Grisham: A Topic-Based Search, Visualization, and Exploration System

Authors

Abstract

We describe a demonstration of our system *Grisham*, presenting methods for topic-based paper exploration. We use a popular topic modeling algorithm, Latent Dirichlet Allocation, to derive topic distributions for each scientific paper in a DBLP citation data set. We allow users to specify personal topic distribution to contextualize the exploration experience. We demonstrate three types of exploration *keyword-based search*, *topic-based exploration*, and *citation-lineage search*. In each model we shape the results to be specific to a user specification. We describe the components of our web-based system.

Introduction

In a variety of situations, from literature surveys to legal document collections, people try to organize and explore large amounts of documents. Current technology to search on documents are done based on keywords with minor extensions. Sometimes in order to enable keyword search, external effort—What are they? – CPG— has to be put in to tag the documents with relevant keywords. Keyword-based search is useful when the user knows exactly what he or she is looking for. It is not particularly useful when a user wants to explore or learn a new topic. This is especially important when for example, a researcher wants to find out the state of the art in a particular area or a student would like to create a literature survey. In such situations, *topic-based search* can return more relevant results. Topic-based search is a classification of the search space to highlight topical relevance. In order to accomplish this, the topics underlying the document collection need to be extracted, and then they have to be represented in terms of those topics and ranked based on relevance to a particular topic.

In our system called *Grisham* we present various techniques for topic-based exploration and search of peer reviewed scientific papers. Our work allows user to search for scientific papers using three methods: First, users may perform a traditional search *keyword-based search* for papers. Second, users can specify a topic or topics he or she is interested in and the most relevant papers for that topic will be listed in the order of their relevance by our system. Lastly, users are allowed to explore similar or related documents using a visual graph like interface, i.e., given a paper or an article, a user will be shown a set of papers which cite the original paper in the order of their relevance to the topic. In addition, we allow the user to specify a topic distribution that specifies their interest. In all three methods, we adapt search results to personalize results.

The main contributions of our work are the *user-topic ranking function* which ranks the relevance of documents to a set of topics—I think we need to edit this; a similar ranking function is there in George et al. 2012 – CPG, the *similar document explorations* which is performed by computing the similarity of papers to a topic of interest, and *topic-document visualization* which ranks citations for a paper by the interest of the user.

Topic Models

Topic models are a set of models for the documents in a collection or corpus. They enable us to represent the properties of a large corpus containing numerous words with a small set of *topics*, by extracting the underlying topical structure of the corpus and representing the documents according to these topics. We can then use these representations for organizing, summarizing, and searching the corpus. Traditionally, topic models assume word occurrences within a document are independent of each other—i.e., "bag of words" models. Latent Dirichlet Allocation or LDA (Blei, Ng, and Jordan 2003) is a well known, generative, probabilistic topic model for a corpus. A probabilistic generative model assumes data as *observations* that originate from a generative probabilistic process that includes *hidden* variables. The hidden variable are typically inferred via *posterior inference*, in which one tries to identify the posterior distribution of the hidden variables conditional on the observations. Loosely speaking, one can consider posterior inference as the reverse of the generative process. The generative model of LDA assumes that there exists a set of *latent* (hidden) topics in for a give corpus. A topic is defined as a distribution—they are assumed to be generated from a *Dirichlet* distribution with a set of parameters—over the corpus vocabulary. For example, the *whales* topic typically will have words related to *whales* and correlated topics, e.g., *blue whales*, *killer whales*, *whaling*, etc., with high probability and words related to other uncorrelated topics, e.g., *sports*, *medicine*, etc., with low

probability—assuming the corpus is built from a subset of articles from the topics *whales*, *sports*, and *medicine*. In addition, each document in the corpus is described by a latent topic distribution—they are assumed to be generated from another *Dirichlet* distribution with a set of parameters—and the words in a document are generated from the document specific topic distribution. In real life, we only observe documents and their words. As in any generative probabilistic model, the latent variables in the LDA model are typically identified by posterior inference.

However, in most of these generative models, posterior inference is intractable due to the high dimensionality of the latent variable space, and practitioners typically rely on approximate posterior inference alternatives. For the LDA model, people have used different approximate inference methods such as deterministic *optimization methods* (Blei, Ng, and Jordan 2003) and *sampling methods* (Griffiths and Steyvers 2004) for the inference. Blei, Ng, and Jordan employed variational methods to find approximations to the posterior distribution of latent variables, by posing a family of lower bounds on the log likelihood indexed by a set of variational parameters. The variational parameters are then identified by a deterministic optimization procedure that seeks to find an optimal lower bound. Griffiths and Steyvers's method was based on Gibbs sampling—a Markov chain Monte Carlo method that helps to approximate the intractable posterior integral as an empirical estimate of the samples generated from a Markov chain. In Gibbs sampling, one forms the Markov chain by repeatedly sampling each variable conditional on the most recently sampled values of the other variables (Geman and Geman 1984). In this paper, we use the scalable implementation of the online variational inference algorithm for LDA (Hoffman, Blei, and Bach 2010) by Řehůřek and Sojka 2010.

Due to the fully generative semantics, even at the level of documents, LDA is expected to overcome several drawbacks—e.g., issues such as synonymy and polysemy of words—of earlier models for corpora such as Term-Frequency Inverse Document Frequency (TF-IDF, Salton, Wong, and Yang 1975) and Latent Semantic Analysis (LSA, Dumais et al. 1995). In this paper, we are interested in the LDA model parameters such as the corpus-level latent topic distributions and document-level latent topic distributions. The latent document topic distributions are lower dimensional representations of documents—compared to the term frequency vectors of documents, in which each element of the vector stands for a term in the corpus vocabulary—and very useful for finding and grouping similar documents in a corpus. Similarly, the latent topics in a corpus, which are distributions over the vocabulary terms, are helpful in visualizing the prevalent thematic structure of a corpus and exploring documents related to a specific theme of interest. Now we describe the notation we use in this paper. For a given corpus, let D be the number of documents in the corpus and V be the number of terms in the corpus vocabulary. The number of topics K in the corpus is a constant and known. For $d = 1, 2, \ldots, D$, we denote the K dimensional vector θ_d^* as the estimate of document d's latent distribution on the topics identified via an approximate posterior inference algorithm. In addition, for $j = 1, 2, \ldots, K$, we denote the V dimensional vector β_j^* as the estimate of jth topic distribution. This forms a $K \times V$ topic matrix, whose jth row is the jth topic and each element β_{jt}^* represents term t's probability for the jth topic.

Grisham

Grisham system is built to process articles, provide fast response to user queries and display descriptive results in a user interface. In this section we describe our preprocessing steps to extract topic models from the documents. We then discuss the User Model behind each user search. Finally we discuss our methods for topic-based search and exploration.

Data pre-processing and topic learning

Here we describe the main pre-processing steps we perform on a collection of articles for topic modeling and search. First, we tokenize articles with the help of the python Natural Language Toolkit (NLTK) [1] and a set of predefined regular expressions. Next, we standardize tokens by removing noise and stop-words. We use typical normalization techniques for word tokens such as *stemming*, in particular we use the popular Porter stemming algorithm (Porter 1980) implementation in NLTK. After building a vocabulary of corpus words, each document is represented as a sparse "bag of words". Last, we use the processed documents as input to the topic learning algorithm (Hoffman, Blei, and Bach 2010) which will in turn learn the latent topic structure of a corpus from the term co-occurrence frequencies of the corresponding documents.

User Model

When performing search, exploration and discovery over articles users may bring particular context to their search. Incorporating this information into the search process has been shown to be beneficial to users (Dou, Zhicheng and Song, Ruihua and Wen, Ji-Rong 2007; Ma, Zhongming and Pant, Gautam and Sheng, Olivia R. Liu 2007). We develop a user model that encapsulates the users personal context and integrates it into their search task.

This model is a distribution of weights for each identified topic in a corpus. Formally, given a set of topics β_j^*s the user model is defined as

$$\mathcal{U} = \{u_0, \ldots, u_K\}$$

where $u_j \in [0, 1]$, $\sum_{j=1}^{K} u_j = 1$, K is the number of topics in the corpus. We allow the user to interactively select the weights that correspond to each topic learned over the corpus. This allows the users to change preferences with each query for more desirable results.

Keyword Search

The user model is used in different ways to provide better feedback to the user. After a *keyword-based search*, the document results of the search are re-ranked by calculating the

[1] `http://www.nltk.org/`

KL-divergence of each document and the user model. Formally, given the set of result documents $\mathcal{D}$:

$$KL(\mathcal{U}||\theta_d^*) = \sum_{j=1}^{K} u_j \ln \frac{u_j}{\theta_{dj}^*}. \quad (1)$$

where $d \in \mathcal{D}$ and θ_d^* is the topic proportion for document d from the LDA model.

Topic-Based Search

Grisham provides several ranking choices to let the user find the best articles. One way of ranking is to identify the topics of real interest, by looking at the most *informative terms* of the estimated topics β_j^* for the corpus, and then determine relevant articles given the topic of interest. A trivial approach to identify informative terms in a topic is to determine the most probable words by sorting the vocabulary terms in that topic in the order of their term probabilities, β_{jt}^*s. In the literature, people have looked at several other methods for finding informative terms (Chuang, Manning, and Heer 2012) and evaluating topics (Mimno et al. 2011). In this paper, we use a visualization scheme called *word cloud* (Davis 2013), to visualize the most probable words in a topic. For example, see Figure 1 for the visualizing of a topic that is extracted from a corpus, which is built from a subset of Wikipedia articles under the category *Whales*.

We can exploit the estimated document specific topic distributions θ_d^*s of individual articles in a corpus, to rank them on relevance for given a topic. Let t be the index for the topic of interest. For each document $d = 1, 2, \ldots, D$ in the corpus, we can calculate (George et al. 2012)

$$m(d) = \ln \theta_{dt}^* + \sum_{j \neq t} \ln(1 - \theta_{dj}^*), \quad (2)$$

where $j = 1, 2, \ldots, K$, and sort them to rank them on relevance. Here, we assume that each θ_{dt}^* is normalized, i.e., $\sum_{j=1}^{K} \theta_{dj}^* = 1$. Intuitively, we can see that this equation will give a high value for a document, if the probability of occurring the tth topic is high in that document. This means a document with a higher value of this score is highly relevant for the topic of interest t, and contains a considerable amount of words from that topic t. In the next section, we will see more about *Grisham*'s visualization methods for the estimated topics and ranked documents based on them.

Topic-Based Exploration

Here, we use the estimated document specific topic distribution, θ_d^*, of an article from the LDA model, for visualizing the hidden topical content of the article. For example, Figure 2 shows a Pie Chart[2] visualization for the θ_d^* of the Wikipedia article *Killer Whale*. It is an article listed under the Wikipedia category *Killer Whales*. Different slices of the pie chart represent different topics in the article *Killer Whale* and the size of a slice represents the probability of a topic given that article. For this illustration, we manually labeled all the topic distributions obtained via the LDA posterior inference, on a corpus that is built using a subset of Wikipedia articles under the category *Whales*. We used the topic word clouds and the Wikipedia subcategories under the category *Whales* for labeling. Once we find an interesting topic to pursue, we can explore all the relevant documents under that topic based on the method described in the previous section.

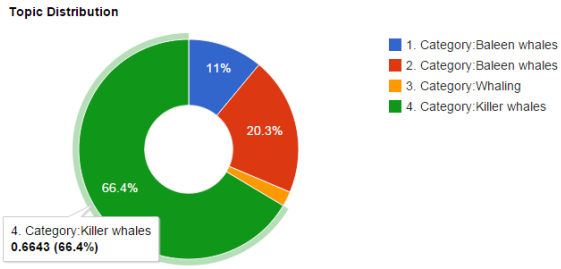

Figure 2: Visualization of the document specific topic distribution for the Wikipedia article *Killer Whale*.

Another way to visualize a document is to look at its *paragraph* or *section* specific topic distributions. Each section or paragraph is written with careful attention is every peer review article or paper. While looking at an article or paper, one can easily identify which section or paragraph is of one's real interest. This intuition can be used to improve topic based exploration. We used the learned LDA model for a corpus for estimating section or paragraph's topic distribution with the help of the Gensim LDA implementation (Řehůřek and Sojka 2010). This is an online task and is performed when a user selects a section or paragraph of an article, which is described in detail in the *Grisham* Demonstration section.

Another interesting option to explore is how we can use an article's topic distribution for searching similar articles of interest. Recall that LDA enables us to transform documents in a corpus into vectors, e.g., θ_d^*s, in a lower dimensional topic space of that corpus. One can then define similarity between two documents via any typical vector space similarities, e.g., cosine similarity. In *Grisham* we call this as article *lineage search*.

Christan: We need to add more about lineage search? – CPG

User Interface

Note this section will be spread out into the other sections. – CEG

To demonstrate *Grisham*'s exploratory search we loaded scientific paper from the DBLP conference (Tang et al. 2008). All the searches in the system require the keep in account the user model. The user model is a profile of preference provided by the user before she makes any search. This is done by specifying weightage to various topics. The *Grisham* website has a list of topics with a slider associated with each of them using which a user can specify the degree of interest in that particular topic. This array of user preference is used as the ranking factor in all the results. The website has three basic functionality which has been classi-

[2] https://developers.google.com/chart

Figure 1: *Topic Word Cloud.* Words with high probabilities for the given topic are larger in size and words with low probabilities for the given topic are smaller in size. From the most probable words, we can infer that the topic is mainly refers the Wikipedia category *Killer Whales*—one of the main categories from which, we downloaded the articles for the corpus.

fied under three different tabs of the same name. They are keyword paper search, Topic explore, and Graph explore.

Keyword Paper Explore This is a universal search facility. The user may enter one or more keywords representing topics, or author names etc. The keywords are searched in the title, abstract, and the author names of all the papers are listed. The listing is ranked based on the user model. In the first page only a few of the papers are displayed in two boxes — one containing matching words in the title, and the other in the abstract. Clicking on any box will open up a more complete list of papers.

Topic Explore The second tab on the website is for topic exploration and it allows a user to click on a specific topic to know more about the papers associated with the topic. Initially, all the extracted topics from the corpus are shown along with their topic words. This list is color coded to distinguish the relevance of topics as indicated by the user model — much like a heat map. The list is clickable and one a topic is clicked, relevant papers to that topic ranked based on the user model and displayed. The user can look at this heat map to adjust their topic preferences. We use equation 1 on the client side to calculate this preference. In the graph explorer the citations for the current paper is ranked using equation 1. The citations of that paper that are most similar to the user model are ranked the highest. A screenshot is shown in Figure 3.

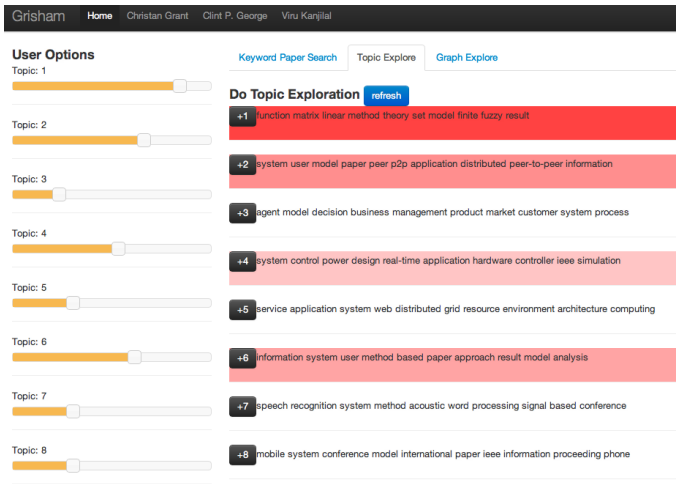

Figure 3: User configured topic exploration

Graph Explore One of the most interesting visualizations in the project is the graph explore which allows a user to conceptually drill down a graph of a paper and its citations in

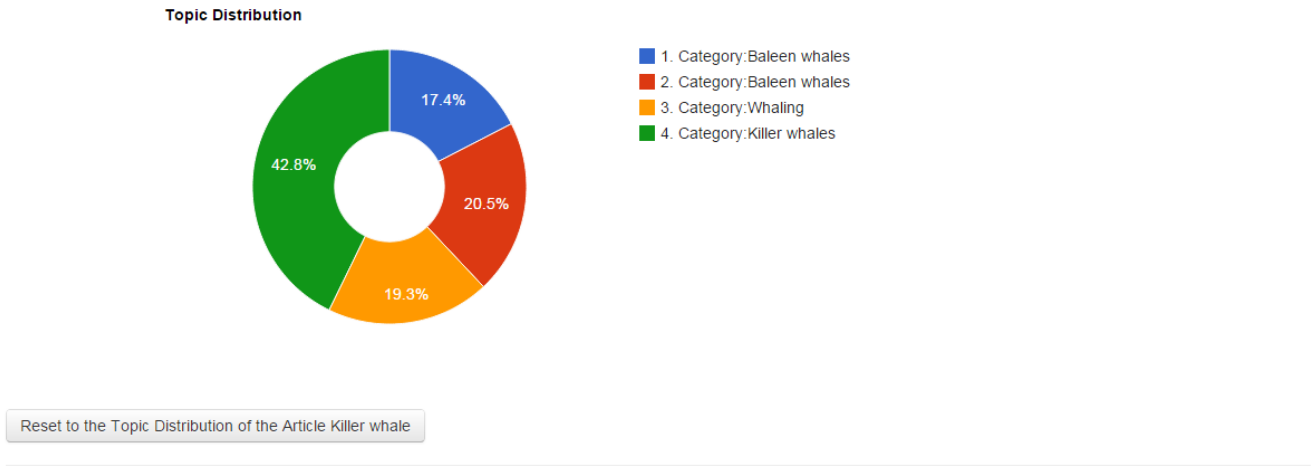

Figure 5: Visualizing the topic distribution of the introduction section of the Wikipedia article *Killer Whale*. See Figure 2 for the topic distribution of the whole article.

a recursive sequence. This is a common method of literature exploration; a user takes up a base paper and then reads up all the papers which have been cited in that base paper. These steps are recursively performed for each subsequent paper until the user has found a sufficient amount of papers or they read all the papers in their collection.

The graph explore functionality allows a user to perform this in a more visually appealing. Once a user decides on a base paper (through keyboard search), the system shows a graph representation of its citations which are ranked based on her profile (user model). This will help her pursue the most relevant papers first. Clicking on any secondary paper will open up the graph further and list it's citations ranked taking into account the user model.

Discussion and Related Work

There have been previous systems to use topic modeling as a basis of search and exploration.

A system of note is Yang et al. (Yang, Torget, and Mihalcea 2011) whom apply topic modeling to collections of historical news papers to assist search. They found that the topics generated from topic models are are generally good, however once the sets of topics are generated, an expert opinion is required to name them. In *Grisham*, we allow users to select numbered topics for article-search based on topically relevant words.

Termite (Chuang, Manning, and Heer 2012) provides a visual analytic tool for assessing topic quality that allows for comparison of terms within and across latent topics. They introduce a saliency measure that enables the selection of relevant terms. For a particular topic, the system provides the word frequency distribution relative to the full corpus and shows the most representative terms according to the saliency measure.

A lot of previous work (Chang et al. 2009; Mimno et al. 2011; Newman et al. 2010) depended heavily on experts examining lists of the most probable words in the topic and validating the models. Hall et al. (Hall, Jurafsky, and Manning 2008) applied unsupervised topic modeling to study historical trends in computational linguistics across 14,000 publications. The work required experts to validate the quality of the results. Only 36 out of 100 topics were retained, and there were 10 additional topics that were not produces by the model and had to be manually inserted.

Leake et al (Leake, Maguitman, and Reichherzer 2003) provide methods to aid concept mapping by suggesting relevant information in the context of topics models, represented as concept maps.

For visualizing the results, previous work (Chuang, Manning, and Heer 2012; Bertin 1983; Henry and Fekete 2007) matrix views to surface the relation ships between a large number of terms. However, it requires an appropriate ordering. Interacting with such visualizations can be complex because the user should already have some idea about the results in advance to be able to generate such orderings.

Conclusion

We describe *Grisham*, a system for topic-based paper search and exploration given a user model. This is a demonstration of a promising new search paradigm. Any research who

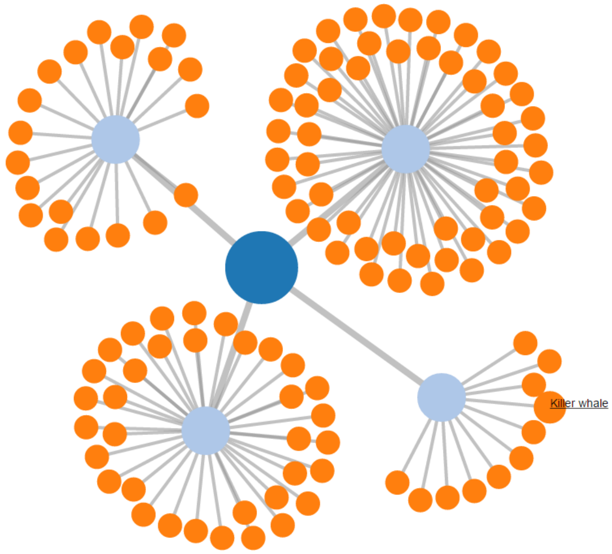

Figure 4: Visualizing of Topic-Based Search in *Grisham*, orange circles represent documents and blue circles represent the estimated topics in a corpus.

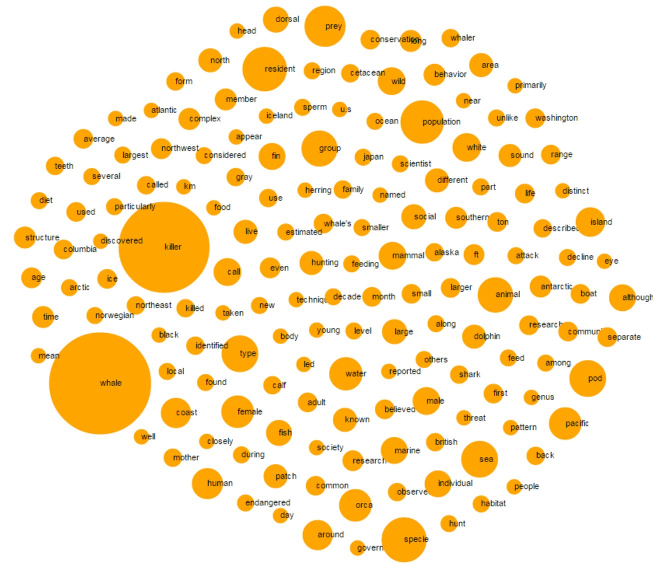

Figure 6: Visualizing the relative frequencies of terms in the Wikipedia article *Killer Whale*. I think we need to decide whether we should keep this figure in the paper or not. – CPG

would like to do exploratory search or a literature reviews will find the system beneficial. During the demonstration we will allow participants to freely interact with the system. We will discuss the algorithm and formulas that drive *Grisham* with attendees.

References

Bertin, J. 1983. Semiology of graphics: diagrams, networks, maps.

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. Journal of Machine Learning Research 3:993–1022.

Chang, J.; Gerrish, S.; Wang, C.; Boyd-graber, J. L.; and Blei, D. M. 2009. Reading tea leaves: How humans interpret topic models. In Advances in neural information processing systems, 288–296.

Chuang, J.; Manning, C. D.; and Heer, J. 2012. Termite: Visualization techniques for assessing textual topic models. In Advanced Visual Interfaces.

Davis, J. 2013. D3: Word cloud. online.

Dou, Zhicheng and Song, Ruihua and Wen, Ji-Rong. 2007. A large-scale Evaluation and Analysis of Personalized Search Strategies. In Proceedings of the 16th international conference on World Wide Web, WWW '07, 581–590. New York, NY, USA: ACM.

Dumais, S.; Furnas, G.; Landauer, T.; Deerwester, S.; Deerwester, S.; et al. 1995. Latent semantic indexing. In Proceedings of the Text Retrieval Conference.

Geman, S., and Geman, D. 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. IEEE Transactions on Pattern Analysis and Machine Intelligence 6:721–741.

George, C.; Wang, D.; Wilson, J.; Epstein, L.; Garland, P.; and Suh, A. 2012. A machine learning based topic exploration and categorization on surveys. In Machine Learning and Applications (ICMLA), 2012 11th International Conference on, volume 2, 7–12.

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

Hall, D.; Jurafsky, D.; and Manning, C. D. 2008. Studying the history of ideas using topic models. In Proceedings of the conference on empirical methods in natural language processing, 363–371. Association for Computational Linguistics.

Henry, N., and Fekete, J.-D. 2007. Matlink: Enhanced matrix visualization for analyzing social networks. In Human-Computer Interaction–INTERACT 2007. Springer. 288–302.

Hoffman, M.; Blei, D. M.; and Bach, F. 2010. Online learning for latent dirichlet allocation. In NIPS.

Leake, D. B.; Maguitman, A. G.; and Reichherzer, T. 2003. Topic extraction and extension to support concept mapping. In FLAIRS Conference, 325–329.

Ma, Zhongming and Pant, Gautam and Sheng, Olivia R. Liu. 2007. Interest-based Personalized Search. ACM Trans. Inf. Syst. 25(1).

Mimno, D.; Wallach, H. M.; Talley, E.; Leenders, M.; and McCallum, A. 2011. Optimizing semantic coherence in topic models. In Proceedings of the Conference on Empirical Methods in Natural Language Processing,

EMNLP '11, 262–272. Stroudsburg, PA, USA: Association for Computational Linguistics.

Newman, D.; Noh, Y.; Talley, E.; Karimi, S.; and Baldwin, T. 2010. Evaluating topic models for digital libraries. In Proceedings of the 10th annual joint conference on Digital libraries, 215–224. ACM.

Porter, M. 1980. An algorithm for suffix stripping. Program: electronic library and information systems 14(3):130–137.

Řehůřek, R., and Sojka, P. 2010. Software Framework for Topic Modelling with Large Corpora. In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, 45–50. Valletta, Malta: ELRA.

Salton, G.; Wong, A.; and Yang, C. S. 1975. A vector space model for automatic indexing. Commun. ACM 18(11):613–620.

Tang, J.; Zhang, J.; Yao, L.; and Li, J. 2008. Extraction and mining of an academic social network. In Proceedings of the 17th international conference on World Wide Web, WWW '08, 1193–1194. New York, NY, USA: ACM.

Yang, T.; Torget, A.; and Mihalcea, R. 2011. Topic modeling on historical newspapers. ACL HLT 2011 96.