CS388 NLP Project Report

# Text Mining for Hidden Relations and Trending

C. Vic Hu
vic@cvhu.org

Ali Unwala
aliunwala@gmail.com

***Abstract*—robotics spanning over 30 years.**

## I. Introduction

In this paper, we developed a data visualization technique and a topic prediction algorithm. By parsing the US patents in the field of 'robot' and 'robotic', we obtained 10,000 text data spanning over 33 years, from 1981 all the way to 2013. These patents were separated into years and passed to Mallet [**?**] for latent Dirichlet analysis [**?**] to obtain a collection of topic distributions. Our methods are then built on top of the results from this topic modeling algorithm.

To help observe the topic trending from year to year, we first developed a visualization technique to aid our analysis. Topics obtained from the text data intrinsically create twisted structures inherited from the previous years. To better understand the underlying patterns, we made a visualization technique, known as the Tree Convergence Graph (TCG). A TCG shows how topics split up, merge together, appear and disappear over a period of time. Our visualization technique superimposes topics from year to year until they converge, creating a bidirectional tree structure that is meaningful to read. We found this style of visualization novel and immensely useful in understanding topic relationships as they grew.

Next, we proposed a topic prediction algorithm called Thematic Particle Clustering (TPC), which creates a collection of sampling particles, that climb towards the closest topics and form a cluster with other topics iteratively. We then used the clustering patterns to quantitatively describe how the future topics will be like. We made two assumptions that topics are conditionally dependent on previous topics, and that the topics modeling algorithm can be simulated with clustering methods. The efficacy of TPC was then demonstrated with the experimental results on the patent dataset.

In section 2, we give a brief background on the topic modeling algorithms, specifically how Latent Dirichlet Analysis (LDA) works. Section 3 describes the algorithms for TCG and TPC. Section 4 gives an example of TCG and a rigorous evaluation of TPC. Section 5 and 6 include our related work and how we can improve our current work, followed by a conclusion in section 7.

## II. Background

Given a collection of text-based patent documents, one intuitive idea to find trending patterns is to examine the underlying thematic structures hidden in the text. Based on the vocabulary distribution, we want to know what the intrinsic topics are implied in the given context. One known way to do this is through Probabilistic Topic Models formalized by Blei et al. [**?**]

### A. Probabilistic Topic Models

The main objective of topic modeling is to automatically discover the unobserved hidden structures – the topics, per-document topic distributions, and the per-document per-word topic assignments, while a collection of text documents is the only observable variables.

For example in Fig. **??**, we have annotated a selection of words, with topics distinguished by colors. In the orange topic, we get words like flange, surface and extending, which could be interpreted as the attachment of hardware components. Similarly, the blue and green topics could be translated into topics about installation and mounting respectively. By looking at the text, human beings can infer what the patent data in Fig. **??** is about, and accordingly highlight the relevant keywords that compose such topics.

Nonetheless, the efficiency and accuracy of human labors does not scale when the size or complexity of these patent documents increases. The objective of probabilistic topic modeling is to automate this inference process and to provide hidden insights and meaningful intelligence to big data. If we are able to successfully construct a probable thematic structure from a large archive of text data for each time slice in a sequence, we could infer how these topics inherit or inspire each other, and most excitingly, predict the most likely topics in the future.

### B. Latent Dirichlet Allocation

Latent Dirichlet allocation, or LDA, is the simplest topic model [**?**]. LDA assigns each word in the documents a distribution over a fixed number of topics. Instead of having a hard boundary between topic collections, LDA provides a distribution of topics per document, giving the likelihood of a mixed proportion of topic assignments. Namely, all text documents share the same set of topic collection but with different proportions to each topic. For instance in Fig. **??**, although there are $K = 100$ topics overall, only a few topics were actually activated in this particular patent.

| | |
|---|---|
| $\beta_k,\ k = 1 \cdots K$ | The K topics, represented by a distribution over words. |
| $\theta_d,\ d = 1 \cdots D$ | Topic proportions for document d, where $\theta_{d,k}$ is the topic proportion of topic k for document d. |
| $z_d,\ d = 1 \cdots D$ | Topic assignments for document d, where $z_{d,n}$ is the topic assignment for the n-th word in document d. |
| $w_d,\ d = 1 \cdots D$ | The observed words for document d, where $w_{d,n}$ is the n-th word in document d. |

TABLE I: Topic modeling notations

To build the generative probabilistic model, we compute the joint distribution and use it to estimate the posterior probability. With the notation specified in Table **??**, the LDA generative process can be formalized as the following joint probability of both hidden and observed random variables:

$$p(\beta_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D}) = \prod_{i=1}^{K} p(\beta_i) \prod_{d=1}^{D} p(\theta_d) \left( \prod_{n=1}^{N} p(z_{d,n}|\theta_d) p(w_{d,n}|\beta_{1:K}, z_{d,n}) \right)$$

It is important to note that there are several conditional dependencies implied in the graphical models, which reflects the main principles of how LDA assumes the documents are generated:

1) Randomly pick a distribution $\theta_d$ over topics.
2) For each word in the document
    a) Randomly choose a topic from the previously-chosen distribution $\theta_{d,n}$.
    b) Randomly choose a word from the corresponding distribution $Z_{d,n}$.

Assuming this generative process is how our documents are created, now LDA uses the graphical model in Fig. **??** to infer the posterior probability of the hidden structures given our observable words:

$$p(\beta_{1:K}, \theta_{1:D}, z_{1:D}|w_{1:D}) = \frac{p(\beta_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D})}{p(w_{1:D})}$$

The computation of possible topic structures is often intractable and the posterior distribution can only be approximated in most cases. To form an approximation algorithm, topic modeling can generally be categorized as a sampling-based and variational based algorithm. The most popular sampling method for topic modeling is Gibbs sampling, which introduces a sequence of random variables to construct a Markov chain and collects samples from the limiting distribution to estimate the posterior. Instead of using samples to approximate the posterior, variational methods find the closest parameterized distribution candidate by solving optimization problems [**?**] [**?**].

### C. Limitations & Potential Improvements

Although LDA provides a powerful perspective to browsing and interpreting the implicit topic structures in our patent corpus, there are limitations it imposes against further discoveries. An extensive amount of research has been focused on relaxing some of the assumptions made by LDA to make it more flexible and suitable for more sophisticated information retrieval.

LDA is essentially a bag-of-words probabilistic model. Namely, it constructs a word-frequency vector for each document but disregards the word ordering and the neighboring context. Although this assumption looses the syntactic information and sometimes seems unrealistic when processing natural language, it is usually good enough when capturing the document semantics and simplifying hidden structural inferences. Nonetheless, for more sophisticated tasks such as language generation or writing style modeling, the bag-of-words assumption is insufficient and needs to be relaxed. In these cases, there are variants of topic models that generate topic words conditioned on the previous word [**?**], or switches between LDA and hidden Markov models (HMM) [**?**].

The LDA graphical model in Fig. **??** is invariant to the ordering of our patent documents. Which is an inappropriate assumption if the hidden thematic structure is actually dependent on sequential information such as years published. To discover how the topics change over time, the dynamic topic model [**?**] treats topics as a sequence of distributions over

words and track how they change over time.

In either LDA or more sophisticated dynamic topic models [**?**], the number of topics $\beta_{1:K}$ is determined manually and assumed to be fixed. One elegant approach provided by the Bayesian non-parametric topic model [**?**] is to find a hierarchical tree of topics, in which new documents can imply previously undiscovered topics.

To include additional attribute information associated with the documents such as authorships, titles, geolocation, citations and many others, an active branch of research has been formed to incorporate meta-data in topic models. The author-topic model [**?**] associates author similarity based on their topic proportions, the relational topic model [**?**] assumes document links are dependent on their topic proportion distances, and more general purpose methods such as Dirichlet-multinomial regression models [**?**] and supervised topic models [**?**].

Many other extensions of LDA are available, including the correlated topic model [**?**], pachinko allocation machine, [**?**], spherical topic model [**?**], sparse topic models [**?**] and bursty topic models [**?**].

## III. Problem Definition and Algorithm

### A. Task Definition

The primary objective of this paper is to learn from a given set of text documents and try to predict the upcoming themes. In particular, we focused on the US patents with the search results of 'robot' and 'robotic', consisting of over 10,000 documents from 1981 to 2013. To formalize the idea of how the thematic structures change in a sequence of text data, we applied LDA [**?**], the simplest Topic Modeling we introduced in the previous section, to each time slice and found a collection of topics for each year. As mentioned previously, each topic is essentially a distribution over a set of words, which can be represented as a weighted vocabulary vector $\beta_i^y$ for the i-th topic in year $y$. Now we have a collection of topic vectors over a sequence of time, $\beta_{1:K}^{1:Y}$, our goal is to use this information to estimate the most likely topic vectors in the upcoming year, $\beta_{1:K}^{Y+1}$, from which we can interpret a reasonable prediction and quantitatively evaluate our performance.

### B. Tree Convergence Graph Algorithm

Before we were able to parse the topic models we needed to be able to understand the data we were looking at. Our TCG algorithm followed the following steps

```
1       Initialize array named PATHS
2       foreach YEAR:
3       |     foreach TOPIC:
4       |     |     PATHS = return path from
        |     |     current node to the end
        |     |     of years
5       |     end
6       end
7
8       foreach PATH in PATHS:
9       |     PATH = merge PATH with
10      |     other other PATHS
11      end foreach path in paths
12
13      plot(paths)
```

### C. Thematic Particle Clustering Algorithm

Built on top the results obtained from the LDA topic models, our Thematic Particle Clustering (TPC) algorithm aims to make topic predictions for the upcoming year based on what the algorithm had seen in the past. Our goal is to formalize a set of particles $\mathbf{w}_{1:N}$ inferred of the topics from previous years, cluster them and use the results to describe what we think the upcoming topics $\beta_{1:K}$ will be. Before jumping into the details of TPC, let's go over some of the fundamental concepts and definitions we will use.

*1) Distance Functions:* The idea of a particle is essentially a sampled instance of the topic distribution over a time sequence, represented by a vector $\mathbf{w}_i \in \mathbb{R}^{|V|},\ i \in \{1, \cdots, N\}$, where $|V|$ is the total vocabulary size of all the topic words appeared. Since one of our intermediate objectives is to formalize clusters between these particles, we need to first define how we will measure the similarity or distance between any pair of particles $\mathbf{w}_i,\ \mathbf{w}_j$.

**Minkowski**:

$$d = \sqrt[p]{\sum_{k=1}^{|V|} |w_{ik} - w_{jk}|^p}$$

Note that when $p = 1$, the Minkowski reduces to the city block distance, while $p = 2$ gives the Euclidean distance and $p = \infty$ yields the Chebychev distance.

**Cosine**:

$$d = 1 - \frac{\mathbf{w}_i\, \mathbf{w}_j^T}{|\mathbf{w}_i|_2 |\mathbf{w}_j|_2}$$

**Correlation**:

$$d = 1 - \frac{(\mathbf{w}_i - \overline{\mathbf{w}}_i)(\mathbf{w}_j - \overline{\mathbf{w}}_j)^T}{|(\mathbf{w}_i - \overline{\mathbf{w}}_i)|_2 |(\mathbf{w}_j - \overline{\mathbf{w}}_j)|_2}$$

where

$$\overline{\mathbf{w}}_i = \frac{1}{|V|}\sum_{k=1}^{|V|} w_{ik},\ \overline{\mathbf{w}}_j = \frac{1}{|V|}\sum_{k=1}^{|V|} w_{jk}$$

**Jaccard**:

$$d = \frac{\#\left[(w_{ik} \neq w_{jk}) \cap ((w_{ik} \neq 0) \cup (w_{jk} \neq 0))\right]}{\#\left[(w_{ik} \neq 0) \cup (w_{jk} \neq 0)\right]}$$

*2) Clustering Algorithms:* LDA is essentially finding the 'principal components' from the text documents, reducing a large archive of data into just $K$ representative topics $\beta_{1:K}$. Therefore, our assumption is that by grouping similar particles induced from past topics, we can observe the clustering patterns and make reasonable predictions on how the future topics will act.

Now we have a collection of well-defined distance functions, we can start looking at clustering methods to group our particles together accordingly, and here are a set of common clustering algorithms we consider:

**K-Means:** As the name itself suggests, the K-Means clustering algorithm consists of K cluster centroids and moves these means iteratively towards the center of its closest neighbors, until they no longer change. Although K-Means has been proved to be guaranteed for convergence [**?**], its clustering performance is often correlated to how the seeds are initialized, and the optimal choice of $K$ is often not apparent (in our case, $K$ is the same as the number of topics.)

```
1. Initialize the means by picking
   K samples at random
2. Iterate
2.a. Assign each instance to
     its nearest mean
2.b. Move the current means to
 the center of all the
 associated points
```

**Hierarchical Agglomerative Clustering (HAC):** This algorithm starts with treating every instance as individual cluster, and iteratively joins pairs of similar clusters repeatedly until there is only one cluster. If we take the merging history and form a hierarchical binary tree, it will look like the dendrogram in Fig. **??**.

Although we have defined the distance functions in the previous section, we have not yet formalized the similarity functions between clusters. In our method, we will focus on the following four similarity functions:

- **Single-linkage:** Also known as the nearest neighbor, computes the distance between the two closest elements from two clusters
- **Complete-linkage:** The conjugate of **single-linkage**, also known as the farthest neighbor, computes the distance between the two farthest elements (maximum distance) from two clusters
- **UPGMA:** Unweighted Pair Group Method with Averaging calculates the distance between two clusters, $C_i \& C_j$, by averaging all distances between any pair of objects from the two clusters.

  $$dist(C_i, C_j) = \frac{1}{|C_i||C_j|}\sum_{w_i \in C_i}\sum_{w_j \in C_j} dist(w_i, w_j)$$

  Now, let's call this newly-formed cluster $C_{ij}$ and compare its distance with another cluster $C_k$:

  $$dist(C_{ij}, C_k) = \frac{|C_i| dist(c_i, c_k) + |C_j| dist(c_j, c_k)}{|C_i| + |C_j|}$$

  where $c_i,\ c_j,\ c_k$ are the centroids for clusters $C_i,\ C_j,\ C_k$
- **WPGMA** Weighted Pair Group Method with Averaging is similar to UPGMA except that the cluster distance is now calculated as:

  $$dist(C_{ij}, C_k) = \frac{dist(c_i, c_k) + dist(c_j, c_k)}{2}$$

The behavior of HAC is often dominated by the chosen similarity function. While each variant has a different set of clustering patterns it is good at capturing, all of them have trade offs.

*3) Putting It Altogether:* Now we have gone through all the necessary concepts we used, the concepts of our Thematic Particle Clustering (TPC) algorithm can be discussed. The full TPC algorithm can be found in Table. **??**.

To simulate how the topic flow splits and merges over time, we introduce the concept of *particles*. One key assumption we made here is that when the topics carry over to the next year, each topic from the current year actually splits up itself

and redistribute its sub-components to merge with those from other topics. Namely, if we focus on just one topic from a year and the path it takes, what we get is a tree-like structure that describes how such topics merge together with other topics to form future topics, or simply gradually vanishes. If we take all these tree-like structures from every topics and study how they behave over the years, what we end up with is a complex directional acyclic graph (DAG) (see Fig. **??**)

Looking at the complex DAG in Fig. **??**, or what we call the Topic Convergence Graph (TCG), as a river system, the idea of the particles can be thought of as a school of fishes released upstream. As fish swim downstream, they start eating diffrent food depending on the path they took down the river system. When the fish have finally reached the bottom of the river system, we can randomly sample them, and extract information about its traveled trajectory by checking what vitamins the fish has or is lacking. Using a method simiar to this we hope to observe an accumulated momentum that topics have that will help us to predict the topics in the future.

The above are the intuitions behind the Thematic Particle Clustering algorithm. We start by initializing a collection of particles uniformly assigned to each topic in the first year, with gaussian noise added to include stochasticity (controlled by a parameter $\alpha$.) For each time step (sliced by year), we apply the clustering algorithms mentioned above to group similar particles together. Here we made another assumption that LDA could be simulated by clustering similar particles. Once forming the $K$ clusters, we look at the topics from the next year and pick a topic that is the closest to each cluster. Finally, we do a hill-climb for all the particles within each cluster towards the corresponding optimal topic with a discount rate $\gamma$ to anneal old weights and the same gaussian noise controlled by $\alpha$ (see Step 5 in Table. **??**), and repeat the same process for the next time step.

## IV. Experimental Evaluation

### A. Methodology

Before starting, we preprocessed the results obtained from LDA and constructed each topic in each year as vectors in the space of the entire vocabulary $\beta_{1:K}^{1:Y}$, as shown in Fig. **??**. We can clearly observe the patterns of how the vocabulary evolved over the years, including repeated occurrence of certain words and the frontier curve that indicates newly-emerged word, which is also represented in Fig. **??**.

| Step | Action |
|---|---|
| 1 | Initialize $\mathbf{w}_{1:N} = \beta_i^1 + \mathbf{N}(0, \sigma(\beta_i^1))$<br>// Randomly assign each particle the i-th topic from the first year $\beta_i^1$ |
| 2 | For $y = 2 \cdots (yo - 1)$<br>// Iterate the following steps through each of the following years in the time sequence until reaching right before the predicting year $yo$ |
| 3 | $\mathbf{c} = cluster(\mathbf{w}_{1:N}, K, dist, link)$<br>// Get the cluster index $\mathbf{c}$ from the clustering algorithm with the designated distance function $dist$ and the linkage $link$ |
| 4 | For $k = 1 \cdots K$ |
| 5 | $\mathbf{w}_k = \gamma\, \mathbf{w}_k + \beta_j^y + \alpha \mathbf{N}(0, \sigma(\beta_j^y))$<br>// Iterate through each of the $K$ clusters to find the closest topics from the current year $\beta_j^y$, where $\mathbf{w}_k$ is defined as the particles assigned to cluster $k$. Apply $\gamma$, the discount rate, to the old weights and shift the new weights toward $\beta_j^y$ with an $\alpha$-strong gaussian noise |
| 6 | End the for loops for both Step 2 and 4 |

TABLE II: The TPC Algorithm

*1) Tree Convergence Graph:* The TCG was created out of necessity to gain intuition about the underlying data structure. TCG helped us formalize our ideas which led to the TPC method. TCG very quickly shows when large groups of topics appear. This structure gave us the intuition that topics could be word biased. Since topics with the most popular words for an year may be more important over time than just an random topics. TCG helped us gain an intuition for evaluating the TPC method by comparing how well a word biased error does with respect to an unbiased error.

*2) Thematic Particle Clustering:* We were successfully able to implement TPC. TPC was implemented twice with two clustering algorithms HAC and K-Means. We performed rigorous testing against various parameter values and with respect to different clustering algorithms and distance functions described in section 3.

Our TPC using HAC (TPC-HAC) evaluation has 4 components. First, we tested $\gamma$, the discount rate parameter, to see how annealing the weights from older years influences the learning performance. Second, we tested TPC-HAC while varying the noise factor, $\alpha$, which allowed us to study effects of adding stochasticity in $\mathbf{w}$ vectors. Third, we compared our clustering algorithms with a collection of distance functions. Finally, we looked at how the linkage functions in HAC affect our results.

Our evaluation of TPC using K-Means (TPC-KM) had 3 components. All of which were in a similar style to the TPC-HAC evaluation, except for the linkage function comparison, which only applies to HAC methods.

### B. Results

*1) Tree Convergence Graph:* For readability, we present an example of a TCG in the Appendix A. On the X axis we show 10 years of patents and on the Y axis we have 10 topics per year. Each line on the graph shows the most probable link between a set of patents between 2 years. It is important to notice the many "dangling lines" that start midway though the graph (for example the green line at the bottom in 1985). These "dangling lines" represent new topics appearing. It is also important to note that once a group of topics have converged to an single topic they will never again diverge.

Appendix A shows only 10 of the 31 years of data we pulled from 1982 to 1991. We let the graph continue without adding adding any new "dangling lines" and by 1998 you can see every topic from between 1982-1991 has converged to just two separate topics. Each time a line changes color in Fig. **??** this represents a new topic usurping an older one.

We make the assumption that every year a topic must go to a topic the next year. We believe this is a reasonable assumption for patents since inventions build on each other and would rarely skip a whole year before something new came out.

*2) Thematic Particle Clustering:* First, we compared TPC-HAC with various $\gamma$ values to see the effect of discount rate. In Fig. **??** we have a baseline method that simply models each year's topics using only the ones from the previous year. We can immediately observe some improvements imposed by $\gamma = 0.75,\ 0.5$, and inferior performance led by $\gamma = 1.0,\ 0.0$ for no discount or learning at all. For the following experiments, we hold $\gamma = 0.75$ as our optimal discount rate.

Both TPC-HAC and TPC-KM constantly outperformed the baseline method in almost every year in our patent data, with an average improvement of 11% and 12% respectively over 33 years. In the next section we also give a reason for spike in error from years 21 to 24.

### C. Discussion

*1) Tree Convergence Graph:* TCGs are a great tool to visualize data but have trade offs. For example a TCG will not directly show you why two topics decided to merge. Nor does it tell you what words a topic is made up of. Or how strongly the topics converged. These issues could be added to the TCG graph but would make the data less parseable. Some other features like word counts for the entire year would just not make sense in the TCG graph. Regardless of some of these drawbacks we still believe visualizing this data provided an important starting point for building intuition about the data.

*2) Thematic Particle Clustering:* Based on our experimental results, the TPC algorithm has a promising potential in predicting topics obtained from LDA. They are both dependent on the choice of discount rate $\gamma$, with little bias introduced when choosing different noise level $\alpha$. Although both TPC-HAC and TPC-KM perform better than the baseline method in most years, the learning curve doesn't go down monotonically with more years of data, as one might expect. All of our results consistently have rising errors. In fact, TPC is even worse than the baseline methods between year 21 and 24. One way to explain this result is large amount of new topic additions in certain years, which couldn't be predicted using the topics from the past. Since our patent data was pulled from the robotic-related literature, it makes sense that we would see large innovations in the late 90s and after 2000. If we were to use TPC on a more established field, the performance would be more reliable and promising, which remains as a subject for our future work.

## V. Related Work

Our method for choosing the most probable path of topics builds on the ideas of the Viterbi Algorithm (VA) [**?**]. Our predictions for topics are made by finding the error between topics from year to year. In any two given years our algorithm will find the lowest error for every topic from one year to the next. With this information we can build a forward trellis structure from every starting topic.

The dynamic topic models (DTM) developed by Blei et al. [**?**] builds a generative graphical model to study how the topics change over time, are very useful for trend prediction. However, DTM assumes a fixed set of topics throughout the entire time sequence, and it only samples how the proportions of the same set of topics change over time. Although we also assume that the number of topics is fixed over the years, our topic models were calculated by a conventional LDA individually for each year, which reflects a more direct perspective on the true topics formed by each time slice. Becasue of these diffrences we could not use Blei's work as

a base line comparison to ours.

## VI. Future Work

Building on this work there are many paths that could lead additional research.

First, our implementation of TPC biases particles with respect to word counts. There could be better features to try and bias results by, for example authors or document length. Finding a better set of features to bias words by could be done by creating an large set of features for each document and seeing which set of features are most correlated from year to year. Correlations could be found through tools like Weka.

Another way to modify TPC's could be through using multi-layer neural network compression techniques on the particles. In the TPC algorithm $N > K$ where $N$ is the number of particles and K is the number of topics. We have to provide a transform to fit $N$ into $K$ so it can be used from year to year. Using a neural network with an appropriately defined error function we could transform $N$ particles into $K$ particles with multiple features weighting the distribution.

Finally, more sophisticated methods can be introduced when measuring the similarity between two topics. Techniques such as Kullback-Leibler divergence, information entropy gains and TF-IDF could be very useful directions to consider for future research and development.

## VII. Conclusion

To study the hidden relations in a large archive of patent text data, we applied the topic modeling algorithm, LDA, years of patent data and built the Topic Convergence Graph to help us visualize topics trending and their inheritance relations. Then, we introduced a topic prediction algorithm, Thematic Particle Clustering, which generates a number of sampling particles to capture information spillovers between years and uses their clustering patterns to forecast future topics. Compared with a baseline method, we were able to make a decent improvement of 11-12% on average. Although there are many ways we can improve upon our method to provide a more reliable and accurate prediction, we believe our works opened a promising door for further research in the field and demonstrated the possibility of predicting topic trends in text data.

## VIII. Acknowledgement

We are very grateful to the valuable feedback and suggestions given by Professor Mooney at the University of Texas at Austin. We would also like to thank Blei et al. for building up the ground works behind topic modeling, and McCallum et al. for the development of MALLET.

## References

[1] Blei, D. Introduction to Probabilistic Topic Models. Princeton University. 2011.

[2] Blei, D., Ng, A. and Jordan, M. Latent Dirichlet allocation. Journal of Machine Learning Research, 3:993-1022, January 2003.

[3] Hoffman, M., Blei, D. and Bach, F. On-line learning for latent Dirichlet allocation. In Neural Information Processing Systems, 2010.

[4] Wallach, H. Topic modeling: Beyond bag of words. In Proceedings of the 23rd International Conference on Machine Learning, 2006.

[5] Griffiths, T., Steyvers, M., Blei, D. and Tenenbaum, J. Integrating topics and syntax. In L. K. Saul, Y. Weiss, and L. Bottou, editors, Advances in Neural Information Processing Systems 17, pages 537-544, Cambridge, MA, 2005. MIT Press.

[6] Blei, D. and Lafferty, J. Dynamic topic models. In International Conference on Machine Learning, pages 113-120, New York, NY, USA, 2006. ACM

[7] Teh, Y., Jordan, M., Beal, M. and Blei, D. Hierarchical Dirichlet process. Journal of the American Statistical Association, 101(476):1566-1581, 2006.

[8] Rosen-Zvi, M., Griffiths, T., Steyvers, M. and Smith, P. The author-topic model for authors and documents. In Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence, pages 487-494. AUAI Press, 2004.

[9] Chang, J. and Blei, D. Hierarchical relational models for document networks. Annals of APplied Statistics, 4(1), 2010.

[10] Mimno, D. and McCallum, A. Topic models conditioned on arbitrary features with Dirichlet-multinomial regression. In Uncertainty in Artificial Intelligence, 2008.

[11] Blei, D. and McAuliffe, J. Supervised topic models. In Neural Information Processing Systems, 2007.

[12] Blei, D. and Lafferty, J. A correlated topic model of Science. Annals of Applied Statistics, 1(1):17-35, 2007.

[13] Li, W. and McCallum, A. Pachinko allocation: DAG-structured mixture models of topic correlations. In International Conference on Machine Learning, pages 577-584, 2006.

[14] Reisinger, J., Waters, A. ,Silverthorn, B. and Mooney, R. Spherical topic models. In International Conference on Machine Learning, 2010.

[15] Wang, C. and Blei, D. Decoupling sparsity and smoothness in the discrete hierarchical dirichlet process. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, Advances in Neural Information Processing Systems 22, pages 1982-1989. 2009.

[16] Doyle, G. and Elkan, C. Accounting for burstiness in topic models. In International Conference on Machine Learning, pages 281-288. ACM, 2009.

[17] Selim, S. Z. and Ismail, M. A. (1984). K-means-type algorithms: a generalized convergence theorem and characterization of local optimality. Pattern Analysis and Machine Intelligence, IEEE Transactions on, (1), 81-87.

[18] McCallum, A. K. (2002). Mallet: A machine learning for language toolkit.

[19] Forney Jr, G. D. (1973). The viterbi algorithm. Proceedings of the IEEE, 61(3), 268-278.

## IX. Appendix

## X. Appendix A