Sparse Word Graphs:
A Scalable Algorithm for Capturing Word Correlations in Topic Models

Submitted for Blind Review

Abstract

Statistical topic models such as the Latent Dirichlet Allocation (LDA) have emerged as an attractive framework to model, visualize and summarize large document collections in a completely unsupervised fashion. One of the limitations of this family of models is their assumption of exchangeability of words within documents, which results in a ‘bag-of-words’ representation for documents as well as topics. As a consequence, precious information that exists in the form of correlations between words is lost in these models.

A bigram topic model [23] is a natural extension of LDA that relaxes the bag-of-words assumption, but one of its main bottlenecks is scalability, since it requires estimating \( V(V - 1) \) number of parameters for each topic, where \( V \) is the vocabulary size. Another problem with bigram models is that they do not consider long-distance correlations between words that may be very useful for summarization and topic visualization.

In this work, we adapt recent advances in sparse modeling techniques to the problem of modeling word correlations within topics. The new algorithm we present, called Sparse Word Graphs, works in two stages: first, the standard LDA model is run on the data to obtain topic assignments to all the words in all the documents. Next, we define a sparse Markov Random Field between words for each topic and learn the sparse structure through an \( l_1 \)-norm regularized conditional likelihood of the topic assignment data obtained from LDA.

Our experiments on AP corpus reveal both long-distance and short-distance word correlations within topics that are semantically very meaningful. In addition, the new algorithm is highly scalable to large collections as it captures only the most important correlations in a sparse manner.

1 Introduction

In the recent past, large scale electronic document collections have become ubiquitous. Representative examples include PubMed\(^1\), a collection of Medical documents, JS-\(TOR^2\), a large corpus of articles from major science and arts journals, Google’s searchable database of US patents\(^3\), archives of news web sites such as New York Times\(^4\), community based encyclopedias such as Wikipedia\(^5\), not to mention the gigantic world-wide web.

The sizes of these collections have reached such enormous proportions that it is no longer practical for humans to analyze, interpret or mine their contents in a reasonable time. It is becoming increasing evident that we need automated techniques to support information analysts in summarizing their contents and mining interesting patterns from these large-scale collections.

In the recent past, statistical topic modeling has become very popular as a completely unsupervised method to help summarize and visualize the contents of these collections [6, 11, 13, 4, 7, 23]. These models use simple surface features such as word occurrences within documents to reveal surprisingly meaningful semantic content of documents [5].

The basic version of this family of models is called Latent Dirichlet Allocation or LDA in short [6]. In this model, each document \( d \) is assumed to be generated from a \( K \)-component mixture model, where the mixing probabilities \( \theta_d \) for each document are governed by a global Dirichlet distribution with parameter \( \alpha \). Each of the words \( w_i \) in the document is generated by a repetitive process of sampling a mixture component \( z_i \) from \( \theta_d \) and then sampling the word itself from a multinomial distribution over the entire vocabulary \( \beta_{z_i} \), associated with the mixture component \( z_i \). The generative process and the corresponding graphical representation of LDA are shown in figure 1 and table 1 respectively. Given a collection of documents, it is possible to automatically infer topic multinomials \( \{ \beta_k \}_{k=1}^K \) and mixing probabilities \( \{ \theta_d \}_{d=1}^M \) in an entirely unsupervised fashion either using an approximate variational inference algorithm [6] or through Gibbs sampling [10].

Each of the component multinomial distributions \( \beta_k \) assigns a high probability to a specific set of words that are semantically coherent. Hence these distributions over the

\(^{2}\)http://www.jstor.org/
\(^{3}\)http://www.google.com/patents
\(^{4}\)http://pqasb.pqarchiver.com/nytimes/advancedsearch.html
\(^{5}\)http://www.wikipedia.org/
For each document $d = 1, \cdots, M$
- sample topic mixing proportions $\theta_d \sim \text{Dirichlet}(\cdot | \alpha)
- For each position $i = 1, \cdots, N_d$
  * sample topic $z_{di} \in \{1, \cdots, K\} \sim \text{Multinomial}(\cdot | \theta_d)$
  * sample word $w_{di} \in \{1, \cdots, V\} \sim \text{Multinomial}(\cdot | \beta_{z_{di}})$

Table 1. Generative process of LDA: $V$ is the vocabulary size and $K$ is the number of topics while $\alpha$ is the symmetric Dirichlet parameter and $\{\beta_1, \cdots, \beta_K\}$ are the multinomial topic parameters. The only input to the model is $K$, the number of topics. All the parameters can be estimated by approximate variational techniques or Gibbs sampling.

For each document $d = 1, \cdots, M$
- sample topic mixing proportions $\theta_d \sim \text{Dirichlet}(\cdot | \alpha)$
- For each position $i = 1, \cdots, N_d$
  * sample topic $z_{di} \in \{1, \cdots, K\} \sim \text{Multinomial}(\cdot | \theta_d)$
  * sample word $w_{di} \in \{1, \cdots, V\} \sim \text{Multinomial}(\cdot | \beta_{z_{di}})$

Figure 1. Graphical representation of LDA: each plates represents repeated sampling as many times as the number in the right-hand bottom corner of the plate.

vocabulary are referred to as topics and represent a concise summary of the contents of the document collection. A few representative topics learnt from the AP corpus are displayed in figure 2. In addition, the mixing probabilities $\theta_d$ of each document can be considered as the document’s representation in a low-dimensional semantic “topic-space”, allowing us to efficiently compute similarity between documents, thus aiding various information retrieval tasks [6].

In the recent past, there have been several extensions to LDA. Notable among them are the Dynamic Topic Model, [7] which models the evolution of topic content with time; HMM-LDA [11] in which semantic analysis of LDA is combined with the syntax analysis of HMMs; Pachinko Allocation, [13] that models a hierarchy of topics; Correlated Topic Model [4] which captures the correlations between topics by replacing the Dirichlet prior with a logistic-normal distribution; and finally the Dirichlet Process Mixture Model [3] which discovers the number of topics $K$ automatically.

Broadly speaking, the motivation behind all these models is twofold: to be able to model text accurately and be able to predict unseen data well, and to provide the user (in this case, probably an information analyst) a better visualization of topics. Despite their additional features in comparison to LDA, one component remains the same among all these models, namely, exchangeability of words. In other words, word occurrences are treated conditionally independent of each other, given their topics. In information retrieval parlance, this is referred to as modeling documents as “bags of words”. In doing so, much of the valuable information that exists in terms of correlations between words is lost. For example, a topic representation that simply assigns high weight to the words ‘white’ and ‘house’, while completely ignoring the phrase ‘white-house’ may not immediately reveal to the user that the topic is about the president of the United States. Apart from phrases, there could also be interesting long-distance correlations that occur between words. Capturing such relationships explicitly in the model would go a long way in better visualization and summarization of topics. The aim
of this work is to capture such correlations between words in topic models. We present a new scalable algorithm called Sparse Word Graphs that addresses this problem.

The rest of the paper is organized as follows. In section 2 we discuss the problems with bigram topic models in capturing word correlations and argue the case for algorithms that model correlations in a sparse manner. Then, we present the $l_1$ norm minimization technique used in sparse modeling of data. We also describe two published algorithms that discuss how to learn the sparse structure of a graphical model from data. In section 3, we describe the new Sparse Word Graphs algorithm that casts the word correlations problem as that of structure learning problem in Markov Random Fields and applies the $l_1$ norm minimization technique to solve the problem. Section 4 presents some of the sparse word graphs generated for a few representative topics and compares them with the LDA topic representation. In section 5, we compare and contrast the new algorithm with other word-correlation models and finally chart out directions for future work in section 6.

2 Past Work

Since the vocabulary of any document collection of reasonable size consists of thousands of words, it is a significant challenge to estimate the correlations between all pairs of words in each topic. Researchers have focused on modeling correlations between topics [4, 5] which is also very interesting, but is not as big a challenge since typically there are only a few hundreds of topics at most. There is very little work available in the literature on modeling word correlations within topics, which we believe is mainly owing to the complexity of the problem.

The only topic model we are aware of that attempts to go beyond the “bag of words” approach is the bigram topic model by Wallach [23]. In this model, each word is sampled conditioned on the previous word and the current topic. Accordingly, the model has $VK(V-1)$ parameters, where $V$ is the vocabulary size and $K$ is the number of parameters, as compared to $VK$ parameters of LDA. The quadratic complexity in vocabulary size may not appear too daunting until we look at some numbers: imagine a small document collection with a vocabulary of just 10,000 words. For a bigram model with 50 topics, the number of parameters would be roughly 5 billion. Thus, it is not very surprising that the author in [23] limited her experiments to very small collections consisting of not more than 2500 unique words.

Bigram models are very relevant in the context of language modeling for speech recognition where local word dependencies play a major role [18]. However, they may not be very appropriate in the context of topic modeling for two important reasons: firstly, bigram models capture only phrases and local dependencies such as “United States” and “White House”, while a more attractive model would be one that also reveals unexpected long range dependencies within topics, in addition to the traditional phrases. Secondly, for the purposes of topic visualization and summarization, it is enough to capture the most important correlations between words that help reveal the semantics of a topic. Explicit modeling of correlations between all pairs of words, as done in the bigram topic model, is simply overkill. Thus, a model that models both short-distance as well as long distance correlations between words in a sparse manner with fewer number of parameters, is more appropriate for topic modeling.

2.1 Enforcing sparsity using $l_1$-norm minimization

Regularization of optimization problems using a penalty function is a standard approach in machine learning. This is done either to improve generalizability or to encode prior information about the problem into the model. Recent research has found that regularizing an optimization problem by an $l_1$-norm penalty of the parameter vector leads to sparse solutions [21, 9, 17, 20].

The $l_1$-norm of a d-dimensional vector $u$, is defined as $\|u\|_1 = \sum_{i=1}^{d} |u_i|$. Figure 3 plots the $l_1$ norm of a 1-dimensional vector $u$. We also compare it with the more traditional square of the $l_2$ norm penalty, defined as $\|u\|_2^2 = \sum_{i=1}^{d} u_i^2$ in the same plot. The plot shows that for small values of $u < 1$, $\|u\|_1 \gg \|u\|_2^2$. Therefore, if one were to minimize the $l_1$-norm of a vector, it drives many of the small components to zeros, unlike the square of the $l_2$-norm penalty, which is more tolerating to small residuals [8]. Hence, one can enforce sparsity of a model in an optimization problem by using a regularization term that minimizes the $l_1$-norm of the parameter vector.
As a simple illustration, the problem of linear regression involves computing a least squares solution as shown below.

\[
\hat{\theta} = \arg \min_{\theta} n \sum_{i=1}^{n} (y_i - \theta^T x_i)^2
\] (1)

where \( \{ (y_i, x_i) \}_{i=1}^{n} \) are the input data, \( \theta \) is the parameter vector. A popular variant of this problem is ridge-regression, which regularizes the above optimization problem with the square of the \( l_2 \)-norm of the parameter vector in order to improve generalizability [2]:

\[
\hat{\theta} = \arg \min_{\theta} n \sum_{i=1}^{n} (y_i - \theta^T x_i)^2 + \rho \| \theta \|_2^2
\] (2)

where \( \rho > 0 \) is a weight associated with the regularization term. An equivalent lasso-regression problem can be formulated by replacing the \( l_2 \)-norm-square regularization term above with an \( l_1 \)-norm as shown below.

\[
\hat{\theta} = \arg \min_{\theta} n \sum_{i=1}^{n} (y_i - \theta^T x_i)^2 + \rho \| \theta \|_1
\] (3)

Lasso-regression defined in this manner results in sparse solutions for the vector \( \theta \) [20].

Although there are other norms that enforce sparsity, the attractiveness of the \( l_1 \)-norm lies in its convexity, permitting us to apply standard convex-optimization techniques to the problem [8].

In the remaining part of this section, we examine two pieces of work that show how to learn the sparse structure of graphical models using \( l_1 \)-regularized optimization problems.

### 2.1.1 Sparse Structure Learning in Gaussian Random Fields

If \( \mathbf{X} \) denotes a set of \( p \) random variables governed by a multi-variate Gaussian distribution given by \( \mathcal{N}(\mu, \Sigma) \), then the independence relations among the variables are governed by the zero pattern in the inverse covariance matrix \( \Sigma^{-1}. \) More precisely, if we form a graph \( G(\Sigma) = (\mathcal{V}, \mathcal{E}) \) with vertices \( \mathcal{V} \) corresponding to the random variables and edges \( \mathcal{E} \) satisfying \( s, t \in \mathcal{E} \) if and only if \( S_{st} \neq 0. \) If \( \mathcal{N}(s) \) denotes the neighbors of \( s \) in the graph, then the independence relation \( X_s \perp X_t \mid X_{\mathcal{V} \setminus s} \) holds for any vertex \( u \not\in \mathcal{N}(s). \)

Recent work of Meinshausen and Bühlmann [16] showed how to give an asymptotically correct estimator of the graph \( G(\Sigma) \) from data \( \{ x^{(i)} \}_{i=1}^{n} \) drawn from the unknown distribution \( \mathcal{N}(\mu, \Sigma). \) The strategy is to regress each variable \( X_s \) onto other variables, imposing an \( l_1 \) penalty on the parameters to encourage sparsity. The non-zero components then serve as an estimate of the neighbors of \( s \) in the graph. In more detail, the optimization problem for each vertex would be:

\[
\hat{\lambda}_s = \arg \min_{\lambda_s} \sum_{i=1}^{n} (x_s^{(i)} - \lambda_s^T x_{-s}^{(i)})^2 + \rho \| \lambda_s \|_1
\] (4)

where \( \lambda_s = (\lambda_{s1}, \cdots, \lambda_{sp}) \) are the parameters of the lasso fit and \( x_{-s} \) denotes the set of all variables with \( x_s \) replaced by 1 while \( \lambda_{-s} \) denotes the vector \( \lambda_s \) with the component \( \lambda_{ss} \) removed. The estimated set of neighbors is given by

\[
\hat{\mathcal{N}}(s) = \{ t : \hat{\lambda}_{st} \neq 0 \}
\] (5)

### Application to Topic Models

This scheme was adopted by Blei and Lafferty [5] in estimating a graph of sparse correlations between topics in their Correlated Topic Model (CTM) [4]. The main idea of the model is as follows: In LDA, the mixing proportions of all the topics \( \{ \theta_{dk} \}_{k=1}^{K} \), sampled from the Dirichlet distribution, are almost independent of each other. This is not realistic, since related topics such as ‘health’ and ‘diet’ or ‘science’ and ‘technology’ tend to exhibit strong correlations. The CTM is an extension of LDA in which the Dirichlet prior is replaced by a logistic normal distribution\(^6\) that allows correlations between topics.

The covariance matrix of the logistic Normal distribution models correlations between all pairs of topics in general. However, for the purposes of visualization, it is enough to capture only the most interesting correlations in a sparse manner. To accomplish this objective, one can easily apply the algorithm described above as follows: note that the natural parameters of the mixing proportions\(^7\) are governed by a Gaussian Random Field. One can therefore use the values of these parameters, estimated from a variational technique, as input data to the sparse structure learning problem. Blei and Lafferty [5] ran this algorithm on CTM to obtain some very meaningful sparse topic structure.

### 2.1.2 Sparse Structure Learning in Markov Random Fields

In a very related work, Wainwright et al [22] showed how to estimate a sparse graph structure of a discrete pairwise Markov Random Field (MRF) wherein, the neighborhood of any vertex in the graph is estimated by performing an \( l_1 \)-regularized logistic regression on the rest of the vertices. Their algorithm is as follows:

Let \( G = (\mathcal{V}, \mathcal{E}) \) with vertex set \( \mathcal{V} \) of size \( |\mathcal{V}| = p \) and edge set \( \mathcal{E}. \) Let \( \mathbf{X} = (X_1, \cdots, X_p) \) be a set of binary random variables associated with the vertices of the graph. Let

\(^6\)A Gaussian distribution whose domain \( \eta \in \mathbb{R}^p \) is mapped to a simplex by a logistic transformation \( \theta = \exp(\eta)/\exp(\sum_{i=1}^{p} \eta_i). \)

\(^7\)The true samples \( \eta \) from the Gaussian before the logistic tranformation.
the joint probability of the random variables be given by the Ising model as follows:

\[
P(x|\lambda) = \exp(\sum_{s \in V} \lambda_s x_s + \sum_{(s,t) \in E} \lambda_{st} x_s x_t - A(\lambda))
\]  

(6)

where the parameters \( \{\lambda_{st}\}_{(s,t) \in E} \) capture the correlation between the variables \( X_s \) and \( X_t \). \( A(\lambda) \) is the log-normalizing constant of the distribution. Given \( n \) samples \( \{x^{(1)}, \ldots, x^{(n)}\} \) such that each \( x^{(i)} \in \{0, 1\}^p \) drawn from an unknown distribution \( P(x|\lambda^*) \), the goal is to estimate the structure of the graph, that is to estimate \( \hat{E} \) such that \( \lim_{n \to \infty} P(\hat{E} = E) = 1 \).

The authors show that the following algorithm asymptotically converges to the true structure as the data size \( n \) increases: we maximize the \( l_1 \) regularized conditional likelihood of each variable \( X_s \) conditioned on all the other variables \( X_{-s} \). For a pairwise MRF with binary variables, this leads to the following \( l_1 \)-regularized logistic regression problem:

\[
\hat{\lambda}_s = \arg\max_{\lambda} \sum_{i=1}^n \log P(x^{(i)}|x^{-s}^{(i)}, \lambda_s) - \rho \|\lambda_{-s}\|_1
\]

\[
= \arg\max_{\lambda_s} \sum_{i=1}^n x^{(i)}_s \lambda_s^T x^{-s}_s - \log(1 + \exp(\lambda_s^T x^{-s}_s)) - \rho \|\lambda_{-s}\|_1
\]

(7)

where \( \lambda_s = (\lambda_{s1}, \cdots, \lambda_{sp}) \) are the parameters of the \( l_1 \) logistic regression and \( x^{-s} \) denotes the set of all variables with \( x_s \) replaced by 1 while \( \lambda_{-s} \) denotes the vector \( \lambda \) with the component \( \lambda_s \) removed. Similar to the case of Gaussian Random Fields, the estimated set of neighbors is given by:

\[
\hat{N}(s) = \{t : \hat{\lambda}_{st} \neq 0\}
\]

(8)

One could then define the set of edges \( E \) as a union or an intersection of neighborhood sets \( \{\hat{N}(s)\}_{s \in V} \) of all the vertices. Wainwright et al [22] showed that both definitions would converge to the true structure asymptotically.

In the next section, we will cast our problem as that of structure learning of a pairwise Markov Random field, which allows us to apply the algorithm of Wainwright et al [22] described above.

### 3 Sparse Word Graphs

In order to be able to define a binary pairwise MRF structure learning problem, we first convert the topic assignments generated by LDA into topic-specific binary data as follows:

As described in table 1, the Latent Dirichlet model assigns a topic to each word-occurrence in a document. Given a document collection, these latent topic assignments to all the words can be computed using a variational algorithm or Gibbs sampling. The starting point of our algorithm is the topic-assignment data generated by LDA. In LDA, each document \( d \) is represented as a vector of words \( w_d = \{w_{d1}, \ldots, w_{dN_d}\} \), where \( w_i \in \{1, \cdots, V\} \) is one of the \( V \) unique words in the vocabulary and \( N_d \) is the document length. LDA generates corresponding topic assignment vector \( z_d = (z_{d1}, \cdots, z_{dN_d}) \) where each \( z_i \in \{1, \cdots, K\} \) is one of \( K \) topics.

First, we convert the LDA topic-assignment vector \( z_d \) to a set of \( K \) binary vectors \( \{x^{(d)}_k\}_{k=1}^K \), where each \( x^{(d)}_k = (x^{(d)}_{k1}, \cdots, x^{(d)}_{KV}) \) is of length \( V \) as follows:

\[
x^{(d)}_{kv} = \begin{cases} 
1 & \text{if } z_j = k \text{ and } w_j = v \\
0 & \text{otherwise}
\end{cases}
\]

(9)

In other words, each variable \( x^{(d)}_{kv} \) associated with a word \( v \) for a topic \( k \) is assigned a value of 1 if the word occurs in the document and is assigned the topic \( k \) by LDA and 0 otherwise. Note that each word can occur multiple times in a document. Technically, LDA permits a different topic assignment to each occurrence of the word in the document. However it turns out that, in the maximum likelihood estimation setting, LDA assigns the same topic to all occurrences of a given word. Hence the assignment of \( x^{(d)}_{kv} \) in eqn. (9) is a well-defined function.

Now we assume that each topic \( k \) is associated with a random vector \( X_k = (X_{k1}, \cdots, X_{KV}) \) whose joint probability is defined under the following pairwise Markov Random Field defined as follows:

\[
P(x_k|\lambda_k) = \exp(\sum_{s \in V} \lambda_{ks} x_{ks} + \sum_{(s,t) \in E} \lambda_{kst} x_{ks} x_{kt} - A(\lambda_k))
\]

(10)

where the vertices \( V \) in the MRF correspond to the words in the vocabulary with an unknown sparse edge structure \( E \). The task is now to learn the sparse structure \( E \) of the topic-specific MRF using the data \( \{x^{(d)}_{k}\}_{d=1}^M \) where \( M \) is the number of documents in the collection. It is clear that one can directly apply the algorithm of Wainwright et al [22] described in section 2.1.2 to this problem.

We run \( V \) \( l_1 \) regularized logistic regression problems for each topic as shown in table 2. We define the set of edges as the union of all the neighborhoods of all vertices. As a heuristic, we also estimate the strength of the correlation between two words as the sum of the two parameter values obtained from the logistic regression problems corresponding to the two words, as shown in table 2. The algorithm is also illustrated in the form of a flow chart in figure 4.
For each topic $k \in \{1, \ldots, K\}$

- **Input:** LDA binary data $\{x_k^{(d)}\}_{d=1}^M$
- For each word $v \in \{1, \ldots, V\}$
  * Compute $\lambda_{kv} = \arg \max V \sum_{d=1}^M x_k^{(d)} \lambda_{kv} x_{k,v} - \log(1 + \exp(\lambda_{kv} x_{k,v})) - \rho \|\lambda_{k,v}\|_1$
  - For each word pair $(v, v')$ s.t. $\lambda_{k,v'} \neq 0$ or $\lambda_{k,v'} \neq 0$; $\phi_{k,v'v} = \lambda_{k,v'} + \lambda_{k,v'}$
- **Return:** Topic specific edge weights $\phi_k$

Table 2. Sparse Word Graphs Algorithm: $\lambda_{kv} = (\lambda_{k,v1}, \ldots, \lambda_{k,vV})$ is the parameter vector associated with the $l_1$ logistic regression problem of word $v$ in topic $k$, $x_{k,v}^{(d)}$ is the binary vector $x_{k,v}^{(d)}$ with $x_{k,v}^{(d)}(d) = 1$ and $\lambda_{k,v}$ is obtained by removing the component $\lambda_{k,v}$ from the vector $\lambda_{k,v}$.

Figure 4. Flow chart of the Sparse Word Graphs algorithm

3.1 Scalability

As described above, for each topic, we run $V$ $l_1$ regularized logistic regression problems, each of which is of size $V$. Hence it appears that the complexity of the problem is still $O(V^2K)$. This is technically true in terms of a loose upper-bound, but in practice, the new algorithm is very efficient in terms of both computational time as well as storage costs compared to a traditional bigram model for the following reasons:

1. Although the size of data vectors $x_k^{(d)}$ is $V$, the number of non-zero components is strictly upper-bounded by $N_{d_{max}}$, the maximum document length in the collection, which is typically much less than $V$. Thus, the input data to logistic regression is extremely sparse, making the learning very efficient.

2. A bigram model typically needs to estimate and store $KV(V - 1)$ parameters. The new algorithm estimates and stores only the edge weights of the sparse structure, which needs much less computation and smaller storage space in practice.

3. Since each of the problems is independent, it is possible to run the $V$ problems in parallel, resulting in a speed-up of computation.

4. Recent work by Koh et al [12] proposed a new, fast interior point solution for the $l_1$-regularized logistic regression problem, which makes it very scalable and practical for large dimensional problems.

4 Experiments

For our experiments, we used the small AP corpus consisting of $M = 2,246$ documents and $V = 10,473$ unique words. We ran a 10 topic LDA model on this document set to obtain the topic-assignment data $\{z_d\}_{d=1}^M$. Next, for each topic, we generated binary data $\{x_k^{(d)}\}_{d=1}^M$. We filtered out those documents for which the mixing proportion for this topic $\theta_{dk}$ is less than 0.25, to remove noisy data. The only parameter in the algorithm is the regularization weight $\rho$ (see table 2) which can be used to control the degree of sparsity: higher values of $\rho$ will result in more sparsity. We used $\rho = 0.1$ in our experiments. Then, for each topic, we ran the fast, scalable, interior point implementation of $l_1$ regularized logistic regression [10][12] for each of the 10,473 words and merged the resulting sparse neighborhoods by a union operation. On average, it took us just about 45 minutes per topic to compute the sparse graph structure on an Intel Xeon 1.86GHz processor with 4GB of RAM.

We present the sparse word correlation structure of two of the topics we labeled “Business” and “War” respec-

---

9 We used an efficient C-implementation downloaded from http://www.cs.princeton.edu/~blei/lda-c/
10 Downloadable from http://www.stanford.edu/~boyd/l1_logreg
tively in figures 5 through 8. The topic “Business” resulted in 128,074 edges, while the topic “War” has only 35,588 edges (as compared to the total 109,683,729 possible edges). Since it is practically impossible to display all the edges in each topic, we presented two views of each topic. In figures 5 and 7, we presented truncated neighborhoods of the top ranking words in LDA. We define the truncated neighborhood of a word as the top two edges in its neighborhood, where the ranking is done in the descending order of the edge-strength $\phi_k$ (see table 2). In figures 6 and 8, we display the truncated neighborhoods of the top ranked edges from the full set of edges.

It is clear from figures 5 and 7 that the Sparse Word Graphs representation of topics is more expressive and meaningful than the LDA representation. The Sparse Word Graphs algorithm succeeds in not only capturing phrases (such as ‘New-York’ and ‘vice-president’ in topic “Business” and ‘United-States’ and ‘air-force’ in topic “War”), but also semantically coherent, long distance correlations (such as ‘inc-acquire’ and ‘pay-receive’ in “Business” and ‘Washington-American’ and ‘Iraq-Kuwait’ in “War”). We also notice that each of the connected components in the figures corresponds to a distinct “concept” within the topic.

The top ranking edges from the two topics and their truncated neighborhoods, displayed in 6 and 8, contain mostly phrases and full names of people. This is not surprising, since these are the word pairs that display strongest correlations within topics. This information is quite useful as well, since a quick glance tells us who are the major players in the respective topics. For example ‘Shearson-Lehman’, ‘Wall-Street’, ‘Ford-Motor’ certainly play a major role in “Business” while ‘Saudi-Arabia’, ‘Saddam-Hussein’, United-States, ‘Ferdinand-Marcos’, ‘Contras-Sandinista’ are all associated with one war or the other. It is also interesting that the name of an anti-war proponent such as ‘Corazon-Aquino’ has cropped up in the topic. Also, the truncated neighbors of the top ranked edges exhibit interesting and meaningful associations that are not necessarily phrases (e.g.: Wall-street-McDonnell’, ‘UAL-Uniteds-unions’, ‘Ford-motors-Chrysler’, ‘San-Francisco-Kohberg’, etc. in “Business” and ’Saudi-Arabia-Hezbollah’, ‘Saddam-Hussein-Baghdad’, United-States-Washington’, ‘Tariq-Hussein-minister’, etc. in “War”).

5 Discussion

5.1 Relation to other word-correlation models

We note that there are other approaches in the past that addressed the problem of modeling word-correlations. Most notable among them is the Hyperspace Analog to Language (HAL) model [15, 14]. This technique models correlation between a pair of words as a weighted count of the number of times they co-occur within a window of fixed length. The weight of each co-occurrence is given by the inverse of the number of words between them. The model enforces sparsity by not considering word pairs that never co-occur within the window length in the entire collection. In [14], the authors show that this algorithm can be implemented on document collections with vocabulary as large as 70,000 words. Our work is very similar to the HAL algorithm in spirit, but the main difference is the following: HAL unearths *global* correlations between words, while Sparse Word Graphs can capture topic-specific, *semantic* word correlations. For example, the words ‘bank’ and ‘river’ may exhibit high correlation in the topic of “Geography” but will exhibit almost no correlations in “Business” (‘bank’ is a common word in “Business” but ‘river’ is not). HAL does not recognize this distinction, but Sparse Word Graphs can. Note that Sparse Word Graphs can approximately produce HAL output by using document binary vectors as input instead of the LDA topic-assignments data.

Another technique that is similar in spirit to our work is the popular idea of query expansion in information retrieval [1]. In this approach, the original short query from the user is first issued to the database to fetch top ranking documents. Words that highly co-occur with the query words in these documents are returned as candidates for query expansion. When the original short query is replaced by the new expanded query and re-issued to the system, the performance typically improves significantly. One can think of this approach as a *dynamic* version of Sparse Word Graphs, in which the neighborhood of query words in the query-specific topic graph are generated as the output. However, query expansion is not a document summarization tool as it requires the queries (topics) to be pre-specified.

Wordnet\(^{11}\) is another effort at constructing a semantic network of words. However, this is a completely human supervised effort, and as such is not directly related to our completely unsupervised algorithm.

5.2 Applications of Sparse Word Graphs

Our experiments demonstrated that the algorithm captures both short distance correlations such as bigrams and phrases as well as semantically meaningful long distance correlations in topics. Therefore this algorithm serves as a better visualization and summarization tool for document collections than LDA.

The algorithm could also be used for word sense disambiguation. Given a word such as ‘bank’, one could identify its different senses in terms of its neighborhoods in various topics such as “Geography”, “Business”, etc. Some preliminary work on this idea using the LDA model already shows

\(^{11}\)http://wordnet.princeton.edu/
Figure 5. Comparison of LDA topic representation with truncated neighborhoods of top ranking LDA words for topic “Business”: top ranked LDA words are bold-faced.

Figure 6. Truncated neighborhoods of top ranked edges in the topic “Business”: top ranked edges are displayed in bold-faced with thick lines
Figure 7. Comparison of LDA topic representation with truncated neighborhoods of top ranking LDA words for topic “War”: top ranked LDA words are bold-faced.

Figure 8. Truncated neighborhoods of top ranked edges in the topic “War”: top ranked edges are displayed in bold-faced with thick lines.
promise [19].

Another related application is query expansion: this technique can sometimes be misled by polysemous words. One could use Sparse Word Graphs as an intermediate step to disambiguate the query as follows. Using the same running example, if the user types the query ‘bank’, then the query-expansion algorithm could first specify the neighborhoods of ‘bank’ from different topics. The user could pick one of neighborhoods, which could then be used to expand the query unambiguously and perform a second retrieval. Alternatively, the system could exploit the session contextual information to automatically pick the right topic and then expand the query based on its neighborhood in the topic.

6 Conclusions and Future Work

We have presented a new algorithm that combines LDA with sparse structural learning methods to successfully capture short and long distance within-topic correlations between words. The algorithm is highly scalable to large collections. For an interested data-mining practitioner, efficient implementations of its components (LDA and $l_1$ regularized logistic regression: URLs displayed in section 4 in the paper) are readily available.

We however note that, our algorithm is not a unified probabilistic model for capturing within-topic word correlations. Building a comprehensive topic model for this problem is a very challenging and complex problem, since these correlations are not explicitly observed (unlike in a model like HAL), but contained within latent variables called topics. Hence, in this work, we simplified the problem by using a two-step process of running LDA first and then using its output in learning the structure of the sparse MRF for each topic. We believe that our work is a significant first step towards solving this challenging problem of modeling sparse word correlations in the topic modeling framework.

As part of our future work, we hope to be able to construct a unified statistical topic model that addresses this problem. We also intend to evaluate the efficacy of this algorithm on specific tasks such as word-sense disambiguation and query-expansion for information retrieval.

References


12words with multiple meanings