixture models of topic correlations. In *ICML*, pp. 633–640, 2007. 1, 2

McCallum, A.K. Mallet: A machine learning for language toolkit, 2002. http://mallet.cs.umass.edu. 2, 5

Mimno, David, Wallach, Hanna, Talley, Edmund, Leenders, Miriam, and McCallum, Andrew. Optimizing semantic coherence in topic models. In *EMNLP*, 2011. 5.1

Nascimento, J.M. P. and Dias, J. M. B. Vertex component analysis: A fast algorithm to unmix hyperspectral data. *IEEE TRANS. GEOSCI. REM. SENS*, 43:898–910, 2004. 4

Stevens, Keith, Kegelmeyer, Philip, Andrzejewski, David, and Buttler, David. Exploring topic coherence over many models and many topics. In *EMNLP*, 2012. 5.1

Thurau, C., Kersting, K., and Bauckhage, C. Yes we can simplex volume maximization for descriptive web–scale matrix factorization. In *CIKM–10*, 2010. 4

Wallach, Hanna, Murray, Iain, Salakhutdinov, Ruslan, and Mimno, David. Evaluation methods for topic models. In *ICML*, 2009. 5.1

Yao, Limin, Mimno, David, and McCallum, Andrew. Efficient methods for topic model inference on streaming document collections. In *KDD*, 2009. 5.2