The Pennsylvania State University
The Graduate School

CLUSTERING AND TOPIC DISCOVERY IN SCIENTIFIC LITERATURE

A Thesis in
Computer Science and Engineering
by
Levent Bolelli

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2007

The thesis of Levent Bolelli was reviewed and approved* by the following:

C. Lee Giles
Professor of Information Sciences and Technology
Thesis Advisor, Chair of Committee

Wang-Chien Lee
Professor of Computer Science and Engineering

Padma Raghavan
Professor of Computer Science and Engineering

Alan MacEachren
Professor of Geography

Raj Acharya
Head of the Department of Computer Science and Engineering

*Signatures are on file in the Graduate School.

Abstract

Discovery of latent semantic groupings and identification of intrinsic structures in data collections is a crucial task for many data analysis needs. Various unsupervised machine learning algorithms have been devised to accomplish this task for a vast number of applications, including topic identification in text databases, clustering similar images in web search, disease identification in medical fields to name a few. Most algorithms, however, have been designed for *homogeneous* data where the algorithm works on a uniform set of attributes that represent the data objects. Real-world datasets, on the other hand, are richer in structure and contain multiple levels of connectivity among the data objects, such as hyperlinks in web pages, textual annotations or text contexts surrounding images on the web and citations and authorship information of scientific literature. Utilizing only a single information source provides a narrow focus of view into the real nature of the relationships between data objects. Each additional dimension of connectivity increases our understanding of the semantic characteristics of the collection and improves our ability to detect distinct groups of objects where the objects in each group exhibit similar properties.

This thesis presents three algorithms that merge multiple sources of information for clustering and topic discovery in collections of academic papers. The first algorithm combines textual content of academic papers with the information extracted from the citation relationships between the papers for finding scientific topic clusters in the data collection. The second algorithm integrates authorship information of documents into the text-based clustering process to yield improved clustering solutions. Based on the validation from these two algorithms that it is possible to improve topic discovery by utilizing additional dimensions of similarity among data objects, we provide a generative model that merges citation relationships, authorship information, user queries, user tags and the timestamps of documents for discovering scientific topics in collections of academic papers. Further, the generative process can effectively model the evolutionary characteristics of document collections and can discover the change of the popularity of scientific topics over time.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to express my gratitude to my thesis advisor, Prof. C. Lee Giles, for making the years with him rewarding and memorable. This journey became a pleasant experience with his valuable insight, constant guidance and friendly accessibility. I am also grateful to my committee members, Prof. Wang-Chien Lee, Prof. Padma Raghavan and Prof. Alan MacEachren for their useful suggestions and feedback. Special thanks go to the members of the Intelligent Information Systems (IIS) Research Laboratory for technical collaboration and discussions. I also thank my supervisors and colleagues at Ask.com and Google, Inc. for fruitful discussions and the valuable experience that they have provided.

I am indebted to my parents for providing me a sound educational foundation and for their unwavering support and encouragement in my academic pursuits. I am grateful to them for planting the seeds of the life that I'm enjoying today.

Finally, and most importantly, I would like to thank my beautiful wife, Şeyda. She is the most essential contributor to my well being and my achievements in life. She has not only given me constant love and support, but she also has been one of my main research collaborators throughout my Ph.D. studies. I have greatly benefited from her expertise in statistics and machine learning. This thesis would not be possible without her.

Dedication

To my love, Şeyda.

Chapter 1

Introduction

The amount of digital information produced each year has been growing exponentially for many years. It has been estimated [1] that the *digital universe* was 161 billion gigabytes (161 exabytes) in the year 2006 and is projected to grow six fold to 988 exabytes by the year 2010. The same study reports that over 95% of the digital universe is unstructured data and only less than 10% of organizational information is "classified" in some way. The World Wide Web(WWW) has followed a faster growth rate in the past decade. The indexable web [2], known as *"the part of the Web which is considered for indexing by the major engines"*, was estimated to be at least 320 million pages [3] in 1998. This estimate was brought up to more than 11 billion pages [4] in 2005 and Yahoo![1] announced that they have indexed 20 billion pages the same year. One of the many reasons why the web has been growing so fast can be attributed to the fact that the mass reach of the WWW has made it a popular medium for sharing information and many types of media content, both on commercial and personal websites. From the early stages of the web, which only contained static html pages connected through hyperlinks,

[1] http://www.yahoo.com

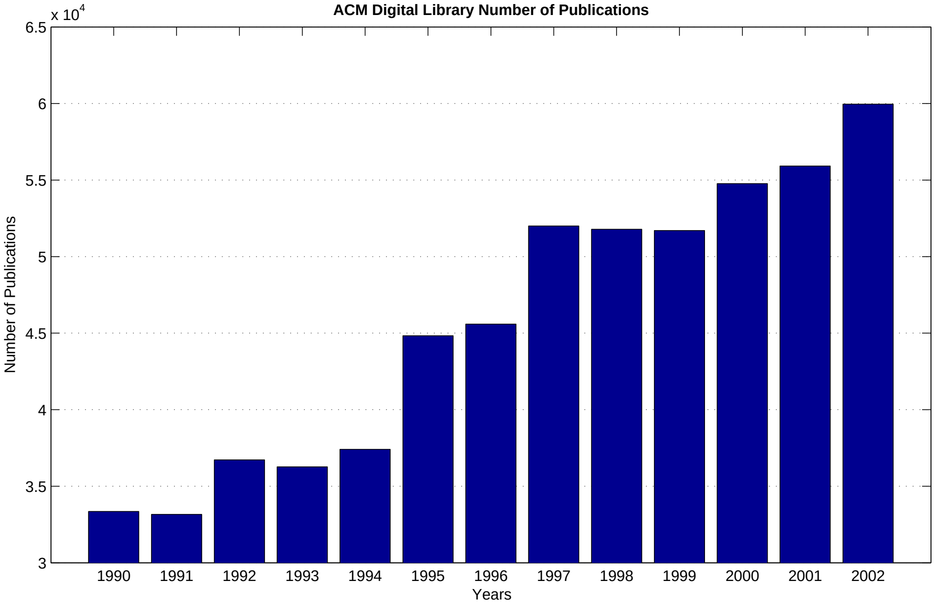


Figure 1.1. Number of Publications added to the repository of ACM Digital Library each year between 1990 and 2002

today's web is much richer in content where audio, video and documents in various formats are increasingly becoming available online. For instance, in the domain of academic papers, the number of papers on the web has been constantly increasing. In addition to the fact that earlier publications are becoming increasingly available on the web, the number of papers published each year has been also increasing over years. Figure 1.1 shows the number of publications added each year to the digital library of Association for Computing Machinery(ACM), one of the major publishers in Computer Science and related fields. This growth rate, in turn, reflects to the number of papers available online, increasing the number of papers indexed by digital libraries that collect academic papers from the web.

It is clear that the pace of growth of the amount of digital information available in many fields has surpassed our ability to perform any kind of manual data analysis tasks with reasonable accuracy and in reasonable amount of time. Organizing the ever growing amount of digital information, therefore, requires developing algorithms that enable us to analyze, organize and retrieve information in an accurate and practical manner. In this thesis, we present new algorithms for detecting and tracking the evolution of scientific topics in repositories of scientific literature.

Many popular search engines today can be considered to be effective for retrieving relevant web pages for most user queries. However, many information needs can not be directly addressed by the *general focus* of search engines that often lack sufficient domain knowledge in order to accurately retrieve, rank and present the data in specific domains. For instance, search engines crawl and index academic papers as any other document type found on the web and treat the textual content of these papers as any other type of document. Thus, the structure of the papers are lost and the semantic components of the papers (i.e. title, authors, citations, etc.) are not identified. This approach is not only detrimental to the effectiveness of the search engine for academic document retrieval, but also limits the extent of data analysis tasks that can be performed, such as grouping papers based on authors, topics and institutions, or discovering scientific topics and their evolution over time.

1.1 Problem Statement

Digital libraries (or niche search engines) are specialized systems that have in-depth understanding of data characteristics in a specific domain with custom architectures that enable them to effectively find, organize and present relevant information in that domain. In the domain of scientific literature, harvesting academic papers from the web and organizing them based on the needs of the researchers helps researchers find papers in their fields, discover relevant research through citation analysis and bibliographic coupling, and analyze research trends. In order to provide the necessary tools to the users to achieve such tasks, digital libraries need ways to identify and extract metadata and citations from papers and understand the distinct topics that the papers address. The feasibility of accomplishing such tasks manually has diminished with the vast amount of academic papers that is available on the web, especially in the field of Computer Science. Managing the growing number of scientific publications requires new tools for automatically organizing, searching and browsing large collections. A prerequisite to effectively achieve these goals involves understanding the semantics of the papers through the process of *topic detection*. In the absence of metadata of the papers and the citations between them, we can only rely on the textual content of the documents and apply traditional data mining algorithms to find groups of papers that are on the same subject. On the other hand, extracted metadata and citations of papers provide us many additional layers of similarity between papers, including common authorship of documents, common affiliations, co-authorship networks and bibliographic coupling, which can substantially improve the quality of the analysis of data mining solutions. However, these additional dimensions can not readily be integrated into a unified solution, since many supervised and unsupervised learning algorithms are designed with the assumption that the data objects only have homogeneous components with uniform representations. In order to account for the heterogeneous nature of the academic papers and to utilize the additional dimensions effectively, we need custom algorithms that can integrate heterogeneous components of papers into a unified topic discovery framework.

This thesis presents two clustering algorithms that address the aforementioned problem outlined above. The algorithms integrate the heterogeneous components of academic papers into unified clustering solutions. Hence, they utilize multiple document similarity viewpoints and thus, do not suffer from the limitations of traditional clustering algorithms that only work on a single dimension of the documents.

Only after the identification of the distinct topics of papers we can gather sufficient knowledge to organize scientific collections, which presents opportunities to conduct in depth analysis in this *semantic* space. For instance, rich collections of research articles shed light into the evolution of human knowledge. Every novelty of today builds upon past discoveries, advances our knowledge, and eventually becomes surpassed by novelties of tomorrow. Understanding the temporal characteristics of topics of scientific publications and identifying topical trends has the potential to assist researchers to understand past topical trends to adapt to new research directions. It is therefore crucial to accurately model the temporal nature of documents that define the topical trends over time. Trend analysis in text collections has become a new research focus due to its applicability to and impact on many domains in addition to scientific literature, including news articles [60], weblogs [58] and social interaction of authors [61, 62] to name a few. A methodology that accurately models the evolution of document collections while utilizing the richness of the documents leads to a better corpus model that captures the true nature of the temporal characteristics of the documents.

This thesis describes a generative model of documents that incorporates the temporal dimension of the collection of academic papers for topic discovery and trend analysis, while integrating the heterogeneous sources of information from the papers to improve the accuracy of the generative process.

1.2 Organization of This Thesis

The rest of this thesis is organized in four parts. The next chapter (Chapter 2) presents a clustering algorithm that incorporates textual content of academic papers and the citations between the papers. Chapter 3 investigates the authorship of papers and describes a method that combines the authorship information of documents with text-based clustering. The third part (Chapter 4) presents a generative model of documents, that brings together many of the knowledge sources that we can gather in scientific corpus for topic discovery and trend analysis. Chapter 5 concludes the thesis with concluding remarks and future research directions.

- **Clustering Heterogeneous Datasets using Citation Analysis (Chapter 2)** This chapter presents a clustering method that merges citation-based and content-based semantic evidence to improve topical clustering of documents. An information theoretic approach is employed to evaluate the importance of terms in documents based on the citation graph of the collection. In particular, we consider both the existence and lack of citation between the papers to determine the *topicality* of the words in the corpus, and augment the text-based similarity space of documents with the topical weights of words identified from the citation relationships. The experimental results depict the effectiveness of the citation graph for detecting the topics of papers.

- **K-SVMeans Clustering of Documents (Chapter 3)** This chapter presents K-SVMeans, a hybrid clustering algorithm that simultaneously clusters documents based on their distribution over words and learns a Support Vector Machine (SVM) classifier based on the authors' distribution over documents, without the need for a manually labeled training set. The clustering decisions are based on both the textual content of the documents and the global view of the authorship of documents obtained from the learners of the clusters. The experimental results in two distinct text collections show the benefits of authorship analysis for topic detection in text documents.

- **A Generative Model for Scientific Literature (Chapter 4)** This chapter presents a generative model for topic detection and topical trend analysis in scientific literature. The generative model captures the dynamics of research interests of researchers and integrates citations, authors, queries and tags to improve the accuracy of the generative process. As opposed to traditional generative models of documents that ignore the temporal characteristics of document collections, the generative model presented in this chapter divides the collection into multiple time segments, and learns the topics in the collection iteratively, starting from the first time segment to the last. At the beginning of the iteration in each time segment, the results of previous time segments are propagated to the current one as prior knowledge, modeling the real-world temporal order of the documents.

- **Conclusions and Future Work (Chapter 5)** This chapter presents concluding remarks and provides further research directions for topic discovery in collections of scientific literature.

Chapter 2

Clustering Heterogeneous Datasets using Citation Analysis

2.1 Introduction

Discovery of latent semantic groupings and identification of intrinsic structures in datasets is a crucial task for many data analysis needs. Various application domains, with significantly varying characteristics of underlying data, look for effective and efficient algorithms to facilitate this task. Although the data type of interest can take many forms (e.g. text, images, gene sequences, graph, web logs, etc), the idea remains the same: Understanding the characteristics of the data and detecting its natural groupings for better utilization of the data. This process is known as *Clustering* although it has been named differently in different contexts, such as *unsupervised learning* in pattern recognition and *partitioning* in graph theory. Clustering is the task of dividing the data into distinct groups such that objects in the same cluster are similar and objects in different clusters are dissimilar, where the definition of *similarity* can take various forms and is domain dependent. For instance, textual documents may be considered similar if they contain many overlapping words/phrases, or if they are authored by the same person. On the other hand, two images may be thought of as being similar if they bear similar low-level visual cues, or higher level semantics. Due to the differences of characteristics of data and the varying needs in different domains, it is a common practice to tailor clustering algorithms to take advantage of the specifics of the underlying datasets.

2.2 Classification of Clustering Algorithms

Clustering is a division of data into groups of similar objects. Each group, also known as a cluster, consists of objects that are similar between themselves and dissimilar to objects of other groups, where the definition of similarity is dependent on the characteristics of the data objects. A general classification of clustering algorithms is as follows.

- **Partitional / Hierarchical Clustering**: Partitional clustering algorithms [14, 15] create one set of clusters whereas hierarchical clustering algorithms [16, 17, 18] build a tree of clusters, also known as a *dendogram*. The dendogram can be obtained either in a bottom-up(agglomerative hierarchical clustering) or top-down(divisive hierarchical clustering) fashion.

- **Hard / Soft Clustering**: Hard clustering algorithms assign each data object to one and only one cluster. Soft clustering algorithms, on the other hand, may assign the data objects to many clusters by finding a probability distribution of data objects on the clusters.

- **One-way / Co-Clustering**: The main difference between one-way clustering and co-clustering is that one-way clustering only clusters the data objects where co-clustering simultaneously clusters the data objects and their attributes. In document clustering, co-clustering corresponds to clustering documents based on the words, and words based on the documents. This duality between data object clustering and attribute clustering can help improve the clustering solution.

2.3 Issues Related to Clustering in Scientific Literature

The rapidly growing world wide web and the increasing volume of scientific literature available in digital format on the web has stimulated supervised and unsupervised data mining research to focus on linked documents. For linked documents, in addition to the textual content similarity, which can be thought of as an *implicit similarity*, we also have the link graph of the documents that depicts the *relatedness information* conveyed by the authors of the documents. Conventional clustering algorithms use attribute information to group documents under the assumption that two documents are related to each other if they have similar attribute values. However, relational data are richer in structure, hence provide more information available to disambiguate groupings. Link structure analysis has been studied extensively and has shown to be a significant aid for both supervised and unsupervised data analysis tasks [19, 20, 21]. In this chapter, we focus on clustering in the collection of scientific literature to discover topical groupings of papers using the textual content of papers combined with the information gathered from the citation graph of the collection. In a citation graph, papers are represented as vertices of the graph and citations are represented as directed edges between citing and cited documents. The importance of citations for topical analysis of documents has been known for decades and the concepts of bibliographic coupling [22] and co-citations [23] have been studied extensively. Bibliographic coupling represents the connection between two documents through common citations. The bibliographic coupling strength is proportional to the number of overlapping citations to other documents. Co-citation is defined as the frequency with which two documents are cited together. The similarity of two documents is proportional to the number of times that they are cited from other documents. In this work, we investigate the citations in the context of direct citation linkage and we are only interested in which documents are cited from which documents. The papers and citation graph have been obtained from CiteSeer's[1] repository.

CiteSeer [24] is a scientific literature digital library that has grown to index over 760.000 academic publications in Computer Science and related fields. Citations of the papers are extracted and linked to cited papers by Autonomous Citation Indexing (ACI) [25]. The citation graph that is constructed through this process provides a network of scientific connectivity since citations in research publications represent an important knowledge source regarding the context of scientific work. The citation relationships have been shown to be a valuable resource for a number of tasks such as ranking search results, identification of related research documents, trend analysis and social network analysis. Besides topical relevance, a number of factors that influence citations have been identified, including the desire to publicize own research [26] and promote own field, author's ability to access the document [27] and to read the language that it is written in [28]. Regardless of the

[1] http://citeseer.ist.psu.edu

reason for citations, comparatively, citation relationships between scientific documents convey more valuable information than a collection of linked web documents [21]. However, the citation graph itself can have limited clustering performance in digital libraries due to the following issues:

1.) *Cited Document Availability.* CiteSeer collects the papers by crawling the web. Thus, the citations of a paper (i.e. target papers) may not be locally available in CiteSeer's repository due to several reasons: a) the citations may not be available on the web, b) they may just not have been crawled, or c) they may not be related to Computer Science or a similar field and may not be added to CiteSeer's repository. In any of these cases, the citations point to virtual metadata records that is identified by the extracted fields of the citation, including title, authors, publication venue, etc. However, the unavailability of the textual content of the cited papers prevents detailed analysis on the semantic similarity between the citing and cited papers. That is, we will not know *why* the paper is being cited by the citing document.

2.) *Identity Uncertainty.* Citations are references to unique documents, but their representations may vary. Finding the best matches for citations is a problem known as identity uncertainty [29]. The task of ACI is to uncover the identity of the paper that a citation refers to in order to group together similar citations to the same document, and to link citations to real documents – those that exist inside the ACI system and those that are yet to be crawled. Although ACI has been highly effective, it is still possible that distinct representations of the same citation may be mapped to different documents, or two citations to different papers be connected to the same target paper.

The aforementioned reasons lead us to use only the citations where both the cited and citing documents are available in the collection, which sparsifies the link graph significantly. In this section, we show that taking an information theoretic approach towards textual content analysis of pairs of documents with citation relationships provides a significant improvement in the discovery of document clusters. Further, we believe that the methodology presented here is applicable to web document collections where similar link constraints can be observed. One example is hierarchical clustering of documents where lower level taxonomies may not exhibit strong connectivity. Another application domain is search engine result clustering [30], an often employed technique to facilitate users' quick browsing through search results. Both applications suffer from the lack of sufficient links between the documents in a given subspace of the entire collection, which can be addressed by the algorithm proposed here.

Documents on similar topics exhibit specific characteristics that separate them from documents that focus on other topics. Similar documents cite each other and they contain some level of textual similarity, measured by the amount of overlapping words/phrases. Some of those words - such as *experiment*, *result*, *compute* - are considered to be general words and they are not beneficial to the clustering algorithm. Other words like - *image*, *security*, *database* -, on the other hand, are correlated with the topics of the papers and they are very valuable for classifying the papers and identifying topical clusters in the collection. Although both textual content and link structure can be used independently for topical clustering, an algorithm that merges both sources of heterogeneous data has the potential to yield better clustering solutions than using either data source alone for clustering purposes. If the link structure of the documents is dense enough, then link based clustering, augmented by textual content, will generally yield well separated clusters. On the other hand, in situations where link graph is sparse, access to linking and/or linked documents is limited, or there is some sort of ambiguity in the link structure itself, the link graph can not be used as the dominant source of clustering. Thus, it is crucial to find a text-based clustering solution that incorporates information from the available link structure. Our work addresses this problem and provides an algorithm that bridges the disconnect between textual content and citations of papers by discovering the topicality of words from the citation relationships and the quantitative topicality measure that is found from the citations is utilized to augment the text-based clustering solution.

2.4 Related Work

As discussed in Section 2.2, due to its wide range of applications, numerous clustering algorithms have been proposed, varying in the way that the clusters are obtained (hierarchical or partitional clustering), cluster membership of instances (hard or soft clustering) and the entities that are being clustered (one-way or co-clustering). In this work, we are interested in partitioning the set of academic papers into distinct scientific topics. Therefore, our focus is to design a partitional, one-way, hard clustering algorithm.

Document clustering algorithms for linked documents can be broadly categorized as text-based [18, 31, 32], link-based [33, 34] and hybrid [35, 36, 37, 38] approaches. In the domain of linked documents, link analysis for clustering and classification purposes has generally been studied in the context of web pages. PageRank [39] and HITS [40] are two of the most popular algorithms showing the importance of link structure for analyzing associations between documents.

In order to merge text-based and link-based information, [41] and [42] use generative probabilistic models of document content and connectivity. He et al. [36] use the hyperlink structure to cluster web pages using spectral graph partitioning. In their work, the link graph is used as the dominant source of similarity between documents, and the link-based similarity measures are augmented by textual content similarity and co-citation similarity. [38] proposes a probabilistic model of link structure based on the cluster membership. The model is optimized based on observed data where the attributes determine the group membership and group membership determines the link structure. Modha et al. [43] propose an algorithm for clustering hypertext documents by using both the document contents and link structure. The algorithm uses an extended version of the classical Euclidean K-means clustering algorithm that performs clustering based on word similarity, in-link similarity and out-link similarity. The effect of each similarity is controlled by a parameter, which needs to be explicitly set by the user.

A number of algorithms have been proposed for *link prediction*, which is the task of identifying the missing entity or entities of a partially observed link by using the existing observation of the data sample available in the domain. [44] uses directed graphical models (Bayesian Networks and Probabilistic Relational Models) to represent a probabilistic model of both links and data object attributes. A comparison of various machine learning approaches for link prediction/completion is given in [45]. One major drawback of model-based link prediction is the dependence on the training data. That is, the learner builds a probabilistic model on the training data, and it will lack confidence in the probabilities of the entities that have not been included in the training set.

2.5 Algorithm

We start with describing the notation used in this section. Let $d_i \in \mathcal{R}^n$ denote the m documents in the collection where n is the number of words in the corpus and C denote the non-symmetric citation matrix where each row and column of C correspond to a document and $C_{ij} = 1$ if d_i cites d_j, and zero otherwise. Each document is represented as a vector in the feature space. Following L_2 normalization of the document vectors so that each $||\vec{d_i}|| = 1$, we generate a similarity matrix S from the cosine similarities of each document pair:

$$S_{ij} = cos(d_i, d_j) = \frac{\vec{d_i^T} \cdot \vec{d_j}}{||\vec{d_i}|| \cdot ||\vec{d_j}||} \tag{2.1}$$

We then calculate, for each citing document, the average distance of its citations using the similarity matrix S and the citation graph as follows:

$$\mathcal{D}_i = \frac{\sum_{j=1}^{n} S_{ij} \cdot C_{ij}}{k_i} \tag{2.2}$$

where $\mathcal{D}_i$ represents the Average Citation Distance(ACD) of document d_i, and k_i is the total number of citations of document d_i that is present in the collection. In this definition, only the citations in the collection can contribute to the ACDs, since, for missing citations, we do not have the text of the document and hence, S_{ij} will be zero. We are interested in evaluating the significance of the words by comparing each word's popularity in document pairs connected with citations against document pairs that do not have citation relationships. To achieve this goal, the ACDs enable us to view the document space from these two perspectives by populating the following two sets: The first set, G^A is the *Actual Citation Graph* and is populated with the citing papers and their citations. This set is

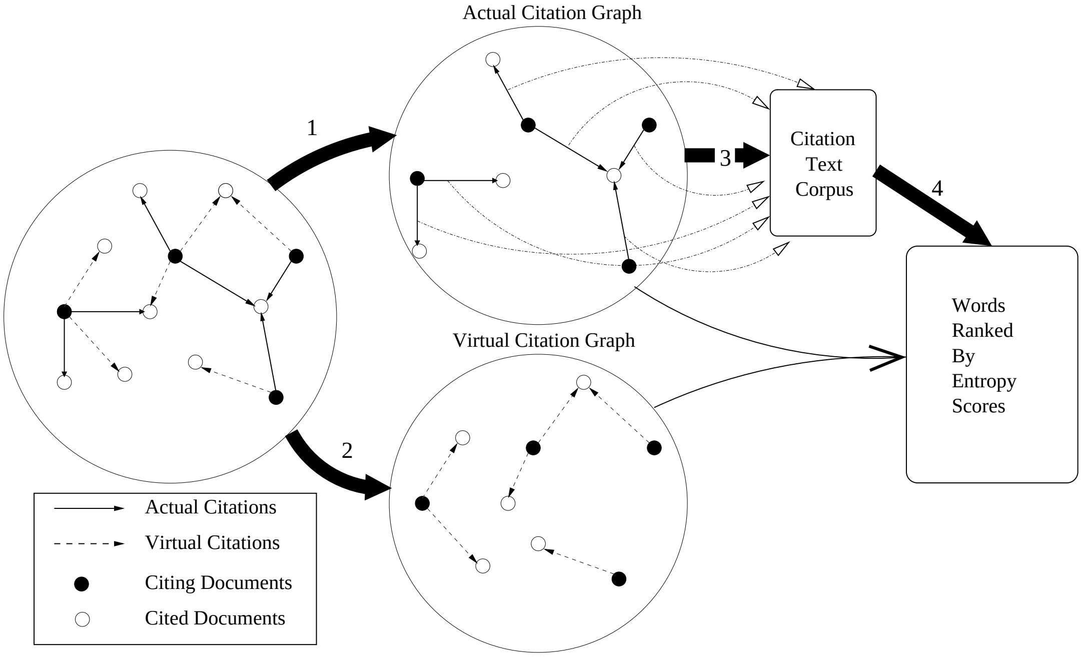


Figure 2.1. Schematical view of the algorithm. The ACDs are used to find the virtual citations and the given link structure is split into Actual Citation Graph G^A and Virtual Citation Graph G^V (Steps 1 & 2). The set of words appearing in both in citing and cited documents in $G^A$are inserted in the Citation Text Corpus T (Step 3). For each word in T, we use the link and word co-occurence information from G^A and G^V to calculate the expected entropy loss scores (Step 4)

the collection of documents that form the citation graph. The second set, G^V, is the *Virtual Citation Graph* and it is populated using the $\mathcal{D}_i$'s in the following fashion. For each document d_i having k_i citations (i.e. the citing documents in G^A), we select k_i documents that are **not** cited by d_i and is separated from d_i by a distance closest to a radius $\mathcal{D}_i$. That is, for each citing document d_i, we find k_i documents such that their content-wise similarity to d_i suggests that d_i should also be citing these documents, but no such citation exists in the graph for d_i. This set of documents form the *Virtual Citation Graph*; the citations in the Virtual Citation Graph are not actual citations but they are *inferred* citations based on the textual similarity between the citing and virtually cited documents.

Algorithm Non-uniform Feature Weighting

```
Populate $G^A$ with the documents in the citation graph
Initialize $G^V \leftarrow \emptyset$, $T \leftarrow \forall t_{ij}$ for $C_{ij} = 1$
    for each citing document $d_i$ in $G^A$ with $k_i$ citations do
        $G^V \leftarrow G^V \cup \{k_i$ not-cited documents of $d_i$ closest to $\mathcal{D}_i\}$
    end
    for each $t_p \in T$ do
        $E_p =$ Entropy loss calculated from equation 2.6.
        $\bar{w}(d_i, t_p) \leftarrow (1 - \lambda) \cdot w(d_i, t_p) + \lambda \cdot E_p, \forall d_i \in \{d_1, d_2, \cdots, d_m\}$
    end
```

After populating G^V with the citing documents in G^A and their respective *virtual citations*, the sets G^A and G^V have exactly the same number of edges, since we restrict G^V to contain the same linking vertices as G^A and insert exactly the same number of (virtual) citations to it. This way, we enable each vertex (i.e. citing document) to be equally represented both in G^A and G^V. We then collect the common words between citing and cited documents in G^A and denote the set of the words as T. We do not consider the shared terms in the document pairs that are in G^V, since our aim is to identify the importance of the terms that appear in actual citation relationships.

We use expected entropy loss measure [46] to calculate the amount of information that each term in T conveys about citations. Our intuition is to find a numerical representation of the importance of each feature that is shared by the documents that are linked together. This also enables us to estimate what makes document A cite document B and not cite C, although B and C may also be

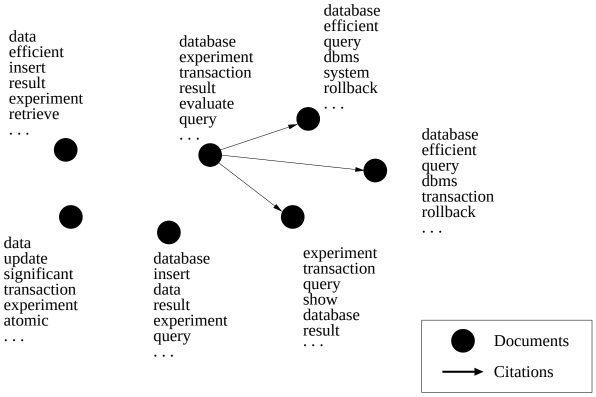


Figure 2.2. A sample set of documents and words from each document. Words that are common to cited and citing documents, and not common in not-cited documents are topically important.

similar based on textual content.

If a word occurs frequently between citing and cited documents in G^A, but not in the virtual citations in G^V, this word is regarded as a good candidate for being a topical word and is emphasized in the clustering algorithm. Figure 2.2 depicts this case for the topic ”databases”. The topical terms ”*database*”, ”*transaction*” and ”*query*” appear more frequently in the documents with citation relationships than the rest of the documents. The non-topical words, such as ”*experiment*” and ”*result*” appear in both cited and not-cited documents. The citations enable us to detect the relative topical importance of each word in the corpus. This approach serves as a means of eliminating one shortcoming of clustering algorithms; that is, each feature is weighted based on some corpus statistics and almost all clustering algorithms treat the attributes of data objects uniformly. We break this uniformity by reflecting the information obtained from the citation graph by scoring the shared terms of the citations using expected entropy loss.

2.5.1 Expected Entropy Loss

Given a text corpus with n distinct features and k categories, expected entropy loss measures amount of categorical discriminative power of each feature in the dataset. For instance, for two categories "Computer Vision" and "Computer Security", the set of terms {*image, segmentation, visual*} are discriminative for Computer Vision and {*attack, rsa, secure*} are discriminative for Computer Security, whereas terms like *algorithm* or *data* are less descriptive of either category, hence have lower expected entropy loss scores. Expected entropy loss enables us to detect this distinction among the words in the corpus.

In formal definition, let C^A and C^V be the events of a sample being a member of the specified class, where the superscripts A and V refer to the actual and virtual citation graphs, respectively. A sample in our case is a shared term between citing and cited documents. The prior entropy of the class distribution is

$$e = -P(C^A)lgP(C^A) - P(C^V)lgP(C^V) \tag{2.3}$$

The posterior entropy of the class distribution when feature f is present in the citation text corpus is

$$e_f = -P(C^A|f)lgP(C^A|f) - P(C^V|f)lgP(C^V|f) \tag{2.4}$$

The posterior entropy of the class distribution when feature f is absent in the corpus is denoted as $e_{\overline{f}}$ and can be found in a similar manner.

$$e_{\overline{f}} = -P(C^A|\overline{f})lgP(C^A|\overline{f}) - P(C^E|\overline{f})lgP(C^E|\overline{f}) \tag{2.5}$$

Thus, the posterior expected entropy is $e_f P(f) + e_{\overline{f}} P(\overline{f})$ and expected entropy loss of feature f is defined as

$$Ent.Loss(f) = e - (e_f P(f) + e_{\overline{f}} P(\overline{f})) \tag{2.6}$$

which is guaranteed to be positive for every feature f.

2.5.2 Feature Weight Adjustment

The citation text corpus T contains the shared words between citing and cited documents in G^A (which is a subset of the original feature space) and we use this subset to realign the document vectors. Expected entropy loss based ranking of the most and least informative words in the corpus T is given in Table 2.1. It can be noted that more meaningful and topic bearing terms rank higher than less informative terms and the expected entropy loss scores act as a metric of topicality of the words. Hence, by integrating the entropy loss information into the document vector representations, it is possible to achieve better separation of the distinct clusters. For each word in T, we update each document vector containing that feature as follows:

$$\bar{w}(d_i, f_j) \leftarrow (1 - \lambda) \cdot w(d_i, f_j) + \lambda \cdot Ent.Loss(f_j) \tag{2.7}$$

for $i = [1 \cdots n]$, $\forall f_j \in \vec{d_i}$. $w(d_i, f_j)$ represents the original Term Frequency-Inverse Document Frequency (TF-IDF) weight of feature f_j in document d_i. This weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The term frequency (TF) of a word in the given document is the number of times a given word appears in that document, normalized by

Table 2.1. Features ranked by decreasing expected entropy loss.

Rank	**Feature**
1.	automata
2.	radio
3.	collapse
4.	realtime
5.	switchboard
6.	tcp
7.	molecular
8.	fluctuate
9.	grayscale
10.	dendogram
...	...
...	...
...	...
5547.	statement
5548.	quinlan
5549.	roth

the total number of terms in the document, given as $TF_i = \frac{n_i}{\sum_k n_k}$. The inverse document frequency (IDF) is a measure of the general importance of the word in the whole collection, defined as $IDF_i = \log \frac{N}{n}$ where N is the total number of documents in the collection and n is the number of documents that contain the word i at least once. TF-IDF weight is the product of TF and IDF scores of the word.

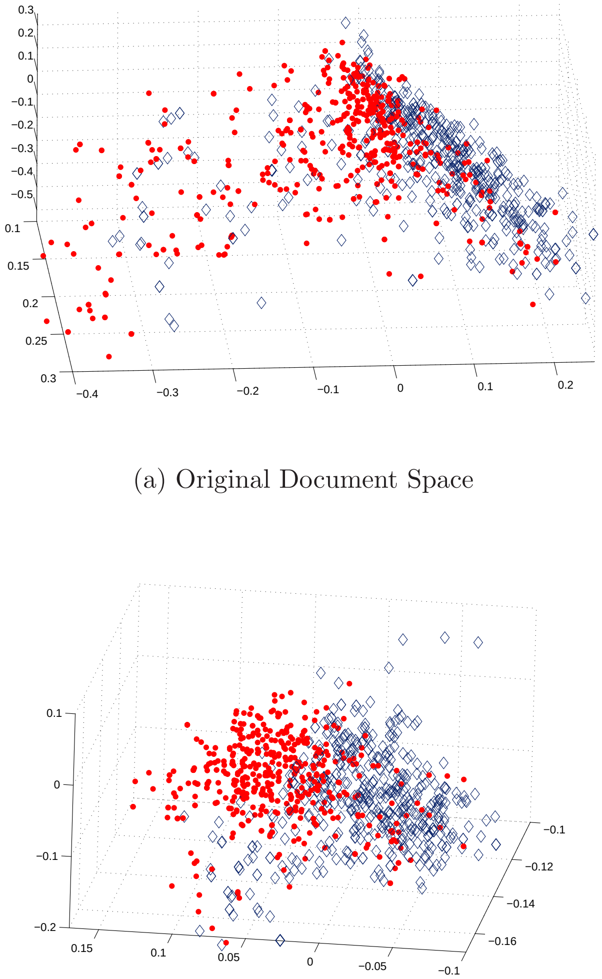
In equation 2.7, λ is a parameter that adjusts the effect of the information gain of the feature on the final weight, which can also be thought of as relative bias of that term in the document. $\lambda = 0$ refers to the original weighting scheme with no topical boosting of the words and $\lambda = 1$ corresponds to purely entropy score based weighting. Hence, λ has the effect of proportionally reducing the significance of the features that don't exist in the citation corpus.

Following the weight updates of the features of all documents, the document vectors are re-normalized to unit length. We then perform clustering on the updated document vectors. A visual representation of the effects of the weight readjustment is shown in Figure 2.3 for categories 1 and 4 of our dataset, which are the two most difficult clusters to separate. Please note that since the orientation of the document vectors are are changed after feature reweighting, the factorization of the term-document matrix obtained from Singular Value Decomposition yields a different mapping of the document vectors onto the 3D space. The representations of the document space before and after the weight readjustment are given from the viewpoint that maximally separates the two clusters and shows the change of the separation of the clusters after augmenting the documents with the expected entropy loss scores of the words in T.

Computationally, given a dataset with N documents, C citations and a text corpus of T, the complexity of generating the similarity matrix and formation of the virtual citation graph is $O(TN^2)$ and the calculation of the expected entropy losses is bounded by $O(CT)$. So the overall complexity of the algorithm is $O(T(N^2+C))$.

2.6 Experiments

We used a selection of 7227 papers from CiteSeer's repository as our dataset. We selected the first 1000 words of each paper, resulting in a text corpus of 9601 distinct features after preprocessing the text by stemming, stop word and infrequent word removal. The clustering is performed both using the original TF-IDF scores of words and the scores augmented by the entropies of the words. A total of 4404 citation relationships exist between the papers in the dataset. The text corpus T of the citation relationships consists of distinct 5549 words. We used the Cluto [47] clustering toolkit in our experiments. Cluto implements some of the most widely



(a) Original Document Space

(b) Document vectors adjusted by Entropy loss

Figure 2.3. Effect of integrating entropy scores of citation corpus with $\lambda = 0.1$. Documents are mapped to 3D space by Singular Value Decomposition (SVD). The SVD transformation reorients the document vectors in the 3D space in Figure 2.3(b). Both viewpoints are selected to show the maximal separation of the clusters.

used clustering algorithms in the literature, including agglomerative, divisive and graph-based techniques and hence, provides good baseline comparisons.

Table 2.2. Dataset Venue Distribution in the CiteSeer Collection. Each Cluster represents a separate topical category.

Cluster 1		Cluster 2		Cluster 3	
Venue	Samples	Venue	Samples	Venue	Samples
AAAI	662	POPL	599	ICCV	682
IJCAI	599	PLDI	664	CVPR	830
ICTAI	232				
total	1493	*total*	1263	*total*	1512

Cluster 4		Cluster 5	
Venue	Samples	Venue	Samples
ICML	990	VLDB	1049
ECML	211		
ML	80		
KDD	629		
total	1910	*total*	1049

2.6.1 Evaluation Metrics

The clustering performance is evaluated by comparing the predicted cluster of each document with the categorical labels (venues) from the document corpus. Since it is not feasible for us to manually label the topic(s) of each paper, the publication venues of the papers are utilized to infer the topics. The papers are split into 5 topical groups based on their publication venues where the venue-group assignments are based on the focus similarity of research topics of the venues. The categorical distribution of the publication venues is shown in Table 2.2. The categories used in our experiments are *Artificial Intelligence*, *Programming Languages*, *Computer Vision*, *Machine Learning* and *Very Large Databases.* To assess the clustering performance of the algorithm, we used the standard F_1 and entropy measures as our evaluation criteria.

2.6.1.1 F_1 Measure

F_1 measure combines precision (p) and recall (r) with equal weight in the formulation given as

$$F_1(p,r) = \frac{2.p.r}{p+r} \tag{2.8}$$

We report results both on Macro-averaged F_1 and Micro-averaged F_1 scores. The key difference between those two F_1 measures is that macro-averaging gives equal weight to each cluster, whereas micro-averaging equally weights each document.

2.6.1.2 Cluster Entropy

The cluster entropy measure shows the distribution of various classes of documents within each cluster. For each cluster C_i of size n_i, the entropy of this cluster is defined as

$$E(C_i) = -\frac{1}{log\ k} \sum_{j=1}^{k} \frac{n_i^j}{n_i} log \frac{n_i^j}{n_i} \tag{2.9}$$

where k is the number of classes in the dataset and n_i^j is the number of documents of the i^{th} class that were assigned to the jth cluster. The entropy of the entire clustering solution is the average of the cluster entropies adjusted by their respective sizes, given by

$$\sum_{i=1}^{k} \frac{n_i}{n} E(C_i) \tag{2.10}$$

A smaller entropy score indicates better clustering solution over the entire dataset.

Table 2.3. Results on four different clustering criterion functions. For each clustering, the left column corresponds to the original similarity space S and the right column represent the citation augmented space $\mathbf{S}^*$

	Internal Similarity		External Similarity	
	$I_{sim}(S)$	$I_{sim}(\mathbf{S}^*)$	$E_{sim}(S)$	$E_{sim}(\mathbf{S}^*)$
$F_1(Micro)$	80.7%	**85.7%**	80.8%	**81.9%**
$F_1(Macro)$	81.5%	**86.7%**	81.4%	**82.3%**
$Entropy$	36.7%	**28.7%**	37.5%	**33.5%**

	Hybrid		Graph-based	
	$H_{sim}(S)$	$H_{sim}(\mathbf{S}^*)$	$G_{sim}(S)$	$G_{sim}(\mathbf{S}^*)$
$F_1(Micro)$	76.1%	**82.5%**	81.5%	**84.4%**
$F_1(Macro)$	76.8%	**83.2%**	81.8%	**84.8%**
$Entropy$	41.8%	**32.8%**	36.3%	**34.3%**

2.7 Results on Four Criterion Functions

We evaluated our algorithm using the following four different similarity criterion functions. Each criterion function represents the objective that we try to optimize for discovering clusters. The first criterion, I_{sim}, is an *internal* similarity metric that tries to maximize the similarity between each document and the centroid of its assigned cluster. The second criterion function, E_{sim}, is an *external* approach that tries to separate the documents of each cluster from the entire collection. The *hybrid* approach, H_{sim}, tries to find a clustering solution by optimizing the inter-cluster (I_{sim}) and intra-cluster (E_{sim}) similarity metrics simultaneously. The final criterion, G_{sim}, uses the similarity graph of the documents and tries to find the optimum cuts of the graph using MinMaxCut algorithm.

$$maximize\ I_{sim}(S) = \sum_{r=1}^{k} \sum_{d_i \in S_r} cos(d_i, C_r) \tag{2.11}$$

Table 2.4. Confusion matrix of the clustering solution without citation augmentation of text corpus

$\lambda = 0$					
	AI	PL	CV	ML	DB
Cluster 1	**1434**	0	26	84	8
Cluster 2	1	**1197**	13	26	7
Cluster 3	65	3	**1004**	190	146
Cluster 4	11	34	579	**1146**	163
Cluster 5	1	29	288	47	**725**

Table 2.5. Confusion matrix of the clustering solution after citation augmentation of text corpus for $\lambda = 0.15$

$\lambda = 0.15$					
	AI	PL	CV	ML	DB
Cluster 1	**1492**	1	139	95	1
Cluster 2	1	**1233**	4	23	9
Cluster 3	9	1	**1188**	276	31
Cluster 4	3	26	326	**1050**	4
Cluster 5	7	2	253	49	**1004**

$$\mathit{minimize}\ E_{sim}(S) = \sum_{i=1}^{k} n_i \frac{\sum_{v \in S_i, u \in S} cos(v,u)}{\sqrt{\sum_{v,u \in S_i} cos(v,u)}} \tag{2.12}$$

$$\mathit{maximize}\ H_{sim}(S) = \frac{I_{sim}}{E_{sim}} \tag{2.13}$$

$$\mathit{minimize}\ G_{sim}(S) = \sum_{m=1}^{k} n_m^2 \frac{cut(S_m, S - S_m)}{\sum_{d_i, d_j \in S_m} cos(d_i, d_j)} \tag{2.14}$$

The results of the clustering solutions using the four criterion functions are given in Table 2.3 for $\lambda = 0$ and $\lambda = 0.15$. S and $\mathbf{S}^*$ refer to the original and updated document similarities, respectively.

In all four criterions, we were able to achieve better clustering solutions using the entropy-based weight adjustments of the features. The most benefit can be

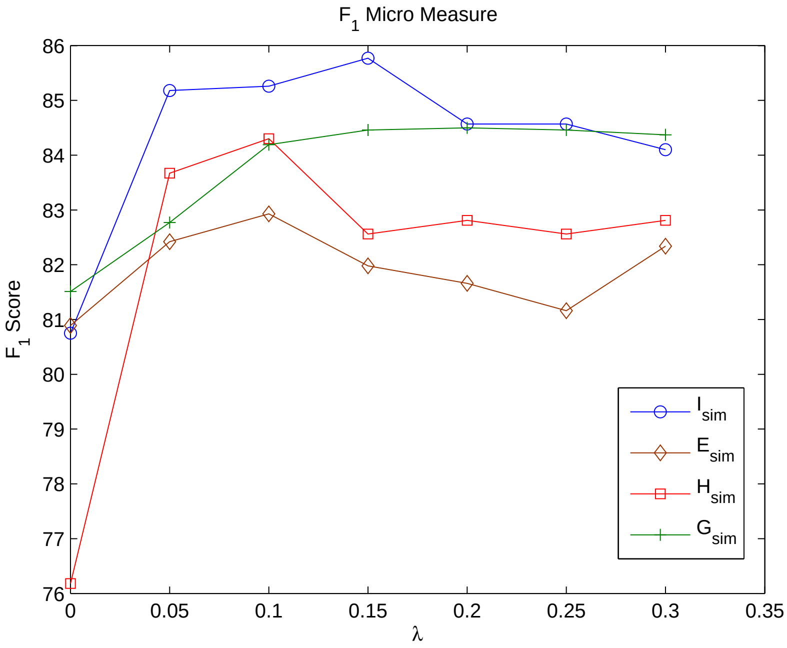


Figure 2.4. F_1 Micro score variation based on λ

observed for I_{sim} and H_{sim} similarity metrics, indicating that similar documents are grouped into more compact clusters. This behavior is expected since the citations we used were mostly to the papers that are in the same category, hence we boosted the weights of the terms that collectively define their respective categories, hence maximizing the internal similarity of the documents of the same cluster. In Figures 2.4 and 2.5, we show the effect of varying λ on F_1 and entropy scores of the clustering solution for all criterion functions. Even for the $\lambda = 0.05$ case which indicates only a slight support from the entropies on the feature values, all four criterion functions achieve significant accuracy improvement. Further increasing λ over a certain point either has no, or negative effect on the clustering solution. Since the entropy values are needed for the *bias* effect on feature weights, increasing λ beyond a certain point starts to cause a dominating effect on the document vectors. In that case, the documents containing just a couple of common topical words (i.e.

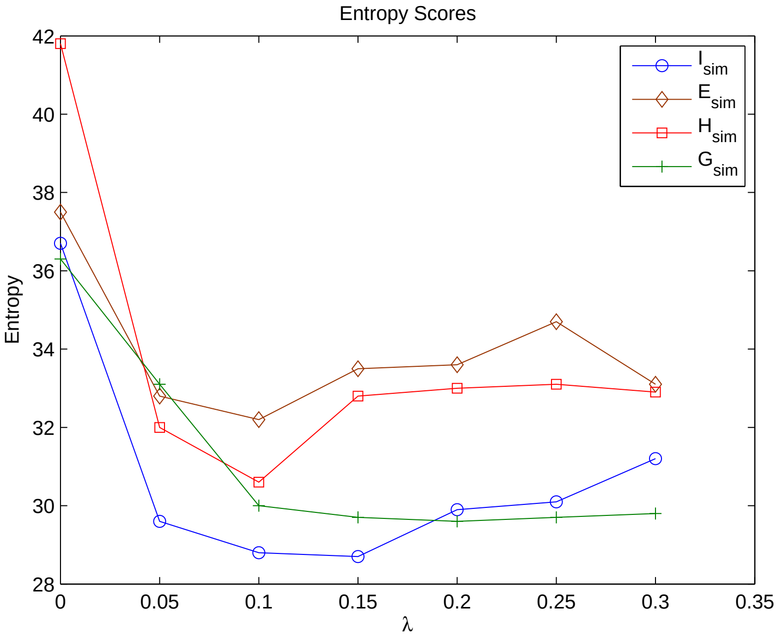


Figure 2.5. Entropy score variation based on λ

"network", "tcp", "learning") tend to group together, causing an adverse effect. It is therefore desirable to keep λ at values that is sufficient enough to boost the importance of the topical words without dominating the content of the papers.

2.8 Concluding Remarks

Most clustering algorithms assume that the components of data objects are independent and identically distributed. This assumption has led to the design of numerous supervised and unsupervised learning algorithms to work on such "flat" data, where each data instance is represented as a fixed length vector of attribute values. For data sets where the data set has richer structure, such as hyperlinks in web documents and citations in scientific literature, an efficient and effective solution to incorporate the connectivity information in the clustering solution yields better clustering performance. In this section, we presented an algorithm that incorporates the citation graph of a collection of scientific literature to the clustering solution to better identify distinct groups of documents. The existence and nonexistence of citation relationships of papers are used to identify the most important topic-bearing words in the papers, based on expected entropy loss measure. We have shown that a feature weighting scheme incorporating the citation-based extraction of topically significant words and applying partial bias for those terms can effectively discover clusters of similar papers.

Chapter 3

Clustering Heterogeneous Datasets using Authorship Information

3.1 Introduction

The task of discovering latent semantic groupings and identification of intrinsic structures in datasets falls in the field of unsupervised learning, also known as clustering, where the goal is to find distinct groups of instances within a data collection. In broad terms, clustering is an optimization problem that tries to find a partition of the data collection such that the items that belong to the same cluster are as similar as possible (cluster compactness) and the discovered clusters are as separate as possible (cluster distinctness) based on a specified (dis)similarity metric within the high dimensional space that the data objects exist. Research in the field of clustering predates the creation and the vast popularity of the world wide web. Due to the size and the characteristics of web-based data, it is desirable to tailor traditional clustering algorithms so that we can process web-based data efficiently and effectively.

Traditional clustering algorithms work on ”flat” data, making the assumption that the data instances to be clustered can only be represented by a set of homogeneous and uniform features, like words in text documents or visual features in image collections. If the data objects have multiple attributes in heterogeneous dimensions, a naïve solution would try to cluster the data objects in each dimension separately and combine the individual clusterings of objects. However, this approach would discard the inherent relationships among the distinct data types and, as we have shown in the previous section, a unified framework that incorporates the heterogeneous types into a single clustering solution has the potential to discover distinct clusters more effectively.

Most real-world data, especially data available on the web, possess rich structural relationships. One of the most pronounced characteristics of web based data is that the data contains *heterogeneous* components; that is, they have multiple types of information that describe the entities that we are interested in clustering, such as authorship and citation graph of scientific documents [35], hyperlinks in web connectivity graph [36] and surrounding text around images in web pages [48]. Discarding the additional sources of information or not utilizing the interplay between multiple aspects of the data can potentially decrease our understanding of the data collection. For example, scientific papers, email messages, blog and newsgroup posts can be directly clustered based on the textual content of the documents using traditional clustering algorithms. One problem with this approach is that it discards the *global view* of the authorship of the documents. Figure 3.1 depicts the document and authorship spaces of a sample document collection. In addition to the textual content of the documents, authors can be represented by the collection of words of the documents they have authored. Since documents authored by the same person tend to be topically similar, we can use this information as an additional dimension of similarity in the clustering process. Combining these two dimensions has the potential to yield better clustering solutions than investigating a single source of similarity in isolation.

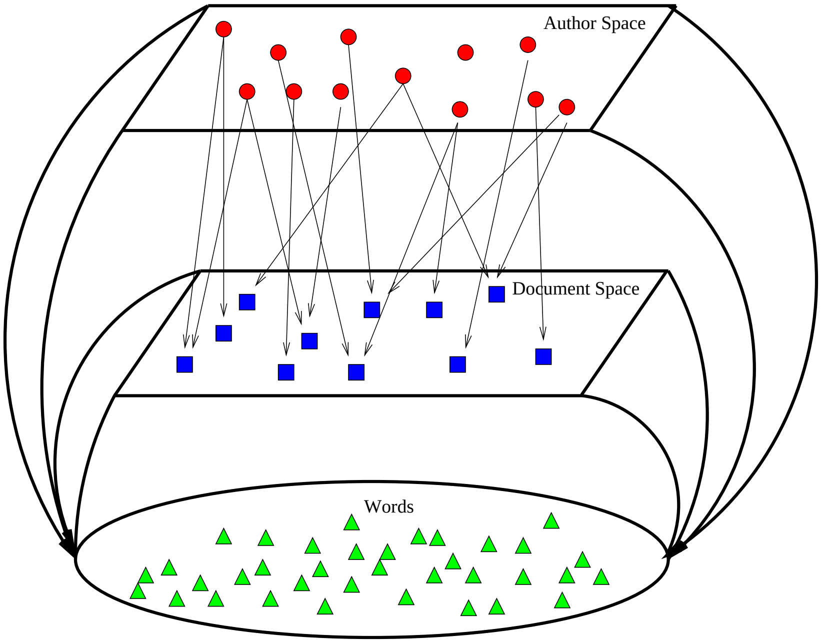


Figure 3.1. Author space and Document space. In addition to the textual representation of documents, each author can be represented as the collection of the words of the documents they have (co)authored.

In this chapter, we present K-SVMeans, a clustering algorithm that integrates the well known K-Means clustering with the highly popular Support Vector Machines(SVM), a machine learning algorithm that has been shown to be highly effective, especially for text classification tasks. K-SVMeans simultaneously clusters documents along one dimension of the data while learning a classifier in another dimension, which, in turn effects the intermediate cluster assignment decisions in the original dimension. The hybrid property of K-SVMeans comes from the fact that it merges and unsupervised learner with a supervised learning algorithm, while eliminating the need for labeled training instances for SVM training. The fundamental difference between unsupervised learning and supervised learning in data mining is that unsupervised learners try to *discover* latent groups of objects without any prior information whereas supervised learners attempt to find a model that can best represent the class assignments of known observations so that the classes of unseen observations can be *predicted* accurately. Thus, in supervised learning, the *supervision* is referred to as the process of obtaining user provided true labels for a set of observations for the training phase. Our clustering framework eliminates the need for labeled training instances for SVM learning. The cluster assignments of K-Means are used to train an Online SVM in a separate data type, and the SVM effects the clustering decisions of K-Means in the primary clustering space. This ping-pong style clustering of heterogeneous datasets effectively increases the clustering performance compared to clustering using a single homogeneous data source.

3.2 Related Work

Although clustering is a decades old problem, research in multivariate data clustering where the data can be represented by multiple interrelated components has gained momentum only in the past couple of years due to its applicability in many domains. The initial directions towards multivariate clustering started with the simultaneous clustering of both rows and columns of contingency tables, also known as coclustering, biclustering, or block clustering. A spectral graph bipartitioning algorithm is proposed in [49] that clusters documents based on words, and words based on documents by finding the normalized cut of the bipartite graph. The same problem has been addressed in [50] by taking an information-theoretic approach. The proposed solution attempts to minimize the loss in mutual information between the original and the clustered contingency tables. [51] computes a partial singular value decomposition of the edge-weight matrix of the bipartite graph to cocluster words and documents. Although these works have laid the foundations of multivariate data clustering, these algorithms can not handle multi-type interrelated data objects.

A multi-type extension of the bipartite spectral graph partitioning has been proposed in [52] for textual datasets and then for images and surrounding texts in web environment [48]. The data objects form a tripartite graph, and the tripartite graph is treated as two separate bipartite graphs. The spectral partitioning of the bipartite graphs is obtained by minimizing the cuts of both bipartite graphs using semi-definite programming in $m + n + t$ dimensional space where each dimension represents the dimension of a separate data type. The high-dimensionality of the problem space is prohibitive and prevents its applicability to real-world datasets of big sizes. [53] provides a multi-way clustering framework that maximizes the mutual information between the clusters of multiple data types based on representation of the interaction between each pair of data types as a contingency table of co-occurence counts. The generation of clusters is performed by a combination of agglomerative (bottom-up) and partitional (top-down) clusterings of different data types. The decision as to which types will be clustered agglomeratively or partitional, and the order of their clusterings is determined by a clustering schedule determined beforehand of the clustering process, and the clustering schedule needs to be provided by the user.

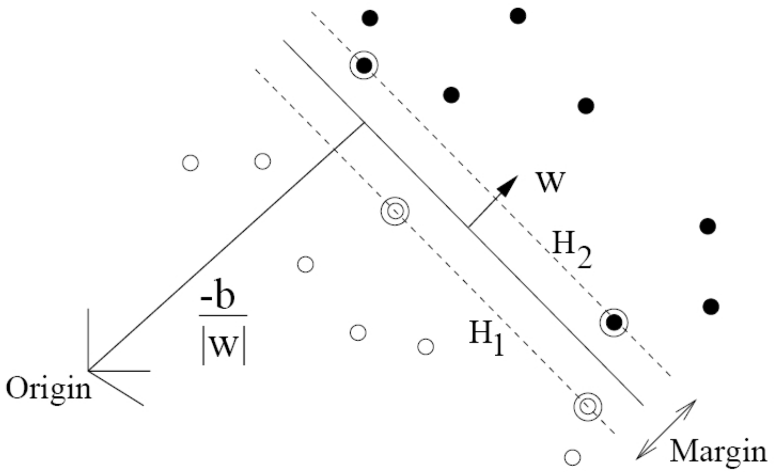


Figure 3.2. A representation of Support Vector Machines. The solid line is the hyperplane which separates two classes with maximum margin. The circled instances are the support vectors, which define the hyperplane.

3.3 Background on Support Vector Learning

A key feature of K-SVMeans is the integration of Support Vector Machines with K-Means clustering. This subsection provides background on SVMs and Online Support Vector Learning, which is a key component in clustering decisions made by K-SVMeans.

SVM is a supervised machine learning algorithm that is well known for its generalization performance and ability to handle high dimensional data which is a common case in document classification problems. Considering the binary classification case, let $((\mathbf{x}_1, y_1) \cdots (\mathbf{x}_n, y_n))$ be the training dataset where $\mathbf{x}_i$ are the feature vectors representing the observations and $y_i \in (-1, +1)$ be the labels of each observation. For example, in spam e-mail filtering applications, spam and regular e-mails may be labeled as +1 and −1, respectively. From these observations, SVM builds an optimum hyperplane – a linear discriminant in the kernel transformed higher dimensional feature space – that maximally separates the two classes by the widest margin by minimizing the following objective function

$$\min_{\mathbf{w},\mathbf{b},\xi_\mathbf{i}} \frac{1}{2}\mathbf{w}\cdot\mathbf{w}^T + C\sum_{i=1}^{N}\xi_i \quad \text{subject to} \quad \begin{cases} \forall i \; y_i(\mathbf{w}^T\mathbf{x_i} - b) \geq 1 - \xi_i \\ \forall i \; \xi_i \geq 0 \end{cases} \tag{3.1}$$

where $\mathbf{w}$ is the norm of hyperplane, b is offset, $y(\mathbf{x}_i)$ are the labels and ξ_i are the slack variables that permit the non-separable case by allowing misclassification of training instances. In practice, the SVM solution is obtained by solving the dual of the convex optimization problem in equation 3.1 by defining the following dual objective function

$$W(\alpha) = \sum_i \alpha_i y_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j K(x_i, x_j) \tag{3.2}$$

and solving the SVM Quadratic Programming (QP) problem:

$$\max_\alpha W(\alpha) \quad \text{with} \quad \begin{cases} \sum_i \alpha_i = 0 \\ A_i \leq \alpha_i \leq B_i \\ A_i = \min(0, Cy_i) \\ B_i = \max(0, Cy_i) \end{cases} \tag{3.3}$$

where α_i are the Lagrange multipliers and $K(\cdot)$ is the Kernel function, which can be an inner product, polynomial, RBF, or any other function that obeys Mercer's condition [54, 55]. The convex quadratic programming (QP) problem in equation 3.3 can be solved by Sequential Minimal Optimization(SMO) [56]. SMO works by optimizing two α_i's at a time while holding the remaining α_i's fixed, solving the optimization problem analytically. The most significant benefits of being able to solve the QP problem analytically is increased efficiency of SVMs, the ability to handle massive datasets and thus make it practical for real-world applications, and make way to *online* learning methods.

LASVM [57] is an online SVM algorithm that does not require that all the labeled training instances be presented to the learner before the SVM training phase, hence it can incrementally build a learner as opposed to batch learning. Based on the SMO optimization of Eq. 3.3, LASVM updates its model whenever new observations are available to the learner. Once a training instance is presented to LASVM, the learner searches its set of existing support vectors that maximizes the dual function, and adjusts both α's by the maximal step size of the optimization, while maintaining the constraint $\sum_{i=1}^{N} \alpha_i = 0$.

3.4 K-SVMeans for Document Clustering

The ability to use SVM in an online setting enables us to integrate it into unsupervised learning algorithms, and as will be shown later, this approach eliminates the need to use labeled training data for SVM although that is a *requirement* for supervised learning algorithms, hence the term *supervision*. In this section, we present K-SVMeans, a K-Means-based clustering algorithm that clusters documents based on their textual content where cluster assignment decisions are based on documents' distances to the clusters as well as the classification of the documents' authors to the SVM learner of each cluster.

3.4.1 K-Means Clustering

The original formulation of K-Means algorithm first initializes k clusters with N documents and then assigns each document $d_i,\ 1 \leq i \leq N$ to a cluster $c_i,\ 1 \leq i \leq k$ where d_i's distance to the representative of its assigned cluster c_i is minimum. Variants of K-Means algorithm differ in the initialization of clusters (e.g. random or maximum cluster distance initialization), the definition of similarity (e.g. Eucledian or Kullback-Leibler Divergence), or the definition of cluster representativeness (e.g. mean, median or weighted centroid vector). K-SVMeans algorithm is independent of any of those variations, but for brevity, we describe the algorithm for Spherical K-Means with random initialization that represents each cluster by its centroid vector. Given n documents $\mathbf{x_1}, \mathbf{x_2} \cdots \mathbf{x_n}$ $\forall \mathbf{x_i} \in \mathbf{R}^w$ where w is the size of the text corpus and each $\mathbf{x_i}$ is normalized such that $||\mathbf{x_i}|| = 1$. K-Means partitions the document $\mathbf{x_i}$ into k disjoint clusters $\pi_1, \pi_2, \cdots, \pi_k$ so that

$$\bigcup_{i=1}^{k} \pi_i = \{\mathbf{x_1}, \mathbf{x_2}, \cdots \mathbf{x_n}\} \quad \text{where} \quad \pi_i \cap \pi_j = \emptyset, \;\; i \neq j$$

where the centroid c_i of each cluster π_i is defined as

$$c_i = \frac{\sum_{\mathbf{x_k} \in \pi_i} \mathbf{x_j}}{|| \sum_{x_j \in \pi_i} \mathbf{x_j} ||} \tag{3.4}$$

The goal of the clusterer is to maximize the similarity between the data objects and their assigned clusters, hence, the objective function becomes

$$\max Q = \sum_{j=1}^{k} \sum_{\mathbf{x_i} \in \pi_\mathbf{j}} \mathbf{x_i^T} \cdot \mathbf{c_j} \quad \forall \pi_i \;\; 1 \leq i \leq k \tag{3.5}$$

K-Means optimizes the objective function iteratively by following two steps: A cluster assignment step, where each document is assigned to a cluster with the closest centroid, followed by a cluster centroid update step.

Original K-Means Algorithm

input: $\mathbf{X}=(\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_n})$ *Documents*

N : *Number of Clusters*

output: C=$\{c_1, c_2, \cdots, c_k\}$ *Cluster Centroids*

m: $X \rightarrow C$ *Cluster Assignments*

Procedure K-Means

```
Randomly assign documents to clusters and initialize cluster centroids.
Initialize iteration count $t = 0$
 Do
   $t \leftarrow t + 1$
   For each $x_i \in X$
     $m(x_i) = arg\, \max_{j=[1 \cdots k]} c_j^{(t)} \cdot x_i$
   End
   For each cluster $\pi_i$ with centroid $c_i^{(t)}$
     $c_i^{(t)} = \frac{\sum_{x_j \in \pi_i} \mathbf{x_j}}{||\sum_{x_j \in \pi_i} \mathbf{x_j}||}$   // Update centroids
   End
 While Obj. func. change > threshold
```

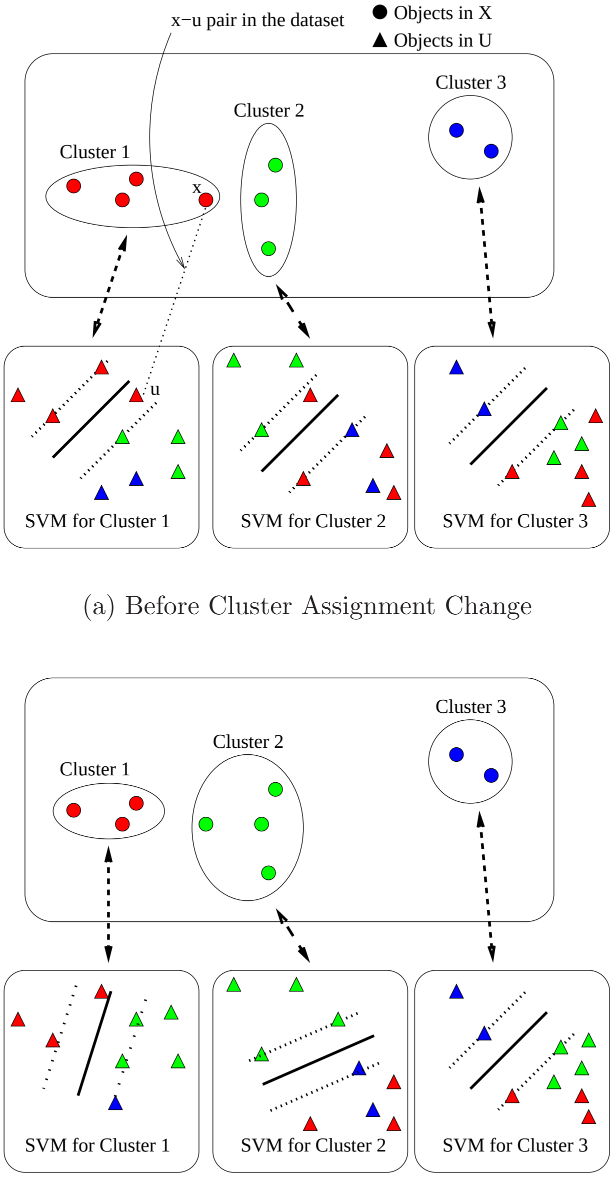
The algorithm terminates when the change in the objective function value between two successive iterations is below a given threshold. Upon the termination of the algorithm, each document belongs to one of the k clusters. This partitioning, however, is done on a single dimension, that is, the documents are clustered based on the textual content of the documents. In K-SVMeans, we integrate the support vector learning into the iterations where intermediate cluster assignments are decided by both the documents' distances to the cluster centroids and the SVM learners' belief that the document authors are part of that cluster as well.

3.4.2 K-SVMeans Clustering

Consider that the instances in the set $X = (x_1, x_2, \cdots, x_n)$, which we want to obtain a clustering solution, are related to another set $U = (u_1, u_2, \cdots, u_m)$ in some way. Each x_i can be related to one or multiple u_j's in a $X \to U$ mapping where objects in U denote a unique property of x_i. The reverse map $U \to X$ lets us represent each u as a mixture of the x_i's that are connected to it. This definition can be visualized in Figure 3.1 where the documents are represented as the objects in X and the authors are represented as the objects in U.

During the clustering process, the intermediate cluster assignments in K-SVMeans are determined by two conditions. First, a data object x_i is assigned to a cluster when both its similarity to the cluster centroid and the SVM learners (of the cluster x_i belongs to and the new candidate cluster) agree on that cluster assignment. Secondly, in case the candidate cluster's SVM learner decides that the objects in U that are connected to x_i do not belong to that cluster (i.e. the decision values of the u's are negative), then we apply a penalty term ($\lambda > 1$) on the distance function of K-Means so that the similarity between x_i and the candidate cluster centroid must be strong enough to warrant a cluster assignment change of x_i. The penalty term also ensures us that the SVM classifiers are not adversely effected by incorrect clustering decisions of K-Means that result in mislabeling of the u_j. Only highly similar x_i are allowed a cluster change in case the SVM classification decision is not trusted.

A representation of the clustering in K-SVMeans is given in Figure 3.3. Each cluster has an associated SVM learner that is trained during the clustering process. In the graphical example, the object x is closer to Cluster 2, and its mapping u is misclassified in the SVM of Cluster 1, and correctly classified in the SVM of Cluster



(a) Before Cluster Assignment Change

(b) Document assigned to Cluster 2. The document's author added as a positive observation to Cluster 2, and negative observation to the rest of the clusters.

Figure 3.3. Cluster assignment and SVM update in K-SVMeans for three clusters

2. The algorithm, therefore, assigns x to Cluster 2, updates the learners of all learners to reflect the cluster change of x. Depending on the model characteristics

of each cluster's SVM learner, the change of status of u (moving from one cluster to another) may or may not effect the learners. Note that the label change of u can only potentially effect either the old class of u (where it's label changed from +1 to -1), or it's new class (where the label changed from -1 to +1). The learners of rest of the clusters are not effected by this cluster reassignment.

K-SVMeans can be run in multiple iterations where the SVM learner initialization is performed by using the clustering solution generated in the previous run. In the first iteration, we run standard K-Means algorithm to yield a clustering based on the primary space X. This iteration has two purposes. First, we use the clustering result from this step as a baseline for comparison. Second, and more importantly, it generates the labeled initialization set for the SVM learners of K-SVMeans. In the beginning of an iteration $t+1$, we look at each cluster π_i^t generated in the previous run and select m objects closest to the centroid of π_i^t and use their associated u_i for SVM initialization. We use one-against-rest classification in the SVMs, so the u's become positive observations for their respective clusters, and negative observations for the rest of the clusters. Although it is possible that the previous iteration may have assigned some of the x_i's to incorrect clusters, the x_i's that are closest to the centroids are more likely to be correctly assigned to their correct clusters whereas the incorrect assignments tend to appear towards the boundaries of the clusters. One thing to note about K-SVMeans is that the optimization of the objective function of K-Means retains its non-increasing characteristics, and the algorithm is guaranteed to converge to a local minima.

K-SVMeans Cluster Assignment

Definitions:

x_i: *Objects to be clustered*

d_{ij}: *distance of object x_i to cluster π_j*

$m(i)$: *assigned cluster of x_i*

$l(\pi_i)$: *SVM learner of cluster π_i*

$\hat{y}(u,\pi) = \sum_{z=1}^{n} \alpha_z^{\pi} \mathbf{K}^{\pi}(u, u_z^{\pi}) + b^{\pi}$ *SVM decision value for u for cluster π*

λ: Penalty term

```
For each $x_i \in X$
  $d = x_i \cdot c_{m(i)}$ , $\pi_i \leftarrow m(i)$
  $s = \sum_{\forall u_k ,T_{ik}=1} \hat{y}(u_k, \pi_i)$
    For each cluster $\pi_j$, $i \neq j$
      $\hat{d} = x_i \cdot c_j$
      $\hat{s} = \sum_{\forall u_k ,T_{ik}=1} \hat{y}(u_k, \pi_j)$
        If ($\hat{d} < d$ and $s < 0$ and $\hat{s} > 0$) or
          ($\hat{d} \cdot \lambda < d$ and $\hat{s} < 0$)
          Remove u's related with $x_i$ from $l(\pi_i)$
          Insert u's related with $x_i$ to $l(\pi_j)$ as +1
          Insert u's related with $x_i$ to $l(\pi_p)$ as -1, $j \neq p$
          $m(x_i) \leftarrow \pi_j$
        End
    End
End
```

3.5 Datasets and Evaluation Metric

We conducted experiments on a subset of CiteSeer's paper collection and on the 20 Newsgroup dataset - a collection of approximately 20,000 newsgroup documents, partitioned across 20 different newsgroups - to evaluate the clustering performance of K-SVMeans by comparing the predicted cluster of each document with the categorical labels from the document corpus. The labels are inferred from the publication venues for the CiteSeer collection, and from the newsgroup directories of the postings for the 20 Newsgroup dataset. In both datasets, we are interested in the effect of the authorship of documents for the topical clustering of documents. We use the standard micro-averaged F_1 measure as our evaluation criteria, which gives equal weight to each document, regardless of the cluster sizes. The characteristics of the datasets and the results are presented in the following subsections. We selected the RBF kernel for the online SVM and ran experiments with the SVM parameters $C = 100$ and $\gamma = 0.001$ after 10-fold cross-validation. For the K-Means clustering section of K-SVMeans algorithm, we used the Gmeans clustering toolkit [49], which we integrated with the LASVM package [57].

3.5.1 CiteSeer Dataset

The first dataset we used is a collection of scientific papers obtained from CiteSeer's repository. The categorical distribution of the subset of papers from CiteSeer's collection we used in our experiments is given in Table 3.1. From each paper, we extracted the title, abstract and keyword sections, and removed the stop words. We also removed the words that appear less than three times in the whole collection. In the corpus that we used, there are a total of 7623 papers that have been authored by 5623 distinct authors. Each author is represented as a collection of the words

CITESEER DATASET					
Cluster 1		Cluster 2		Cluster 3	
Venue	Samples	Venue	Samples	Venue	Samples
AAAI	606	SIGCOMM	680	EUROCRYPT	379
IJCAI	961	INFOCOM	1109	CRYPTO	265
ICTAI	207			ASIACRYPT	145
total	1774	*total*	1789	*total*	789

CITESEER DATASET			
Cluster 4		Cluster 5	
Venue	Samples	Venue	Samples
POPL	803	KDD	607
ASPLOS	300	PKDD	134
ECOOP	316	CIKM	392
ICLP	296	SIGIR	423
total	1715	*total*	1556

Table 3.1. CiteSeer Dataset Venue Distribution among five topical clusters that are used in our experiments

in the documents that he/she has (co)authored. Since there is a one to many relationship between the documents and the authors, we integrated the effect of order of authorship in the representation of authors in vector form. The weight of feature f_i of author vector $\vec{a_i}$ is

$$\vec{\mathbf{a}}_i^{f_j} = \sum_{a_i \in d_k} \frac{1}{Rank(a_i, d_k)} \cdot w(f_j, d_k) \tag{3.6}$$

where $Rank(a_i, d_k)$ is the rank of authorship of author a_i in document d_k and $w(f_j, d_k)$ is the TF-IDF score of feature f_j in d_k. The author vectors are L_2 normalized to eliminate the effects of different document lengths and different number of authored documents.

20 NEWSGROUP DATASET		
Cluster 1	Cluster 2	Cluster 3
Religion	**Hardware**	**Politics**
talk.religion.misc soc.religion.christian alt.atheism	sci.electronics comp.sys.mac.hardware comp.sys.ibm.pc.hardware	talk.politics.guns talk.politics.mideast talk.politics.misc
2424 posts	2924 posts	2625 posts

20 NEWSGROUP DATASET	
Cluster 4	Cluster 5
Software	**Sports**
comp.os.ms-windows.misc comp.graphics comp.windows.x	rec.sport.hockey rec.sport.baseball
2938 posts	1993 posts

Table 3.2. 20 Newsgroup Topic Distribution among five topical clusters that are used in our experiments

3.5.2 20 Newsgroup Dataset

The second collection we used is the 20 Newsgroup dataset. Each message is authored by one author and is about a single topic. We used the reduced version of the dataset[1] where the cross-posts in the collection are removed, and the messages only contain the *From* and *Subject* fields in addition to the message body. We combined 14 of the newsgroups in 5 categories for a total of 12,904 messages submitted by 5,992 people. The list of newsgroup topics we used and their categorical groupings are given in Table 3.2. Each unique person is identified from the email address found in the *From* fields of the messages. Since each message has only one sender, each author has a straightforward representation of the cumulative collection of words found in all of that person's messages. Each message is cleaned from stop words, infrequent words less than three occurences are removed,

[1] Dataset available at `http://people.csail.mit.edu/jrennie/20Newsgroups/`

and vectors normalized to unit length. The author vectors are constructed from their corresponding document vectors, and the lengths of author vectors are also normalized to one to account for different number of postings of authors.

3.6 Experimental Results

We report results on each dataset for two clustering criterion functions for K-Means, averaged over ten runs. The first clustering algorithm is the Euclidean K-Means that makes the cluster assignment decisions based on the Euclidean distances between the document vectors. The second algorithm we used is the Spherical K-Means that uses the cosine distances between documents as the similarity metric.

For both clusterings, we experimented with two separate initialization schemes. In the first scheme, each document is assigned a random cluster ID to initialize the clusters. The second scheme chooses one of the cluster centroids as the farthest point from the center of the whole data set, and all cluster centroids are well separated.

In each experiment, following the completion of K-Means, K-SVMeans initializes each cluster's SVM learner with the authors of 50 documents that are closest to the cluster centroids in the first run. In each successive iteration, we increase our confidence in the clustering achieved in the previous K-SVMeans run, and we increase the number of authors that are used for SVM initialization by %50 of the previous run. The penalty term that accounts for SVM misclassification of authors for the clustering distance function of the documents is empirically set to 1.5.

From Table 3.3, it can be seen that we were able to outperform K-Means clustering results, regardless of the clustering criterion function, or the initialization

CiteSeer Dataset				
Distance / Cl. Init.	**K-Means**	**K-SVMeans (x1)**	**K-SVMeans (x2)**	**K-SVMeans (x3)**
Spherical / Random	68.418	73.318	76.102	**76.194**
Spherical / Well Sep.	69.306	75.243	77.713	**80.596**
Euclidean / Random	55.945	60.284	61.575	**62.082**
Euclidean / Well Sep.	58.712	64.392	65.941	**66.746**

20 Newsgroup Dataset				
Distance / Cl. Init.	**K-Means**	**K-SVMeans (x1)**	**K-SVMeans (x2)**	**K-SVMeans (x3)**
Spherical / Random	70.792	75.486	76.918	**77.315**
Spherical / Well Sep.	72.314	77.263	78.368	**78.764**
Euclidean / Random	52.623	54.978	55.711	**56.013**
Euclidean / Well Sep.	53.747	55.549	56.292	**56.426**

Table 3.3. Experimental Results based on the F_1 scores of the clusterings. Clustering results for K-Means are used for baseline comparison as well as generating the labeled initial author seeding set for K-SVMeans(x1). The two initialization schemes, random and well separated centroids, are used for K-Means initialization and each successive iteration uses the same initial clusters for documents.

scheme of K-Means. K-SVMeans(x2) and K-SVMeans(x3) are the second and third iterations of K-SVMeans clustering, respectively. The inclusion of more and more authors to the SVM initialization set in each successive iteration enables the learners to build accurate models earlier in the clustering solution, and thus, increases the clustering accuracies. It can be observed that we have obtained higher improvement in clustering accuracies for CiteSeer dataset than the Newsgroup collection. In scientific publications, researchers generally target the publication venues that lie within their research interests. Therefore, it is easier for the SVM learner to predict the category that a particular author is interested in. On the other hand, since the newsgroup messages are comparably more random, and are based on personal interests which may span multiple topics, a person's messages may be more distributed across topics, making it difficult for the classifier to make

K-Means					
	AI	**COMM**	**CRYPT**	**PL**	**DM**
Cluster 1	**681**	15	11	23	667
Cluster 2	13	**1697**	3	200	23
Cluster 3	428	20	**762**	263	33
Cluster 4	86	23	0	**1103**	43
Cluster 5	566	34	13	126	**790**

K-SVMeans(x3)					
	AI	**COMM**	**CRYPT**	**PL**	**DM**
Cluster 1	**1271**	8	10	70	69
Cluster 2	17	**1658**	1	113	22
Cluster 3	25	56	**770**	42	8
Cluster 4	132	23	1	**1444**	43
Cluster 5	329	44	7	46	**1414**

Table 3.4. CiteSeer confusion matrix of clustering for standard K-Means and K-SVMeans(x3) for a sample run. The topical clusters are AI : Artificial Intelligence, COMM: Communications, CRYPT: Cryptography, PL: Programming Languages, DM: Data Mining

as accurate predictions as the case for scientific domain. Even in that case, the SVM classification of newsgroup authors was able to assist the clustering decisions made by the K-Means and improve the clustering accuracies. In Tables 3.4 and 3.5, we show the confusion matrices of CiteSeer and Newsgroup datasets where K-SVMeans has outperformed K-Means by a wide margin. In the CiteSeer dataset, as can be seen in the confusion matrices, clusters Artificial Intelligence and Data Mining contain many documents that have been incorrectly assigned to each other. The same problem can be observed with the Hardware and Software categories of the Newsgroup dataset. The problem stems from the narrow focus of K-Means which only looks at each document in isolation. The presence of common terms between those categories misleads the K-Means clusterer to make incorrect clustering decisions. A global view that considers the main interests of an author by

K-Means					
	REL	**HW**	**POL**	**SW**	**SP**
Cluster 1	**2121**	10	153	11	9
Cluster 2	33	**2165**	114	2041	36
Cluster 3	175	691	**1132**	165	20
Cluster 4	93	41	1206	**711**	1
Cluster 5	2	17	20	10	**1927**

K-SVMeans(x3)					
	REL	**HW**	**POL**	**SW**	**SP**
Cluster 1	**2144**	10	205	21	10
Cluster 2	29	**2478**	39	314	23
Cluster 3	228	50	**2367**	27	4
Cluster 4	17	371	5	**2572**	28
Cluster 5	6	15	9	4	**1928**

Table 3.5. Newsgroup confusion matrix of the clustering for standard K-Means and K-SVMeans(x3) for a sample run. The topical clusters are REL : Religion, HW: Hardware, POL: Politics, SW: Software, SP: Sports

looking at all of the content generated by that person helps us correctly model his/her interests and enables us to gain better understanding of the nature of the information produced by that author.

3.7 Concluding Remarks

Traditional clustering algorithms do not handle rich structured data well by either focusing on a single homogeneous type or by not considering the interrelationships between the multiple aspects of the data. Therefore, those algorithms are not sufficient to deal with the existing (and emerging) data that is heterogeneous in nature, where relationships between objects can be represented through multiple layers of connectivity and similarity. In this chapter, we presented a novel clustering algorithm, K-SVMeans, which is designed to perform clustering on rich structured multivariate datasets. We have shown that the applicability of Support Vector Machines are not limited to classification problems and SVM classification can greatly effect the performance of clustering algorithms for multivariate datasets. Even in the absence of labeled training instances for SVM, generating labels on-the-fly effectively increases clustering performance. Our experimental results on the integration of authorship analysis with topical clustering of documents show substantial improvements over traditional K-Means and confirms that there is great benefit in incorporating additional dimensions of similarity into a unified clustering solution.

Chapter 4

Generative Model for Topic Discovery in Scientific Literature

In the previous chapters, we presented clustering algorithms for document collections with multi-type interrelated components and showed the effectiveness of integrating multiple sources of information for unified clustering solutions. In this chapter, we present a generative model of documents that utilizes the heterogeneous components for topic discovery in scientific literature, with a special emphasis on the temporal ordering of the documents.

Automatic identification of semantic content of documents has become increasingly important due to its effectiveness in many tasks, including information retrieval, information filtering and organization of documents collections in digital libraries. Identifying the topics that a document addresses increases our understanding of that document, the characteristics of the collection as a whole and the interplay between distinct topics. In domains where the temporal ordering of documents is not of importance, studying a snapshot of the collection at any given time is sufficient to deduct as much information as possible about the various topics of interest in the collection. However, many document collections exhibit temporal relationships and analysis of the temporal dimension of the collections has become an important field of study in many applications, including weblog topic mining [58], evolution of author and paper networks [59], news event analysis [60] and social interaction of researchers [61, 62]. Scientific literature is one of the fields that exhibits strong temporal relationship between the documents and the popularity of the topics that are addressed in the papers change over time. Figure 4.1 shows the number of times that three Computer Science terms have been mentioned in the abstracts of the articles published by ACM from 1990 until 2004. Among those terms, the popularity of *expert systems* has been on a steady decline, whereas number of papers that mention *Hidden Markov Model* and *Support Vector Machine* has been constantly increasing, where *Support Vector Machine* is virtually non-existent in the collection until 1997, according to ACM repository. It is clear that popularity of topics vary over time, new topics emerge while some cease to exist. Thus, simulating such dynamics through the integration of the temporal order of documents into the topic discovery process can potentially yield more accurate topics.

Latent Dirichlet Allocation (LDA) [63] has been shown to be a highly effective unsupervised learning methodology for finding distinct topics in document collections. LDA is a generative process that models each document as a mixture of topics where each topic corresponds to a multinomial distribution over words. The learned document-topic and topic-word distributions are then used to identify the best topics for the documents and the most descriptive words for each topic. However, the original formulation of LDA focuses on analyzing the snapshot of collections and treat the collection as being generated at a single point in time. This approach makes the simplifying assumption of the *exchangeability* of documents,

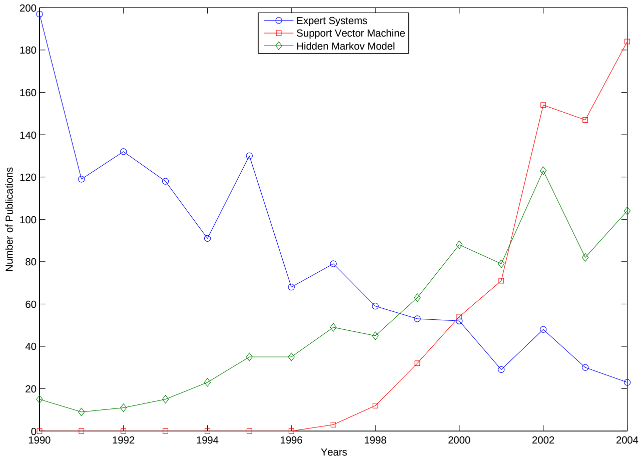


Figure 4.1. Analysis of the popularity of sample key phrases mentioned in academic papers published by ACM over 15 years.

meaning that the documents do not exhibit any particular order. The discovered document-topic and topic-word distributions are obtained as a result of the generative process on the whole corpus at once and the temporal ordering of the documents are ignored in the model. Finding the evolutionary characteristics of topics then involves going back to the probability distributions of documents over topics obtained from the generative process and mapping out topic probabilities over time based on the timestamps of documents. This approach can potentially suffer from *topic dilution*, where all topics are discovered based on our current observations. Clearly, following the observation from Figure 4.1, it would be easier for the generative process to discover expert systems as a topic, or at least a strong component of a more general topic, in the year 1990 alone, than finding it in the whole collection of articles published over 15 years.

In this chapter, we present a generative model of documents, namely Segmented Author-Topic Model (S-ATM), that utilizes the temporal ordering of documents to assist the process of topic discovery. S-ATM is based on the Author-Topic Model

[64, 65] and extends it to integrate the temporal characteristics of the document collection into the generative process. Although S-ATM is equally applicable to the standard Topic-Model [63], as we have shown in Chapter 3, authorship of documents is an effective ingredient for topic discovery in document collections [66], since the authorship layer of the collection adds another layer of similarity between documents and acts as a glue that connects different documents with similar topics. S-ATM first segments the document collection into desired time intervals and iteratively builds a model starting from the earliest date to the latest time segment. Once the topics for a time segment are discovered, the generative process for the next time segment starts with the initialization of the parameters that depend on the characteristics of the topics discovered in the previous time segments.

Most of the algorithms that are concerned with document topic analysis consider only the textual content of documents. In our work, we augment the text corpus of the articles with user queries from CiteSeer and user assigned tags to the papers from CiteULike[1]. Further, we utilize the citation relationship between papers to discover the *topicality* of words and boost the weight of those words to improve the quality of the discovered topics.

4.1 Related Work

Numerous statistical approaches for modeling text documents have been proposed for modeling text documents. Probabilistic latent semantic analysis (pLSA) [67] models the generation of each document through activating multiple topics where each topic is a multinomial distribution over words. This model improves upon

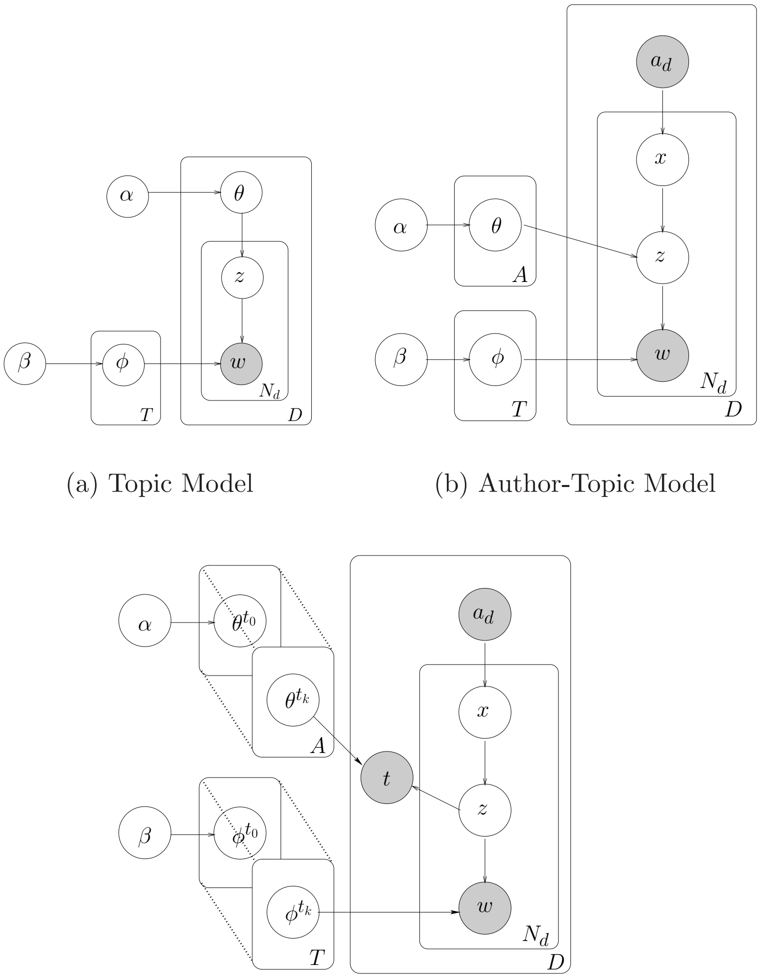
[1] http://www.citeulike.org

the singular value decomposition based latent semantic analysis (LSA) [68] which can not handle polysemy. pLSA, on the other hand, uses a distribution indexed by the training documents, leading to the fact that the number of parameters to be estimated grows linearly with the number of training documents in the collection. Thus, practical applications with large training documents are susceptible to overfitting with this model.

Latent Dirichlet Allocation (LDA) [63] is another generative model that has become popular in recent years that overcomes the overfitting problem of pLSA by using a Dirichlet distribution for modeling the distribution of topics for each document. The generative process of LDA is shown in Figure 4.2(a) in plate notation [69] where boxes represent replicates, the light-colored circles are latent variables and the shaded circles are the observed variables. Given a collection of text documents $D = \{d_1, \cdots, d_D\}$ with words from a corpus $W = \{w_1, \cdots, w_{N_d}\}$ and T topics, the generation of a document follows a three step process: First, for each document d_i, LDA samples a Dirichlet distribution of over topics for that document. Second, for each word w_j in d_i, a topic is chosen from the topic distribution θ, and finally, a word is sampled from the multinomial distribution ϕ of topics over words in W. Such a generative model has been shown [70] to be effective in terms of discovering topics in scientific literature.

The author-word model [71] takes an author-focused approach for modeling each document. Given a set of authors a_d as the authors of document d, to generate each word in d, an author is randomly sampled from a_d and a word is chosen from an author-specific distribution over words for that author. Subsequent works [64, 65] provide an author-topic model which combines the author-word model with topic-word model. The author-topic model states that a generated document is a product of the mixture of the topics of its authors, where each word is generated by the activation of one of the topics of an author of that document. To generate a word w_j in document d_i, an author x is drawn uniformly from a_d and a topic z is generated from the multinomial distribution θ_x of this author. Finally, a word is generated from z's distribution over words. The author-topic model subsumes the author-word and topic-word models as special cases, where the author-word model represents the condition of each author having a unique topic and the topic-word model represents the case where each document has a single author. Erosheva et al. [72] define a mixed-membership model that merges paper abstracts with bibliographic citations to discover categories of publications. All these aforementioned models only consider a snapshot of the document collection generated at the time of the modeling process. The topic dynamics through the temporal characteristics, therefore, is not accounted in the model, which leads to discovering *synthetic* topic evolution by looking at the timestamps of the documents after the topical modeling stage.

Recent work by Blei and Lafferty [73] capture topic dynamics through defining an iterative process that learns the mixture parameters of topics for each time slice in the collection, and propagates the topic distributions to the next iteration by evolving the distributions with Gaussian noise. An approximate inference on the parameters is difficult using Gibbs sampling, hence the parameters are estimated either by variational Kalman filtering or variational wavelet regression. Each stage in the training process is dependent on the set of parameters estimated in the prior stage and the model assumes that topics follow a natural evolution themselves. Topics over Time (TOT) [74] is another LDA-based generative process that models time jointly with word co-occurence patterns. The non-Markov continuous time approach enables the algorithm to not discretize time, which also results in the generation of a timestamp for each word in the document, leading to



(a) Topic Model

(b) Author-Topic Model

(c) Segmented Author-Topic Model

Figure 4.2. The graphical representation of the Topic Model, Author-Topic model and Segmented Author-Topic Model using plate notation.

multiple distinct timestamps for each document. These models do not consider the authorship information of the documents and focus on the evolution of the topics in isolation. In this paper, we take an author-specific approach for modeling topic evolutions in scientific literature and model the state transitions as a mixture of the parameters estimated in prior iterations.

4.2 Segmented Author-Topic Model for Document Collections

The Segmented Author-Topic Model (S-ATM) eliminates the exchangeability assumption of the traditional Author-Topic Model (ATM) by sequential modeling of the documents. The model segments the document collection into time slices $\mathsf{t}_k = \mathsf{t}_0 + k\Delta\mathsf{t}$ where t_0 denotes the earliest timestamp, $\Delta\mathsf{t}$ is the size of time slice and $k = [0 \cdots n]$ is a particular time segment.

In S-ATM, the generation of a document starts with a group of authors $\mathbf{a_d}$ deciding on writing a document d. Each topic has a multinomial distribution over words and each author has a multinomial distribution over topics. A document with multiple authors has a distribution over topics that is a mixture of the topic distributions of authors. For each word w in document d, an author of d is chosen uniformly from the set of authors a_d of the document, and a word is generated through sampling a topic from the multinomial distribution of the chosen author over all topics. In the model, author-topic distributions θ have a symmetric Dirichlet prior with a hyperparameter α and word distributions of topics ϕ have a symmetric Dirichlet prior with a hyperparameter β. The generative process in S-ATM is conceptually similar to ATM in which we extend ATM to maintain a ”memory” of learned distributions from past observations and utilize θ and ϕ distributions from earlier iterations as prior knowledge for subsequent iterations. The next section provides details on Gibbs sampling for S-ATM and the propagation of the model parameters.

4.2.1 Gibbs Sampling for Estimation of the Model Parameters

Gibbs sampling is an algorithm to approximate the joint distribution of multiple variables by drawing a sequence of samples. As a special case of the Metropolis-Hastings [75] algorithm, Gibbs sampling is a Markov chain Monte Carlo algorithm and is usually applied when the conditional probability distribution of each variable can be evaluated. Rather than explicitly parametrizing the distributions for variables, Gibbs sampling integrates out the parameters and estimates the corresponding posterior probability.

In Gibbs sampling, a Markov chain is formed, the transition between successive states of which is simulated by repeatedly drawing a topic for each observed word from its conditional probability on all other variables. In S-ATM, the algorithm goes over all documents word by word. For each word w_i, the topic z_i and the author x_i responsible for this word are assigned based on the posterior probability conditioned on all other variables: $P(z_i, x_i | w_i, \mathbf{z_{-i}}, \mathbf{x_{-i}}, \mathbf{w_{-i}}, \mathbf{a_d})$. $\mathbf{z_i}$ and $\mathbf{x_i}$ denote the topic and author assigned to w_i, while $\mathbf{z_{-i}}$ and $\mathbf{x_{-i}}$ are all other assignments of topic and author, excluding current instance. $\mathbf{w_{-i}}$ represents other observed words in the document set and $\mathbf{a_d}$ is the observed author set for the document.

A key issue in using Gibbs sampling for distribution approximation is the evaluation of conditional posterior probability. Given T topics and V words, $P(z_i, x_i | w_i, \mathbf{z_{-i}}, \mathbf{x_{-i}}, \mathbf{w_{-i}}, \mathbf{a_d})$ is estimated by:

$$P(z_i = j, x_i = k | w_i = m, \mathbf{z_{-i}}, \mathbf{x_{-i}}, \mathbf{w_{-i}}, \mathbf{a_d}) \propto \tag{4.1}$$

$$P(w_i = m | x_i = k) P(x_i = k | z_i = j) \propto \tag{4.2}$$

$$\frac{C_{mj}^{WT}+\beta}{\sum_{m'} C_{m'j}^{WT}+V\beta}\frac{C_{kj}^{AT}+\alpha}{\sum_{j'} C_{kj'}^{AT}+T\alpha} \tag{4.3}$$

where $m' \neq m$ and $j' \neq j$, α and β are prior parameters for topic and word Dirichlets, C_{mj}^{WT} represents the number of times that word $w_i = m$ is assigned to topic $z_i = j$, C_{kj}^{AT} represents the number of times that author $x_i = k$ is assigned to topic j.

The transformation from Eq. 4.1 to Eq. 4.2 drops the variables, $\mathbf{z_{-i}}$, $\mathbf{x_{-i}}$, $\mathbf{w_{-i}}$, $\mathbf{a_d}$, because each instance of w_i is assumed independent of the other words in a document.

For any sample from this Markov chain, we can then estimate $P(w_i = m|z_i = r)$ and $P(z_i = r|x_i = q)$ from the topic-word distribution ϕ and author-topic distribution θ, respectively:

$$P(w_i = m|z_i = r) \propto \frac{C_{mr}^{WT}+\beta}{\sum_{m'} C_{m'r}^{WT}+V\beta} \tag{4.4}$$

$$P(z_i = r|x_i = q) \propto \frac{C_{rq}^{AT}+\alpha}{\sum_{r'} C_{r'q}^{AT}+T\alpha} \tag{4.5}$$

The iteration at time $\mathfrak{t}_0$ starts with random initialization of author-topic assignments C^{AT} and topic-word assignments C^{WT}, which at the end of the training, yields us the author-topic distributions $\theta^{\mathfrak{t}_0}$ and topic-word distributions $\phi^{\mathfrak{t}_0}$. Each subsequent iteration then utilizes the distributions obtained in the previous iterations to initialize the assignments for the current time segment. That is, initialization of the author-topic assignments for a time segment $\mathfrak{t}_k, k > 0$ becomes

$$C_{rq,\mathfrak{t}_k}^{AT} = \lambda\Re(C^{AT}) + (1-\lambda)\sum_{i=0}^{k-1}(\frac{1}{2})^{k-i}\theta_{rq}^{t_i} \tag{4.6}$$

and the initialization of the topic-word assignments becomes

$$C^{WT}_{mr,\mathsf{t}_k} = \lambda\mathfrak{R}(C^{WT}) + (1-\lambda)\sum_{i=0}^{k-1}(\frac{1}{2})^{k-i}\phi^{t_i}_{mr} \tag{4.7}$$

where $\mathfrak{R}(\cdot)$ adds random noise to the initialization by assigning topics to authors in Eq. 4.6 and words to topics in Eq. 4.7 independent of the prior knowledge obtained from $(\theta^0, \theta^1, \cdots, \theta^{k-1})$. The initialization places higher emphasis to recent distributions than earlier ones through a decay component. This enables S-ATM to integrate all prior knowledge the learner has gathered so far with varying levels of confidence based on the influence that they may have, based on the temporal distance of the distribution to the current time segment. Since we train the model on each time segment while propagating knowledge from previous segments, the distributions θ^{t_k} and ϕ^{t_k} only contain the topic-probabilities of authors and topic probabilities of words seen so far. Hence, at the start of the initialization of a new segment t_{k+1}, the model may find a new author a', or a new word w', in which case the distributions $\theta^{\mathsf{t}_i}_{a'm}$ and $\phi^{\mathsf{t}_i}_{mw'}$, $i = [0, \cdots, k]$ $m = [1, \cdots, T]$ will be zero denoting that we don't have prior knowledge for that observation.

The parameter λ determines the amount of prior knowledge that we want to propagate to subsequent segments. $\lambda = 1$ corresponds to treating each time segment with no prior knowledge and obtaining θ and ϕ distributions solely based on the documents that belong to that particular time segment, which won't be able to capture trends over time. The other extreme case of $\lambda = 0$ corresponds to full propagation of prior knowledge with no randomness involved, which may bias the learner towards already discovered topics and preventing the discovery of new topics. In our experimental settings, we do not estimate the hyperparameters α and β and empirically set fixed smoothing parameters at $50/T$ and 0.01, respectively, where T is the number of topics we we would like to discover.

4.3 Evaluation Metrics

Perplexity is a standard measure for estimating the generalization performance of a probabilistic model. Given a set of test words $(\mathbf{w}_d|\mathbf{a}_d)$,$d \in D^{\text{test}}$ where D^{test} is the set of test documents, perplexity is defined as the exponential of the negative normalized predictive likelihood under the probabilistic model of the collection, which is formulated as

$$\text{perplexity}(\mathbf{w}_d|\mathbf{a}_d) = \exp\left[-\frac{\log(\mathbf{w}_d|\mathbf{a}_d)}{N_d}\right] \tag{4.8}$$

The perplexity analysis on the CiteSeer dataset is performed on randomly sampled %10 of the documents in the entire collection. In addition to the perplexity analysis, we were also interested in the accuracy of the model on a dataset with ground truth labels of the documents. For the purpose of quantitative analysis in terms of prediction accuracy, we created a synthetic dataset that simulates a set of documents sampled from a predefined set of topics and authors of those topics. The characteristics of the synthetic dataset and the experimental evaluation of S-ATM is given in the next section.

4.4 Experiments

4.4.1 Synthetic Dataset

We generated a synthetic dataset to assess the author-topic model's ability to capture the topics of authors and the accuracy of representing the topic shifts of

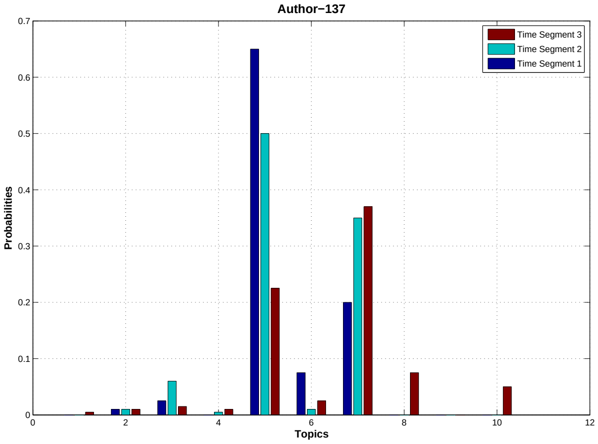


Figure 4.3. Topic distributions of a sample author from the synthetic dataset. The distributions are over three time segments, where the author's main topic changes from Topic 5 to Topic 7 and time segment 2 is the transition stage in the community.

authors.

4.4.1.1 Data Generation

The synthetic dataset consists of five communities with each having a unique distribution over ten topics. In the collection, each topic is modeled as a distribution over 200 words. For each author, we randomly sampled a community and the author generates words that follows the topic distribution of that community. In order to model topic dynamics, each community is modeled over three consecutive time segments, where the community's topic distributions vary over those segments. At each time segment, each author generates words that follow the topic weights of his/her community for that particular time segment. The dataset consists of 1000 authors that we have observations over those three time segments. The topic distribution of a sample author over three time segments is given in Figure 4.3.

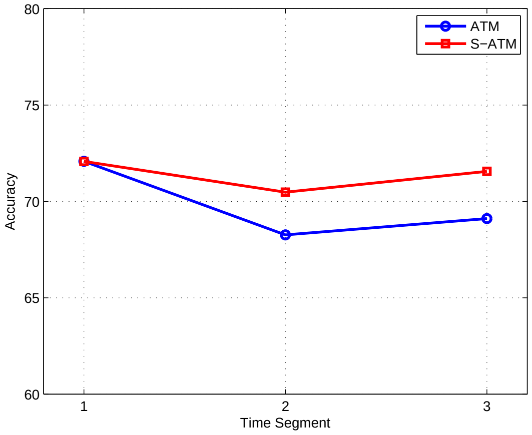


Figure 4.4. Accuracy comparison of ATM and S-ATM on the synthetic dataset for three time segments.

4.4.1.2 Results on the synthetic dataset

We ran the author-topic model on each time segment to achieve a baseline comparison for S-ATM. ATM randomly initializes the author-topic and topic-word distributions at the beginning of each time segment where the S-ATM partially integrates the prior knowledge, softened through random noise. In order to reduce the effect of random start at the first time segment, the results of both algorithms are averaged over 100 runs with 1000 iterations of Gibbs Sampling and with $\lambda = 0.5$. We compare precision-at-1 accuracies to assess whether the authors have been correctly classified to their respective topics. The comparative results for both algorithms are given in Figure 4.4. S-ATM is theoretically the same as ATM for the first time segment since there is no prior knowledge that it can utilize. Hence, both algorithms are initialized randomly and yield similar results. In time segments 2 and 3 where the communities transition from one topic to another, ATM tends to lose its accuracy as it starts to predict incorrect topics for authors, since they do not have distinctly pronounced topic memberships as time segment

1. S-ATM, on the other hand, utilizes the knowledge about the past memberships of authors to communities and hence, classifies authors to their respective communities with higher accuracy.

4.4.2 Experiments on the CiteSeer Collection

We conducted experiments on the scientific papers obtained from the CiteSeer repository. In addition to the textual content of the papers, we utilized many additional sources of information to improve the accuracy of the topic discovery process. Namely, in addition to the abstracts of the papers, we used the query logs from CiteSeer, user tags from CiteULike and term boosting from the citation graph of the documents in the collection. The following subsections describe the characteristics of the dataset that we used in our experiments.

4.4.2.1 Data Preparation

Our collection of CiteSeer documents consists of a set of scientific articles published over 15 years, between 1990 and 2004. In total, there are 41,540 documents published by 35,314 authors. We used the title, abstract and keywords fields from the documents and preprocessed the text by removing all punctuation and stop words, yielding a vocabulary size of 25,101 distinct words. The yearly breakdown of number of documents, authors and distinct words given in Table 4.1.

In the collection, it is possible that different authors may have common names, or an author may be represented by variations of the same name (such as D. Johnson or David Johnson or David B. Johnson), a problem known as ”name disambiguation”. We cross-referenced the documents in our collection with the metadata obtained from the ACM repository through exact paper title match of

Year	# Docs	# Authors	# Words
1990	782	1231	7131
1991	718	1337	6716
1992	1143	1968	8716
1993	1228	2134	9335
1994	1918	3178	11298
1995	2191	3837	12100
1996	2482	4304	13117
1997	3010	5267	14364
1998	3099	5459	14714
1999	3490	6228	15226
2000	3660	6854	15578
2001	3999	7479	15581
2002	5037	9056	16273
2003	4989	9205	15699
2004	3794	7708	14139

Table 4.1. Dataset characteristics with the number of documents and distinct number of authors and words for each year.

the documents in both collections. The ACM metadata contains a unique identifier for each author. We used the ACM author identifiers for the papers and obtained the publication year of papers from the ACM metadata.

4.4.2.2 Augmenting the Text Corpus

Once a scientific paper is published, it becomes an immutable document in the sense that the content of the document does not change over time. However, the environments that the documents reside in, such as search engines and digital libraries, continue to gather additional semantic information for the document in terms of *queries* and *tags*. As can be seen in Table 4.2 for sample articles, queries and tags can be effectively used for topic discovery since the keywords often times are concise descriptions of the paper and highlight the key terms in the document.

The generative process of the author topic model in Figure 4.2(b) can also be viewed from the search engine users' (or researchers') point of view. A researcher r has a distribution θ over scientific topics. When formulating a query, to generate each term, the researcher samples a topic z from θ_r and each term is generated from z's distribution over words. In this process, the set of topics in θ and the query terms are assumed to be same as the corresponding author topic model. The words in query terms need to be in the text corpus of the documents in order to retrieve a set of documents that result in a user click, and the user's topics should be similar to the topics in the collection so that the user interacts with the search system. Further, the queries concisely define the topics of the documents. An investigation of the CiteULike tags for Blei's paper on Latent Dirichlet Allocation [63] has most popular tags of bayesian, clustering, dirichlet, machine learning, and topic detection where similar queries have been submitted to CiteSeer. Therefore, due to the similar generative processes of documents and queries (and tags as well), we integrate the queries and tags with document contents that leads to a boosting effect of topical terms.

4.4.2.3 User Queries

Search engines heavily utilize query-click pairs of users logs for improving ranking of search results[76, 77] and recommending/rewriting queries [78, 79]. The action of a user click - and staying at the target page longer than a predefined timespan- as a result of a query is a good indicator that the terms in the query are descriptive of the topic(s) of the target page. Many additional signals, such as user's query history, session length, number of result clicks and their positions can offer valuable insight into the context of query-click pair observations, but many data analysis tasks can be effectively performed by solely investigating collections of queries. For instance, we have developed [80] a probabilistic hierarchical model

Table 4.2. Sample CiteSeer queries and CiteULike tags of five papers in the dataset for our experiments.

Paper Title	**CiteSeer Queries**	**CiteULike Tags**
Partitioning-Based Clustering for Web Document Categorization	[*document categorization*] [*soft clustering of web pages*]	[*clustering*] [*partitioning*] [*web*]
A Call-By-Need Lambda Calculus	[*online lambda reduction*] [*lambda syntax*]	[*algorithm*] [*lisp*] [*lambda calculus*]
Item-based Collaborative Filtering Recommendation Algorithms	[*item recommendation*] [*recommender algorithms*]	[*collaborative filtering*] [*recommender systems*] [*social networks*]
Using Web Structure for Classifying and Describing Web Pages	[*view web structure*] [*classifying web pages*]	[*automatic classification*] [*information organization*]
A Min-max Cut Algorithm for Graph Partitioning and Data Clustering	[*min max clustering*] [*data clustering bisection*]	[*clustering*] [*spectral*] [*mincut*]

that, for a given query, predicts the search result the user is most likely to click on through query segmentation. The two extreme cases of the segmentation for model learning are 1) Segment the queries in the logs to the word level, treat each word as an independent observation, learn a Bayesian model on the words, and 2) No segmentation - Each query in the logs is treated as a single unit and learn a Bayesian model based on query-click observations. In the former case, when a new query is submitted, the likelihood of a document click is estimated based on the words the query contains, whereas in the latter case, the model tries to predict a document-click action based on its previous observation on the same exact query. The word level segmentation suffers from *prediction accuracy*, since isolation of each individual term loses the meaning inherent in the query and decreases the accuracy of the model performance. No segmentation, on the other hand, suffers from *predictability*, meaning that due to the low probability of observing the same exact queries in the logs, the model can not predict clicks for a significant number of queries. Our model compromises between predictability and prediction accuracy by building a conditional probability hierarchy that starts with individual words in the queries and hierarchically combines conditional probabilities of words to estimate the meaningful and most likely phrases. Our findings from the experimental evaluation of the conditional probability hierarchy model not only show that we can outperform both extreme cases, but the strong tie between documents and queries potentially can assist topic discovery as well.

From the access logs of CiteSeer from January to March 2007, we selected the user queries to the documents in our collection and kept the queries with less than 50 characters. Our observations from the analysis of logs indicate that longer queries are navigational in nature and tend to look for exact title match of the papers. The shorter queries, on the other hand, are informational queries with only a couple of concise key terms that can highlight the topical nature of the clicked documents. After the preprocessing of the user logs, we identified 246,902 queries for the documents in our sample collection.

4.4.2.4 User Tags

A *Tag* can be defined as a simple form of metadata consisting of one or more freely chosen keywords that is assigned to digital objects by web users. As a matter of fact, the concept of tagging is not new and has been used for many years and the HTML 2.0 specification supports META keywords for the content authors to provide descriptive terms for web pages. The concept of *sharing* tags, however, has stimulated interest in collaborative tagging systems.

Until recently, queries were the only way of *implicit* description of document content by users of search engines and digital libraries. The popularity of tagging systems, which has also coined the term *Folksonomies* (short for Folk Taxonomies), provide an *explicit* medium to specify labels for web objects by users. Based on the observation that both queries and tags serve inherently similar purpose of locating web objects, we don't make a distinction between queries with document clicks and tags.

The CiteULike tag dataset contains 6527 unique tags for all of the documents in CiteSeer's repository. In the sample dataset for our experiments, we identified 2919 tags with 1012 unique words which we integrated into the corpus.

4.4.2.5 Boosting the topical terms using citation relationships

Citations in research publications represent an important knowledge source regarding the context of scientific work. They have been used to facilitate information search and retrieval in scientific digital libraries and they have been shown to be valuable for tasks such as ranking search results, identification of related research document, trend analysis, topic analysis and social network analysis. The presence of a citation between two papers indicates a level of topical similarity between those papers, which can influence the topic discovery process. We adopt the citation-based topical term identification from Chapter 2 and rank the most important topic-bearing terms based on the existence and absence of citations between papers using Expected Entropy Loss measure [46].

Table 4.3 shows the top ten words ranked by the expected entropy loss measure based on the citation graph. It is evident that these terms are highly valuable for topic discovery and placing higher emphasis on the words with high expected entropy loss scores will yield in the reduction of the noise caused by common words with less topical quality. From the ranked list of terms, we selected the top %10 of

Rank	Term
1	queries
2	authentication
3	programming
4	database
5	classifier
6	disk
7	image
8	matrix
9	robot
10	tree

Table 4.3. Top ten topical words identified from the citation relationships, ranked by expected entropy loss.

the vocabulary as the cut-off for the boost effect. Overall, we assigned twice the weight to queries, tags and top %10 topical words than words in the rest of the document contents.

4.4.3 Experimental Results for CiteSeer

The application of S-ATM to CiteSeer dataset provides insight into the distinct topics in the collection, most influential authors for the topics and the popularity trends of the topics over time. The collection contains papers from CiteSeer repository published between 1990 and 2004 in ACM conferences. Our quantitative analysis on the CiteSeer dataset yields %8.2 decrease in the perplexity of S-ATM (with perplexity 11,876) compared to ATM (with perplexity 12,937). Table 4.4 shows examples of 4 topics that are learned by S-ATM for the CiteSeer dataset. The topics are extracted from a single sample at the 1000th iteration of the Gibbs sampler with a model distribution propagation parameter $\lambda = 0.5$. For each topic, we provide the top 5 topical words most likely to be generated conditioned on the topic, and the top 5 most likely authors to have generated a word conditioned on the topic, for three distinct time segments. Clearly, the evolution of the topics in the fields of memory systems, network architectures, information retrieval and image processing is captured by S-ATM where the top authors who tend to produce the most words for topics also change over time.

We show the popularity trends of sample topics discovered by S-ATM in Figure 4.5. The popularity of topics are calculated by the fraction of words assigned to each topic for a year for all topics and for each year from 1990 to 2004. It can be seen that the popularity of machine learning has been steadily increasing over those years, due to widespread interest of its applications in many research areas. This can also be evidenced from the evolution of Topic 92 in Table 4.4. The supervised learning term *classifiers* emerges as one of the top words for the image processing topic. We observe the stabilization of the popularity of Digital Library and Processor Architectures topics. On the other hand, the topics Programming Languages and Operating Systems have been declining in popularity in our dataset, which also agrees with the analysis provided in [64], where our results show a more smooth decline for the popularity of these topics. One of the reasons might be attributed to the fact that the knowledge propagation in S-ATM causes the model to be less sensitive to the minor fluctuations in the topic popularities at each year and presents a smoothed topic trend analysis. Obviously it is possible to control the amount of smoothing that we desire using the λ parameter, which compromises between sensitivity and amount of prior knowledge that we utilize for learning learning the model parameters in each time segment.

Topic 8					
1990		**1994**		**2004**	
memory	.11255	memory	.11907	dynamic	.08096
random	.07198	dynamic	.07533	memory	.07993
disk	.06544	storage	.05643	access	.06774
access	.06369	access	.04582	random	.04634
consistency	.05017	shared	.04079	low	.03792
Author	**Prob.**	**Author**	**Prob.**	**Author**	**Prob.**
Patterson_D	.04036	Larus_J	.03709	Kandemir_M	.02275
Chen_P	.03814	Grunwald_D	.03683	Dubois_M	.01885
Soffa_M	.02478	Ball_T	.02689	Jouppi_N	.01817
Gibson_G	.02359	Davidson_J	.02453	Pande_S	.01705
Reed_D	.02155	Calder_B	.02323	Zhuang_X	.01341

Topic 23					
1990		**1994**		**2004**	
graph	.14944	graph	.16844	networks	.12200
process	.09876	routing	.08812	search	.08946
routing	.06919	process	.07256	graph	.08486
architecture	.06688	architecture	.06580	routing	.07542
computation	.04859	networks	.06108	process	.06778
Author	**Prob.**	**Author**	**Prob.**	**Author**	**Prob.**
Kaiser_G	.03190	Ranka_S	.07624	Wang_J	.04217
Perry_D	.02717	Mehtora_K	.06770	Sen_S	.04186
Gupta_R	.01883	Lilja_D	.05937	Morris_R	.02637
Gupta_A	.01675	Reif_J	.03605	Estrin_D	.01985
Rothberg_E	.01619	Blum_A	.03524	Liu_J	.01922

Topic 48					
1990		**1992**		**2004**	
databases	.25395	retrieval	.39651	mining	.42257
transactions	.15872	databases	.24174	users	.12466
dbms	.06802	users	.08319	retrieval	.06109
users	.04534	dbms	.03241	databases	.05730
heterogeneous	.03174	transactions	.03115	heterogeneous	.04007
Author	**Prob.**	**Author**	**Prob.**	**Author**	**Prob.**
Özsu_T	.09058	Fuhr_N	.09886	Sanderson_M	.06653
Chung_C	.04816	Croft_B	.06304	Younas_M	.05426
Perry_D	.04081	Li_J	.05036	Allan_J	.05129
Tsur_S	.03174	Delis_A	.04964	Pei_J	.02330
Zaniolo_C	.02914	Krovetz_R	.03916	Li_Q	.01765

Topic 92					
1990		**1994**		**2004**	
voronoi	.19580	segmentation	.14452	web	.39106
segmentation	.11918	diffuse	.08563	segmentation	.03875
texture	.11918	relaxation	.04478	regions	.02783
lighting	.05107	voronoi	.04170	classifiers	.02276
textures	.04256	interior	.03342	texture	.02137
Author	**Prob.**	**Author**	**Prob.**	**Author**	**Prob.**
Shields_M	.06129	Max_N	.03917	Antonacopoulos_A	.01654
Hanrahan_P	.01900	Nayar_S	.03006	Silva_A	.00865
Ware_C	.01702	Oren_M	.02931	Soatto_S	.00711
McKinley_P	.01373	Salesin_D	.01695	Ma_W	.00668
Liu_J	.01180	Jiang_X	.01332	Zaki_M	.00207

Table 4.4. Evolution of Four Sample Topics for S-ATM

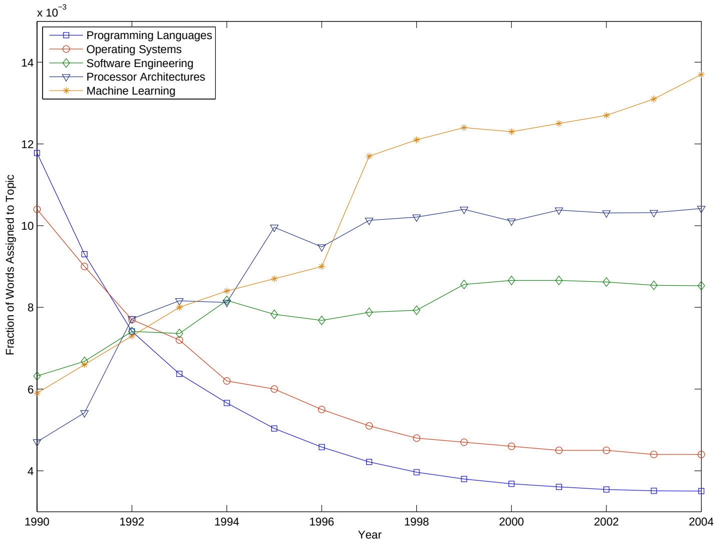


Figure 4.5. Topic trends for five research topics in Computer Science discovered in CiteSeer collection.

4.5 Concluding Remarks

Many real-world text collections exhibit temporal relationships where the temporal aspects of these collections present valuable insight into the topical structure of the collections. Temporal topic discovery requires an understanding of the characteristics of the data based on the temporal evolution of the topics in the collection. In this chapter, we present S-ATM, a generative model of documents that iteratively learns author-topic and topic-word distributions for scientific publications while integrating the temporal order of the documents into the generative process. The model parameters are estimated using Gibbs sampling and the distributions learned for each year are used as priors for the probabilities in the subsequent years. In addition to the textual content of the papers, we utilize user queries, tags and employ citation-based topic boosting to improve the topic discovery process. Our quantitative evaluation on a synthetic dataset as well as the application of S-ATM to a sample dataset from CiteSeer repository indicates that we can effectively discover scientific topics and most influential authors for the topics, as well as the evolution of topics over time.

Chapter 5

Conclusions and Future Work

The amount of academic documents on the World Wide Web has been steadily growing for many years. General purpose search engines treat those documents as any other flat file that is indexed based on the textual content and retrieved based on textual relevance to user queries. Search engines, therefore, under-utilize the information contained in academic documents since they ignore the structure of the documents and mix text segments that belong to different sections of the documents, such as title, authors, affiliation and citations. Digital libraries such as CiteSeer, on the other hand, utilize algorithms that parse the metadata and citations of academic documents and create a repository of academic documents that can be linked through many levels, including citations, authors, institutions, publication venues and publication years.

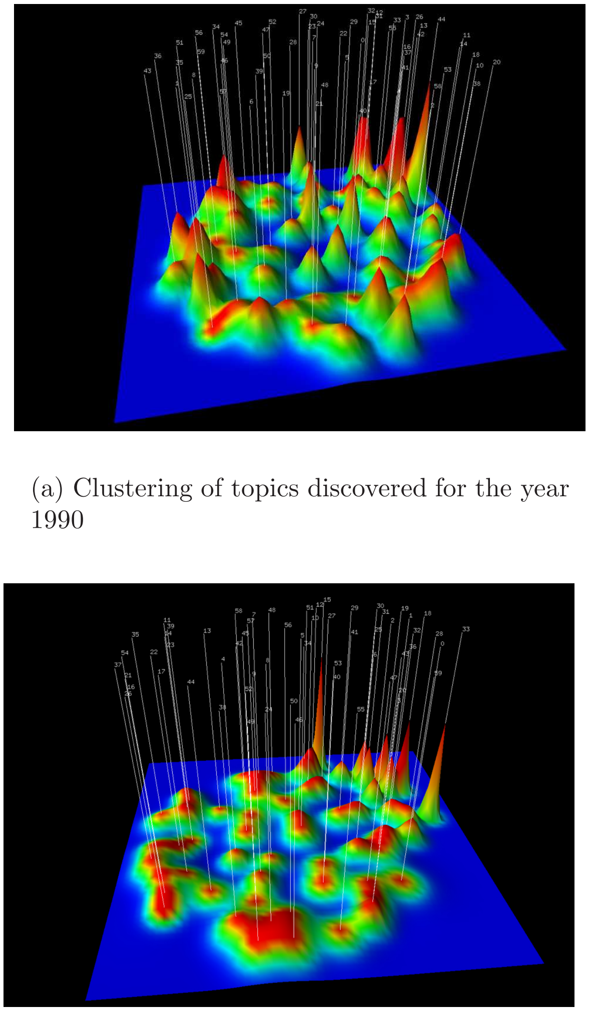
Mining scientific topics from academic papers is a crucial step for the identification of the content of documents. The discovered topics can be used to organize the collection and enable users to browse the collection based on the topics of the documents, recommend similar papers and improve the ranking of documents. Traditional *homogeneous* clustering algorithms try to group topically similar documents by relying only on textual features. This approach significantly under-utilizes the information that is extracted from the documents. This thesis proposes a topic identification framework for scientific literature that combines all the heterogeneous sources of information that are available in a scientific paper repository and search engine environment. The heterogeneous sources can be classified as *static* information (e.g. paper metadata and citations) from the papers and *dynamic* information (e.g. user queries and tags) from the search engine environment. Chapters 2 and 3 introduce novel clustering algorithms for clustering using the static heterogeneous information and the experimental evaluation of the algorithms validate our claim that the heterogeneous information sources can and should be utilized to achieve better clustering solutions. Chapter 4 describes a generative model that utilizes both static and dynamic information for topic discovery process.

In Chapter 2, we presented a methodology to utilize the citation graph to separate informative terms from uninformative ones. Traditional clustering algorithms treat textual content of documents uniform. That is, each word in a document is equally important from the perspective of the clustering algorithm. Ideally the similarity of documents needs to be assessed based on the amount of common ”topical terms” between papers. Finding the topical terms in document collections, on the other hand, requires domain knowledge and has to be updated constantly to account for emerging concepts. Many clustering solutions only filter out ”stop-words” - words that occur frequently in documents that are insignificant, such as function words ”the”, ”he” and ”if”. However, many common non-topical words (in academic papers), such as ”experiment” or ”result” will still be present in the corpus and adversely effect the clustering performance. This chapter presented an information theoretic approach that quantifies the topical weight of the words in the corpus based on the citation relationships between the papers. Since the presence of a citation between two academic papers is an indicator of relevance, the terms that are common to both citing and cited papers are likely to be on the same topic. However, it is likely that many non-topical words will appear in both documents and filtering out those words becomes a challenge. Our proposed solution both considers the presence and lack of citations. In addition to the ”actual” citation graph, we define a ”virtual” citation graph - virtual citations between papers based on the textual similarity. These two classes of citation graphs are then used to quantify the *topicality* of each word based on expected entropy loss. Terms with higher expected entropy loss scores have more topical characteristics, since they are more closely tied to the actual citation graph than the virtual citation graph. We then augment the TF-IDF based term weights with the expected entropy loss scores in order to break the uniformity of the words in the corpus. The similarity between two documents, therefore, becomes more dependent on the shared topical words than common ones. This non-uniform weighting of the words based on the topicality substantially improves the accuracy of the clustering algorithm.

Chapter 3 investigates the effect of authorship of documents for identification of the scientific topics of papers. Authors' research interests can be characterized by the topics that are addressed in their published articles. The task of topic analysis, without the authorship information, has a narrow focus when each document is treated as a separate observation. In reality, however, the authorship of scientific articles often times offers valuable insight into the topics of the articles. We propose a hybrid clustering algorithm, K-SVMeans, that merges the heterogeneous data sources of document text and authorship of papers to improve the clustering accuracy. The novelty of K-SVMeans can be attributed to the fact that it integrates the well known unsupervised clustering algorithm, K-Means with the highly popular supervised learning algorithm, namely support vector machines. A ping-pong style clustering is performed on the collection that simultaneously clusters the documents based on text of the documents and learns a classifier for the authors of each cluster, where each author is represented as a mixture of the documents that he/she has (co)authored. We use an efficient online SVM implementation that makes this approach practical and our algorithm eliminates the need for a separate manually labeled training set for the SVM, which is a requirement for batch SVM learners. In K-SVMeans, the class labels for authors are dynamically generated during the clustering process. At any stage in K-SVMeans clustering, each cluster contains a set of documents that are assigned to that cluster, and an associated SVM learner for that cluster. The authors of the documents in a cluster belong to the positive class, and the authors of the documents assigned to the rest of the clusters belong to the negative class. During the clustering process, a document is assigned to another cluster under the following conditions: 1) The document's textual similarity to the candidate cluster is higher than the original cluster and the authors of the documents are classified to the candidate cluster, or 2) The authors of the document are incorrectly classified to the candidate cluster but the textual similarity of the document to the candidate cluster is higher than a specified threshold. Once the cluster assignment of a document is updated, the authors of the document are inserted into the svm training set of the new cluster as positive observations, and the svm's of the remaining clusters as negative observations. We show on two separate datasets that the authorship of documents is indeed an important resource which can not be neglected for topic analysis of documents. A clustering algorithm, working at the document level, loses focus on the global view of the document collection, whereas the authorship of documents provides a wider perspective, achieving better clustering solutions.

Chapters 2 and 3 shed light into the richness of the content of scientific publications and how topic analysis based on textual content can be augmented by the additional dimensions of similarity between the documents. Chapter 4 describes a generative model of documents that discovers distinct topics in a scientific corpus by harnessing the static information of Chapters 2 and 3 with user queries and tags, which can be thought of as *dynamic* content generated by users. In addition to integrating the heterogeneous sources of information into the topic discovery process, we also integrate the temporal dimension of the repository. Both the LDA topic model and the Author-Topic Model(ATM) make the simplifying assumption of the exchangeability of the documents, meaning that the documents in the collection can be sampled interchangeably without considering the temporal order of the publications. The generative model we propose - Segmented Author-Topic Model (S-ATM)- on the other hand, learns probabilistic author-topic and topic-word distributions for each time segment defined in the generative process. The distributions learned for one time segment is then used as the prior knowledge for estimating the distributions in the subsequent time segments, with an exponential decay factor. Our experiments on a synthetic dataset shows that S-ATM outperforms ATM, and the experiments on the CiteSeer dataset shows the scientific topics in the repository and the evolution of the topics over time.

The future work involves determining the number of topics to be discovered in each time segment. Due to the dynamic nature of topics, it is likely that the number of topics in the collection varies over time, because of the advances in science and the increase in the number of documents added to the collection each year. Figure 5.1 depicts a set of the topics discovered for the CiteSeer dataset for our experiments in Chapter 4. Figure 5.1(a) is the distribution of a set of topics discovered for the year 1990, and Figure 5.1(b) represents the distributions of the same topics in the year 2000. We observe that some of the topics found for the papers published in 1990 tend to merge and collapse into a single topic in 2000. Forcing the number of topics to be discovered each year is not only unrealistic, but may be detrimental to the performance of the generative process as well.



(a) Clustering of topics discovered for the year 1990

(b) Clustering of topics discovered for the year 2000

Figure 5.1. Two Clusterings of the discovered topics

Perplexity is one of the most widely used measures for determining the optimum number of topics in generative models of documents. However, determining the number of topics by performing perplexity analysis each separate year would be computationally expensive and this approach would not guarantee the level of distinctness of the discovered topics. Some of the discovered topics may be very similar, where some topics may be too general that they may not correspond to any real-world topics. To address this problem, we might be interested in the post-processing of the results of the generative process that splits/merges the discovered topics in the model. The topics can be represented as a vector of authors where each author's weight is the probability of the topic for that author. We can then perform clustering on the matrix $M = [\mathbf{t_1 t_2} \cdots \mathbf{t_k}]$ where the columns are the topics and the rows are the authors. A second approach would be to investigate the top topics of each author and generate a graph that captures the co-occurences of the topics in authors' topic distributions. In the graph representation, the vertices represent the topics and the edges represent the association strengths of the topics, given as the co-occurence of the topics in the authors' topic probabilities. Defining $I_a(t_i, t_j)$ as the indicator function that takes the value 1 if author a's top topics contain t_1 and t_2 and the value 0 otherwise, the association strength of topics i and j would then be found as

$$e_{ij} = \frac{\sum_{\forall a \in \mathbf{A}} I_a(t_i, t_j)}{\sum_{\forall a \in \mathbf{A}} I_a(t_i) + \sum_{\forall a \in \mathbf{A}} I_a(t_j)} \tag{5.1}$$

This resulting graph can then be partitioned to identify similar topics in the set of topics discovered by the LDA. Whether we cluster the topic vectors of authors, or the similarity graph of topics, it is possible to estimate the optimal number of clusters using various cluster validity indices [81] for each year, yielding a more accurate estimate of the number of topics and a model that follows the real-world characteristics of the evolution of topics.

Bibliography

[1] Gantz, J. F., D. Reinsel, C. Chute, W. Schlichting, J. McArthur, S. Minton, I. Xheneti, A. Toncheva, and A. Manfrediz (2007) *The Expanding Digital Universe: A Forecast of Wordwide Information Growth through 2010*, EMC Corporation.

[2] Selberg, E. (1999) *Towards Comprehensive Web Search*, Ph.D. thesis, University of Washington.

[3] Lawrence, S. and C. Giles (1998) "Searching the world wide web," *Science*, **280**, pp. 98–100.
URL `http://citeseer.comp.nus.edu.sg/lawrence98searching.htm%l`

[4] Gulli, A. and A. Signorini (2005) "The indexable web is more than 11.5 billion pages," in *WWW '05: Special interest tracks and posters of the 14th international conference on World Wide Web*, ACM Press, New York, NY, USA, pp. 902–903.

[5] Zhuang, Z., R. Wagle, and C. L. Giles (2005) "What's there and what's not?: focused crawling for missing documents in digital libraries," in *JCDL*

'05: Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries, ACM Press, New York, NY, USA, pp. 301–310.

[6] Diligenti, M., F. Coetzee, S. Lawrence, C. L. Giles, and M. Gori (2000) "Focused Crawling Using Context Graphs," in *VLDB '00: Proceedings of the 26th International Conference on Very Large Data Bases*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 527–534.

[7] Forsyth, D. A., J. Malik, M. M. Fleck, H. Greenspan, T. K. Leung, S. Belongie, C. Carson, and C. Bregler (1996) "Finding Pictures of Objects in Large Collections of Images," in *ECCV '96: Proceedings of the International Workshop on Object Representation in Computer Vision II*, Springer-Verlag, London, UK, pp. 335–360.

[8] Han, H., C. L. Giles, E. Manavoglu, H. Zha, Z. Zhang, and E. A. Fox (2003) "Automatic document metadata extraction using support vector machines," in *JCDL '03: Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries*, IEEE Computer Society, Washington, DC, USA, pp. 37–48.

[9] Jaiswal, A. R., C. L. Giles, P. Mitra, and J. Z. Wang (2006) "An architecture for creating collaborative semantically capable scientific data sharing infrastructures," in *WIDM '06: Proceedings of the eighth ACM international workshop on Web information and data management*, ACM Press, New York, NY, USA, pp. 75–82.

[10] Bolelli, L., X. Lu, Y. Liu, A. Jaiswal, K. Bai, I. Councill, P. Mitra, J. Z. Wang, K. Mueller, J. Kubicki, B. Garrison, J. Bandstra, and C. L. Giles (2007) "ChemXSeer: A Chemistry Web Portal for Scientific

Literature and Datasets," in *OR2007: The Second International Conference on Open Repositories*.

[11] Lagoze, C., S. Payette, E. Shin, and C. Wilper (2005) "Fedora: An architecture for complex objects and their relationships," *International Journal on Digital Libraries*, **6**(2), pp. 124–138.

[12] Staples, T., R. Wayland, and S. Payette (2003) "The fedora project: An open source digital object repository system," *D-Lib Magazine*, **9**.

[13] "DSpace Digital Repository System," .
URL `http://www.dspace.org`

[14] Kaufman, L. and P. Rousseeuw (1990) *Finding Groups in Data: An Introduction to Cluster Analysis*, John Wiley and Sons, New York, NY.

[15] Ng, R. T. and J. Han (1994) "Efficient and Effective Clustering Methods for Spatial Data Mining," in *VLDB '94: Proceedings of the 20th International Conference on Very Large Data Bases*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 144–155.

[16] Zhang, T., R. Ramakrishnan, and M. Livny (1996) "BIRCH: an efficient data clustering method for very large databases," in *SIGMOD '96: Proceedings of the 1996 ACM SIGMOD international conference on Management of data*, ACM Press, New York, NY, USA, pp. 103–114.

[17] Guha, S., R. Rastogi, and K. Shim (1999) "ROCK: A Robust Clustering Algorithm for Categorical Attributes," in *ICDE '99: Proceedings of the 15th International Conference on Data Engineering*, IEEE Computer Society, Los Alamitos, CA, USA, pp. 512–521.

[18] ——— (1998) "CURE: an efficient clustering algorithm for large databases," in *SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, ACM Press, New York, NY, USA, pp. 73–84.

[19] Calado, P., M. Cristo, M. A. Goncalves, E. S. de Moura, B. Ribeiro-Neto, and N. Ziviani (2006) "Link-based similarity measures for the classification of Web documents," *Journal of the American Society for Information Science and Technology*, **57**(2), pp. 208–221.

[20] Calado, P., M. Cristo, E. Moura, N. Ziviani, B. Ribeiro-Neto, and M. A. Goncalves (2003) "Combining link-based and content-based methods for web document classification," in *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, ACM Press, New York, NY, USA, pp. 394–401.

[21] Couto, T., M. Cristo, M. A. Goncalves, P. Calado, N. Ziviani, E. Moura, and B. Ribeiro-Neto (2006) "A comparative study of citations and links in document classification," in *JCDL '06: Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, ACM Press, New York, NY, USA, pp. 75–84.

[22] Kessler, M. (1963) "Bibliographic Coupling between Scientific Papers," *American Documentation*, **14**, pp. 10–25.

[23] Small, H. (1973) "Co-citation in the Scientific Literature: A new measure of relationship between two documents," *Journal of the American Society for Information Science*, **24**(4), pp. 265–269.

[24] Giles, C. L., K. Bollacker, and S. Lawrence (1998) "CiteSeer: An Automatic Citation Indexing System," in *The 3rd ACM Conference on Digital Libraries*, pp. 89–98.

[25] Lawrence, S., C. L. Giles, and K. Bollacker (1999) "Digital Libraries and Autonomous Citation Indexing," *IEEE Computer*, **32**(6), pp. 67–71.

[26] Hayland, K. (2003) "Self-citation and self-reference: credibility and promotion in academic publication," *Journal of the Academic Society for Information Science*, **54**(3), pp. 251–259.

[27] Lawrence, S. (2001) "Online or invisible," *Nature*, **411**(6837), p. 521.

[28] Yitzhaki, M. (1998) "The language preference in sociology: measurements of 'language self-citation', 'relative own language preference indicator' and 'mutual use of languages'," *Scientometrics*, **41**, pp. 243–254.

[29] Pasula, H., B. Marthi, B. Milch, S. Russell, and I. Shpitser (2002) "Identity uncertainty and citation matching," in *Advances in Neural Information Processing*, pp. 1401–1408.

[30] Zeng, H.-J., Q.-C. He, Z. Chen, W.-Y. Ma, and J. Ma (2004) "Learning to cluster web search results," in *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM Press, New York, NY, USA, pp. 210–217.

[31] Chiu, T., D. Fang, J. Chen, Y. Wang, and C. Jeris (2001) "A robust and scalable clustering algorithm for mixed type attributes in large database environment," in *KDD '01*, pp. 263–268.

[32] Xu, W., X. Liu, and Y. Gong (2003) "Document clustering based on non-negative matrix factorization," in *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM Press, New York, NY, USA, pp. 267–273.

[33] Wang, Y. and M. Kitsuregawa (2001) "Use Link-Based Clustering to Improve Web Search Results," in *WISE '01: Proceedings of the Second International Conference on Web Information Systems Engineering (WISE'01) Volume 1*, IEEE Computer Society, Washington, DC, USA, p. 115.

[34] Hou, J. and Y. Zhang (2003) "Utilizing hyperlink transitivity to improve web page clustering," in *Proc. of 14th Australasian database conference on Database technologies*, pp. 49–57.

[35] Bolelli, L., S. Ertekin, and C. L. Giles (2006) "Clustering Scientific Literature Using Sparse Citation Graph Analysis," in *10th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 30–41.

[36] X. He, C. D., H. Zha and H. Simon (2002) "Web Document Clustering Using Hyperlink Structures," *Computational Statistics and Data Analysis*, **41**, pp. 19–45.

[37] Neville, J. M. and D. Jensen (2003) "Clustering relational data using attribute and link information," in *Text Mining and Link Analysis Workshop, 18th International Conference on Artificial Intelligence*.

[38] Kubica, J., A. Moore, J. Schneider, and Y. Yang (2002) "Stochastic link and group detection," in *Eighteenth national conference on Artificial intelligence*, American Association for Artificial Intelligence, Menlo Park, CA, USA, pp. 798–804.

[39] Brin, S. and L. Page (1998) "The anatomy of a large-scale hypertextual Web search engine," in *WWW7: Proceedings of the seventh international conference on World Wide Web 7*, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, pp. 107–117.

[40] Kleinberg, J. (1999) "Authoritative sources in a hyperlinked environment," *Journal of the ACM*, **46**(5), pp. 604–632.

[41] Getoor, L., N. Friedman, D. Koller, and B. Taskar (2002) "Learning Probabilistic Models of Link Structure," *Journal of Machine Learning Research*, pp. 679–708.

[42] Cohn, D. and T. Hofmann (2000) "The missing link - a probabilistic model of document content and hypertext connectivity," in *NIPS '00: Advances in Neural Information Processing Systems* (T. K. Leen, T. G. Dietterich, and V. Tresp, eds.), MIT Press, Cambridge, MA.

[43] Modha, D. S. and W. S. Spangler (2000) "Clustering hypertext with applications to web searching," in *HYPERTEXT '00: Proceedings of the eleventh ACM on Hypertext and hypermedia*, ACM Press, New York, NY, USA, pp. 143–152.

[44] Getoor, L., N. Friedman, D. Koller, and B. Taskar (2001) "Learning probabilistic models of relational data," in *ICML'02: Proceedings of the Nineteenth International Conference on Machine Learning*, pp. 170–177.

[45] Goldenberg, A., J. Kubica, P. Komarek, A. Moore, and J. Schneider (2003) "A Comparison of Statistical and Machine Learning Algorithms on the Task of Link Completion," in *KDD Workshop on Link Analysis for Detecting Complex Behavior*.

[46] Glover, E., G. Flake, S. Lawrence, W. P. Birmingham, A. Kruger, C. L. Giles, and D. Pennock (2001) "Improving Category Specific Web Search by Learning Query Modifications," in *Symposium on Applications and the Internet, SAINT*, IEEE Computer Society, Los Alamitos, CA, San Diego, CA, pp. 23–31.

[47] Karypis, G. (2002), "CLUTO," .
URL http://glaros.dtc.umn.edu/gkhome/views/cluto/

[48] Gao, B., T.-Y. Liu, T. Qin, X. Zheng, Q.-S. Cheng, and W.-Y. Ma (2005) "Web image clustering by consistent utilization of visual features and surrounding texts," in *MULTIMEDIA '05: Proceedings of the 13th annual ACM international conference on Multimedia*, ACM Press, New York, NY, USA, pp. 112–121.

[49] Dhillon, I. S. (2001) "Co-clustering documents and words using bipartite spectral graph partitioning," in *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM Press, New York, NY, USA, pp. 269–274.

[50] Dhillon, I. S., S. Mallela, and D. S. Modha (2003) "Information-Theoretic Co-Clustering," in *Proceedings of The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD-2003)*, pp. 89–98.

[51] Zha, H., X. He, C. H. Q. Ding, M. Gu, and H. D. Simon (2001) "Bipartite Graph Partitioning and Data Clustering," in *Conference on Information and Knowledge Management*, pp. 25–32.

[52] Gao, B., T.-Y. Liu, X. Zheng, Q.-S. Cheng, and W.-Y. Ma (2005) "Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data co-clustering," in *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, ACM Press, New York, NY, USA, pp. 41–50.

[53] Bekkerman, R., R. El-Yaniv, and A. McCallum (2005) "Multi-way distributional clustering via pairwise interactions," in *ICML '05: Proceedings of the 22nd international conference on Machine learning*, ACM Press, New York, NY, USA, pp. 41–48.

[54] Cristianini, N. and J. Shawe-Taylor (2000) *An Introduction to Support Vector Machines*, Cambridge University Press.

[55] Vapnik, V. (1995) *The Nature of Statistical Learning Theory*, Springer-Verlag.

[56] Platt, J. C. (1999) *Fast training of support vector machines using sequential minimal optimization*, MIT Press, Cambridge, MA, USA.

[57] Bordes, A., S. Ertekin, J. Weston, and L. Bottou (2005) "Fast Kernel Classifiers with Online and Active Learning," *Journal of Machine Learning Research*, **6**, pp. 1579–1619.

[58] Mei, Q., C. Liu, H. Su, and C. Zhai (2006) "A probabilistic approach to spatiotemporal theme pattern mining on weblogs," in *WWW '06: Proceedings of the 15th international conference on World Wide Web*, ACM Press, New York, NY, USA, pp. 533–542.

[59] Brner, K., J. T. Maru, and R. L. Goldstone (2004) “The simultaneous evolution of author and paper networks.” *Proceedings of the National Academy of Sciences*, **101 Suppl 1**, pp. 5266–5273.
URL `http://dx.doi.org/10.1073/pnas.0307625100`

[60] Mei, Q. and C. Zhai (2005) “Discovering evolutionary theme patterns from text: an exploration of temporal text mining,” in *KDD ’05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, ACM Press, New York, NY, USA, pp. 198–207.

[61] Zhou, D., X. Ji, H. Zha, and C. L. Giles (2006) “Topic evolution and social interactions: how authors effect research,” in *CIKM ’06: Proceedings of the 15th ACM international conference on Information and knowledge management*, ACM Press, New York, NY, USA, pp. 248–257.

[62] Backstrom, L., D. Huttenlocher, J. Kleinberg, and X. Lan (2006) “Group formation in large social networks: membership, growth, and evolution,” in *KDD ’06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM Press, New York, NY, USA, pp. 44–54.

[63] Blei, D. M., A. Y. Ng, and M. I. Jordan (2003) “Latent dirichlet allocation,” *Journal of Machine Learning Research*, **3**, pp. 993–1022.

[64] Steyvers, M., P. Smyth, M. Rosen-Zvi, and T. Griffiths (2004) “Probabilistic author-topic models for information discovery,” in *KDD ’04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM Press, New York, NY, USA, pp. 306–315.

[65] Rosen-Zvi, M., T. Griffiths, M. Steyvers, and P. Smyth (2004) "The author-topic model for authors and documents," in *AUAI '04: Proceedings of the 20th conference on Uncertainty in artificial intelligence*, AUAI Press, Arlington, Virginia, United States, pp. 487–494.

[66] Bolelli, L., S. Ertekin, D. Zhou, and C. L. Giles (2007) "A clustering method for web data with multi-type interrelated components," in *WWW '07: Proceedings of the 16th international conference on World Wide Web*, ACM Press, New York, NY, USA, pp. 1121–1122.

[67] Hofmann, T. (1999) "Probabilistic Latent Semantic Analysis," in *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, Morgan Kaufmann, San Francisco, CA, pp. 289–296.

[68] Deerwester, S. C., S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman (1990) "Indexing by Latent Semantic Analysis," *Journal of the American Society of Information Science*, **41**(6), pp. 391–407.

[69] Buntine, W. L. (1994) "Operations for Learning with Graphical Models," *Journal of Artificial Intelligence Research*, **2**, pp. 159–225.

[70] Griffiths, T. L. and M. Steyvers (2004) "Finding scientific topics." *Proceedings of the National Academy of Sciences*, **101 Suppl 1**, pp. 5228–5235. URL http://dx.doi.org/10.1073/pnas.0307752101

[71] McCallum, A. (1999) "Multi-label text classification with a mixture model trained by EM," in *AAAI Workshop on Text Learning*.

[72] Erosheva, E., S. Fienberg, and J. Lafferty (2004) "Mixed-membership models of scientific publications." *Proceedings of the National Academy of*

Sciences, **101 Suppl 1**, pp. 5220–5227.
URL `http://dx.doi.org/10.1073/pnas.0307760101`

[73] Blei, D. M. and J. D. Lafferty (2006) "Dynamic topic models," in *ICML '06: Proceedings of the 23rd international conference on Machine learning*, ACM Press, New York, NY, USA, pp. 113–120.

[74] Wang, X. and A. McCallum (2006) "Topics over time: a non-Markov continuous-time model of topical trends," in *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM Press, New York, NY, USA, pp. 424–433.

[75] Robert, C. P. and G. Casella (2005) *Monte Carlo Statistical Methods (Springer Texts in Statistics)*, Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[76] Joachims, T. (2002) "Optimizing search engines using clickthrough data," in *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM Press, New York, NY, USA, pp. 133–142.

[77] Agichtein, E., E. Brill, S. Dumais, and R. Ragno (2006) "Learning user interaction models for predicting web search result preferences," in *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM Press, New York, NY, USA, pp. 3–10.

[78] Kraft, R. and J. Zien (2004) "Mining anchor text for query refinement," in *WWW '04: Proceedings of the 13th international conference on World Wide Web*, ACM Press, New York, NY, USA, pp. 666–674.

[79] Cui, H., J.-R. Wen, J.-Y. Nie, and W.-Y. Ma (2002) “Probabilistic query expansion using query logs,” in *WWW ’02: Proceedings of the 11th international conference on World Wide Web*, ACM Press, New York, NY, USA, pp. 325–332.

[80] Zhou, D., L. Bolelli, J. Li, C. L. Giles, and H. Zha (2007) “Learning User Clicks in Web Search.” in *International Joint Conference on Artificial Intelligence(IJCAI ’07)*, pp. 1162–1167.

[81] Günter, S. and H. Bunke (2003) “Validation indices for graph clustering,” *Pattern Recognition Letters*, **24**(8), pp. 1107–1113.

Vita

Levent Bolelli

Levent Bolelli received the B.Sc degree in Electrical and Electronics Engineering from Orta Doğu Teknik Üniversitesi, Ankara, Turkey. Upon graduation, he joined *Cyber*Soft as the group leader of Tax Offices Complete Automation Project (VEDOP), which is known to be the largest IT project in Turkey by that time. Following the completion of the project, he came to the U.S. to pursue a graduate degree in Computer Science. He got his M.Sc degree in Computer Science from Center for Advanced Computer Studies at UL Lafayette and enrolled in the Ph.D program in Computer Science and Engineering at the Pennsylvania State University in August 2002.

After he joined Penn State, he did some work on multimodal human computer interaction with applications to Geographic Information Science (GIS). He developed software architectures, namely Dave_G and GCCM, that enable geographically distributed users to collaborate through GIS using natural speech and gestures in front of large screen displays, using laptops and hand-held devices. In 2004, he served as a software consultant for Video Mining, Inc. (formerly known as Advanced Interfaces, Inc.) for the development of GeoMIP, a Multimodal Interface Platform for Geographical Information Systems.

His research interests include information retrieval, data mining and machine learning and their applications to search engines. He did an internship at Ask.com in 2005, focusing on information extraction from web sites. In 2006, he went to Google as an intern, and worked on projects related to personalized search. At Penn State, he worked on the design and development of CiteSeer^X and Chem^{X}Seer, two niche search engines and digital libraries in the fields of Computer Science and Chemistry, respectively. He had also taken on the responsibilities of development, maintenance and co-administration of CiteSeer and SmealSearch with his colleagues under the supervision of Prof. C. Lee Giles.

He will join Google, Inc. in 2007.