

DISCOVERING LATENT TOPICAL PHRASES IN DOCUMENT
COLLECTIONS AND NETWORKS WITH TEXT COMPONENTS:
LEVERAGING TEXT MINING AND INFORMATION NETWORK
ANALYSIS FOR HUMAN ORIENTED APPLICATIONS

BY

MARINA GRIGORYEVNA DANILEVSKY

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2014

Urbana, Illinois

Doctoral Committee:

Professor Jiawei Han, Chair, Director of Research
Professor ChengXiang Zhai
Assistant Professor Julia Hockenmaier
Doctor Eunye Koh, Adobe Labs

Abstract

One of the major challenges of mining topics from a large corpus is the quality of the constructed topics. While phrase-generating approaches generally produce high quality output, they do not scale very well with the size of the data. Thus, the state of the art solutions usually rely upon scalable unigram-generating methods, which do not produce high quality human-readable topics, or are forced to use external knowledge bases. Furthermore, while document collections naturally contain topics at different levels of granularity (general vs. specific), very few traditional methods focus on generating high quality hierarchical topic structures.

This dissertation presents a series of approaches that directly addresses these challenges of generating high quality phrase-based topics, both as a flat set and organized as a hierarchy, as well as some potential applications. First, we describe a framework that generates high-quality topics represented by integrated lists of mixed-length phrases. The key is adapting a phrase-centric view towards the construction and ranking of topical phrases. The approach is domain-independent, and requires neither expert supervision nor an external knowledge base. The framework is initially constructed to work on collections of short texts, such as titles of scientific documents. However, we then show how the framework can be easily and robustly extended to work on collections of longer texts, and demonstrate its applicability to human needs with a task-centric evaluation.

The dissertation then addresses the need to move beyond generating a flat set of topics, and present an approach to constructing hierarchical topics, which extends the phrase-centric approach to create high quality phrases at varying levels of granularity. Another application of this technique is then presented: the task of entity role discovery. By tying entities in a community to topical phrases, users are able to explicitly understand both how and why individual entities are ranked within a specific community. A final extension is then described, which is a combined approach for constructing the hierarchy, which uses entity link information to improve the hierarchy quality.

To Yosha, for everything

Acknowledgments

I am profoundly grateful to my advisor, Dr. Jiawei Han, for his wisdom and guidance that has shaped my graduate school experience. Every conversation with him is a guarantee of learning something new, and his sharp insight and practical approach to research continues to inspire me. He is a dedicated advisor, a brilliant manager of a large research group, and a tireless advocate for his students.

My husband Josh deserves more thanks than I could possibly give for his unwavering mental, emotional, and academic support. It is a known fact that I would not have made it through this program without him. I am incredibly lucky to have such an amazing partner, and I know we're both excited to find out what happens when neither of us is in grad school any longer. Josh, I love you very, very much.

Mom and Dad, thank you for always supporting my goals and my interests, and for teaching me that sanity checks are as necessary in life as they are in coding. The apple did not fall far from the tree and I am very proud to carry on a family tradition of working in the sciences.

Brent and Sue, I am so grateful to you for your encouragement and advice, it has been a great help from before I even started my Ph.D. and I'm sure will continue to be for many years to come.

I am thrilled to have been part of a strong and collaborative research group. The Data Mining group at the University of Illinois is distinguished by its members' constant readiness to support each other's research endeavors. The honor of the individual is indeed the honor of the group, and vice versa. Particular thanks are due to Chi Wang, without whose collaboration the material on which this thesis is based would never have been so well-developed. My thanks go out to Nihit Desai, Xiang Ren, Fangbo Tao, Jialu Liu, Xiao Yu, Ming Ji, Yizhou Sun, and many others.

I greatly appreciate the guidance and support of my Ph.D. committee, Dr. ChengXiang Zhai, Dr. Julia Hockenmaier, and Dr. Eunye Koh, and am grateful to them for their useful comments on my dissertation as well as the wealth of knowledge they have imparted to me during my Ph.D.

Thank you Donna and Karyl, for the emotional support that cannot be properly put into words, and that can only be known by those of us who are the same person. We all made it!

A great thanks is due to Dr. Daniel Garcia-Swartz, for being the first person to pull me into writing research papers and thus smooth my way to graduate school, and for always encouraging me to pursue my interests and strengths.

There will always be a special place in my heart for Paul Sally Jr. for his unwavering encouragement of ‘the girl’ in his high school AP Computer Science class. The girl is still doing computer science!

I am very grateful to the National Science Foundation for supporting me with a Graduate Research Fellowship grant (NSF DGE 07-15088) in the final 3 years of my doctoral program. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or other sponsors.

Table of Contents

Chapter 1	Introduction	1
Chapter 2	Literature Review	5
Chapter 3	Automatically Constructing Topical Keyphrases from Short Text	9
Chapter 4	A New Domain, A New Dataset, A New Evaluation	24
Chapter 5	Extending the Framework to Construct Hierarchical Topical Phrases	41
Chapter 6	Using Hierarchical Topical Phrases for Entity Community Discovery	57
Chapter 7	Beyond Text: Constructing Hierarchical Topical Phrases from Heterogeneous Networks	66
Chapter 8	Conclusions	85
Appendix A	Instructions for KERT User Study on DBLP Dataset	88
Appendix B	Instructions for KERT User Studies on Goodreads Dataset	90
Appendix C	Instructions for CATHY User Study on DBLP Dataset	92
References		94

Chapter 1

Introduction

Yes, But Whats It *Really* All About, Then, When You Get Right Down To It, I *Mean* Really!

(Didactylos the Ephebian, Small Gods)

Let thy speech be short, comprehending much in few words.

(Ecclesiasticus 32:8)

‘What is it about?’ is a familiar question to everyone. Whether you are an avid reader looking to discover a new book, or a researcher looking for terminology and prior work that is relevant to your research, the need to characterize a document, or a collection of documents, pervades many areas of life. The latter problem - the need to characterize a collection of documents - is most often addressed via querying or classification tasks. There has been much work devoted to finding the most similar document to a query phrase, or to classify a document as belonging to one or another label. Modeling the topics, or concepts, present in the document collection is often an integral part of these tasks. However, few researchers have traditionally been interested in constructing a high quality representation which would be easily interpretable by human users. This problem is precisely the focus of this dissertation.

Most current approaches to topic construction yield ranked lists of unigrams to represent topics. However, it has long been known that unigrams account for only a small fraction of human-assigned index terms [66]. Furthermore, a person who is unfamiliar with the topic may not be able to easily view unigrams, and automatically combine them into ‘true’ phrases. For example, a person completely unfamiliar with the topic of Machine Learning would not be able to know that the unigram list {‘support’, ‘vector’, ‘machine’} should actually be transformed into the phrase ‘support vector machine.’ Therefore, in order to construct high quality keyphrases for a given topic, it is important to provide n-gram keyphrases rather than unigram keywords.

Table 1.1 presents one example of how the topic of Machine Learning might be denoted, within the universe of Computer Science areas. The topic is represented by a list of high quality mixed-length phrases, of which the top ranked are shown. Unlike existing unigram-centric methods, a person is not required to combine highly ranked unigrams into phrases (e.g., if you are not familiar with Machine Learning, you may not be able to combine ‘support’, ‘vector’, and ‘machine’ into the name of a well-known classification technique.) The topic is also not limited to

Table 1.1: The topic of Machine Learning, automatically constructed from the titles of CS papers published in DBLP.

learning
support vector machines
reinforcement learning
feature selection
conditional random fields
classification
decision trees
:

Table 1.2: The genre of Philosophy, automatically constructed from the descriptions of books designated as belonging to the Philosophy genre on Goodreads.

twentieth century
ayn rand
human nature
philosophy
work
thought
theory
:

phrases of a given length (top unigrams, top bigrams, etc). This also mimics the natural behavior of a person asked to generate topical phrases, since the constraint of limiting such phrases to any particular length is unnatural, and should therefore not be imposed by a successful algorithm. A single phrase to serve as a label for the entire topic is often too strong of a requirement, just as an expert on a topic is likely to use several phrases to describe it. On the other hand, the general concepts of the topic should become clear after perusing just the few top ranked generated phrases, rather than needing to peruse a very long list.

Scientific literature is not the only area which benefits from high-quality topical phrase construction. The categorization of books (or other forms of entertainment, such as music) into genres is a complicated, subjective, and very necessary task. Therefore automatically constructing useful representations of concepts such as genres which human beings can then use in such tasks is a valuable contribution. Table 1.2 presents one example of how the Philosophy genre might be described, represented by the top few phrases generated automatically from the descriptions of books which are known to belong to this genre.

A different direction that is necessary to consider is that a flat representation of the topics (or concepts) in a dataset may not always be as useful as a hierarchical organization at different levels of granularity. Therefore it is necessary to be able to also construct high quality topical hierarchies from texts, where each topic is represented by a ranked list of topical phrases, such that a child topic is a subset of its parent topic. For example, the topic of query processing and optimization may be described by the phrases {'query processing', 'query optimization',...}, while its parent topic of general problems in databases may be described by {'query processing', 'database systems', 'concurrency

control', ... } A sample of such a topical hierarchy is shown in Figure 7.1b

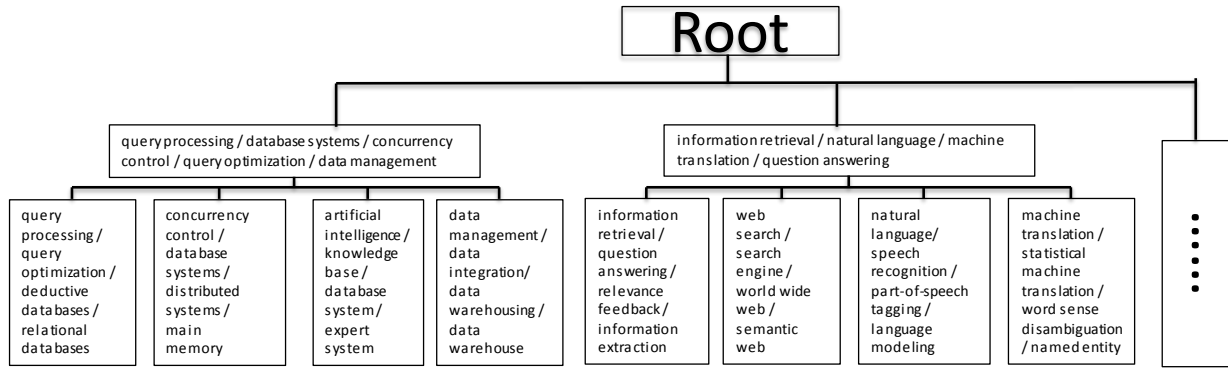


Figure 1.1: Topical hierarchy, constructed from the titles of computer science papers published in DBLP

In order to facilitate tasks such as efficient search, mining and summarization of heterogeneous networked data, it is very valuable to discover and organize the concepts presented in a dataset into a multi-typed topical hierarchy. Such a construction allows a user to perform more meaningful analysis of the terminology, people, places, and other network entities, which are organized into topics and subtopics at different levels of granularity. Therefore, we also consider how the constructed topical phrases may be used to discover the roles of entities which are connected to the discovered topics. We then delve further into network data and expand our framework to automatically construct multi-typed topical hierarchies, such as the one shown in Figure 7.1c

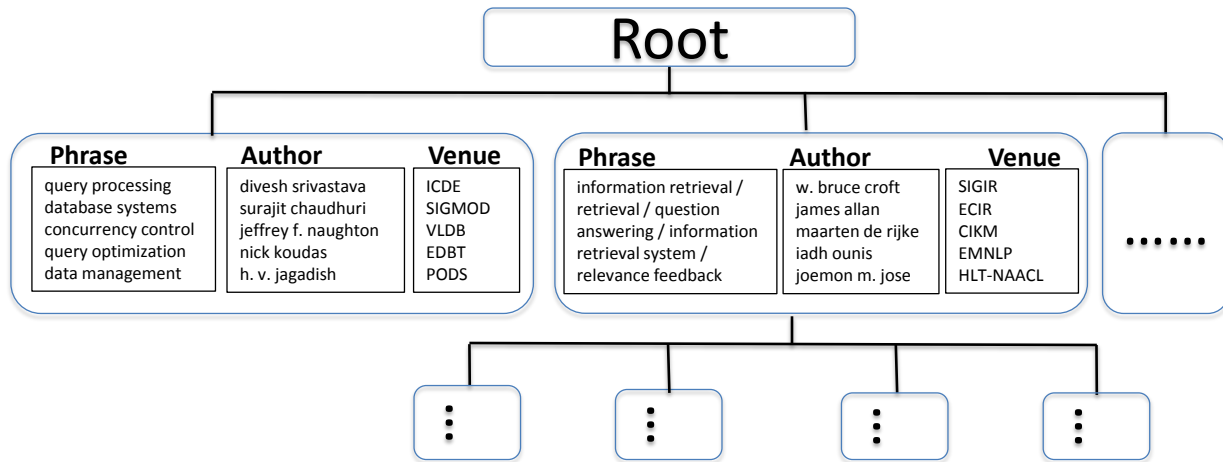


Figure 1.2: Topical hierarchy incorporating heterogeneous entities, constructed from the titles, authors, and publication venues of computer science papers published in DBLP

Throughout this dissertation, we seek to address several particular challenges associated with constructing topical phrases. We focus on characteristics that render the topical phrases to be high quality in the eyes of human judgement, rather than a particular metric (e.g., perplexity). We tackle the resulting non-trivial problem of evaluation through

multiple user studies, developing methods that both inquire about topical phrase quality directly, and assess the quality through task-based evaluations. We also keep a human-centric orientation as we explore how the topical phrase construction techniques presented in this dissertation can be used in various interesting applications.

We also consistently constrain the techniques developed in this dissertation to be mostly unsupervised, and requiring no external knowledge bases in addition to the document collections or networks with text components which are used as data sources. Furthermore, we limit our reliance on linguistic techniques such as parsing as much as possible. These limitations serve to develop methods which may be used with text from a wide variety of domains, or potentially even different languages, and requiring very little in user knowledge or guidance, thus resulting in robust algorithms which may be applied to various generalized tasks.

1.1 Structure of Dissertation

This dissertation presents a series of approaches and experiments addressing the challenges of generating high quality phrase-based topics. The rest of this document is structured as follows:

Chapter 2 presents a literature review of previous approaches to topic and phrase discovery.

Chapter 3 introduces KERT (Keyphrase Extraction and Ranking by Topic), a framework for topical keyphrase generation and ranking on collections of short texts. By altering the steps in the traditional methods of unsupervised keyphrase extraction, KERT is able to directly compare phrases of different lengths, resulting in a natural integrated ranking of mixed-length keyphrases. The effectiveness of KERT is demonstrated on two real world short document collections

Chapter 4 adapts KERT for collections of longer text, outside of the scientific domain. A new, task-centric evaluation of algorithm performance is also presented, effectively demonstrating the robustness and flexibility of the KERT framework.

Chapter 5 moves beyond a flat set of topics and introduces CATHY (Constructing a Topical HierarchY), a phrase-centric framework for topical hierarchy generation via recursive clustering and ranking.

Chapter 6 explores a further application of the CATHY framework to mine entity roles in topical communities, such as the role of an author in a research community, or the contribution of a user to a social network community organized around similar interests.

Chapter 7 unites the aims of chapters 5 and 6 by introducing CATHY HIN (Constructing a Topical HierarchY from a Heterogeneous Information Network), which is both able to construct a hierarchy and incorporate non-textual information during the construction.

Chapter 8 concludes and discusses future research directions.

Chapter 2

Literature Review

In this chapter, we will briefly review the existing literature on topics that are related to the work presented in this dissertation. Chapters 3 and 4 require an overview of topical keyphrase extraction and ranking, as well as a refresher on unigram-generating topic models. Chapters 5-7 are related to the area of topical hierarchy construction. Chapter 6 is interested in role discovery, and the identification of hierarchical topical communities. Finally, Chapter 7 works in the area of mining topics from heterogeneous information networks. The rest of this chapter presents related literature for each of these areas in turn.

2.1 Topical keyphrase extraction and ranking

Keyphrases have traditionally been defined as terms or phrases which summarize the topics in a document [66]. Keyphrase extraction is an important step in many tasks, such as document summarization, clustering, and categorization [44]. More recently, the definition has been expanded to include the notion of topical keyphrases - groups of keyphrases which summarize the topics in a given document, or document collection [42]. Most existing work on keyphrase extraction identifies keyphrases from either individual documents or a collection of long documents [65, 42]. However, recently there has been some interest in working with documents consisting of short text, such as a collection of tweets [72], in order to summarize the document collection.

The state-of-the-art approaches to unsupervised keyphrase extraction have generally been graph-based, unigram-centric ranking methods, which first extract unigrams from text, rank them, and finally combine them into keyphrases. We mainly review the related work in extracting topical keyphrases from document collections rather than keyphrase extraction from single documents. TextRank [48] constructs keyphrases from the top ranked unigrams in a document collection. Topical PageRank [42] splits the documents into topics and creates keyphrases from top ranked topical unigrams. Zhao et al [72] uses one example of short text - microblogging data from Twitter.

Some previous methods have used clustering techniques on word graphs for keyphrase extraction [41, 26], relying on external knowledge bases such as Wikipedia to calculate term importance and relatedness. Barker and Cornacchia [3] use natural language processing techniques to select noun phrases as keyphrases.

Tomokiyo and Hurst [65] take a language modeling approach, requiring a document collection with known topics as input and training a language model to define their ranking criteria. Keyphrases are traditionally extracted as ngrams using statistical modeling [68], or as noun phrases using natural language processing techniques [3].

2.2 Topic Modeling

Topic modeling techniques such as PLSA (probabilistic latent semantic analysis) [29] and LDA (latent Dirichlet allocation) [7] take documents as input, model them as mixtures of different topics, and discover word distributions for each topic.

LDA is perhaps the best known unigram-generating topic modeling technique. Each document in a collection is viewed as having been generated by a mixture of topics, where each word is seen to have been generated by some topic, with some probability. Figure 2.1 shows the plate notation for LDA.

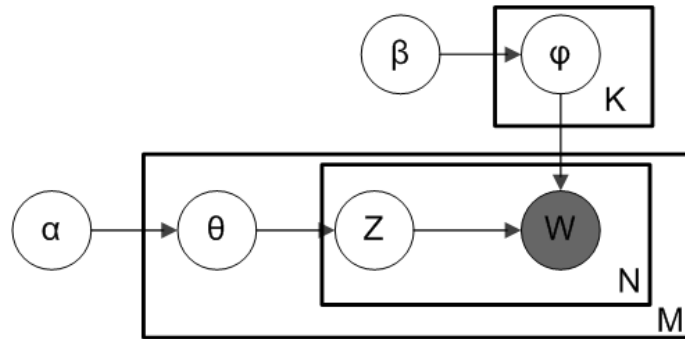


Figure 2.1: Graphical model of LDA.

Formally, let ϕ^t denote the word distribution for topic $t = 1, \dots, k$. Let θ^d be the topic distribution of document d . The generative process is as follows:

1. Draw $\phi^t \sim Dir(\beta)$, for $t = 0, \dots, k$.
2. For each document $d \in D$,
 - (a) draw $\theta^d \sim Dir(\alpha)$.
 - (b) for each word position i in d
 - i. draw topic $z_i \sim Multi(\theta^d)$
 - ii. draw $w_i \sim Multi(\phi^{z_i})$

The values α and β are Dirichlet prior parameters for θ and ϕ . The goal is to learn the set of topics, their associated word probabilities, the topic of each word, and the particular topic mixture of each document. This is done with some

kind of inference technique, such as Gibbs sampling[23]. Of particular interest to the work in this dissertation is that each word in every document can then be labeled with the topic that generated it.

There are numerous extensions to LDA. For example, correspondence LDA [5] is proposed to model annotated data, such as captioned images, having e.g., a generative model for image regions (Gaussian) and another generative model for the caption text (multinomial). To capture the syntax of text, LDA has also been extended to capture text syntax [25] by introducing a dependency between the topics that generate adjacent words. using a hidden Markov model. The author-topic model [61] extends LDA by including the authorship information. In general, the area of unigram-generating topic models is varied and well-studied.

2.3 Topical hierarchy construction

Topical hierarchies, concept hierarchies, ontologies, *etc.*, provide a hierarchical organization of data at different levels of granularity, and have many important applications, such as in web search and browsing tasks [22]. Although there has been a substantial amount of research on ontology learning from text, it remains a challenging problem (see [69] for a recent survey). The learning techniques can be broadly categorized as statistics-based or linguistic-based. Statistics-based methods are like those presented in [46], where LDA is extended to learn a topic hierarchy. Many studies are devoted to mining subsumption ('is-a') relationships [36], either by using lexico-syntactic patterns (e.g., 'x is a y') [60, 51] or statistics-based approaches [71, 18]. Chuang and Chien [12] and Liu *et al.* [40] generate taxonomies of given keyword phrases by supplementing hierarchical clustering techniques with knowledge bases and search engine results. However, unlike most of these approaches, all of the methods presented in this dissertation function without resorting to external knowledge resources such as WordNet or Wikipedia.

2.4 Role Discovery and Community Detection

Role analysis has its root in sociology. Sociologists use a notion of equivalence to assign nodes to different roles. For example, *automorphic equivalence* requires nodes in the same equivalence class (role) to have equivalent neighbors [20]. Network analysts use various notions of 'centrality' to find roles such as authorities and hubs [34]. Most works in computer science present a very similar flavor of role definition. As a result, the techniques of role discovery are all essentially either clustering [45, 30, 28], or ranking [31, 67, 9]. However, community knowledge has been shown to be useful in role analysis [59, 11], and Costa and Ortale [16] recently developed a Bayesian approach to jointly model the links generated by both communities and roles. They do not analyze the roles in communities defined by a textual hierarchy.

Link-based community detection has been studied extensively in the past decade. A multitude of methodologies

have been proposed for the community detection, or graph clustering problem in network analysis (see [57] and [21] for comprehensive surveys). For hierarchical community discovery, both deterministic [70] and probabilistic methods [14] have been proposed. Lancichinetti *et al.* [35] presents the first algorithm that finds both overlapping communities and the hierarchical structure. Agglomerative, or bottom-up approaches are the most common in these studies.

Some recent work shows that topic modeling can be used to augment community discovery [73, 43, 8, 74]. These approaches can find mixed membership for entities in various topical communities. However, they also do not study hierarchical communities. We perform top-down discovery of hierarchical topical communities, represented by phrases mined from the text component of a heterogeneous information network, and then infer communities of entities via their links to the text in the network.

2.4.1 Mining topics in heterogeneous networks

Basic topic modeling techniques such as PLSA [29] and LDA [7] described above take documents as input, and output word distributions for each topic. Recently, researchers have studied how to mine topics when documents have additional links to multiple typed entities [63, 62, 17, 10, 33, 64]. These approaches make use of multiple typed links in different ways. *iTopicModel* [62] and *TMBP-Regu* [17] use links to regularize the topic distribution so that linked documents or entities have similar topic distributions. Chen *et al.* [10] and Kim *et al.* [33] extend LDA to use entities as additional sources of topic choices for each document. Tang *et al.* [64] argue that this kind of extension has a problem of ‘competition for words’ among different sources when the text is sparse. They propose to aggregate documents linked to a common entity as a pseudo document, and regularize the topic distributions inferred from different aggregation views to reach a consensus.

Chapter 3

Automatically Constructing Topical Keyphrases from Short Text

3.1 Introduction

¹Keyphrases have traditionally been defined as terms or phrases which summarize the topics in a document [66]. Keyphrase extraction is an important step in many tasks, such as document summarization, clustering, and categorization [44]. More recently, the definition has been expanded to include the notion of topical keyphrases - groups of keyphrases which summarize the topics in a given document, or document collection [42].

Most current approaches to topic construction yield ranked lists of unigrams to represent topics. However, it has long been known that unigrams account for only a small fraction of human-assigned index terms [66]. Furthermore, a person who is unfamiliar with the topic may not be able to easily view unigrams, and automatically combine them into ‘true’ phrases. For example, a person completely unfamiliar with the topic of Machine Learning would not be able to know that the unigram list {‘support’, ‘vector’, ‘machine’} should actually be transformed into the phrase ‘support vector machine.’ Therefore, in order to construct high quality keyphrases for a given topic, it is important to provide n-gram keyphrases rather than unigram keywords.

On the other hand, it is inappropriate to discard all unigrams when approaching this task, or in fact to demonstrate a bias towards any particular phrase length. For instance, consider that the unigram ‘classification’ and the trigram ‘support vector machines’ are both high quality topical keyphrases for the machine learning topic in the domain of computer science. It is also not ideal to present separate ranked lists of topical phrases of each length, since when people are asked to characterize topics, they do not limit themselves to e.g. listing only bigrams, but rather provide a set of relevant phrases with no regard for phrase length. We should therefore also be able to perform integrated ranking of mixed-length phrases in a natural way.

In this chapter we present KERT (Keyphrase Extraction and Ranking by Topic), a framework for topical keyphrase generation and ranking. By altering the steps in the traditional methods of unsupervised keyphrase extraction, we are able to directly compare phrases of different lengths, resulting in a natural integrated ranking of mixed-length

¹This chapter contains materials from the following previously published work: Marina Danilevsky, Chi Wang, Nihit Desai, Xiang Ren, Jingyi Guo, and Jiawei Han. Automatic Construction and Ranking of Topical Keyphrases on Collections of Short Documents. Proc. of 2014 SIAM Int. Conf. on Data Mining (SDM’14), Philadelphia, PA, April 2014. 2014 Copyright SIAM. Reprinted with permission.

keyphrases.

We demonstrate the effectiveness of our approach on two real world short document collections. The first is the DBLP dataset, a collection of titles of recently published Computer Science papers in the areas related to Databases, Data Mining, Information Retrieval, Machine Learning, and Natural Language Processing. These titles come from DBLP,² a bibliography website for computer science publications. The second is the arXiv dataset, a sample of titles of physics papers published within the last decade. This collection of titles comes from arXiv,³ an online archive for electronic preprints of scientific papers.

3.2 Framework

A key aspect of our framework is that we do not follow the traditional unigram-centric approach of keyphrase extraction, where words are first extracted and ranked independently, and then combined to create phrases. Instead, we construct topical phrases immediately after clustering the unigrams. By shifting from a unigram-centric to a phrase-centric approach, we are able to extract topical keyphrases and implement a ranking function that can directly compare keyphrases of different lengths.

3.2.1 KERT Algorithm Overview

When humans are asked to generate phrases that could represent a topic, the aspect of phrase length is seldom considered, except as a reasonable maximum for the considered topic (e.g. phrases in computer science generally do not consist of more than four words). Therefore, we should generate topical keyphrases in such a way as to be able to directly compare keyphrases of mixed lengths during the ranking step. We refer to this characteristic as exhibiting the **comparability property**. For example, the keyphrases ‘classification,’ ‘decision trees,’ and ‘support vector machines’ should all be ranked highly in the integrated list of keyphrases for the Machine Learning topic, in spite of having different lengths.

Traditional probabilistic modeling approaches, such as language models or topic models do not have the comparability property. They can model the probability of seeing an n-gram given a topic, but the probabilities of n-grams with different lengths (unigrams, bigrams, etc) are not well comparable. These approaches simply find longer n-grams to have much smaller probability than shorter ones, because the probabilities of seeing every possible unigram sum up to 1, and so do the probabilities of seeing every possible bigram, trigram, etc. However, the total number of possible n-grams grows following a power law ($O(|V|^n)$), and ranking functions based on these traditional approaches invariably favor short n-grams. While previous work has used various heuristics to correct this bias during post-processing

²<http://www.dblp.org/>

³<http://arxiv.org>

steps, such as using a penalization term with respect to the phrase length [65, 72], our approach is cleaner and more principled.

The key to KERT exhibiting the comparability property is representing the random event $e_t(p)$ = ‘seeing a phrase p in a random document with topic t ’. With this definition, the events of seeing n -grams of various lengths in different documents are no longer mutually exclusive, and therefore the probabilities no longer need to sum up to 1. In order to do this, we must therefore first discover the phrase ‘ p ’ and then calculate $e_t(p)$. There are two ways to discover keyphrases: either extract them from the text (as sequences of words which actually occur in the text), or to automatically construct them [6], an approach which is regarded as both more intelligent and more difficult [8, 16]. We must therefore clearly define our process of keyphrase discovery.

Like [72], our framework focuses on constructing topics from a collection of short documents. There are many cases where the full text of a document collection is not available, or is too noisy, for the desired task of topic discovery from the collection. The documents may also be a mix of multiple topics, in spite of their short length. In this work we primarily evaluate our performance on two collections of scientific paper titles, which fit this criteria well. While our framework could technically be applied to a collection of noisy short documents such as tweets, it would require at least a reduction of noise in the documents as a pre-processing step in order to perform well, and this is out of the scope of this work

When working with a short document, extracting phrases directly from the text is quite limiting as this approach is too sensitive to the caprices of various writing styles. For instance, consider that two computer science papers titles, one containing mining frequent patterns and the other containing frequent pattern mining, are clearly discussing the same topic, and should be treated as such. A keyphrase may also be separated by other words: a document containing mining top- k frequent closed patterns also contains the topic of frequent pattern mining, in addition to incorporating the secondary topics of top- k frequent patterns, and closed patterns. Therefore, we define a phrase to be an order-free set of words appearing in the same document and must construct our phrases.⁴

Definition 1 A *phrase* p with length n is defined as an order-free set of n words appearing in the same document.

Definition 2 A *topical keyphrase* p with length n belonging to topic t is defined as an order-free set of n words appearing in the same document where each word in the set is assigned to topic t .

We are interested particularly in constructing *topical keyphrases*, which are phrases whose words all belong to the same topic. Therefore, before we are able to construct these topical keyphrases, we must first assign every word in every document to a topic. Algorithms that solve traditional topic modeling problems such as LDA [7] are particularly

⁴This is a simple way of saying that words may belong to the same phrase if they are collocated within the same window, and setting window size to be the entire document. This is a reasonable thing to do when working with short documents such as scientific paper titles. In the next chapter we explore how to adapt KERT to longer documents.

well suited to this task precisely because they provide these topic assignments, in addition to estimating the topic distributions. They are also widely used and well-understood, and can therefore be easily incorporated into our framework for the purpose of assigning words to topics.

Moving backwards through the above overview, our steps for topical keyphrase generation and ranking are therefore as follows:

Step 1. Given a collection of short texts, cluster words in the dataset into topics, using an appropriate unigram-generating topic model method.

Step 2. Construct candidate keyphrases for each topic according to the word topic assignments specified by the topic model output.

Step 3. Rank the keyphrases in each topic $t \in 1, \dots, k$ according to a well-defined ranking function.

In the following sections we detail the methodology for each of these steps.

3.2.2 Clustering Words using Topic Modeling

The first step of KERT is to cluster every unigram in every document to a topic. LDA [7] and its extensions have been shown to be effective for modeling textual documents. In this chapter we specifically work with one modification of the LDA [7] model, bLDA, which includes an additional background topic $z = 0$. This allows us to relax the assumption, previously introduced in Section 3.2.1 that every word is informative, and allow to relegate uninformative unigrams to a background topic, which effectively removes them from further consideration by KERT. At the same time, using the relatively simple model of bLDA rather than a more specialized model allows us to focus on the performance of the mining and ranking steps of KERT, which comprise our main contributions, rather than the unigram clustering step. However, our framework can easily incorporate the output of many topic models, such as PLSA [29], LDA [7], PY [56] and SAGE [19].

As an example, Table 3.1 shows the outputs of LDA and bLDA on the DBLP dataset, mentioned in Section 3.1. The topical quality of unigrams generated by LDA and bLDA appear comparable, though bLDA is better at filtering out common, background words which should not belong to any topic. In the rest of this chapter, we refer to bLDA as the topic model component of our framework.

bLDA models each document by a mixture of the foreground topics $z = 1, \dots, k$ and the background topic. For each word in a document, we decide if it belongs to the background topic or one of the foreground topics, and then choose the word from the appropriate distribution.

Formally, let ϕ^t denote the word distribution for topic $t = 0, \dots, k$. Let θ^d be the topic distribution of document d . Let λ denote a Bernoulli distribution that chooses between the background topic and foreground topics. The generative process is as follows:

Table 3.1: Outputs of LDA and bLDA on the DBLP dataset, T=5.

(a) **LDA**. Top unigrams for each topic

T1	T2	T3	T4	T5
data	web	data	learning	model
mining	information	query	knowledge	models
clustering	search	database	approach	text
algorithm	retrieval	systems	networks	language
efficient	system	databases	reasoning	semantic

(b) **bLDA**. Top unigrams for each topic

T1	T2	T3	T4	T5
data	information	system	learning	model
mining	web	database	classification	knowledge
efficient	search	systems	models	language
queries	retrieval	distributed	selection	reasoning
query	text	management	algorithm	logic

1. Draw $\phi^t \sim Dir(\beta)$, for $t = 0, \dots, k$.
2. For each document $d \in D$,
 - (a) draw $\theta^d \sim Dir(\alpha)$.
 - (b) for each word position i in d
 - i. draw $y_{d,i} \sim Bernoulli(\lambda)$
 - ii. if $y_{d,i} = 0$, draw $w_{d,i} \sim Multi(\phi^0)$, otherwise
 - A. draw topic $z \sim Multi(\theta^d)$
 - B. draw $w_{d,i} \sim Multi(\phi^z)$

where α and β are Dirichlet prior parameters for θ and ϕ .

We use a collapsed Gibbs sampling method for model inference. We iteratively sample the topic assignment $z_{d,i}$ for every word occurrence $w_{d,i}$ in each document d until convergence. In traditional topic modeling tasks, the sampled topic assignments are mainly used to estimate the topic distribution ϕ^z . In our case, we are more interested in the topic assignments for the words in each document, because these values are the foundation of our topical keyphrase generation step, as described in the next section. We use the *maximum a posteriori* (MAP) principle to label each word occurrence: $z_{d,i} = \arg \max_{z_{d,i}=0,\dots,k} P(z_{d,i}|W)$.

3.2.3 Candidate Keyphrase Generation

As described in Section 3.2.1, we define a topical keyphrase to be an order-free set of words appearing in the same document and belonging to the same topic. After the clustering step described in Section 3.2.2 is completed, each word $w_{d,i}$ in each document d has a topic label $z_{d,i} \in \{0, \dots, k\}$. If a set of words in a document d are assigned a common foreground topic $t > 0$, these words may comprise a topical phrase in t (e.g. {‘frequent’, ‘pattern’, ‘mining’} in the topic of ‘Data Mining’). If this set occurs in many documents with the topic label t , it is likely to be a good candidate keyphrase for that topic.

We use frequent pattern mining approaches to mine the candidate topical phrases. For each topic t , we construct a topic- t word set $p_d^t = \{w_{d,i} | z_{d,i} = t\}$ consisting of the words with the topic label t for each document d . We unite all the topic- t word sets into the topic- t set $D_t = \{d | p_d^t \neq \emptyset\}$. We may then mine frequent word sets from $p_{D_t}^t = \{p_d^t | d \in D_t\}$ using any efficient pattern mining algorithm, such as FP-growth [27]. We require a good topical keyphrase to have enough topical support $f_t(p) > \mu$ in order to filter out some coincidental co-occurrences (where $f_t(p)$ denotes the frequency of the word set p in topic t).

We thus define a *candidate topical keyphrase* for topic t to be a set of words $p = \{w_1 \dots w_n\}$ which are simultaneously labeled with topic t in at least μ documents, as discovered by the frequent pattern mining step. We then move on to ranking the candidate topical keyphrases within each topic.

3.2.4 Ranking Keyphrases for Topic Representation

Our goal is to construct a ranking function to evaluate the quality of topical keyphrases. Such a ranking function must successfully represent human intuition for judging what constitutes a high quality topical keyphrase. We now present the characteristics that such a ranking function should have, together with the criteria that represent these characteristics. As a running example, consider constructing and ranking keyphrases for topics in Computer Science.

Characteristic 1 *A representative keyphrase for a topic should cover many documents within that topic.*

Criterion 1 Coverage: Coverage, which may be referred to by other names such as frequency, or importance, is the most basic criteria, required by every ranking function that tackles this same problem. *Example: ‘information retrieval’ has better coverage than ‘cross-language information retrieval’ in the Information Retrieval topic.* A keyphrase that is not frequent in a topic should never be highly ranked as being representative of that topic, regardless of its length, or its value according to any other criteria. This further suggests that the combined ranking function should be constructed in such a way that a topical keyphrase with low coverage is guaranteed to have a lower rank.

Characteristic 2 *A keyphrase should rank high in a topic if it is representative of that one particular topic, and is not representative of other topics, thus helping to distinguish between topics.*

Criterion 2 Purity: A keyphrase is pure in a topic if it is only frequent in documents belonging to that topic and not frequent in documents within other topics. *Example: ‘query processing’ is more pure than ‘query’ in the Database topic.* Like Coverage, some version of this criterion is also present in most ranking functions, though it might be referred by other names, such as ‘informativeness’ [65] or ‘relevance’ [72].

Characteristic 3 *Since a keyphrase may consist of more than a single word, it must in some sense be a ‘real’ phrase that may be interpreted by a person, and not a random set of words.*

Criterion 3 Phraseness: Since KERT constructs phrases as word sets, this criterion evaluates the candidate topical keyphrases discovered in the previous step, somewhat following the intuition in [65]. A group of words should be combined together as a keyphrase if they co-occur significantly more often than the expected chance co-occurrence frequency, given that each term in the phrase occurs independently. *Example: ‘active learning’ is a better phrase than ‘learning classification’ in the Machine Learning topic, since the latter simply combines two unigrams which are frequent in the topic.* It is of course possible for both a unigram, and a word set containing that unigram to both be reasonable topical keyphrases, for example as with ‘learning’ and ‘reinforcement learning’ for the topic of Machine Learning.

Criterion 4 Completeness: A keyphrase is not complete if it is a subset of a longer keyphrase, in the sense that it rarely occurs in a document without the presence of the longer keyphrase. *Example: ‘support vector machines’ is a complete phrase, whereas ‘vector machines’ is not because ‘vector machines’ is almost always accompanied by ‘support’.*

The combination of Phraseness and Completeness take the place of other NLP criteria that have been used by approaches which perform phrase extraction rather than phrase construction, such as part-of-speech tagging (e.g. is the phrase a noun group? [41, 42]) or suffix sequences [54]. The Phraseness and Completeness ranking criteria together embody the characteristic of a phrase having ‘semantic coherence’: being ‘understandable’ to people [41] or having ‘term cohesion’ [13]. We also do not go the route of evaluating this characteristic by calculating semantic relatedness between words, which has been used in other approaches using knowledge bases such as Wikipedia [41]. This is because we wish to avoid relying on external information which may possibly be unavailable, or not relevant for a particular dataset, for instance if the dataset is domain-specific.

The step of generating candidate topical keyphrases allows us to represent the random event $e_t(p) =$ ‘seeing a phrase p in a random document with topic t .’ The KERT algorithm therefore exhibits the *comparability property* and is able to directly compare phrases of any length, according to the ranking criteria. Formally, the probability of $e_t(p)$ is defined to be $P(e_t(p)) = \frac{f_t(p)}{|D_t|}$. In the subsequent sections we define our measures representing the 4 criteria of coverage, purity, phraseness, and completeness using quantities related to this probability. Because we

work with relatively short documents, which we assume to be uniformly informative, we do not consider criteria based on document structure, or the location of the phrase in the document.

Coverage

A representative keyphrase for a topic should cover many documents within that topic. For example, ‘information retrieval’ has better coverage than ‘cross-language information retrieval’ in the topic of Information Retrieval. We directly quantify the coverage measure of a phrase as the probability of seeing a phrase in a random topic- t word set $p_d^t \in D_t$:

$$\pi_t^{cov}(p) = P(e_t(p)) = \frac{f_t(p)}{|D_t|} \quad (3.1)$$

Purity

A phrase is pure in topic t if it is only frequent in documents about topic t and not frequent in documents about other topics. For example, ‘query processing’ is a more pure keyphrase than ‘query’ in the Databases topic.

We measure the purity of a keyphrase by comparing the probability of seeing a phrase in the topic- t collection of word sets and the probability of seeing it in any other topic- t' collection ($t' = 0, 1, \dots, k, t' \neq t$). A reference collection $D_{t,t'} = D_t \cup D_{t'}$ is a mix of topic- t and topic- t' documents. If there exists a topic t' such that the probability of $e_{t,t'}(p)$ = ‘seeing a phrase p in a reference collection $D_{t,t'}$ ’ is similar, or even larger than the probability of seeing p in D_t , the phrase p indicates confusion about topic t and t' . The purity of a keyphrase compares the probability of seeing it in the topic- t collection and the maximal probability of seeing it in any reference collection:

$$\begin{aligned} \pi_t^{pur}(p) &= \log \frac{P(e_t(p))}{\max_{t'} P(e_{t,t'}(p))} \\ &= \log \frac{f_t(p)}{|D_t|} - \log \max_{t'} \frac{f_t(p) + f_{t'}(p)}{|D_{t,t'}|} \end{aligned} \quad (3.2)$$

Phraseness

A group of words should be grouped into a phrase if they co-occur significantly more frequently than the expected co-occurrence frequency given that each word in the phrase occurs independently. For example, while ‘active learning’ is a good keyphrase in the Machine Learning topics, ‘learning classification’ is not, since the latter two words co-occur only because both of them are popular in the topic.

We therefore compare the probability of seeing a phrase $p = \{w_1 \dots w_n\}$ and seeing the n words $w_1 \dots w_n$ independently in topic- t documents:

$$\begin{aligned}\pi_t^{phr}(p) &= \log \frac{P(e_t(p))}{\prod_{w \in p} P(e_t(w))} \\ &= \log \frac{f_t(p)}{|D_t|} - \sum_{w \in p} \log \frac{f_t(w)}{|D_t|}\end{aligned}\tag{3.3}$$

Completeness

A phrase p is not complete if a longer phrase p' that contains p usually co-occurs with p . For example, while ‘support vector machines’ is a complete phrase, ‘vector machines’ is not, as ‘support’ nearly always accompanies ‘vector machines.’

We thus measure the completeness of a phrase p by examining the conditional probability of observing p' given p in a topic- t document:

$$\begin{aligned}\pi_t^{com}(p) &= 1 - \max_{p' \supseteq p} P(e_t(p')|e_t(p)) \\ &= 1 - \max_w P(e_t(p \cup \{w\})|e_t(p)) \\ &= 1 - \frac{\max_w f_t(p \cup \{w\})}{f_t(p)}\end{aligned}\tag{3.4}$$

Keyphrase Topical Quality Ranking Function

We can combine these 4 measures into a comprehensive function representing the quality of a topical keyphrase by viewing them within an information theoretic framework. As described above, the coverage criterion is in some sense the most important, since a keyphrase with low coverage will be necessarily of low quality, regardless of its performance according to the other criteria. We can enforce this by representing the relationships between coverage, phraseness, and purity using two pointwise Kullback-Leibler(KL)-divergence metrics.

Pointwise KL-divergence is a distance measure between two probabilities that takes the absolute probability into consideration, and is more robust than pointwise mutual information when the relative difference between probabilities need to be supported by sufficiently high absolute probability.⁵ The product of coverage and purity, $\pi_t^{cov}(p)\pi_t^{pur}(p) = P(e_t(p)) \log \frac{P(e_t(p))}{P(e_{t,t^*}(p))}$ is equal to the pointwise KL-divergence between the probabilities of $e_t(p)$ and $e_{t,t^*}(p)$. Likewise, the product of coverage and phraseness, $\pi_t^{cov}(p)\pi_t^{phr}(p)$ is equivalent to the pointwise KL-divergence between the probability of $e_t(p)$ under different independence assumptions. We combine these two metrics with a weighted

⁵Note that Tomokiyo *et al.* [65] uses KL-divergence metrics to derive their ranking function as well, but they require an annotated foreground corpus and a background corpus as input. Furthermore, they consider language models only for consecutive ngrams and do not exhibit the comparability property. So their ranking function behaves quite differently from ours.

summation, and implement the completeness criterion as a filtering condition to remove incomplete phrases:

$$Quality_t(p) = \begin{cases} 0 & \pi_t^{com} \leq \gamma \\ \pi_t^{cov}[(1 - \omega)\pi_t^{pur} + \omega\pi_t^{phr}](p) & \text{o.w.} \end{cases} \quad (3.5)$$

Input: Document collection D , topic number K , parameters γ, ω, μ
Output: Topics, represented as ranked lists of mixed-length phrases

Using a unigram-generating topic model, label each word w_d in each document $d \in D$ with a topic label t , $t \in 1 \dots K$

```

foreach topic  $t$  do
  Construct topic- $t$  word sets  $p_d^t = \{w_{d,i} | z_{d,i} = t\}$  for each  $d$ 
  Unite all topic- $t$  word sets into topic- $t$  set  $D_t = \{d | p_d^t \neq \emptyset\}$ 
  foreach word set  $p_d^t | d \in D_t$  do
    if  $f_t(p) > \mu$  then
       $p$  is a candidate topical keyphrase for topic  $t$ 
    end
  end
end

foreach topic  $t$  do
  foreach keyphrase  $p$  do
    if  $\pi_t^{com} > \gamma$  then
       $Quality_t(p) = \pi_t^{cov}[(1 - \omega)\pi_t^{pur} + \omega\pi_t^{phr}](p)$ 
    else
       $Quality_t(p) = 0$ 
    end
  end
  Sort keyphrases  $p$  by  $Quality_t(p)$  in descending order
end

```

Algorithm 1: The KERT Algorithm

Here, $\gamma \in [0, 1]$ controls how aggressively we prune incomplete phrases. $\gamma = 0$ corresponds to ignoring the completeness criterion and retaining all *closed* phrases, where no supersets have the same topical support. As γ approaches 1, more phrases will be filtered and eventually only *maximal* phrases (no supersets are sufficiently frequent) will remain. The other three criteria rank keyphrases that pass the completeness filter.

Although $Quality_t(p)$ is a combination of two metrics, the coverage criterion is a factor in both. This reflects the fact that when $P(e_t(p))$ is small, phrase p has low support, and thus the estimates of purity and phraseness will be unreliable and their role should be limited. When $P(e_t(p))$ is large, phrase p has high support, and magnifies the cumulative effect (positive or negative) of the purity and phraseness criteria.

The relative weights of the purity and phraseness criteria are controlled by $\omega \in [0, 1]$. Both measures are log ratios on comparable scales, and can thus be balanced by a weighted summation. As ω increases, we expect more topic-independent, but common phrases to be ranked higher.

For each topic t , we construct a topic- t word set $p_d^t = \{w_{d,i} | z_{d,i} = t\}$ consisting of the words with the topic label t for each document d . We unite all the topic- t word sets into the topic- t set $D_t = \{d | p_d^t \neq \emptyset\}$. We may then mine frequent word sets from $p_{D_t}^t = \{p_d^t | d \in D_t\}$

Table 3.2: Top 10 ranked keyphrases in the Machine Learning topic by different methods.

Method	Top 10 Topical Keyphrases
kpRelInt*	learning / classification / selection / models / algorithm / feature / decision / bayesian / trees / problem
kpRel	learning / classification / learning classification / selection / selection learning / feature / decision / bayesian / feature learning / trees
KERT_{-cov}	effective / text / probabilistic / identification / mapping / task / planning / set / subspace / online
KERT_{-pur}	support vector machines / feature selection / reinforcement learning / conditional random fields / constraint satisfaction / decision trees / dimensionality reduction / constraint satisfaction problems / matrix factorization / hidden markov models
KERT_{-phr}	learning / classification / selection / feature / decision / bayesian / trees / problem / reinforcement learning / constraint
KERT_{-com}	learning / support vector machines / support vector / reinforcement learning / feature selection / conditional random fields / vector machines / classification / support machines / decision trees
KERT	learning / support vector machines / reinforcement learning / feature selection / conditional random fields / classification / decision trees / constraint satisfaction / dimensionality reduction / matrix factorization

3.3 Experiments

3.3.1 Judging Topical Keyphrase Quality

In our first set of experiments, we use the DBLP dataset - a collection of paper titles in Computer Science - to evaluate the ability of our method to construct topical keyphrases that appear to be high quality to human judges, via a user study. The titles were minimally pre-processed by removing all stopwords, resulting in 33,313 documents consisting of 18,598 unique words. We first describe the methods we used for comparison, and then present a sample of the keyphrases actually generated by these methods and encountered by participants in the user study. We then explain the details of our user study, and present quantitative results.

Ranking Methods for Comparison

We use bLDA, introduced in Section 3.2.2, to perform the word clustering step and create input for all the methods that we compare. We resort to a Newton-Raphson iteration method [50] to learn the hyperparameters, and empirically

set $\lambda = 0.1$, which yields coherent results for our dataset.⁶

To evaluate the performance of KERT, we implemented several variations of the function, as well as two baseline functions. The baselines come from Zhao et al [72], who focus on topical keyphrase extraction in microblogs, but claim that their method can be used for other text collections. We implement their two best performing methods: `kpRelInt*` and `kpRel`.⁷ We also construct variations of KERT where the keyphrase extraction steps are identical, but each of the four ranking criteria is ignored in turn. We refer to these versions as `KERT-cov`, `KERT-pur`, `KERT-phr`, and `KERT-com`.

These variations nicely represent the possible settings for the parameters γ and ω , which are described in Section 3.2.4. In KERT we set $\gamma = 0.5$ and $\omega = 0.5$. `KERT-com` sets $\gamma = 0$ to demonstrate what happens when we retain all *closed* phrases. As γ approaches 1, more phrases will be filtered but a very small number of *maximal* phrases (no supersets are frequent) will not be. `KERT-phr` sets $\omega = 0$ and `KERT-pur` sets $\omega = 1$, to demonstrate the effect of ignoring phraseness for the sake of maximizing purity, and ignoring purity to optimize for phraseness, respectively. For Step 2 of every KERT variation, we set $\mu = 5$.

Qualitative Results

Table 3.2 shows the top 10 ranked topical keyphrases generated by each method for the topic of Machine Learning. `kpRel` and `kpRelInt*` yield very similar results, both clearly favoring unigrams. However, `kpRel` also ranks several keyphrases highly which are not very meaningful, such as ‘learning classification’ and ‘selection learning.’ Removing coverage from our ranking function yields the worst results, confirming the intuition that a high quality keyphrase must at minimum have good coverage. Without purity, the function favors bigrams and trigrams that all seem to be meaningful, although several high quality unigrams such as ‘learning’ and ‘classification’ no longer appear. Removing phraseness, in contrast, yields meaningful unigrams but very few bigrams, and looks quite similar to the `kpRelInt*` baseline. Finally, without completeness, phrases such as ‘support vector’ and ‘vector machines’ are improperly highly ranked, as both are sub-phrases of the high quality trigram ‘support vector machines.’

User Study and Quantitative Results

To quantitatively measure keyphrase quality, we invited people to judge the generated topical keyphrases generated by the different methods. Since the DBLP dataset generates topics in Computer Science, we recruited 10 Computer Science graduate students - who could thus be considered to be very knowledgeable judges in this domain - for a user study.

⁶The learned $\alpha = 1.0$ is smaller than the typical setting due to the nature of our short text, and $\beta = 0.07$ is larger because in our dataset, different topics often share the same words.

⁷Their main ranking function `kpRelInt` considers the heuristics of phrase interestingness and relevance. As their interestingness measure is represented by re-Tweets, a concept that is not appropriate to our dataset, we reimplement the interestingness measure to be the relative frequency of the phrase in the dataset, and we therefore refer to our reimplementation as `kpRelInt*`. `kpRel` considers only the relevance heuristic.

Table 3.3: nKQM@K (methods ordered by performance)

Method	nKQM@5	nKQM@10	nKQM@20
KERT_{-cov}	0.2605	0.2701	0.2448
kpRelInt*	0.3521	0.3730	0.3288
KERT_{-phr}	0.3632	0.3616	0.3278
kpRel	0.3892	0.4030	0.3767
KERT_{-com}	0.5124	0.4932	0.4338
KERT	0.5198	0.4962	0.4393
KERT_{-pur}	0.5832	0.5642	0.5144

We generated five topics from the DBLP dataset. As can be seen from the bLDA output in Section 3.2.2, Topic 5 is very mixed and difficult to interpret as representing just one research area. We discarded it, leaving four topics which are clearly interpretable as Machine Learning, Databases, Data Mining, and Information Retrieval. For each of the four topics, we retrieved the top 20 ranked keyphrases by each method. These keyphrases were gathered together per topic and presented in random order. Users were asked to evaluate the quality of each keyphrase on a 5 point Likert scale. The complete set of instructions for the user study can be found in Appendix C.

To measure the performance of each method given the user study results, we adapted the **nKQM@K measure** (normalized keyphrase quality measure for top-K phrases) from [72], which is itself a version of the *nDCG* metric from information retrieval [32]. We define nKQM@K for a method M using the top-K generated keyphrases:

$$nKQM@K = \frac{1}{T} \sum_{t=1}^T \frac{\sum_{j=1}^K \frac{score_{aw}(M_{t,j})}{\log_2(j+1)}}{IdealScore_K}$$

Here T is the number of topics, and $M_{t,j}$ refers to the j^{th} keyphrase generated by method M for topic t . Unlike in [72], we have more than 2 judges, so we define $score_{aw}$ as the *agreement-weighted* average score for the $M_{t,j}$ keyphrase, which is the mean of the judges’ score multiplied by the weighted Cohen’s κ [15]. This gives a higher value to a keyphrase with scores of (3,3,3) than to one with scores of (1,3,5), though the average score is identical. Finally, $IdealScore_K$ is calculated using the scores of the top K keyphrases out of all judged keyphrases.

Table 3.3 compares the performance across different methods.⁸ The top performances are clearly variations of KERT with different parameter settings. As expected, KERT_{-cov} exhibits the worst performance. The baselines perform slightly better, and it is interesting to note that kpRel, which is smoothed purity, performs better than kpRelInt*, and even slightly better than KERT_{-phr}. This is because kpRelInt* adds in a measure of *overall* keyphrase coverage in the entire collection, which hurts rather than helps for this task. Removing completeness appears to have a very small negative effect, and we hypothesize this is because high-ranked incomplete keyphrases are relatively rare, though very

⁸Although we cannot directly evaluate which differences between the nKQM values are statistically significant, we examined the differences between the distributions of mean judge scores. We found, for example, that the preference for phrases generated by KERT_{-pur} was statistically significant, whereas the difference between KERT and KERT_{-com} was not.

obvious when they do occur (e.g. ‘vector machines’). $KERT_{\text{pur}}$ performs the best - which may reflect human bias towards longer phrases - with an improvement of at least 50% over the kpRelInt^* baseline for all reported values of K .

3.3.2 Maximizing Mutual Information

In order to perform a quantitative evaluation, we use a dataset which, unlike the DBLP collection, has been labeled. In the arXiv dataset, each physics paper title is labeled by its authors as belonging to the subfield of Optics, Fluid Dynamics, Atomic Physics, Instrumentation and Detectors, or Plasma Physics. We minimally pre-processed the titles by removing all stopwords, resulting in 9,722 titles evenly sampled from the specified 5 physics subfields, and consisting of 9,648 unique words. Since the titles are labeled, we can explore which method maximizes the mutual information between phrase-represented topics and titles. As the collection has 5 categories, we set $T=5$.

For each method, we do multiple runs for various values of K (the number of top-ranked phrases per topic considered), and calculate the mutual information MI_K for that method as a function of K . To calculate MI_K , we label each of the top K phrases in every topic with the topic in which it is ranked highest. We then check each paper title to see if it contains any of these top phrases. If so, we update the number of events “seeing a topic t and category c ” for $t = 1 \dots T$, with the averaged count for all those labeled phrases contained in the title; otherwise we update the number of events “seeing a topic t and category c ” for $t = 1 \dots T$ uniformly, where c is the Primary Category label for the paper title in consideration. Finally, we compute mutual information at K :

$$MI_K = \sum_{t,c} p(t,c) \log_2 \frac{p(t,c)}{p(t)p(c)}$$

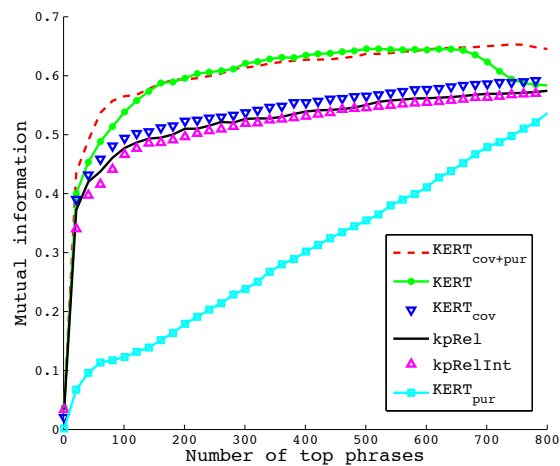


Figure 3.1: Mutual Information at K (MI_K) for various K . Methods in legend are ordered by performance, high to low.

We compare the baselines (kpRelInt* and kpRel), KERT, and variations of KERT where only coverage (KERT_{cov}), only purity (KERT_{pur}), and only coverage and purity (KERT_{cov+pur}) are used in the ranking function. We use the output of bLDA with the same parameter settings as specified in Section 3.3.1. Figure 5.3 shows MI_K for each method for a range of K .

It is clear that for MI_K , coverage is more important than purity, since KERT_{pur} is by far the worst performer. Both baselines perform nearly as well as KERT_{cov}, and all are comfortably beaten by KERT_{cov+pur} (> 20% improvement for K between 100 and 600), which uses our coverage and purity measure. It is interesting to note that adding in the phraseness and completeness measures yields no improvement in MI_K . However, the experiments with the DBLP dataset demonstrate that these measures - particularly phraseness - are very helpful in the eyes of expert human judges. In contrast, while MI_K is definitely improved with the addition of the purity measure, people seem to prefer that this metric not affect the keyphrase ranking. Although we outperform other approaches in both evaluations, these observations show interesting differences between theory-based and human-centric evaluation metrics.

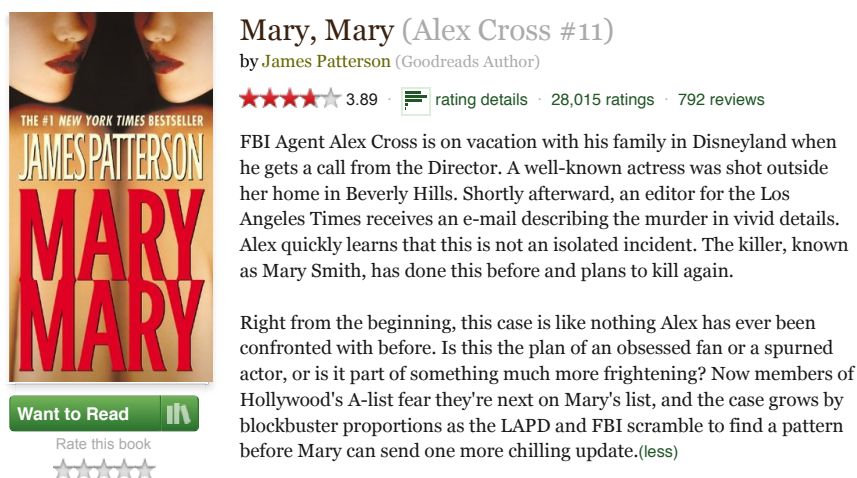
Chapter 4

A New Domain, A New Dataset, A New Evaluation

4.1 Introduction

There are several natural extensions to the KERT framework, which we present in this chapter. First, we would like to examine how well KERT works with long text. Second, it would be beneficial to leave the specific arena of scientific literature. Finally, one of the goals of the work presented in this dissertation is to assist human-centric applications. Therefore it is necessary to find such an application and demonstrate the usefulness of KERT for accomplishing it.

Goodreads¹ is a ‘social cataloguing’ website which allows users to search, organize, and review Goodreads’ database of books (other social cataloguing sites include Last.fm and Flixter.) The nice thing about using ‘social cataloguing’ data is that there is an explicit need for the task of categorization. In the case of Goodreads, books are categorized as belonging to specific genres, by members of Goodreads (For an example of a Goodreads book together with its description, see Figure 4.1.) Genre categorization is itself a subjective task, since each person decides individually which genre a book does or does not belong to. Many genre categorizations spawn active debates, such as whether a book belongs to the genres of science fiction or fantasy (or both - or neither.)



The image shows a screenshot of a Goodreads book entry. On the left is the book cover for 'Mary, Mary' by James Patterson, featuring a close-up of a woman's face. The cover text includes 'THE #1 NEW YORK TIMES BESTSELLER', 'JAMES PATTERSON', and 'MARY MARY'. Below the cover is a green 'Want to Read' button and a 'Rate this book' section with five stars. To the right of the cover, the title 'Mary, Mary (Alex Cross #11)' is displayed, followed by the author 'by James Patterson (Goodreads Author)'. Below the author name is a star rating of 3.89, a 'rating details' link, and statistics for '28,015 ratings' and '792 reviews'. A short synopsis follows: 'FBI Agent Alex Cross is on vacation with his family in Disneyland when he gets a call from the Director. A well-known actress was shot outside her home in Beverly Hills. Shortly afterward, an editor for the Los Angeles Times receives an e-mail describing the murder in vivid details. Alex quickly learns that this is not an isolated incident. The killer, known as Mary Smith, has done this before and plans to kill again.' Below the synopsis is a longer paragraph: 'Right from the beginning, this case is like nothing Alex has ever been confronted with before. Is this the plan of an obsessed fan or a spurned actor, or is it part of something much more frightening? Now members of Hollywood's A-list fear they're next on Mary's list, and the case grows by blockbuster proportions as the LAPD and FBI scramble to find a pattern before Mary can send one more chilling update.(less)'

Figure 4.1: Example of a Goodreads book, together with its description.

¹<http://www.goodreads.com>

Generating topical phrases using books descriptions from Goodreads can therefore yield two natural evaluation metrics. The first is akin to that studied in the previous chapter: How well does this set of phrases describe a particular genre? The second is a new way of evaluating the success of the algorithm: How well does this set of phrases describe a particular book? The idea here is that if a book is designated by many people as belonging to a particular genre, then a high quality genre representation would allow a person to look at the book’s description and decide to also designate the book with this genre. Thus, we are able to construct a classification task, with an existing gold standard (the genre designations of books by the Goodreads community) in order to further evaluate KERT and compare it to existing approaches.

In this chapter we demonstrate the robustness and flexibility of KERT by showing that it can be successfully applied to a collection of longer texts, which are no longer limited to the scientific domain. We present a series of experiments on a labeled dataset of book descriptions to evaluate the quality of automatically generated phrases as representations of genres.

4.2 Adapting KERT for the Goodreads dataset

The KERT framework, as presented in the previous chapter, is as follows:

Step 1. Given a collection of short texts, cluster words in the dataset into topics, using an appropriate unigram-generating topic model method.

Step 2. Construct candidate phrases for each topic according to the word topic assignments specified by the topic model output.

Step 3. Rank the phrases in each topic $t \in 1, \dots, k$ according to a well-defined ranking function.

The Goodreads dataset has several new characteristics that must be considered. The most important one for the KERT framework is that the collection is now of medium-length texts, each text consisting of one or more descriptive paragraphs, and therefore no longer falling under the category of short text. The domain is also no longer that of scientific literature. Finally, this is a labeled dataset, as each book description is characterized as belonging to one or more genres, and which can also serve as an additional source of information

4.2.1 Adapting the clustering step

Background LDA could serve very well for a dataset of long text. However, in this chapter we also seek to demonstrate the ability of KERT to incorporate additional information if it is available. Labeled LDA (LLDA) [55] is a variation of the well-know LDA topic model, with the additional ability to incorporate label information for text corpora. It is particularly well suited to working with Goodreads data because each book in Goodreads is designated as belonging

to genres by Goodreads members. Since the set of possible genres is relatively stable, a genre can therefore be treated as a label for a book description. Since the framework of KERT is modular and flexible, we can easily use LLDA to accomplish the first step, yielding word topic assignments.

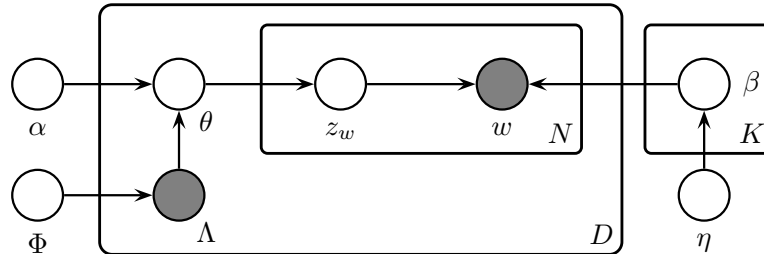


Figure 4.2: Graphical model of Labeled LDA. Unlike traditional LDA, the label set Λ influences the topic mixture θ , in addition to the topic prior α .

Like traditional LDA, Labeled LDA models each document as a mixture of underlying topics with the caveat that the topics for each document are constrained to be only those corresponding to the document's label set. Formally, let each document d with length N be represented by a tuple consisting of a list of word indices $w^d = (w_1, \dots, w_N)$ and a list of binary topic presence/absence indicators $\Lambda^d = (l_1, \dots, l_K)$ where each $l_k \in \{0, 1\}$ and K the total number of unique labels in the corpus, and thus also the number of topics.²

In contrast with the traditional LDA model which draws a multinomial mixture θ^d for every document d over all K topics, LLDA restricts θ^d to be defined only over the topics that correspond to its labels Λ^d . Figure 4.2 presents the plate notation. The generative process is as follows:

1. Draw $\beta_k \sim Dir(\eta)$, for each topic $k = 1, \dots, K$.
2. For each document $d \in D$,
 - (a) For each topic $k = 1, \dots, K$.
 - i. Draw $\Lambda_k^d \in \{0, 1\} \sim Bernoulli(\Phi_k)$
 - (b) Draw $\alpha^d = L^d \times \alpha$
 - (c) Draw $\theta^d \sim Dir(\alpha^d)$
 - (d) For each word position i in d
 - i. Draw topic $z_i \sim Multi(\theta^d)$
 - ii. Draw word $w_i \sim Multi(\beta_{z_i})$

²It is trivial to add a background topic as an additional label present in each document, and we in fact do so, in order to improve phrase quality.

where α and η are Dirichlet prior parameters. The document's labels Λ^d are first generated, with Φ_k as the label prior for topic k . The vector of these labels is then defined as $\lambda^d = k | \Lambda_k^d = 1$. We can now construct L^d , the document-specific label projection matrix of size $M_d \times K$, where $M_d = |\lambda^d|$ as follows: For each row $i \in 1, \dots, M_d$ and column $j \in 1, \dots, K$:

$$L_{ij}^d = \begin{cases} 1 & \text{if } \lambda_i^d = j \\ 0 & \text{otherwise} \end{cases} \quad (4.1)$$

We now use L^d to project the parameter vector of the Dirichlet topic prior α to a lower dimension vector α^d , whose dimensions correspond to the topics represented by the document's labels.

For example, suppose $K = 4$ and that a document d has labels given by $\Lambda^d = 0, 1, 1, 0$ which implies $\lambda^d = 2, 3$, then L^d would be:

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Then, θ^d is drawn from a Dirichlet distribution with the Dirichlet restricted to topics 2 and 3.³

We once again use the *maximum a posteriori* (MAP) principle to label each word occurrence and use that information for the subsequent steps of KERT.

4.2.2 Adapting the phrase construction step

In the previous chapter, we worked with a dataset of short texts and were therefore able to use the entire document as the window size when performing the second step - constructing candidate phrases. When the text is longer, using the entire document is both prohibitive from a scalability standpoint, and incorrect from an interpretation standpoint. Therefore, after the topic assignment step is complete, it is necessary to impose a window restriction on the phrase construction process. A simple way to do this is to break each document into reasonably sized windows, effectively turning the original collection of mid-length texts into a new collection of (many more) short texts. There is a range of options for choosing such windows, and we discuss several variations later in this chapter. A reasonably conservative option that we use on the Goodreads dataset, is to denote punctuation and stopwords as delimiters of breakpoints. For example, the first sentence of the book shown in Figure 4.1, which reads, 'FBI Agent Alex Cross is on vacation with his family in Disneyland when he gets a call from the Director,' would be broken into 6 short texts: {'fbi agent alex cross', 'vacation', 'family', 'disneyland', 'call', 'director'}. As each unigram would have a topic assignment from step 1 of KERT, steps 2 and 3 can now proceed as described in the previous chapter, since the ranking step requires no

³Example adapted from [55]

adaptation.

In this way, by choosing a different unigram-generating topic model, and by introducing the step of breaking the text into windows, KERT is able to easily work with a new and very different dataset as compared to the collections of scientific paper titles used in the previous chapter.

4.3 Experiments

4.3.1 Goodreads dataset

We crawled Goodreads for all books which have been labeled as belonging to a particular genre by at least 100 people, and which had a description written in English. For each book, the title, author, and description was obtained. Each book was also labeled as belonging to a genre if at least 100 users designated it as such. This process yielded a little over 24,000 books.

The book descriptions were stemmed and lemmatized using the Python Natural Language ToolKit [4], and the stopwords were removed. The result was a collection of medium-length texts, with an average length of 67 unigrams. Each text also had one or more labels designating the genres the book belongs to, according to Goodreads members.

We conducted identical experiments on three datasets. For each dataset, we chose three genres, selected that portion of our dataset consisting of books which have been labeled with one or more of these genres, and constructed topical phrases to characterize each genre. The three genre groups chosen were:

- **Humor, Thriller, Philosophy (HPT)** (1,235 books). These genres were chosen because they have almost no overlap (only 3 books belong to more than one genre), and should not be too difficult to distinguish.
- **Children, Young Adult, New Adult (CYN)** (3,384 books). These genres were chosen because they represent an age progression of the reader. There is a respectable amount of overlap between the first two genres (272 books designated as both Children's and Young Adult), and the last two genres (55 books designated as both Young Adult and New Adult), but none between the first and last genre.
- **History, Philosophy, Religion (HPR)** (1,231 books). These genres were chosen because they are reasonably related, with some overlap between each pair of genres (there are 72 books designated as belonging to more than one genre)

Choosing three genres for each experiment (rather than two, four, etc.) such that the sizes of each of the three datasets are comparable allows us to keep algorithm parameters consistent throughout the three experiments, and demonstrate robustness against which particular genres are chosen. Like the setting for the experiments on short texts

in the previous chapter, we set $\alpha = 10$, a bit larger than the typical setting, to reflect the fact that each book description is likely to belong to just a subset of the genres, not all of them. We similarly set $\eta = 0.1$, a bit larger than the typical setting, because different genres may share the same words. We set $\mu = 10$,⁴ $\gamma = 0.9$, and $\omega = 0.2$, though we anecdotally note that the results do not change too much with varying the latter two parameters. We limit the length of possible phrases to 5.

To perform the experiments we used Amazon Mechanical Turk⁵, a crowdsourcing service that provides a platform for human workers to perform intelligence tasks for remuneration. We required that participants completing our study be able to speak English, have already performed over 1,000 other tasks on Mechanical Turk, and have a 95% success rate (reflecting the rate of tasks completed correctly, as judged by those setting the tasks.) The complete set of instructions for both tasks can be found in Appendix B.

We compared the efficacy of KERT with PDLDA [39] and Turbo Topics [6], two phrase-generating approaches that are meant to perform well on longer documents, such as news articles, and which should therefore be well-suited to handling the Goodreads dataset. For each experiment, we use each algorithm to generate three topics, and then manually label each topic with the genre that it best fits, for evaluation purposes. So, for instance, for the experiment on the HPT dataset, we match up one topic to represent the Humor genre, one topic for the Thriller genre, and one for the Philosophy genre. However, study participants never see these manually created genre labels, and are only shown the top phrases for each generated topic.⁶

4.3.2 Qualitative Results

Tables 4.1, 4.2, and 4.3 show the top 15 phrases generated by KERT, Turbo Topics, and PDLDA representing the genres Humor, Thriller, and Philosophy. PDLDA finds a number of well-formed longer phrases (though the unigrams are not at all descriptive), but the topics are fairly mixed, although these particular three genres are not very related. Turbo Topics has only unigrams bubble up to the top, although the topic differentiation is pretty good. On the other hand, KERT demonstrates both a clean differentiation of the different topics and an array of descriptive phrases, including many authors and literary characters that are well-known in the given genre.

4.3.3 Genre Characterization Study

To mimic the first KERT study done on short text, participants were presented with a set of topical phrases asked to rate how well each set describes a particular genre. So, for example, in the HPR experiment (Humor, Thriller,

⁴Recall that this is the minimum support parameter for the phrase construction step, which means that a phrase must exist in 10 *windows*, not in 10 *descriptions* in order to qualify.

⁵<https://www.mturk.com>

⁶As PDLDA separately generates lists of topical n-grams of each length, etc, we use the top 5 unigrams, top 5 bigrams, and top 5 trigrams.

Table 4.1: Top 15 phrases generated by KERT representing the genres Humor, Thriller, and Philosophy

Philosophy	Thriller	Humor
twentieth century	secret service agent	stephanie plum bounty hunter
ayn rand	serial killer	ankh morpork
human nature	dean koontz	joe morelli
english translation	alex cross	dave barry
walter kaufmann	mitch rapp	bill bryson
micHEL foucault	medical examiner	bertie wooster
western philosophy	fbi agent	comic strip
moral philosophy	jack reacher	grandma mazur
marcus aurelius	sara linton	jasper fforde
originally publish	jeffrey tolliver	greg heffley
20th century	united states	terry pratchett
political philosophy	homicide detective	arthur dent
pure reason	kay scarpetta	scott pilgrim
major work	james bond	janet evanovich
philosophical work	jane rizzoli	laurie notaro

Table 4.2: Top 15 phrases generated by TurboTopics representing the genres Humor, Thriller, and Philosophy

Philosophy	Thriller	Humor
work	year	world
life	murder	man
reader	secret	life
human	day	time
philosophy	death	make
story	call	love
write	killer	thing
year	woman	turn
person	begin	family
history	truth	friend
question	kill	live
time	force	end
make	case	child
mind	past	good
show	crime	great

Philosophy), a participant would see the nine sets of topical phrases shown in Tables 4.1, 4.2, and 4.3 (with each column shown independently), and would be asked to rate, on a 5 point Likert scale, how well each topical phrase represents the ‘Humor’ genre.

For each of the three experiments, nine topical phrase groups were generated (since each experiment had three genres). Each set of nine phrase groups was paired with one of the three genres. 20 people were asked to evaluate each such combination. Therefore, this study generated a total of 1,620 data points for evaluation.

Table 4.3: Top 15 phrases generated by PDLDA representing the genres Humor, Thriller, and Philosophy

Philosophy	Thriller	Humor
life	life	life
young woman	united states	serial killer
world war ii	john stuart mill	popular comic strip
find	find	work
jack reacher	san francisco	human nature
boston medical examiner	secret service agent	oliver wendell holmes
world	time	world
year ago	stephanie plum	bill bryson
wee free men	camp green lake	lazy sunday afternoon
man	world	philosophy
medical examiner	alex cross	ankh morpork
commander gray pierce	alex cross series	dr sara linton
time	make	make
grandma mazur	small town	twentieth century
beautiful young woman	supreme court justice	henry david thoreau

Table 4.4: Overall accuracy for Genre Characterization

	Humor Philosophy Thriller	Children’s Young Adult New Adult	History Philosophy Religion
KERT	4.33 > 2.10 ^{††}	3.85 > 2.84 ^{††}	4.28 > 2.43 ^{††}
PDLDA	2.65 ≈ 2.49	3.68 > 3.22 ^{**}	2.82 ≈ 2.62
TurboTopics	3.77 > 2.27 ^{††}	3.55 > 3.20 ^{**}	3.37 > 2.84 ^{***}
	* $p < 0.1$ * $p < 0.05$ * $p < 0.01$	† $p < 0.001$	†† $p < 0.0001$

Evaluating overall accuracy

For each experiment, each algorithm was first evaluated on its overall accuracy. We compared the Likert score distributions for the topical phrase groups whose genre matched the genre being queried, and the Likert score distributions for the topical phrase groups whose genre did *not* match the genre being queried. For example, in the HPR experiment, an algorithm’s overall accuracy compare the scores that the algorithm’s ‘Humor’ topical phrases earned when asked how well these topical phrases describe the ‘Humor’ genre (as well as the algorithm’s ‘Thriller’ topical phrases for the ‘Thriller’ genre, and ‘Philosophy’ topical phrases for the ‘Philosophy’ genre), against the scores that KERT’s ‘Thriller’ and ‘Philosophy’ topical phrases earned when asked how well these topical phrases describe the ‘Humor’ genre (and the same for the other two sets of combinations). If the algorithm performs well, the first set of scores should have a high mean (> 3), representing the fact that they are a good, or very good match for the genre that they are supposed to be describing. The second set of scores should have a low mean (≤ 3), representing the fact that they are a neutral, bad, or very bad matches for the genres that they are *not* supposed to be describing. There should also be a statistically significant difference between the two distributions, which we tested using Wilcoxon Rank Sum.

Table 4.5: Inter-rater agreement for Genre Characterization (weighted κ)

	Humor Philosophy Thriller	Children’s Young Adult New Adult	History Philosophy Religion
KERT	0.72	0.71	0.73
PDLDA	0.70	0.73	0.73
TurboTopics	0.75	0.72	0.74

Table 4.6: Pairwise accuracy for Humor, Philosophy, Thriller)

	Humor	Philosophy	Thriller
KERT vs PDLDA	4.00 > 2.45***	4.70 > 2.40 [†]	4.30 > 3.10***
KERT vs TurboTopics	4.00 > 2.55 [†]	4.70 \approx 4.50	4.30 \approx 4.25
PDLDA vs TurboTopics	2.45 \approx 2.55	2.40 < 4.50 [†]	3.10 < 4.25***
	* $p < 0.1$ * $p < 0.05$ *** $p < 0.01$	[†] $p < 0.001$	^{††} $p < 0.0001$

Table 4.4 presents the results of this study. Similar to what we seen in the previous chapter, PDLDA generally struggles to yield groups of phrases which clearly belong to a particular topic. KERT is able to clearly differentiate between genres in all three experiments. TurboTopics succeeds in two out of three experiments, but the differences between the ‘correct’ and ‘wrong’ distributions are less strong than with KERT.

Evaluating inter-rater agreement

We briefly report the inter-rater agreement for each experiment, calculated using weighted Cohen’s κ [15], as it is well-suited to ordinal data. The κ value does not vary much, regardless of the algorithm or the genre, suggesting that the observed variations in the data are not there due to, for example, there being significantly higher consensus for KERT than for the other algorithms. The κ value remains around 0.70 - 0.75, which does reflect the subjectiveness and noisiness of the task.

Evaluating pairwise accuracy

The second level of analysis directly compared the performances of the three algorithms on each genre, of each of the three experiments, using a paired Wilcoxon Rank Sum test. Tables 4.6, 4.7, and 4.8 present the results of this analysis. So, for example, the first cell of Table 4.6 shows that for the topical phrase sets which were intended to describe the ‘Humor’ genre, the mean score of KERT (4.00) is larger than the mean score of PDLDA (2.45), with the difference between the two distributions being significant at $p < 0.01$.

In the HPT experiment, KERT is clearly more successful than PDLDA within all three genres. In the ‘Humor’ genre, KERT is better than TurboTopics, but the latter is equally good at representing the Philosophy and Thriller genres. All of the algorithms struggle with the CYN experiment, with the only successes being KERT and TurboTopics

Table 4.7: Pairwise accuracy for Children’s, Young Adult, New Adult)

	Children’s	Young Adult	New Adult
KERT vs PDLDA	4.25 > 3.55**	3.60 ≈ 3.90	3.70 ≈ 3.60
KERT vs TurboTopics	4.25 ≈ 4.00	3.60 ≈ 3.40	3.70 ≈ 3.25
PDLDA vs TurboTopics	3.55 < 4.00*	3.90 ≈ 3.40	3.60 ≈ 3.25
	* $p < 0.1$ ** $p < 0.05$ *** $p < 0.01$	† $p < 0.001$	†† $p < 0.0001$

Table 4.8: Pairwise accuracy for History, Philosophy, Religion)

	History	Philosophy	Religion
KERT vs PDLDA	3.60 < 4.10*	4.65 > 2.15†	4.60 > 2.20†
KERT vs TurboTopics	3.60 ≈ 3.80	4.65 > 3.80***	4.60 > 2.50†
PDLDA vs TurboTopics	4.10 ≈ 3.80	2.15 < 3.80***	2.20 ≈ 2.50
	* $p < 0.1$ ** $p < 0.05$ *** $p < 0.01$	† $p < 0.001$	†† $p < 0.0001$

both besting PDLDA in the Children’s genre. PDLDA scores a rare victory over KERT in the History genre of the HPR experiment, but KERT is more successful than both PDLDA and TurboTopics in both of the other genres.

4.3.4 Book Characterization Study

For the second study, participants were presented with a set of topical phrases and asked to rate how well each set describes a particular book. This is akin to labeling a book with appropriate genres. So, for example, in the first experiment (Humor, Thriller, Philosophy), a participant would see the nine sets of topical phrases shown in Tables 4.1, 4.2, and 4.3 (with each column shown independently), and would be asked to rate, on a 5 point Likert scale, how well each topical phrase represents a book description, such as the one shown in Figure 4.1.

For each of the three experiments, nine topical phrase groups were generated (since each experiment had three genres). For each set of nine phrase groups, five books were selected for evaluation for each one of the three genres (where each book was labeled with exactly one genre). 10 people were asked to evaluate each such combination. Therefore, this study generated a total of 4,050 data points for evaluation.

Evaluating overall accuracy

For each experiment, each algorithm was once again first evaluated on its overall accuracy. We compared the Likert score distributions for the topical phrase groups whose genre matched the genre of the presented book and the Likert score distributions for the topical phrase groups whose genre did *not* match the genre of the presented book. For example, in the HPR experiment, an algorithm’s overall accuracy compare the scores that the algorithm’s ‘Humor’ topical phrases earned when asked how well these topical phrases describe a book labeled as belonging to the ‘Humor’ genre (as well as the algorithm’s ‘Thriller’ topical phrases for ‘Thriller’ books, and ‘Philosophy’ topical phrases for

Table 4.9: Overall accuracy for Book Characterization)

	Humor Philosophy Thriller	Children’s Young Adult New Adult	History Philosophy Religion
KERT	3.53 > 2.09 ^{††}	3.93 > 1.87 ^{††}	3.99 > 1.88 ^{††}
PDLDA	2.50 ≈ 2.42	2.11 > 2.28 ^{**}	2.05 ≈ 2.04
TurboTopics	3.11 > 2.42 ^{††}	3.33 > 3.00 ^{**}	3.30 > 2.75 ^{††}

* $p < 0.1$ ** $p < 0.05$ *** $p < 0.01$ † $p < 0.001$ †† $p < 0.0001$

Table 4.10: Inter-rater agreement for Book Characterization (weighted κ)

	Humor Philosophy Thriller	Children’s Young Adult New Adult	History Philosophy Religion
KERT	0.69	0.75	0.76
PDLDA	0.70	0.76	0.77
TurboTopics	0.71	0.75	0.74

‘Philosophy’ books), against the scores that KERT’s ‘Thriller’ and ‘Philosophy’ topical phrases earned when asked how well these topical phrases describe ‘Humor’ books (and the same for the other two sets of combinations). If the algorithm performs well, the first set of scores should have a high mean (> 3), representing the fact that they are a good, or very good match for books belonging to the genre that they are supposed to be describing. The second set of scores should have a low mean (≤ 3), representing the fact that they are a neutral, bad, or very bad matches for books belonging to the genres that they are *not* supposed to be describing.

Table 4.4 presents the results of this study. Once again, PDLDA struggles to yield groups of phrases which clearly belong to a particular topic. Both KERT and TurboTopics are able to clearly differentiate between genres in all three experiments, but the differences between the ‘correct’ and ‘wrong’ distributions for KERT are consistently better than those for TurboTopics. We also see that PDLDA is consistently either equal to or worse than TurboTopics.

Evaluating inter-rater agreement

We again briefly report the inter-rater agreement for each experiment, calculated using weighted Cohen’s κ [15]. Similar to the Genre Characterization study, the κ value does not vary much, and again remains around 0.69-0.76 regardless of the algorithm or the genre.

Evaluating pairwise accuracy

The second level of analysis once again directly compared the performances of the three algorithms on each genre, of each of the three experiments, using a paired Wilcoxon Rank Sum test. Tables 4.6, 4.7, and 4.8 present the results of

Table 4.11: Pairwise accuracy for Humor, Philosophy, Thriller)

	Humor	Philosophy	Thriller
KERT vs PDLDA	3.04 > 2.44**	3.94 > 2.32 [†]	3.60 > 2.74 [†]
KERT vs TurboTopics	3.04 > 2.48*	3.94 > 3.56*	3.60 ≈ 3.28
PDLDA vs TurboTopics	2.44 ≈ 2.88	2.32 < 3.56 [†]	2.74 < 3.28**
	* $p < 0.1$ ** $p < 0.05$ *** $p < 0.01$	[†] $p < 0.001$	^{††} $p < 0.0001$

Table 4.12: Pairwise accuracy for Children’s, Young Adult, New Adult)

	Children’s	Young Adult	New Adult
KERT vs PDLDA	4.32 > 1.66 ^{††}	3.90 > 2.02 ^{††}	3.58 > 2.66 [†]
KERT vs TurboTopics	4.32 > 3.38 ^{††}	3.90 > 3.28**	3.58 ≈ 3.34
PDLDA vs TurboTopics	1.66 < 3.38 ^{††}	2.02 < 3.28 ^{††}	2.66 < 3.34 [†]
	* $p < 0.1$ ** $p < 0.05$ *** $p < 0.01$	[†] $p < 0.001$	^{††} $p < 0.0001$

this analysis. So, for example, the first cell of Table 4.6 shows that for the topical phrase sets which were intended to describe the ‘Humor’ genre, the mean score of KERT on ‘Humor’ books (3.04) is larger than the mean score of PDLDA (2.44), with the difference between the two distributions being significant at $p < 0.05$.

In the HPT experiment, KERT is more successful than PDLDA within all three genres, and better than TurboTopics in all but ‘Thriller’. Given how much the algorithms struggled with the CYN experiment, it is very interesting to see that for this second evaluation task, KERT is able to clearly best the other two algorithms in 5 out of 6 cases (it is as successful as TurboTopics for the ‘New Adult’ genre). In the HPR experiment, KERT is decidedly better than PDLDA (compare with the fact that in the first evaluation, PDLDA was better than KERT for the history genre), and remains either better than or equal to TurboTopics. Finally, we again note that PDLDA is consistently either equal to or worse than TurboTopics across all three experiments.

Characterizing Books Belonging to Multiple Genres

In the third experiment’s dataset (History, Philosophy, Religion), there is a substantial enough amount of books which are labeled with more than one genre that we also ran an experiment seeing whether these books could be correctly categorized. This is a more difficult problem. Table 4.14 presents the overall accuracy for each of the three algorithms.

Table 4.13: Pairwise accuracy for History, Philosophy, Religion)

	History	Philosophy	Religion
KERT vs PDLDA	3.72 > 2.96 [†]	4.10 > 1.64 ^{††}	4.16 > 1.56 ^{††}
KERT vs TurboTopics	3.72 ≈ 3.48	4.10 > 3.90	4.16 > 2.52 ^{††}
PDLDA vs TurboTopics	2.96 > 3.48***	1.64 < 3.90 ^{††}	1.56 < 2.52 ^{††}
	* $p < 0.1$ ** $p < 0.05$ *** $p < 0.01$	[†] $p < 0.001$	^{††} $p < 0.0001$

Table 4.14: Overall accuracy for book characterization for books with multiple labels)

	History Philosophy Religion
KERT	2.96 > 2.28 ^{††}
PDLDA	2.15 ≈ 2.11
TurboTopics	3.19 ≈ 3.27

† † $p < 0.0001$

KERT is the only algorithm to have a statistically significant difference between the two ‘correct’ and ‘incorrect’ score distributions, although it struggles to have a sufficiently high mean for the ‘correct’ score distribution. However, this task appears to be overall too noisy for human judgement

4.4 Scalability

In this section we examine the scalability of KERT, PDLDA, and Turbo Topics. Table 4.15 presents the runtime of generating topical phrases for each of the three genre groups described in this chapter. Turbo Topic’s runtime is clearly the worst, growing exponentially with the size of the dataset. PDLDA is the most efficient, but as the above analysis showed, the generated results are generally unacceptably muddled. KERT’s runtime is not too much worse than PDLDA, and significantly better than Turbo Topics. Over 99% of the runtime is devoted to the step of topic assignment (running LLDA), but once that bottleneck is passed, the rest of the framework runs in 1 or 2 seconds.

Table 4.15: Comparing runtimes of KERT, PDLDA and Turbo Topics, on Goodreads datasets.

	Humor Philosophy Thriller	Children’s Young Adult New Adult	History Philosophy Religion
<i># books*</i>	1,235	3,384	1,231
KERT	10 m 17 s	22 m 29 s	10 m 35 s
PDLDA	2 m 36 s	4 m 50 s	3 m 16 s
TurboTopics	20 m 10 s	5 hr 54 m 50 s	52 m 3 s

*Note that this refers to the number of books, *not* the number of windows constructed from the text.

For comparison, we also examined the performance of Turbo Topics and PDLDA on several collections of short texts, as shown in Table 4.16. In addition to the DBLP dataset and the arXiv dataset, we extracted the title of 13,638 papers from the ACL Anthology Network Dataset,⁷ to form yet another short text collection, referred to in Table 4.16 as the ACL_{title} dataset. Finally, we also created a very large short text dataset using a full collection of the titles available in DBLP ($DBLP_{full}$), consisting of over 1.8 million paper titles.⁸

⁷<http://clair.eecs.umich.edu/aan/index.php>

⁸This DBLP snapshot was constructed on Sept 29, 2012.

Table 4.16: Comparing runtimes of KERT, PDLDA and Turbo Topics, on short texts.

	DBLP_{full}	DBLP	arXiv	ACL_{title}
<i># documents</i>	<i>1,832,469</i>	<i>33,313</i>	<i>9,722</i>	<i>13,638</i>
KERT*	1 h 46 m 1 s	27.57 s	13.1 s	23.3 s
PDLDA	1 d 5 h 17 m 13 s	15 m 32 s	4 m 35 s	12 m 17 s
Turbo Topics	27 d	4 h 58 m	10 m 58 s	9 m 22 s

*For KERT we set $\mu = 10$ for DBLP_{full}; we set $\mu = 5$ for DBLP, arXiv, and ACL_{title}

KERT clearly outperforms both PDLDA and Turbo Topics on every short dataset. On the three smaller collections of short texts (DBLP, arXiv, and ACL_{title}), KERT takes less than half a minute, while PDLDA takes up to 15 minutes, and Turbo Topics takes as long as five hours (on the DBLP dataset). The runtime of Turbo Topics quickly becomes unacceptable, requiring nearly a month to run on DBLP_{full}. PDLDA shows a significant improvement over Turbo Topics as the datasets increase in size, taking a day where Turbo Topics took a month, and 15 minutes where Turbo Topics took 5 hours. However, KERT’s improvement over PDLDA is nearly as great, requiring 8 minutes to PDLDA’s 2 days, and finishing the full DBLP dataset in under two hours. The bulk of KERT’s runtime is once again devoted to the step of topic assignment.

4.5 On Breaking Text Into Windows

We end this chapter with a discussion of the process of breaking text into smaller windows, which is a necessary step for KERT’s analysis of longer text. As described above, in this set of experiments we amend the KERT framework by breaking the text into short windows after the topic assignment step is complete, and before beginning the phrase construction step. A reasonably conservative option that we use on the Goodreads dataset, is to denote punctuation and stopwords as delimiters of breakpoints. However, we considered several other options as well, the effect of which we demonstrate using the phrases that were generated to represent the genres Humor, Thriller, and Philosophy:

- Table 4.17 reproduces an earlier table from this chapter, showing phrases generated by KERT using windows delimited by any kind of punctuation or by stopwords.
- Table 4.18 shows the phrases generated using sentences as windows, so that a phrase may be constructed from any unigrams within the same sentence.
- Table 4.19 shows the phrases generated using only any kind of punctuation as window delimiters, so that a phrase may be constructed from any unigrams not separated by a period, comma, etc.
- Table 4.20 shows the phrases generated using the NLP approach of chunking, which identifies grammatically structured parts of sentences [44]. In particular, we use the Python Natural Language ToolKit [4] to identify

noun phrases, and turn each noun phrase into a window. A noun phrase (NP) is a part of the sentence in which information about the noun is gathered, such as ‘black dog’.

Using only sentences or only punctuation as window delimiters increases the size of the window and allows for unigrams which are more separated to be combined into phrases. Thus, these two approaches can find the phrase ‘critique pure reason,’ which of course represents the seminal philosophical work ‘Critique of Pure Reason’, and which cannot be identified by the more conservative approach, which would split the title into two windows. However, this is then a tradeoff for seeing phrases such as ‘laugh loud’ (which must have originally been ‘laugh out loud’). These differences do not appear to be very severe, although we hypothesize that using a window as large as a whole sentence may turn out to be too permissive for some other datasets, such as scientific literature.

The results generated actually using noun phrases as windows are very similar to those in Table 4.17. However, unlike the other three text breaking approaches, which do not add almost any time to the KERT framework, the method of using noun phrases requires Part-Of-Speech tagging of the text, and is therefore considerably more time-intensive. Therefore, KERT is able to generate basically the same phrases in much less time, and without the need for the external knowledge of grammar (which also needs to change if the dataset domain or language changes), and is a better choice than relying on this NLP technique.

We draw attention to those topical phrases which are agreed upon as belonging in the top 15 by all of the window breaking methods by putting them in bold (e.g., all four methods agree that ‘**twentieth century**’ is a good topical phrase for the Philosophy genre). To be conservative, we do not count phrases that are clearly subsets of one another (e.g., ‘ankh morpork’ and ‘city ankh morpork’). The phrases that meet this criteria are generally of good quality, suggesting a potential approach that would leverage multiple segmentation strategies to ‘vote’ for a good phrase, thus giving extra weight to phrases such as ‘**serial killer**’ for the Thriller genre.

Table 4.17: Top 15 phrases generated using windows delimited by any kind of punctuation or by stopwords, representing the genres Humor, Thriller, and Philosophy (this method was the one used in the above experiments). Phrases in bold are ranked in the top 15 by all four of the window breaking methods discussed in this section.

Philosophy	Thriller	Humor
twentieth century	secret service agent	stephanie plum bounty hunter
ayn rand	serial killer	ankh morpork
human nature	dean koontz	joe morelli
english translation	alex cross	dave barry
walter kaufmann	mitch rapp	bill bryson
michel foucault	medical examiner	bertie wooster
western philosophy	fbi agent	comic strip
moral philosophy	jack reacher	grandma mazur
marcus aurelius	sara linton	jasper fforde
originally publish	jeffrey tolliver	greg heffley
20th century	united states	terry pratchett
political philosophy	homicide detective	arthur dent
pure reason	kay scarpetta	scott pilgrim
major work	james bond	janet evanovich
philosophical work	jane rizzoli	laurie notaro

Table 4.18: Top 15 phrases generated using windows delimited by sentences, representing the genres Humor, Thriller, and Philosophy. Phrases in bold are ranked in the top 15 by all four of the window breaking methods discussed in this section.

Philosophy	Thriller	Humor
critique pure reason	women murder club	stephanie plum bounty hunter
ayn rand	alex cross	hitchhiker guide galaxy
twentieth century	serial killer	bertie wooster gentleman
human nature	dean koontz	joe morelli cop
work philosophy	jack reacher	city ankh morpork
english translation	medical examiner	calvin hobbes
20th century	fbi agent	dave barry
century 19th	mitch rapp	laugh loud
michel foucault	homicide detective	grandma mazur
moral philosophy	secret service	san francisco
marcus aurelius	jason bourne	janet evanovich
alain botton	sara linton	comic strip
social political	rhyme sachs	bill watterson
publish volume	lincoln rhyme	bill bryson
major work	police department	spaceman spiff

Table 4.19: Top 15 phrases generated using windows delimited by **any kind of punctuation**, representing the genres Humor, Thriller, and Philosophy. Phrases in bold are ranked in the top 15 by all four of the window breaking methods discussed in this section.

Philosophy	Thriller	Humor
critique pure reason	alex cross	stephanie plum bounty hunter
ayn rand	medical examiner	bertie wooster gentleman
twentieth century	serial killer	joe morelli cop
english translation	dean koontz	calvin hobbes
human nature	jack reacher	laugh loud
philosophy work	lincoln rhyme	dave barry
michel foucault	fbi agent	jasper fforde
social political	mitch rapp	grandma mazur
include introduction	sara linton	bill bryson
19th century	jason bourne	janet evanovich
moral philosophy	young woman	bill watterson
philosophical work	united states	ankh morpork
alain botton	rhyme sachs	comic strip
literary criticism	kay scarpetta	arthur dent
writing selection	police department	bail bond

Table 4.20: Top 15 phrases generated using windows defined by **identifying noun phrases**, representing the genres Humor, Thriller, and Philosophy. Phrases in bold are ranked in the top 15 by all four of the window breaking methods discussed in this section.

Philosophy	Thriller	Humor
twentieth century	secret service agent	stephanie plum bounty hunter
ayn rand	alex cross	joe morelli
human nature	serial killer	ankh morpork
marcus aurelius	jack reacher	dave barry
western philosophy	dean koontz	comic strip
moral philosophy	medical examiner	bertie wooster
michel foucault	mitch rapp	bill bryson
understand concern	fbi agent	grandma mazur
english translation	homicide detective	greg heffley
major work	young woman	scott pilgrim
walter kaufmann	jane rizzoli	janet evanovich
literary criticism	kay scarpetta	laurie notaro
originally publish	sara linton	bill watterson
20th century	maura isles	calvin hobbes
political philosophy	james bond	jasper fforde

Chapter 5

Extending the Framework to Construct Hierarchical Topical Phrases

5.1 Introduction

¹A different direction that is necessary to consider is that a flat representation of the topics (or concepts) in a dataset may not always be as useful as a hierarchical organization at different levels of granularity. Therefore it is necessary to be able to also construct high quality topical hierarchies from texts, where each topic is represented by a ranked list of topical phrases, such that a child topic is a subset of its parent topic. We are therefore motivated to create a robust framework for constructing high quality topical hierarchies from texts in different domains.

For this task, we return to working with datasets of short texts, as introduced in Chapter 3. Our framework aims to construct a hierarchy where each topic is represented by a ranked list of topical phrases, such that a child topic is a subset of its parent topic. For example, the topic of query processing and optimization may be described by the phrases {‘query processing’, ‘query optimization’,...}, while its parent topic of general problems in databases may be described by {‘query processing’, ‘database systems’, ‘concurrency control’,...}

Our goal has several challenges. Topical phrases that would be regarded as high quality by human users are likely to vary in length (e.g., ‘support vector machines’ and ‘feature selection’ would both be good phrases for a topic about machine learning). Existing phrase extraction and ranking approaches are term-centric and cannot directly compare such mixed-length phrases, highlighting the need for a phrase-centric approach. Globally frequent phrases are not assured to be good representations for individual topics, demonstrating the need to infer the frequency of phrases in each topic. Finally, we must be able to recursively estimate each phrase’s topical frequency for subtopics, in order to construct a topical hierarchy.

Although there exist several topic modeling approaches that can model the hierarchical dependency of unigram-based topics [24, 38, 49], it is challenging to apply these techniques to our scenario because i) since our text is sparse, the distribution estimates are quite brittle [19] when calculating multiple topic levels, and ii) these methods compute the entire hierarchy simultaneously and do not support recursive discovery of subtopics from a topic.

¹This chapter contains materials from the following previously published work: Chi Wang, Marina Danilevsky, Nihit Desai, Yinan Zhang, Phuong Nguyen, Thirvikrama Taula, and Jiawei Han. A Phrase Mining Framework for Recursive Construction of a Topical Hierarchy. Proc. of 2013 ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD13), Chicago, IL, Aug. 2013. Copyright 2013 ACM, Inc. <http://doi.acm.org/10.1145/2487575.2487631>. Reprinted with permission.

In this chapter we present CATHY (Constructing A Topical HierarchY), a phrase-centric framework for topical hierarchy generation via recursive clustering and ranking. CATHY adapts the phrase construction and ranking approaches of the KERT framework; however, unigram-generating topic models are not able to naturally construct a hierarchy of topics, such that a child topic is a subset of its parent topic. Therefore the greatest contribution of this chapter is the transformation of the topic assignment step into one of hierarchical topic assignment. We also demonstrate the efficacy of CATHY on collections of short texts through a series of experiments.

5.2 Problem Formulation

Traditionally, a phrase is defined as a consecutive sequence of terms, or unigrams. However, as discussed in Chapter 3 this definition can be quite limiting as it is too sensitive to natural variations in the term order, or the morphological structure of a phrase. For instance, consider that two computer science paper titles, one containing ‘mining frequent patterns’ and the other containing ‘frequent pattern mining,’ are clearly discussing the same topic, and should be treated as such. A phrase may also be separated by other terms: ‘**mining** top-k **frequent** closed **patterns**’ also belongs to the topic of frequent pattern mining, in addition to incorporating secondary topics of top-k frequent patterns, and closed patterns. Therefore, we continue to define a phrase to be an order-free set of terms appearing in the same document. Our framework can work with alternative definition of phrases as well, such as traditionally defined consecutive ngrams.

Definition 3 (Phrase) *A phrase P with length n is an unordered set of n terms: $P = \{w_{x_1}, \dots, w_{x_n} | w_{x_i} \in W\}$, where W is the set of all unique terms in a content-representative document collection. The frequency $f(P)$ of a phrase is the number of documents in the collection that contain all of the n terms.*

We use phrases as the basic units for constructing a topical hierarchy.

Definition 4 (Topical Hierarchy) *A topical hierarchy is defined as a tree \mathcal{T} in which each node is a topic. The root topic is denoted as o . Every non-root topic t with parent topic $\text{par}(t)$ is represented by a ranked list of phrases $\{\mathcal{P}^t, r^t(\mathcal{P}^t)\}$, where \mathcal{P}^t is the set of phrases for topic t , and $r^t(\mathcal{P}^t)$ is the ranking score for the phrases in topic t . For every non-leaf topic t in the tree, all of its subtopics comprise its children set $C^t = \{z \in \mathcal{T}, \text{par}(z) = t\}$. A phrase can appear in multiple topics, though it will have a different ranking score in each topic.*

To construct a topical hierarchy, we must soft cluster phrases into a hierarchy and find representative phrases for each topic. As an example, consider the task of judging what constitutes high quality phrases for various topics in computer science. There are four criteria for judging the quality of a phrase, previously presented in Chapter 3:

- **Coverage:** A representative phrase for a topic should cover many documents within that topic. *Example: ‘information retrieval’ has better coverage than ‘cross-language information retrieval’ in the Information Retrieval topic.*
- **Purity:** A phrase is pure in a topic if it is only frequent in documents belonging to that topic and not frequent in documents within other topics. *Example: ‘query processing’ is more pure than ‘query’ in the Databases topic.*
- **Phraseness:** A group of terms should be combined together as a phrase if they co-occur significantly more often than the expected chance co-occurrence frequency, given that each term in the phrase occurs independently. *Example: ‘active learning’ is a better phrase than ‘learning classification’ in the Machine Learning topic.*
- **Completeness:** A phrase is not complete if it is a subset of a longer phrase, in the sense that it rarely occurs in a document without the presence of the longer phrase. *Example: ‘support vector machines’ is a complete phrase, whereas ‘vector machines’ is not because ‘vector machines’ is almost always accompanied by ‘support’ in documents.*

The measures which represent these criteria can all be characterized by an important concept: topical frequency.

Definition 5 (Topical Frequency) *The topical frequency $f_t(P)$ of a phrase is the count of the number of times the phrase is attributed to topic t . For the root node o , $f_o(P) = f(P)$. For each topic node in the hierarchy, with subtopics C^t , $f_t(P) = \sum_{z \in C^t} f_z(P)$, i.e., the topical frequency is equal to the sum of the sub-topical frequencies.*

Table 5.1 illustrates an example of estimating topical frequency for phrases in a computer science topic that has 4 subtopics. The phrase ‘support vector machines’ is estimated to belong entirely to the Machine Learning (ML) topic with high frequency, and therefore is a candidate for a high quality phrase. However, ‘social networks’ is fairly evenly distributed among three topics, and is thus less likely to be a high quality phrase. Section 5.3.3 discusses how such candidate phrases are actually ranked, using measures based on estimated topical frequency.

Table 5.1: Example of estimating topical frequency. The topics are assumed to be inferred as machine learning, database, data mining, and information retrieval from the collection

Phrase	ML	DB	DM	IR	Total
<i>support vector machines</i>	85	0	0	0	85
<i>query processing</i>	0	212	27	12	251
<i>world wide web</i>	0	7	1	26	34
<i>social networks</i>	39	1	31	33	104

5.3 CATHY Framework

In order to estimate the topical frequency for each phrase, we need to infer the dataset’s topics. We perform topic inference and estimate topical frequency by analyzing our dataset’s term co-occurrence network.

Formally, every topic node t in the topical hierarchy is associated with a term co-occurrence network G^t . The root node o is associated with the term co-occurrence network G^o constructed from the collection of content-representative documents. G^o consists of a set of nodes W and a set of links E . A node $w_i \in W$ represents a term, and a link (w_i, w_j) between two nodes represents a co-occurrence of the two terms in a document. The number of links $e_{ij} \in E$ between two nodes w_i and w_j is equal to the number of documents containing both terms. For every non-root node $t \neq o$, we construct a subnetwork G^t by clustering the term co-occurrence network of its parent $par(t)$. G^t has all of the nodes from $G^{par(t)}$, but only those links belonging to the particular subtopic t .

We chose to use term co-occurrence network for topic analysis instead of document-term topic modeling because the it naturally supports recursive mining: the clustering result for one topic can be used as the input when further partitioning the topic into subtopics. The CATHY framework generates a topical hierarchy in a top-down, recursive way:

Step 1. Construct the term co-occurrence network $G^o = (W, E)$ from the document collection. Set $t = o$.

Step 2. For a topic t , cluster the term co-occurrence network G^t into subtopic subnetworks $G^z, z \in C^t$, and estimate the subtopical frequency for its subtopical phrases using a generative model.

Step 3. For each topic $z \in C^t$, extract candidate phrases based on estimated topical frequency.

Step 4. For each topic $z \in C^t$, rank the topical phrases using a unified ranking function based on topical frequency. Phrases of different lengths are directly compared, yielding an integrated ranking.

Step 5. Recursively apply Steps 2 - 5 to each subtopic $z \in C^t$ to construct the hierarchy in a top-down fashion.

5.3.1 Clustering: Estimating Topical Frequency

We first introduce the process of clustering for one topic t . We assume C^t contains k child topics, denoted by $z = 1 \dots k$. The value of k can be either specified by users or chosen using a model selection criterion such as the Bayesian Information Criterion [58].

In the term co-occurrence network G^t , we assume every co-occurrence of two terms w_i and w_j is attributed to a topic $z \in C^t = \{1, \dots, k\}$. We represent the total link frequency $e_{i,j}$ between w_i and w_j as a summation of the number of links between w_i and w_j in each of the k topics: $e_{ij} = \sum_{z=1}^k e_{ij}^z$. The goal is thus to estimate e_{ij}^z for $z = 1 \dots k$, which is unlike most network analysis approaches.

We develop a generative model of the term co-occurrence network, and estimate topical frequency $f_z, z \in C^t$.

A generative model for term co-occurrence network analysis

To generate a topic- z link, we first generate one end node w_i following a multinomial distribution $p(w_i|z) = \theta_i^z$, and then generate the other end node w_j with the same multinomial distribution $p(w_j|z) = \theta_j^z$. The probability of generating a topic- z link (w_i, w_j) is therefore $p(w_i|z)p(w_j|z) = \theta_i^z\theta_j^z$.

With this generative assumption for each individual link, we can derive the distribution of topical frequency for any two terms (w_i, w_j) . If we repeat the generation of topic- z links for ρ_z iterations, then the chance of generating a particular topic- z link between w_i and w_j can be modeled as a Bernoulli trial with success probability $\theta_i^z\theta_j^z$. When ρ_z is large, the total number of successes e_{ij}^z approximately follows a Poisson distribution $Pois(\rho_z\theta_i^z\theta_j^z)$.

Now we can write down the generative model for random variables e_{ij}^z with parameters ρ_z, θ^z .

$$e_{ij}^z \sim Poisson(\rho_z\theta_i^z\theta_j^z), z = 1, \dots, k \quad (5.1)$$

$$\sum_{i=1}^{|W|} \theta_i^z = 1, \theta_i^z \geq 0, \rho_z \geq 0 \quad (5.2)$$

The constraints guarantee a probabilistic interpretation. According to the *expectation* property of the Poisson distribution, $E(e_{ij}^z) = \rho_z\theta_i^z\theta_j^z$. Also, according to the *additive* property of expectations,

$$E(\sum_{i,j} e_{ij}^z) = \sum_{i,j} \rho_z\theta_i^z\theta_j^z = \rho_z \sum_i \theta_i^z \sum_j \theta_j^z = \rho_z$$

In other words, ρ_z is the total expected number of links in topic z .

One important implication due to the *additive* property of Poisson distribution is that

$$e_{ij} = \sum_{z=1}^k e_{ij}^z \sim Poisson(\sum_{z=1}^k \rho_z\theta_i^z\theta_j^z) \quad (5.3)$$

So given the model parameters, the probability of all observed links is

$$\begin{aligned} p(\{e_{ij}\}|\theta, \rho) &= \prod_{w_i, w_j \in W} p(e_{ij}|\theta_i, \theta_j, \rho) \\ &= \prod_{w_i, w_j \in W} \frac{(\sum_{z=1}^k \rho_z\theta_i^z\theta_j^z)^{e_{ij}} \exp(-\sum_{z=1}^k \rho_z\theta_i^z\theta_j^z)}{e_{ij}!} \end{aligned} \quad (5.4)$$

In this model, the observed information is the total number of links between every pair of nodes, including zero links and self-links. The parameters which must be learned are the role of each node in each topic $\theta_i^z, w_i \in W, z = 1, \dots, k$, and the expected number of links in each topic ρ_z . The total number of free parameters to learn is therefore $k|W|$.

We learn the parameters by the *Maximum Likelihood* (ML) principle: find the parameter values that maximize the

likelihood in Eq. (7.2). We use an Expectation-Maximization (EM) algorithm that can iteratively infer the model parameters:

$$\text{E-step: } \hat{e}_{ij}^z = e_{ij} \frac{\rho_z \theta_i^z \theta_j^z}{\sum_{t=1}^k \rho_t \theta_i^t \theta_j^t} \quad (5.5)$$

M-step:

$$\rho_z = \sum_{i,j} \hat{e}_{ij}^z \quad (5.6)$$

$$\theta_i^z = \frac{\sum_j \hat{e}_{ij}^z}{\rho_z} \quad (5.7)$$

Intuitively, the E-step calculates the expected number of links \hat{e}_{ij}^z in each topic z between the terms w_i and w_j : the ratio of \hat{e}_{ij}^z to e_{ij} is proportional to its Poisson parameter $\rho_z \theta_i^z \theta_j^z$. The M-step calculates the ML parameter estimates: θ_i^z is the ratio of the total number of links in topic z where one end node is w_i and ρ_z , which is the sum of the total expected number of links in topic z .

We update $\hat{e}_{ij}^z, \theta_i^z, \rho_z$ in each iteration. Note that if $e_{ij} \notin E$, we do not need to calculate \hat{e}_{ij}^z because it equals 0. Therefore, the time complexity for each iteration is $O((|E| + |V|)k) = O(|E|k)$. Like other EM algorithms, the solution converges to a local maximum and the result may vary with different initializations. The EM algorithm may be run multiple times with random initializations to find the solution with the best likelihood. We empirically find that the EM algorithm generally requires hundreds of iterations to converge, although we can improve the efficiency with some acceleration tricks. For example, we do not need to update a parameter in each iteration if it converges before the whole model converges. Similar tricks are used in other generative models such as [2], and we omit the details here.

It is important to note that our method naturally supports top-down hierarchical clustering. To further discover subtopics of a topic, we can extract the subnetwork where $E^z = \{\hat{e}_{ij}^z | \hat{e}_{ij}^z \geq 1\}$ (expected number of links attributed to that topic, ignoring values less than 1) and then apply the same generative model on the subnetwork. This process can be recursively repeated until the desired hierarchy is constructed.

Topical frequency estimation

Using the learned model parameters, we can estimate the topical frequency for a phrase $P = \{w_{x_1} \dots w_{x_n}\}$:

$$f_z(P) = f_{par(z)}(P) \frac{\rho_z \prod_{i=1}^n \theta_{x_i}^z}{\sum_{t \in C^{par(z)}} \rho_t \prod_{i=1}^n \theta_{x_i}^t} \quad (5.8)$$

This estimation is based on two assumptions: i) when generating a topic- z phrase of length n , each of the n terms is generated with the multinomial distribution θ^z , and ii) the total number of topic- z phrases of length n is proportional

to ρ_z . It is easy to see that when $n = 2$, $f_z(\{w_i, w_j\})$ reduces to \hat{e}_{ij}^z .

5.3.2 Topical Phrase Extraction

Since we define phrases to be sets of frequent terms, we develop an algorithm to mine frequent topical patterns. The goal is to extract patterns with topical frequency larger than some threshold *minsup* for every topic z . In contrast to traditional frequent pattern mining problem, the topical frequency of each pattern is unknown and must be estimated. The results from the clustering step in Section 5.3.1 are necessary for our estimation.

To extract topical frequent patterns, one can first mine all frequent patterns with a traditional pattern mining algorithm such as Apriori [1] or FP-growth [27], and then filter them using the topical frequency estimation using Eq. (7.14). The following two properties of topical frequency can be further exploited to speed up this step:

Property 1 *A phrase's topic- z frequency has an upper bound of the topic- z frequency of any of its subphrases.*

Property 2 *A phrase's topic- z frequency has an upper bound $f_{par(z)}(P') \leq \frac{\rho_z \prod_{i=1}^n \theta_{x_i}^z}{\sum_{t \in C^{par(z)}} \rho_t \prod_{i=1}^n \theta_{x_i}^t}$, where $P' \subset P$ is any subphrase of P .*

Note that for only the top level topics $z \in C^o$, the parent topical frequency $f_{par(z)}(P)$ is equal to $f(P)$ and must be counted from the text. However, for all lower levels, the parent topical frequency $f_{par(z)}(P)$ was already calculated when the parent topic was generated, and therefore never needs to be counted.

One problem with the extracted frequent term sets is that every subset of a frequent phrase is also a frequent phrase. However, some of these subphrases should be removed according to the Completeness criterion, previously described in Section 5.2. Like in KERT, for each topic, we remove a phrase P if there exists a frequent phrase P' , such that $P \subset P'$, $f_z(P') \geq \gamma f_z(P)$. The remaining patterns are referred to as γ -maximal patterns ($0 \leq \gamma \leq 1$). When $\gamma = 1$, this is equivalent to a closed pattern, and when $\gamma = 0$, this is a maximal pattern. We empirically set $\gamma \approx 0.5$, which removes a phrase if its topical frequency is no more than twice of some superphrase. In other words, if a phrase co-occurs with a superphrase more than half the time, we consider that it is subsumed by the superphrase, and should be removed. According to Property 1, pruning can be performed by comparing the frequency of a length- n phrase with all of its length- $(n + 1)$ superphrases. The collection of all of the γ -maximal phrases of a topic z forms the candidate phrase set \mathcal{P}^z .

5.3.3 Ranking

As discussed in Section 5.2, topical phrases in \mathcal{P}^z are ranked according to four criteria: coverage, purity, phraseness, and completeness. The last criterion is already employed as a filter for the phrase extraction step, parameterized by γ . So we now combine the remaining three criteria into a ranking function using a probabilistic modeling approach.

The key idea, which remains the same as in the KERT framework, is to consider the *occurrence probability* of ‘seeing a phrase p in a random document with topic t .’ With this definition, the events of seeing n -grams of various lengths in a document are no longer mutually exclusive, and therefore the probabilities no longer need to sum to 1.

We construct estimations for occurrence probability and two *contrastive probabilities* that will be used to compare against the occurrence probability. We use m_z to denote the number of documents that contain at least one frequent topic- z phrase. Similarly, we use m_Z to denote the number of documents that contain at least one frequent topic- z phrase for some topic $z \in Z$. We can then calculate the occurrence probability of a phrase P conditioned on topic z :

$$p(P|z) = \frac{f_z(P)}{m_z} \quad (5.9)$$

The *independent contrastive probability* is the probability of independently seeing every term in phrase

$P = \{w_{x_1}, \dots, w_{x_n}\}$ conditioned on topic z :

$$p_{indep}(P|z) = \prod_{i=1}^n p(w_{x_i}|z) = \prod_{i=1}^n \frac{f_z(w_{x_i})}{m_z} \quad (5.10)$$

and the *mixture contrastive probability* is the probability of a phrase P conditioned on a mixture of multiple sibling topics $Z \subset C^{par(z)}$, $Z \supseteq \{z\}$:

$$p(P|Z) = \frac{\sum_{t \in Z} f_t(P)}{m_Z} \quad (5.11)$$

We can now define the three remaining ranking criteria: coverage, purity, and phraseness. The coverage of a phrase is directly quantified by $p(P|z)$. The phraseness can be measured by the log ratio of the occurrence probability to the independent contrastive probability $\log \frac{p(P|z)}{p_{indep}(P|z)}$. The purity can be measured by the log ratio of the occurrence probability and the mixture contrastive probability $\log \frac{p(P|z)}{p(P|Z)}$. The definition of purity is configurable by altering the makeup of the topic mixture Z . For example, using the mixture of all the sibling topics $C^{par(z)}$ as the topic mixture results in a weaker purity criterion. However, deliberately choosing the subset Z so that the contrastive probability $p(P|Z)$ is maximized, results in a stronger purity criterion.

The three criteria are unified by the ranking function:

$$r^z(P) = p(P|z) \left(\log \frac{p(P|z)}{p(P|Z)} + \omega \log \frac{p(P|z)}{p_{indep}(P|z)} \right) \quad (5.12)$$

where ω controls the importance of the phraseness criterion. This formulation of the ranking function has several desirable characteristics:

- The coverage measure $p(P|z)$ is the most influential, since the other two criteria are represented by log ratios of $p(P|z)$ and a contrastive probability, and the effect of contrastive probability on the ranking score is smaller than the

influence of $p(P|z)$. This is a desirable property because when a phrase P has low support, the estimates of purity and phraseness are unreliable; but their effect is small since the value of $p(P|z)$ would be correspondingly low. Therefore, a phrase with low coverage would inevitably be ranked low, as should be the case for representative phrases.

- The relative importance of the purity and phraseness measures is controlled by ω . Both measures are log ratios on comparable scales, and can thus be balanced by weighted summation. As ω increases, we expect more topic-independent but common phrases to be ranked higher. We therefore restrict $\omega \in [0, 1]$ because our task requires topic-related phrases to be highly ranked.

- The ranking function can also be nicely represented as a pointwise Kullback-Leibler (KL) divergence in an information theoretic framework. Pointwise KL divergence is a distance measure between two probabilities. It is more robust than pointwise mutual information because the former also considers absolute probability. In pointwise KL divergence, the relative difference between probabilities must be supported by a sufficiently high absolute probability. The product $p(P|z) \log \frac{p(P|z)}{p(P|Z)}$ is equivalent to the pointwise KL divergence between the probabilities of $p(P|z)$ and $p(P|Z)$. Likewise, $p(P|z) \log \frac{p(P|z)}{p_{indep}(P|z)}$ is equivalent to the pointwise KL divergence between the probabilities of $p(P|z)$ under different independence assumptions. Therefore, Eq. (7.15) can also be interpreted as a weighted summation of two pointwise KL divergence metrics.

5.4 Experiments

In this section we first introduce the datasets and methods we used for comparison. We then describe our 3-part evaluation: i) we conduct a user study with ‘intruder detection’ tasks to evaluate hierarchy quality; ii) we use category-labeled data to evaluate the mutual information between phrase-represented topics and known topical divisions; and iii) we present several case studies.

5.4.1 Datasets

We analyze our performance on two datasets:²

DBLP. We collected a set of titles of recently published computer science papers in the areas related to Databases, Data Mining, Information Retrieval, Machine Learning, and Natural Language Processing. These titles come from DBLP, a bibliography website for computer science publications. We minimally pre-processed the dataset by removing all stopwords from the titles, resulting in a collection of 33,313 titles consisting of 18,598 unique terms.

Library. We obtain titles of books from the *University of Illinois Library* catalogue database in 6 general cate-

²The datasets are available at <http://illimine.cs.illinois.edu/cathy>

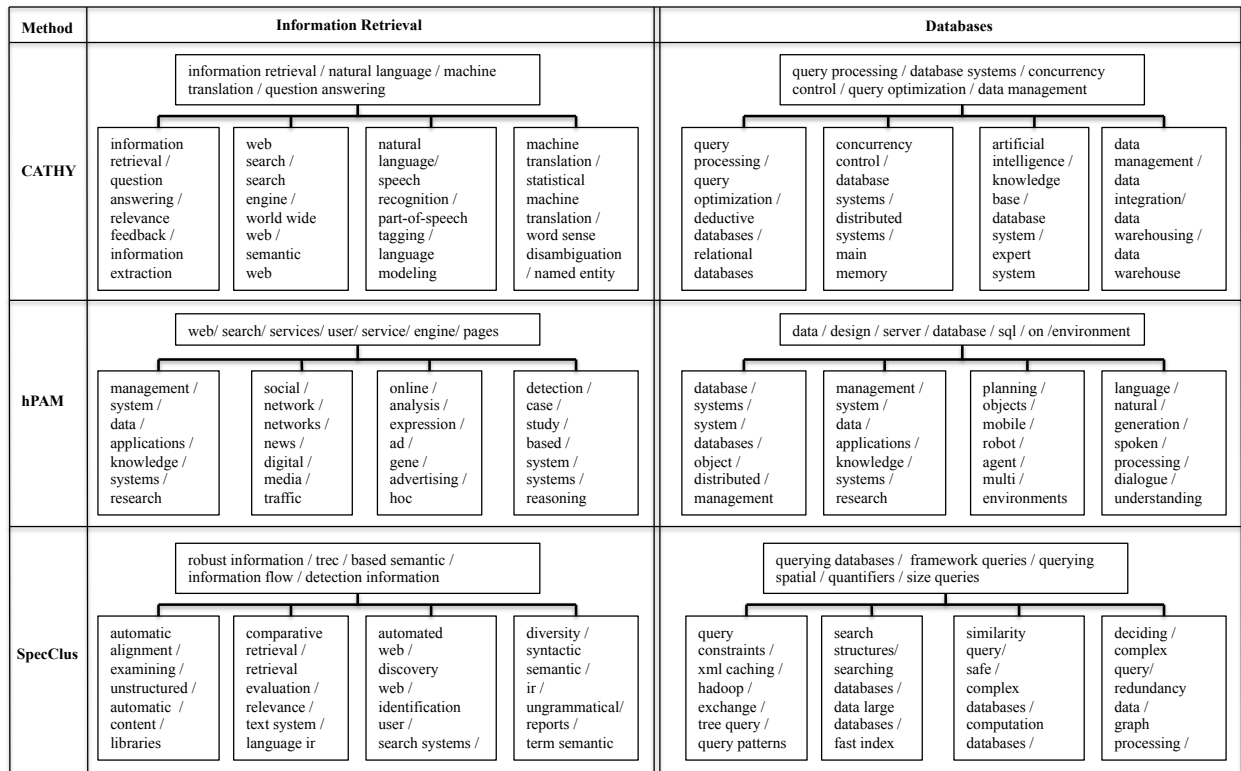


Figure 5.1: Each method generates a hierarchy. For each method, we show the subtrees rooted at Level 2 that are the most likely to represent the topics of Information Retrieval and Databases. The ordering of words in each phrase are determined by the most frequent ordering in the documents, and two phrases only differing in plural/single forms are shown only once.

gories: Architecture, Literature, Mass Media, Motion Pictures, Music, and Theater. We pre-processed the titles by removing all stopwords and terms with frequency less than 5 in the dataset. We also remove titles over 10 words in length, and titles not in English. The resulting dataset contains 33,372 titles consisting of 3,556 unique terms.

5.4.2 Methods for Comparison

As the topical hierarchy construction problem setting that we study is new, there are no directly comparable algorithms. We implement several methods:

SpecClus: As one baseline, we implement a common framework of clustering-based ontology construction, which first extracts all concepts from the text and then hierarchically clusters them. We adapt this to our setting by first mining all frequent phrases using FP-growth [27], a typical pattern mining algorithm. We then implement spectral clustering [53] for the clustering step, where the similarity metric between two phrases is their co-occurrence count in the dataset. This approach uses K-means to perform hard clustering after computing a spectral embedding of the similarity graph. Finally, we rank phrases in each cluster based on their distance from the cluster center. In order to go down in the hierarchy we recursively perform the same clustering and ranking on each cluster of phrases.

hPAM: As a second baseline, we use a state-of-the-art hierarchical topic modeling approach: the hierarchical Pachinko Allocation Model [49]. hPAM takes documents as input and outputs a specified number of supertopics and subtopics, as well as the associations between them. However, it builds a hierarchy for 3 levels simultaneously, not recursively, so we only generate a hierarchy with 3 levels.³

hPAM_{rr}: For each topic, hPAM outputs a multinomial distribution over unigrams. These distributions can be used to calculate the coverage and purity measures in our ranking function (phraseness and completeness do not matter when all candidate phrases are unigrams). We therefore also implement a method that reranks the unigrams in each topic generated by hPAM, with our ranking function adopting the distribution learned by hPAM. We refer to the result as hPAM_{rerank}, or hPAM_{rr}. Note that we cannot rerank SpecClus because it does not generate a probability distribution that can be input into our ranking function.

CATHY_{cp}: In this version of CATHY the ranking function only considers the coverage and purity criteria, and not phraseness or completeness ($\gamma=1, \omega=0$). This allows us to more closely compare the performance of our clustering and mining step with hPAM, using hPAM_{rr}.

CATHY: For evaluation we set $minsup=5, \gamma=\omega=0.5$ for phraseness and completeness criteria, and we use the strong definition of purity, as discussed in Section 5.3.3

³hPAM has several parameters. Our tuning shows that the optimal values of mixture prior between supertopic and subtopic are 1.5 and 1.0 respectively. The mixture prior over topics in the same level is optimal at 1.0 for both levels. The optimal prior for topic distribution over terms is 0.01.

5.4.3 Topical Hierarchy of DBLP Paper Titles

Our first evaluation assesses the ability of our method to construct topical phrases that appear to be high quality to human judges, via a user study. We construct hierarchies with 4 levels from the DBLP dataset. For simplicity we set the number of subtopics for the root node to be 5, for all other non-leaves to be 4, for all of the methods. Since hPAM and hPAM_{IT} only construct 3 levels of the hierarchy, we compare the 3-level hierarchies across all methods, and the full hierarchies for the 3 methods which constructed them.

In the following subsections, we present a sample of the hierarchies actually generated by these methods and encountered by participants in the user study. We then explain the details of our user study, and present quantitative results.

Qualitative Results

Figure 5.1 shows a subset of hierarchies constructed by CATHY and the two baselines, SpecClus and hPAM. In general, CATHY constructs high quality phrases, representing the areas and subareas on both levels. hPAM outputs unigrams that are fair at conveying the top-level topics when considered jointly, but independently are topic-ambiguous (e.g., ‘services’ for IR). hPAM’s second level subtopics are generally more difficult to interpret, and some parent-child relationships are not clearly observed. SpecClus tends to generate phrases with good purity but unsatisfactory coverage and phraseness (e.g., ‘querying spatial’ for DB).

Word and Topic Intrusion User Study

To quantitatively measure topical phrase quality, we invited people to judge the topical phrases generated by the different methods. Since the DBLP dataset generates topics in computer science, we recruited 9 computer science graduate students - who could thus be considered to be very knowledgeable judges - for a user study. We first describe the two tasks administered in the user study, and then discuss the obtained results. The complete set of instructions for both tasks can be found in Appendix C.

In order to evaluate the quality of the generated topical phrases, we adapt two tasks from Chang et al. [8], who were the first to explore human evaluation of topic models. Our first task is Topic Intrusion, which tests the quality of the parent-child relationships in the generated hierarchies. Our second task is Phrase Intrusion, which evaluates how well the hierarchies are able to separate phrases in different topics. Both tasks are depicted in Figure 5.2.

Topic Intrusion Task: Participants are shown a parent topic t and T candidate child topics. $T - 1$ of the child topics are actual children of t in the generated hierarchy, and the remaining child topic is not. Each topic is represented by its top 5 ranked phrases. Participants are asked to select the intruder child topic, or to indicate that they are unable

Question 1/80		Topic Intrusion		
Parent topic	Child topic 1	Child topic 2	Child topic 3	Child topic 4
database systems data management query processing management system data system	web search search engine semantic web search results web pages	data management data integration data sources data warehousing data applications	query processing query optimization query databases relational databases query data	database system database design expert system management system design system

Phrase Intrusion				
Question 1/130	data mining	association rules	logic programs	data streams
Question 2/130	natural language	query optimization	data management	database systems

Figure 5.2: Examples of user study questions. In the Topic Intrusion task (left), participants are asked to select which child topic does not belong to the given parent topic (Child topic 1). In the Phrase Intrusion task (right), participants are asked to select which phrase does not belong with the others (Question 1: ‘logic programs’; Question 2: ‘natural language’)

to make a choice.

Phrase Intrusion Task: Participants are shown T phrases. $T - 1$ of the phrases come from the same topic and the remaining phrase is from a sibling topic. Each phrase is a top-5 ranked phrase in the topic which it represents. Participants are asked to select the intruder phrase, or to indicate that they are unable to make a choice.

For the user study we set $T = 4$, and asked participants 80 Topic Intrusion questions and 130 Phrase Intrusion questions. Questions are generated from the hierarchies constructed by each of the methods. We sample questions from each hierarchy in a uniform way, drawing equally from all topics in each level.

We then calculate the agreement of the user choices with the actual hierarchical structure constructed by the various methods. We consider a higher match between a given hierarchy and user judgment to imply a higher quality hierarchy. For each method, we report the average percent of questions answered ‘correctly’ (matching the method), as well as the average percent of questions that users were able to answer.

Since the hPAM and hPAM_{tr} hierarchies had one fewer level than other methods, we present two analyses. Table 5.2 presents the results from the full set of questions, except the questions generated by hPAM and hPAM_{tr} (4 Levels), and the results from only those questions taken from the shared levels of every method’s hierarchies (3 Levels).

For the Topic Intrusion task, CATHY outperforms all non-CATHY methods by a large margin. CATHY does slightly better than CATHY_{cp} in the 3 level hierarchy, and is significantly better in the 4 level hierarchy, suggesting that participants found the phraseness and completeness criteria to be helpful. SpecClus slightly outperforms hPAM, because hPAM generates broad unigrams with good coverage, which makes the parent-child relationships difficult

Table 5.2: User study results, for 3 level and 4 level hierarchies. Higher values indicate a higher quality constructed hierarchy

	Topic Intrusion		Phrase Intrusion	
3 Levels	Correct	Answered	Correct	Answered
hPAM	34.4%	75.6%	38.8%	78.9%
hPAM_{rr}	32.2%	72.2%	47.8%	77.2%
SpecClus	38.9%	65.6%	36.1%	77.2%
CATHY_{cp}	78.9%	97.8%	57.8%	90.0%
CATHY	82.2%	98.8%	57.2%	88.9%
4 Levels	Correct	Answered	Correct	Answered
SpecClus	34.4%	68.3%	32.9%	77.4%
CATHY_{cp}	61.7%	96.7%	56.7%	88.5%
CATHY	78.3%	97.8%	54.1%	89.3%

to identify, while SpecClus yields phrases with better purity which, when considered jointly, represent a topic more successfully. Participants answered more questions generated by the hPAM variations than by SpecClus, which, combined with the resulting accuracies, suggests that hPAM generated the least well-separated hierarchy (even with reranking).

CATHY and CATHY_{cp} outperform other methods comparably in the Phrase Intrusion task. As hPAM favors high coverage phrases which are often topic-ambiguous, it posted a low performance. hPAM_{rr} considers phrase purity as well as topical coverage, and therefore performs much better on this task. SpecClus favors purity, and thus is more likely to generate seemingly unrelated high ranked phrases in the same topic, which is reflected here by its poor performance. Once again, participants were more likely to answer questions generated by the CATHY variations than by any of the other three methods.

5.4.4 Topical Hierarchy of Book Titles

In this section, we work with the Library dataset. Since the book titles are labeled with their subjects, we examine how well a high quality topical phrase can predict its category, and vice versa. For this, we construct a hierarchy and measure the *coverage-conscious mutual information at K* ($CCMI_K$) of the labels with the top level branches. Our evaluation is based on [49] but we modify their definition of mutual information to also depend on coverage because we represent topics with phrases.

As we saw in Section 5.4.3 that CATHY generally performs equal to or better than CATHY_{cp}, and hPAM_{rr} similarly outperforms hPAM, we simply compare the performances of CATHY, hPAM_{rr}, and SpecClus, with 6 topics ($k = 6$). For each method, we do multiple runs for various values of K (the number of top-ranked phrases per topic considered). To calculate $CCMI_K$, we label each of the top K phrases per topic with the topic in which it is ranked highest. We

then check if each title contains any of these top phrases. If so, we update the number of events “seeing a topic t and category c ” for $t = 1 \dots k$, with the averaged count for all those labeled phrases in the title; otherwise we update the number of events “seeing a topic t and category c ” for $t = 1 \dots k$ uniformly, where c is the category label for the title. Finally, we compute coverage-conscious mutual information at K :

$$CCMI_K = \sum_{t,c} p(t,c) \log_2 \frac{p(t,c)}{p(t)p(c)}$$

Figure 5.3 shows $CCMI_K$ for each method, $K \in [1, 100]$. Since $CCMI_K$ considers the coverage of a phrase as well as its mutual information with a category, its value generally grows with K . Both CATHY and SpecClus demonstrate this by slowly improving over time, although CATHY is consistently much better at differentiating the categories as K increases. $hPAM_{rr}$ prefers unigrams with high coverage, and thus hits an asymptote almost immediately because it is unable to improve on the performance of the first few phrases.

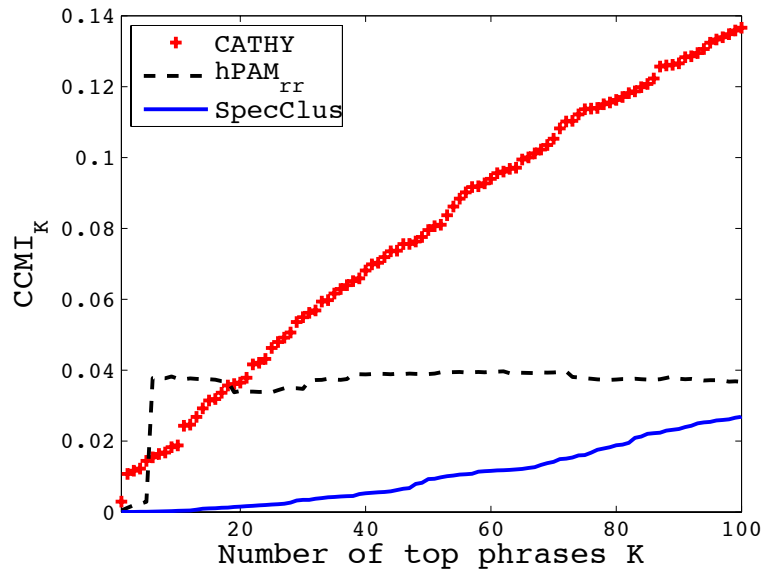


Figure 5.3: $CCMI_K$ values for various methods (methods in legend are ordered by performance, high to low)

5.4.5 On Defining Term Co-occurrence

Term co-occurrence can be defined in many ways: two term may be said to co-occur if they appear in the same sentence, same paragraph, or in a window of N unigrams of each other [47]. Because we worked with collections of short texts, we consistently defined term co-occurrence for our framework to mean co-occurring in the same document. However, most traditional methods of keyphrase extraction only consider phrases to be sequences of terms which explicitly occur in the text. We ran a variation of CATHY which emulates this behavior by defining two terms to

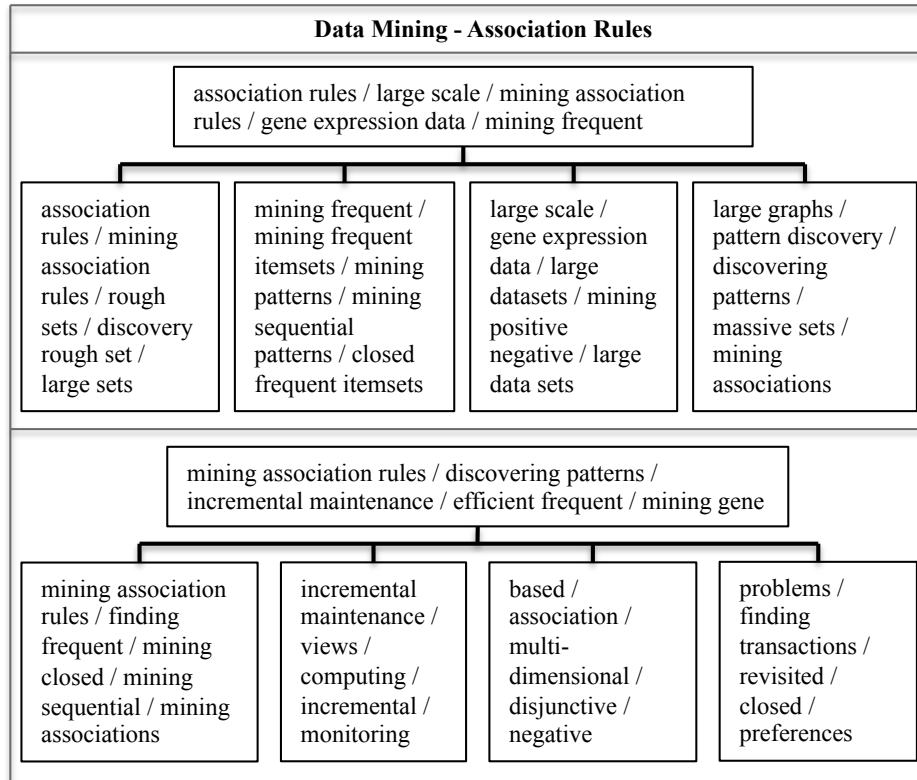


Figure 5.4: A level 3 topic and its level 4 subtopics, from hierarchies constructed by CATHY on two different DBLP term co-occurrence networks: document-based co-occurrence (top) and adjacency-based co-occurrence (bottom). Document-based co-occurrence yields better quality phrases, especially at lower levels

co-occur only if they are actually adjacent in the same title, and constructed a hierarchy on the DBLP dataset.

Using adjacency co-occurrence results in a sparser network and lowers the estimated phrase topical frequencies at every level. As can be seen in Figure 5.4, we observe lower quality phrases in the adjacency-based hierarchy (e.g., the topics which are supposed to be represented by the two rightmost children are very difficult to identify.)

In this work, we address the problem of constructing a topical hierarchy from short, content-representative texts, where topics are represented by ranked lists of phrases. We design a novel phrase mining framework to cluster, extract and rank phrases which recursively discovers specific topics from more general ones, thus constructing a top-down topical hierarchy. A key aspect of our approach involves shifting from a unigram-centric to a phrase-centric view in order to consistently generate high caliber topics over multiple levels. By evaluating our approach on two datasets from different domains, we validate our ability to generate high quality, human-interpretable topic hierarchies.

Chapter 6

Using Hierarchical Topical Phrases for Entity Community Discovery

6.1 Introduction

¹People and social communities are often characterized by the topics and themes they are working on, or communicating about. The roles played by different entities in these communities are of great interest in many contexts of social network analysis. We may be interested in discovering the role of an author in a research community, or the contribution of a user to a social network community organized around similar interests. These types of role discovery tasks center around topical communities mined from social or information networks.

We are also often interested in analyzing such roles at different levels of granularity. In the real world, topical communities - communities built around shared topics - are naturally hierarchical. People participate in large communities, encompassing many interests, as well as small, focused subcommunities. Therefore, in order to analyze the various roles that an entity plays in such different contexts, we must also be able to work with topical communities and subcommunities.

In this chapter we use the CATHY framework to study a new problem of mining entity roles in hierarchical topical communities.² We first use CATHY to detect topical communities from the text component of a social or information network. We can then discover the roles of authors who publish in these communities. The hierarchical structure of the topical communities allows us to distinguish between, e.g., authors who publish on a diverse range of database topics, and authors who are particular experts in query processing. We illustrate our role mining techniques with multiple examples on a real world dataset.

6.2 Role Discovery

We build directly on the results from the previous chapter, working with the DBLP dataset. In this section we illustrate two types of role discovery that can be performed using the topical community hierarchy constructed from this dataset.

¹This chapter contains materials from the following previously published work: Marina Danilevsky, Chi Wang, Nihit Desai, Jiawei Han. Entity Role Discovery in Hierarchical Topical Communities. Proc. of 2013 ACM SIGKDD Int. Workshop on Mining Data Semantics in Heterogeneous Information Networks (MDS'13), Chicago, IL, Aug. 2013. Copyright 2013 ACM, Inc. Reprinted with permission.

²A *topical community* was previously simply called a topic - the new nomenclature reflects only a focus on the role mining application presented in this chapter.

First, given a topical community and an entity type, which entities play the most important roles in the community? For example, an author’s contribution to the topics of a community (by way of published papers) represents the author’s role in that community. Second, for a given topical community and specific entity, what is that entity’s role in the community? For instance, which topics within the community get published in a particular conference? Or, which specific topics within the community does an author contribute to? The topical community hierarchy allows for more nuanced role discovery for a given entity, presenting detailed information at different levels of granularity.

6.2.1 Ranking Community Entities

The role of an entity in a topical community can be interpreted as that entity’s contribution to the community. For example, the role of an author is represented by the work the author does on the community’s topics; the role of a venue is represented by the topics in the community which get published in the venue. Therefore, a natural question to ask is which entities play the roles of top contributors to a particular topical community.

If we consider the role of an entity E in a community z to be that of a contributor of documents (e.g., the role of an author is defined by how many papers he has published on the community’s topics), we can represent the entity’s contribution by estimating the number of documents connected to E which belong to z .

Denote the estimated number of documents in a community z as $|D_z|$, and the set of all documents connected to E as D_E . For example, in the DBLP dataset, the subset D_A is the set of papers authored by A , and D_V is the set of papers published in V . Then, we need to estimate $|D_{E,z}|$, the number of documents attributed to E in community z .

We must first estimate the community frequency of every document $d_E \in D_E$. We described in the CATHY framework how to estimate $f_z(P)$, the community frequency of phrase P . We proceed in a similar top-down recursive fashion in order to estimate the *document* community frequency, $DF_z(d_E)$.

For each document d_E we first perform the intermediate step of calculating the *total phrase frequency* of d_E in community z by adding up the normalized community- z frequencies of all the phrases in d_E :

$$TPF_z(d_E) = \sum_{P \in d_E} \frac{f_z(P)}{\sum_{c \in Ch^{Parent(z)}} f_c(P)} \quad (6.1)$$

The next step is to calculate the normalized *document frequency* of d_E in community z :

$$DF_z(d_E) = \frac{TPF_z(d_E)}{\sum_{c \in Ch^{Parent(z)}} TPF_c(d_E)} DF_{Parent(z)}(d_E) \quad (6.2)$$

The community frequency of a document is distributed among that community’s children, so that the document frequency in a given community is the sum of the document frequencies in the community’s children, $\sum_{c \in Ch^z} DF_c(d) = DF_z(d)$. One exception is that a few documents may contain no frequent topical phrases in any subcommunities

because we filter out infrequent topical phrases. For such documents we do not count their contribution to any subcommunities.

Figure 6.1 shows a hypothetical distribution of document frequency for some document. The document frequency values for every set of subcommunities sum up to the document frequency in the parent community (where the frequency at the root is necessarily 1 for any document).

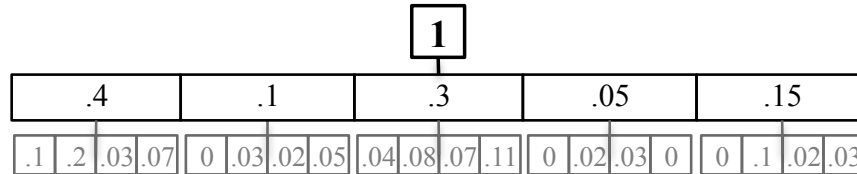


Figure 6.1: A hypothetical distribution of document frequency values for a document, in a hierarchy with 3 levels, beginning at the root.

Finally, we calculate the entity community frequency $EF_z(E)$ by summing up the contributions of all the documents $d_E \in D_E$ to community z :

$$EF_z(E) = \sum_{d_E \in D_E} DF_z(d_E) \quad (6.3)$$

Since some documents may not contribute to any of the subcommunities, the entity frequency in a given community should be equal to or larger than the sum of the entity frequencies in the community's children, $\sum_{c \in Ch^z} EF_c(E) \leq EF_z(E)$. It is clear now that $EF_z(E)$ is precisely our estimate for $|D_{E,z}|$.

Normalizing Phrase Community Frequency

Eq. 6.1 normalizes a phrase's contribution to a document in a given community. Why do we not use the unnormalized $f_z(P)$ which would ensure that a phrase that is more frequent in z influences the document more?

We argue that normalization is better. We would like to fit the document community frequency with the phrase community frequency, i.e., $f_c(P) \approx \sum_{P \in d} DF_c(d)$. Consider a phrase P in document d . The total contribution of $DF_c(d)$ to $f_c(P)$ for all children c of one community z is $\sum_c DF_c(d) = DF_z(d)$. If $DF_c(d) = \frac{f_c(P)}{\sum_c f_c(P)}$ (i.e., the normalized community frequency) holds for all $d \ni P$, then we have exactly $f_c(P) = \sum_{P \in d} DF_c(d)$. However, this is impossible when there are multiple phrases in a document and they have different normalized community frequency. Instead, we can try to minimize the square error $\sum_{P \in d} \sum_c [\frac{f_c(P)}{\sum_c f_c(P)} - DF_c(d)]^2$ with the constraint $\sum_c DF_c(d) = DF_z(d)$. Solving this constrained optimization problem yields the solution in Equation 6.2.

We also evaluated the accuracy of using normalized and unnormalized phrase community frequency. We labeled each document in our collection with the community in which it was most frequent, according to both estimates. We

found that nearly $\frac{1}{3}$ of the documents ended up with different community labels. We sampled a random 1% of these papers, and manually labeled them. We found that the labeling accuracy dropped by 20% from normalizing the phrase contribution to not normalizing. Therefore, normalizing phrase contribution actually does perform better.

Variations in Entity Ranking

Let $|D_z|$ denote the estimated number of documents in a community z . Let D_E denote the set of all documents connected to E where $DF_{Parent(z)}(d_E) \neq 0$, $d_E \in D_E$. Then, $|D_{E,z}|$ denotes the estimated number of documents attributed to entity E in community z (and is precisely equal to $EF_z(E)$, the entity community frequency of E as defined in Equation 6.3). Ranking entities by the value of $|D_{E,z}|$ means only taking into account how much of the topic an entity covers. This ranking would find, for example, the authors who have published the most number of papers on the topics of a particular community. We refer to ranking entities by $EF_z(E)$ as $ERank_{Cov}$.

However, this entity ranking function is not able to discover authors who are more dedicated to their role in a given community than to sibling communities. In order to take this into account, we adapt the notion of purity to apply to entities.

We can calculate the occurrence probability of entity E in community z :

$$p(E|z) = \frac{|D_{E,z}|}{|D_z|} \quad (6.4)$$

and the contrastive probability of seeing E conditioned on a mixture of multiple communities $Z \subset Ch^{Parent(z)}$, $Z \supseteq \{z\}$ (which is analogous to Eq. 5.11):

$$p(E|Z) = \frac{\sum_{c \in Z} |D_{E,c}|}{\sum_{c \in Z} |D_c|} \quad (6.5)$$

We again choose the subset of Z to maximize this probability and strengthen the purity criterion.

We evaluate the purity of entity E in z by comparing the probability of seeing a document E conditioned on community z and the contrastive probability defined by Eq. 6.5.

The criteria of entity purity and coverage can then be unified in an analogous way to Eq. 7.15, with the exception that the notion of phraseness is not applicable to entities. We refer to ranking entities by this value as $ERank_{Cov+Pur}$:

$$ERank_{Cov+Pur}(E, z) = p(E|z) \log \frac{p(E|z)}{p(E|Z)}$$

Table 6.1 shows the top ranked authors in the four subcommunities of Data Mining, based on $ERank_{Cov}$ and $ERank_{Cov+Pur}$. When only coverage is used for ranking, many authors are highly ranked in all communities (e.g. Philip Yu, Jiawei Han, and Christos Faloutsos are top-5 authors in every community). When both coverage and purity

Table 6.1: Top ranked authors in the four subcommunities of Data Mining, based on $ERank_{Cov}$ and $ERank_{Cov+Pur}$.

(a) $ERank_{Cov}$			
{sensor networks, selectivity estimation, large databases, pattern matching, spatio-temporal moving objects, large collections}	{time series, nearest neighbor, moving objects, time series data, nearest neighbor queries}	{association rules, large scale, mining association rules, privacy preserving, frequent itemsets}	{high dimensional, data streams, data mining, high dimensional data, outlier detection}
divesh srivasta	eamonn j. keogh	jiawei han	philip s. yu
nick koudas	philip s. yu	philip s. yu	jiawei han
jiawei han	christos faloutsos	jian pei	charu c. aggarwal
philip s. yu	hans-peter kriegel	christos faloutsos	jian pei
christos faloutsos	jiawei han	ke wang	christos faloutsos
(b) $ERank_{Cov+Pur}$			
{sensor networks, selectivity estimation, large databases, pattern matching, spatio-temporal moving objects, large collections}	{time series, nearest neighbor, moving objects, time series data, nearest neighbor queries}	{association rules, large scale, mining association rules, privacy preserving, frequent itemsets}	{high dimensional, data streams, data mining, high dimensional data, outlier detection}
divesh srivasta	eamonn j. keogh	jiawei han	charu c. aggarwal
surat chaudhuri	jessica lin	ke wang	graham cormode
nick koudas	michail vlachos	xifeng yan	s. muthukrishnan
jeffrey f. naughton	michael j. passani	bing liu	philip s. yu
yannis papakonstantinou	matthias renz	mohammed j. zaki	xiaolei li

criteria are taken into account, only those authors who are significantly more dedicated to one community are highly ranked, resulting in no overlap between communities. Some prolific authors, such as Christos Faloutsos, are no longer highly ranked anywhere, because their contributions are fairly equal among the communities. We are able to easily discover both of these roles.

Table 6.2 shows further examples of ranking authors (using $ERank_{Cov+Pur}$) within two subcommunities of the Database community. By showing the top ranked phrases for each author in a community (we discuss how these are generated in the following section) we are able to see both which authors play the most important roles, and what part of the community each author contributes to.

6.2.2 Entity Role in Community

The second type of role discovery we illustrate is finding out a specific entity's role in a given topical community. In order to represent an entity's role in a community, we want to highlight the subset of the communities which illustrates

Table 6.2: The top ranked authors (using $ERank_{Cov+Pur}$) in two subcommunities of the Database community, along with each author’s top ranked phrases in each community. Each subcommunity is represented by its top-ranked phrases, shown in the first row of each table.

(a) A Database subcommunity

	<i>{query processing / query optimization / deductive databases / materialized views / microsoft sql server / relational databases}</i>
elke a. rundensteiner	query processing / query optimization / materialized views / stream processing / object-oriented databases
hamid pirahesh	query processing / query optimization / materialized views / relational data / relational xml
surajit chaudhuri	query optimization / relational databases / microsoft sql server / materialized views / relational data
jeffrey f. naughton	materialized views / xml query / query processing / relational xml / maintenance view
per-åke larson	materialized views / microsoft sql server / query optimization / materialized maintenance views / relational data
vivek r. narasayya	microsoft sql server / materialized views / relational databases / query management / sql data
serge abiteboul	materialized views / xml data / schemas / query evaluation / materialized maintenance views

(b) A Database subcommunity

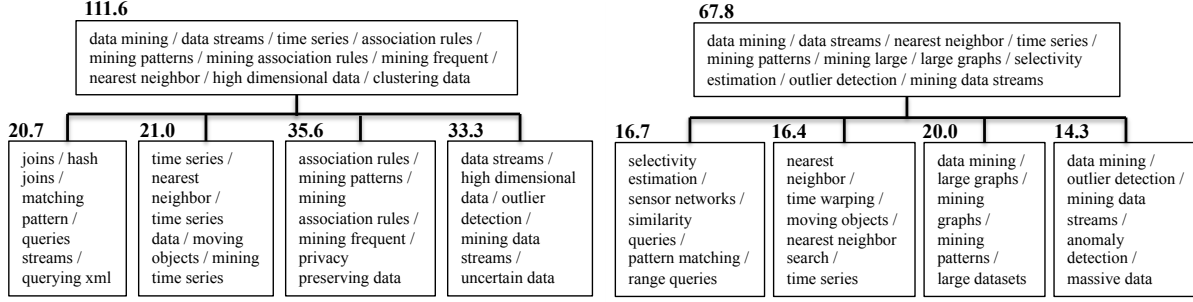
	<i>{concurrency control / database systems / main memory / load shedding / database concurrency control / load balancing}</i>
avi silberschatz	concurrency control / main memory / locking / database systems / transaction management
david b. lomet	recovery / systems recovery / b-trees / transactions recovery / performance access
henry k. korth	concurrency control / database systems / main memory / protocol / transaction systems
bharat k. bhargava	concurrency control / distributed systems / distributed database / recovery / distributed database systems
c. mohan	concurrency control / recovery / locking / data systems / transaction systems
ahmed k. elmagarmid	database systems / concurrency control / distributed database / distributed systems / access control
nancy lynch	concurrency control / locking / nested transactions / control transactions / concurrency transactions

the contribution of the entity. We now therefore introduce a phrase *entity community contribution* ranking function:

$$Cont(P|z, E) = -p(P|z) \log\left(\frac{p(P|z)}{p(P|z, E)}\right) \quad (6.6)$$

where $p(P|z) = \frac{f_z(P)}{|D_z|}$ and $p(P|z, E) = \frac{f_{z, D_E}(P)}{|D_{E,z}|}$.

$f_{z, D_E}(p)$ represents the frequency of phrase P in community z for the document subset D_E . We estimate it as $\sum_{d \in D_E, d \ni P} DF_z(d)$. $Cont(P|z, E)$ has a nice information theoretic interpretation as the pointwise Kullback-Leibler (KL) divergence between the likelihood of seeing phrase P in the documents in community z , and the likelihood of



(a) The roles of Philip S. Yu in Data Mining

(b) The roles of Christos Faloutsos in Data Mining

Figure 6.2: Contrasting the roles of two authors, Philip S. Yu and Christos Faloutsos, in the Data Mining community and subcommunities. The estimate for the number of papers the author contributes to each community is also shown.

seeing phrase P specifically in the documents linked to entity E , in z . Pointwise KL divergence is a distance measure between two probabilities. Therefore, $Cont(P|z, E)$ upranks P if its frequency in the community in conjunction with the entity E is higher than would be expected, based on its overall topical community frequency.

However, using only the Contribution ranking does not give ideal results. Table 6.3 shows the roles of two authors, Philip S. Yu and Christos Faloutsos, in one of the subcommunities of Data Mining subtopics. Using only the contribution ranking function defined in Eq. 6.6 results in poor quality phrases such as ‘fast large.’ On the other hand, using only the phrase quality ranking function defined in Eq. 7.15 - which we refer to here as $Qual(P|z)$ - is also insufficient, as it only evaluates the quality of a phrase, regardless of any entity information. Therefore, we define a *Combined* ranking function for a phrase P which incorporates both the relationship between the entity E and the phrase, as well as phrase quality:

$$Comb(P|z, E) = \alpha Cont(P|z, E) + (1 - \alpha) Qual(P|z) \quad (6.7)$$

The value of $\alpha \in [0, 1]$ can vary. In our experiments, we empirically set $\alpha = 0.5$. Table 6.3 illustrates that the Combined ranking function yields a better list of phrases to represent the roles of the authors.

We can therefore use the Combined ranking function to discover the role of an entity in different topical communities in the hierarchy. As an example, Figure 6.2 shows the roles of Christos Faloutsos and Philip S. Yu in the Data Mining community, and its subcommunities. We also show the entity frequency for each community ($EF_z(E)$), which represents the estimate for the number of papers written by that author in the community.³ The sum of the entity frequencies in the subcommunities do not quite add up to the entity frequency of the parent community because, as discussed in Section 6.2.1, a document does not contribute to the child subcommunities if all of its phrases have become too infrequent in them.

While both authors are prominent in the Data Mining community, Figure 6.2 illustrates how their roles are con-

³It so happens that our dataset contains more papers written by Philip Yu than by Christos Faloutsos, and so the entity topical community frequencies are higher for Philip Yu.

Table 6.3: Using phrase quality, phrase entity community contribution, and a combination of both to represent the roles of Philip S. Yu and Christos Faloutsos in a Data Mining subcommunity

Phrase Quality	P. S. Yu (Contribution)	C. Faloutsos (Contribution)	P. S. Yu (Combined)	C. Faloutsos (Combined)
time series	data indexing	time warping	time series	nearest neighbor
nearest neighbor	data similarity	distance	nearest neighbor	time warping
moving objects	distance	fast time	time series data	moving objects
time series data	fast large	time similarity	moving objects	nearest neighbor search
nearest neighbor queries	similarity indexing	fast large	time series mining	time series
mining time series	time series patterns	fast similarity	time series patterns	distance

trusted in that community, and even more strongly in the subcommunities. For instance, in the third (from left) subcommunity, Philip Yu contributes work on the topics of mining frequent patterns and association rules, whereas the contribution of Christos Faloutsos is more geared towards the topics of mining large datasets and large graphs.

As another example of role discovery, Figure 6.3 shows the role of the SIGIR venue in all 5 top level communities, as well as the subcommunities of Machine Learning and Information Retrieval. The role of a venue in a community is represented by those topics within the community that are published in the venue. Thus, we can see that the Machine Learning topics that get published in SIGIR are techniques related to IR tasks such as feature selection methods that may be used for filtering, and approaches to text categorization and classification problems.

By examining the roles of different venues within a single community, we can also gain some insight to the flavor of each venue. As an example, Table 6.4 compares the roles of three venues - SIGIR, WWW, and ECML - in the general IR community. While both SIGIR and WWW are usually characterized as IR venues, we can clearly see that SIGIR plays a more broad role, publishing most of the topics present in the community, whereas WWW focuses only on those topics that are directly related to the web. On the other hand, ECML is considered to be an ML venue, and its contribution to the IR community is the publishing of papers on topics that use machine learning techniques. Note that all three venues share some high-ranked phrases, illustrating how the roles of all three venues overlap in this community. If we were to strictly label venues, and therefore the papers they publish, as belonging exclusively to one or another community, we would not be able to discover these interesting roles.

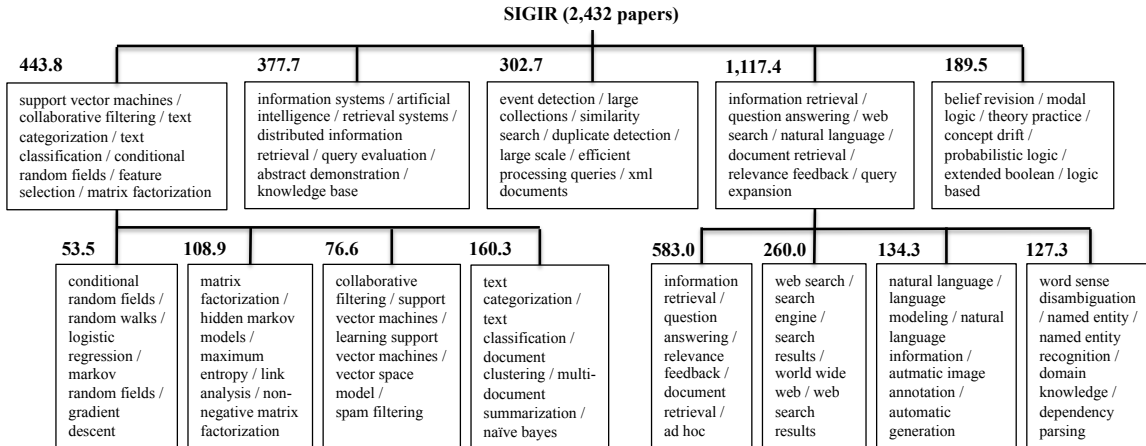


Figure 6.3: The role of the venue SIGIR in several communities and subcommunities. The estimated number of papers published in SIGIR within each community is also shown.

Table 6.4: The roles of three venues - SIGIR, WWW, and ECML - in the general Information Retrieval community

SIGIR	WWW	ECML
information retrieval	web search	word sense disambiguation
question answering	semantic web	world wide web
web search	search engine	information extraction
natural language	question answering	semantic role labeling
document retrieval	web pages	knowledge discovery
relevance feedback	world wide web	query expansion
query expansion	web services	machine translation

Chapter 7

Beyond Text: Constructing Hierarchical Topical Phrases from Heterogeneous Networks

7.1 Introduction

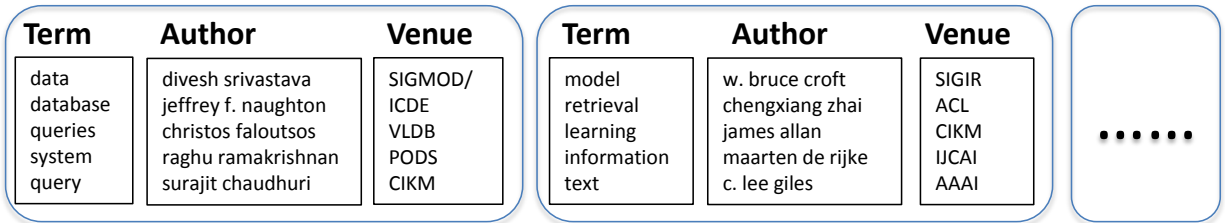
¹ In this final chapter, we extend the work of the previous two chapters into a more robust method of constructing topical hierarchies from heterogeneous information networks. Rather than first constructing a topical hierarchy from the text component, and then using the heuristic method previously presented to mine entity roles, we now describe a combined approach for constructing the hierarchy, which uses entity link information in addition to term co-occurrence.

Few existing approaches for constructing topical hierarchies from text utilize link information from heterogeneous entities that may be present in the data. Conversely, existing methods for heterogeneous network analysis and topic modeling have demonstrated that multiple types of linked entities improve the quality of topic discovery (e.g., NetClus [63]), but these methods are not designed for finding hierarchical structures (See Figure 7.1a for an example output of NetClus). Therefore, there is no existing method that is able to construct a multi-typed topical hierarchy from a heterogeneous network. We recursively construct a topical hierarchy where each topic is represented by ranked lists of phrases and entities of different types. We go beyond the topical hierarchies that are constructed by analyzing textual information alone (e.g., Figure 7.1b), and enrich the topic representation with ranked lists of heterogeneous entities, which provides additional informative context for each topic in the hierarchy (shown in Figure 7.1c). Our approach retains the benefits of NetClus, a recently developed technique for analyzing heterogeneous networks, but is far more robust and well-suited to the task of topical hierarchy construction. Our unified general model is not confined to a particular network schema, and incorporates an inference algorithm that is guaranteed to converge.

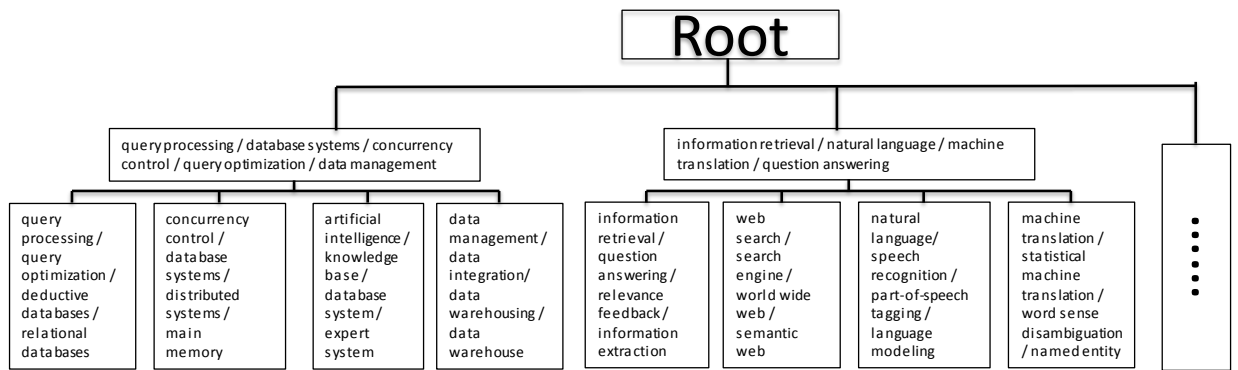
7.1.1 NetClus

As illustrated in Figure 7.2, the input to the NetClus algorithm is a heterogeneous network of star-schema. The example network has one central object type—the star object—and four types of attribute objects (where the type

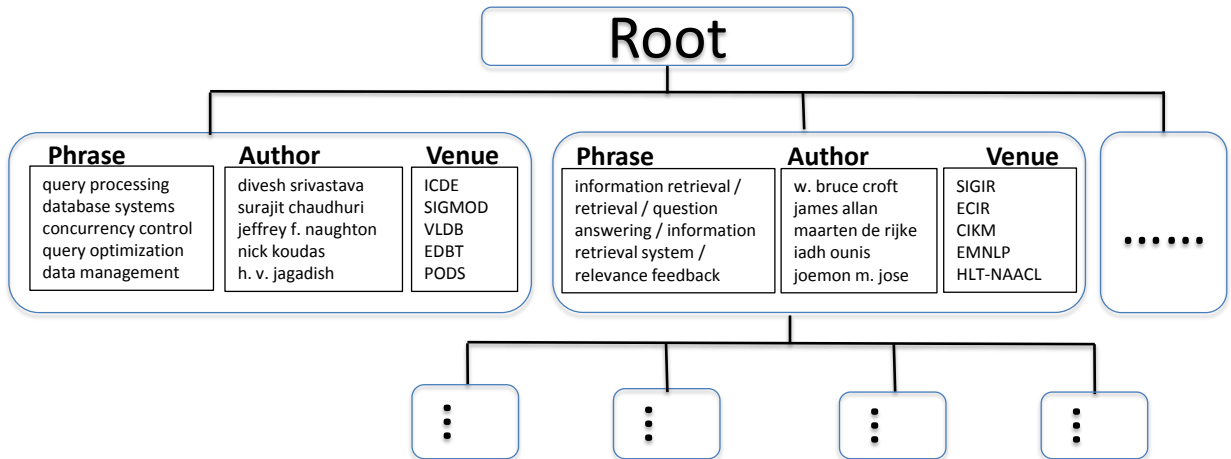
¹This chapter contains materials from the following previously published work: Chi Wang, Marina Danilevsky, Jialu Liu, Nihit Desai, Heng Ji, and Jiawei Han. Constructing Topical Hierarchies in Heterogeneous Information Networks. Proc. 2013 IEEE Int. Conf.on Data Mining (ICDM'13), Dallas, TX, Dec. 2013. Copyright IEEE 2013. Reprinted with permission.



(a) NetClus [63] – clusters of heterogeneous entities. Each rounded rectangle represents one cluster, containing a ranked list of unigrams and two ranked lists of entities



(b) CATHY – topical hierarchy of text only. Each node in the hierarchy contains a ranked list of phrases



(c) CATHYHIN – topical hierarchy of heterogeneous entities. Each node has a ranked list of phrases and two ranked entity pattern lists

Figure 7.1: Sample output from three methods run on a computer science publication network with term, author, and venue attributes

of an object is denoted by its shape and color family). Only links between a star object and an attribute object are allowed. For example, a collection of papers may be transformed into a star schema where each paper is a star object, and attributes such as authors, venues, and terms are attribute objects.

NetClus performs hard clustering on the star objects, and the induced network clusters consist of star objects and their linked attribute objects. Thus, an attribute object may belong to multiple clusters, but each star object is assigned to precisely one cluster. Next, the attribute objects within each subnetwork cluster are ranked via a PageRank-like algorithm, which is based on the structure of the cluster. A generative model then uses the ranking information to infer a cluster distribution for each star object. The cluster memberships of the star objects are then adjusted using a k-means algorithm, and the ranks of attribute objects are re-calculated. Thus, the NetClus algorithm iterates over clustering the star objects based on their inferred membership distribution (as calculated by a generative model based on the existing ranking information), and re-ranking the attribute objects within each newly defined network cluster. The heterogeneous nature of the attribute objects is respected during the ranking step, as only objects of the same type are ranked together, as shown in Figure 7.2.

The iterative clustering and ranking steps of NetClus thus mutually enhance each other. The clustering step provides a context for the ranking calculations, since the ranks of the attribute objects should vary among different clusters (e.g., different areas of computer science). The ranking step in turn improves the quality of found clusters, since highly ranked objects should serve as stronger indicators of cluster membership for their linked star objects.

NetClus can be naturally extended for topical hierarchy construction: after each network is clustered, each of the induced subnetworks are then used as new input, and may thus be recursively clustered and ranked. However, several properties of NetClus render it undesirable for the task of topical hierarchy construction:

1. Topics are represented by ranked lists of terms, and other individual attribute objects. For topics of fine granularity in the hierarchy, this representation may be hard to interpret because single terms and entities may be ambiguous.
2. NetClus assumes a star schema, which hinders its application to more general information networks.
3. NetClus hard clusters star objects, which are usually documents. However, a document is often related to a mixture of topics, especially in the lower levels of a hierarchy. The forced hard clustering can thus result in lost information, as relevant documents fail to appear in relevant subtopics, further decreasing the hierarchy's quality.
4. The iterative algorithm used by NetClus is not guaranteed to converge. The deeper into the hierarchy, the more severe this problem becomes because the output of one level will be input of the next level of the constructed hierarchy.

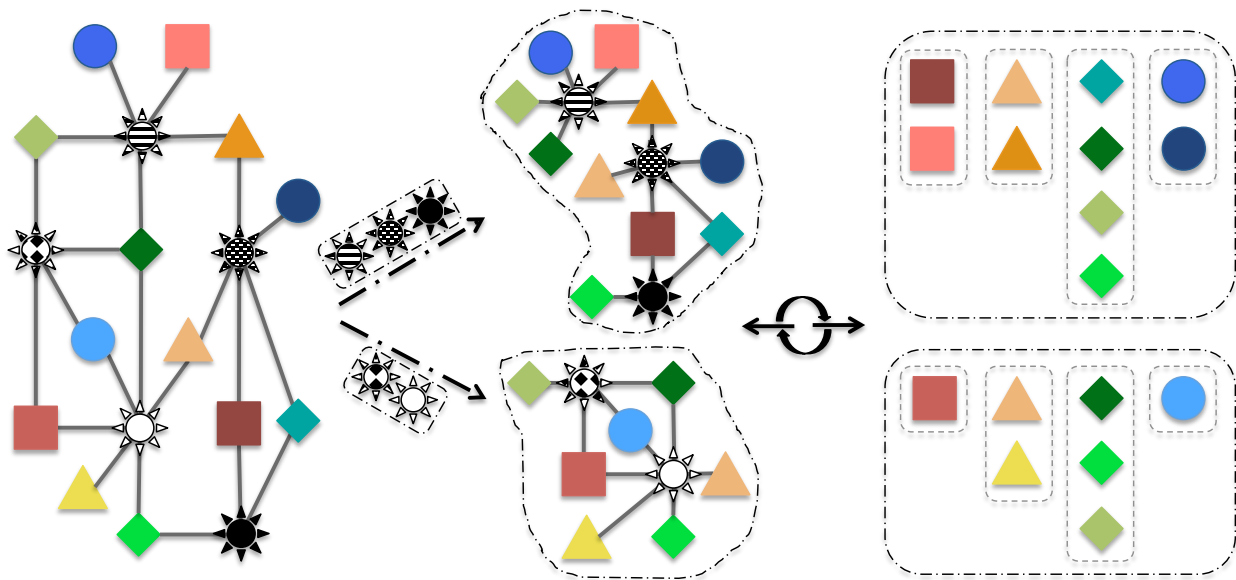


Figure 7.2: An illustration of the NetClus framework. **(L)** NetClus analyses a star schema network, where every link is between a central object and an attribute object of some type (central objects are denoted by stars; attribute objects of the same type are represented by the same shape and color family, with individual objects differentiated by hue). **(M)** The star objects are partitioned into clusters so that each star appears in exactly one cluster. **(R)** Attribute objects (which may appear in multiple clusters) are ranked within each cluster, grouped by type. NetClus iterates over these clustering **(M)** and ranking **(R)** steps, as denoted by the two-way circular arrow symbol

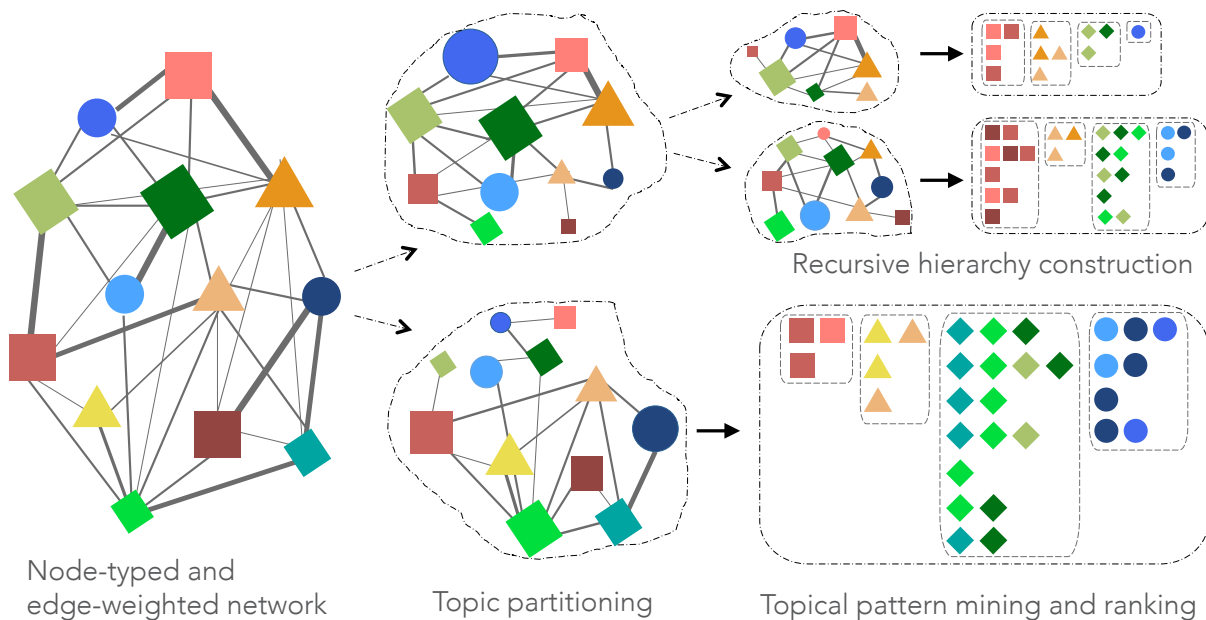


Figure 7.3: An illustration of the CATHYHIN framework. **(L)** Step 1: CATHYHIN analyses a node-typed and edge-weighted network, with no central star objects. **(M)** Step 2: A unified generative model is used to partition the edge weights into clusters and rank single nodes in each cluster (here, node rank within each node type is represented by variations in node size). **(R bottom)** Step 3: Patterns of nodes are ranked within each cluster, grouped by type. **(R top)** Step 4: Each cluster is also an edge-weighted network, and is therefore recursively analyzed. The final output is a hierarchy, where the patterns of nodes of each cluster have a ranking within that cluster, grouped by type.

7.2 CATHYHIN Framework

This section describes our framework CATHYHIN (shown in Figure 7.3), which incorporates the two positive characteristics of NetClus: the utilizing of heterogeneous link types, and the mutually enhancing clustering and ranking steps, while overcoming the disadvantages discussed in Section 7.1.1.

Definition 6 (Heterogeneous Topical Hierarchy) *A heterogeneous topical hierarchy is defined as a tree \mathcal{T} in which each node is a topic. The root topic is denoted as o . Every non-root topic t with parent topic $Par(t)$ is represented by m ranked lists of patterns L_1, \dots, L_m where $L_x = \{P_i^{x,t}\}$ is the sequence of patterns for type x in topic t . The subtopics of every non-leaf topic t in the tree are its children $C^t = \{z \in \mathcal{T}, Par(z) = t\}$. A pattern can appear in multiple topics, though it will have a different ranking score in each topic.*

This definition of the heterogeneous topical hierarchy addresses the first aforementioned criticism of NetClus by representing each topic as multiple lists of ranked patterns, where each list contains *patterns* of objects, rather than individual objects (e.g., phrases rather than unigrams). For instance, the topics in Figure 7.1c each contain 3 lists of patterns.

Our approach does not restrict the network schema, and does not perform hard clustering for any objects. We discover topics by hierarchically soft clustering the links, so that any node may be assigned to multiple topics and subtopics. This removes the restrictions outlined in criticisms 2 and 3 of NetClus. We only require a collection of some kind of information chunks, such as documents, so that each chunk contains multiple objects and we can mine frequent patterns from these chunks.

Formally, every topic node t in the topical hierarchy is associated with an edge-weighted network $G^t = (\{V_x^t\}, \{E_{x,y}^t\})$, where V_x^t is the set of type- x nodes in topic t , and $E_{x,y}^t$ is the set of link weights between type x and type y nodes (x and y may be identical) in topic t . $e_{i,j}^{x,y,t} \in E_{x,y}^t$ represents the weight of the link between node v_i^x of type x and node v_j^y of type y . For every non-root node $t \neq o$, we construct a subnetwork G^t by clustering the network $G^{Par(t)}$ of its parent $Par(t)$. G^t inherits the nodes from $G^{Par(t)}$, but contains only the fraction of the original link weights that belongs to the particular subtopic t . Figure 7.3 visualizes the weight of each link in each network and subnetwork by line thickness (disconnected nodes and links with weight 0 are omitted).

If the original network naturally has a star schema, but the star type is not included in the final topic representation (e.g., the document), we can construct a ‘collapsed’ network by connecting every pair of attribute objects which are linked to the same star object. In the derived network, the link weight $e_{i,j}^{x,y,t}$ between two nodes v_i^x and v_j^y is therefore equal to the number of common neighbors they share in the original star-schema network.

Our framework employs a unified generative model for recursive network clustering and subtopic discovery. The model seamlessly integrates mutually enhanced ranking and clustering while guaranteeing convergence for the infer-

Table 7.1: Notations used in our model

Symbol	Description
G^t	the HIN associated with topic t
V_x^t	the set of nodes of type x in topic t
$E_{x,y}^t$	the set of non-zero link weights of type (x, y) in topic t
$Par(t)$	the parent topic of topic t
C^t	the set of child topics of topic t
z	child topic index of topic t
m	the number of node types
$n_{x,y}$	the total number of links between type- x and type- y nodes
v_i^x	the i -th node of type x
$e_{i,j}^{x,y,z}$	the link weight between v_i^x and v_j^y in topic z
$\phi^{x,z}$	the distribution over type- x nodes in topic z
ϕ^x	the overall distribution over type- x nodes
ρ_z	the total link weight in topic z
θ	the distribution over link type (x, y)
$\alpha_{x,y}$	the importance of link type (x, y)

ence algorithm, thus addressing the final critique of NetClus.

Our framework generates a heterogeneous topical hierarchy in a top-down, recursive way:

Step 1. Construct the edge-weighted network G^o . Set $t = o$.

Step 2. For a topic t , cluster the network G^t into subtopic subnetworks $G^z, z \in C^t$ using a generative model.

Step 3. For each subtopic $z \in C^t$, extract candidate topical patterns within each topic, and rank the patterns using a unified ranking function. Patterns of different lengths are directly compared, yielding an integrated ranking.

Step 4. Recursively apply Steps 2 - 3 to each subtopic $z \in C^t$ to construct the hierarchy in a top-down fashion.

We describe steps 2 and 3 in the following subsections.

7.2.1 Topic Discovery in Heterogeneous Information Networks

Given a topic t and the associated network G^t , we discover subtopics by performing clustering and ranking with the network. We now describe our unified generative model and present an inference algorithm with a convergence guarantee. We further extend our approach to allow different link types to play different degrees of importance in the model (allowing the model to, for example, decide to rely more on term co-occurrence information than on co-author links).

The generative model

We first introduce the basic generative model, which considers all link types to be equally important. For a given topic t , we assume C^t contains k child topics, denoted by $z = 1 \dots k$. The value of k can be either specified by users

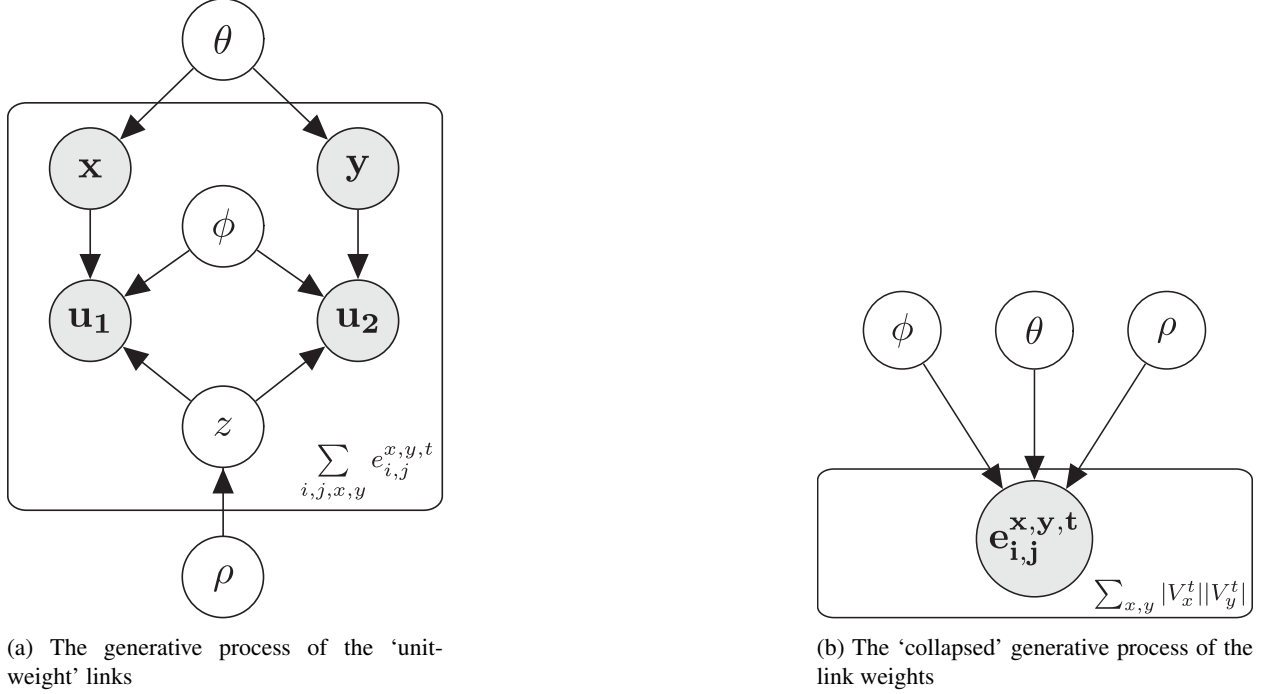


Figure 7.4: Two graphical representation of our generative model for links in a topic t . The models are asymptotically equivalent.

or chosen using a model selection criterion.

In general, the network G^t contains m node types and $\frac{m(m+1)}{2}$ link types.² Similar to NetClus, we assume each node type x has a ranking distribution $\phi^{x,z}$ in each subtopic $z \in C^t$, such that $\phi_i^{x,z}$ is the importance of node v_i^x in topic z , subject to $\sum_i \phi_i^{x,z} = 1$. Each node type x also has a ranking distribution $\phi^{x,0}$ for the background topic, as well as an overall distribution ϕ^x , where ϕ_i^x is proportional to the degree of node v_i^x . In contrast to NetClus, we softly partition the link weights in G^t into subtopics. We model the generation of links so that we can simultaneously infer the partition of link weights (clustering) and the node distribution (ranking) for each topic.

To derive our model, we first assume the links between any two nodes can be decomposed into one or multiple unit-weight links (e.g., a link with weight 2 can be seen as a summation of two unit-weight links). Later we will discuss the case where the link weight is not an integer. Each unit-weight link has a topic label, which is either a subtopic $z \in C^t$, or a dummy label 0, implying the link is generated by a background topic and should not be attributed to any topic in C^t .

The generative process for a topic- z link, $z \in C^t$ (or background topic link, resp.) with unit weight is as follows:

1. Generate the link type (x, y) according to a multinomial distribution θ .
2. Generate the first end node u_1 from the type- x ranking distribution $\phi^{x,z}$ (or $\phi^{x,0}$, resp.).

²We assume the network is undirected, although our model can be easily extended to directed cases.

3. Generate the second end node u_2 from the type- y ranking distribution $\phi^{y,z}$ (or ϕ^y , resp.).

Note that when generating a background topic link, the two nodes i and j are not symmetric, so that we attribute half of it to $i \rightarrow j$ and the other half to $j \rightarrow i$. The first end node is a background node, and can have a background topic link with any other nodes based simply on node degree, irrespective of any topic. Highly ranked nodes in the background topic tend to have a link distribution over all nodes that is similar to their overall degree distribution. See Figure 7.4a for a graphical representation of the model.

With these generative assumptions for each unit-weight link, we can derive the distribution of link weight for any two nodes (v_i^x, v_j^y) . If we repeat the generation of topic- z unit-weight links for ρ_z iterations, then the process of generating a unit-weight topic- z link between v_i^x and v_j^y can be modeled as a Bernoulli trial with success probability $\theta_{x,y} \phi_i^{x,z} \phi_j^{y,z}$. When ρ_z is large, the total number of successes $e_{i,j}^{x,y,z}$ asymptotically follows a Poisson distribution $Pois(\rho_z \theta_{x,y} \phi_i^{x,z} \phi_j^{y,z})$. Similarly, the total number of background topic links $e_{i,j}^{x,y,0}$ asymptotically follows a Poisson distribution $Pois\left(\rho_0 \theta_{x,y} \frac{\phi_i^{x,0} \phi_j^{y,0} + \phi_i^x \phi_j^{y,0}}{2}\right)$.

One important implication due to the *additive* property of Poisson distribution is:

$$e_{i,j}^{x,y,t} = \sum_{z=0}^k e_{i,j}^{x,y,z} \sim Poisson(\theta_{x,y} s_{i,j}^{x,y,t}) \quad (7.1)$$

where $s_{i,j}^{x,y,t} = \sum_{z=1}^k \rho_z \phi_i^{x,z} \phi_j^{y,z} + \rho_0 \frac{\phi_i^{x,0} \phi_j^{y,0} + \phi_i^x \phi_j^{y,0}}{2}$.

This leads to a ‘collapsed’ model as depicted in Figure 7.4b. Though we have so far assumed the link weight to be an integer, this collapsed model remains valid with non-integer link weights (due to Property 3, discussed later).

Given the model parameters, the probability of all observed links is:

$$p(\{e_{i,j}^{x,y,t}\} | \theta, \rho, \phi) = \prod_{v_i^x, v_j^y} \frac{(\theta_{x,y} s_{i,j}^{x,y,t})^{e_{i,j}^{x,y,t}} \exp(-\theta_{x,y} s_{i,j}^{x,y,t})}{e_{i,j}^{x,y,t}!} \quad (7.2)$$

We learn the parameters by the *Maximum Likelihood* (ML) principle: find the parameter values that maximize the likelihood in Eq. (7.2). We use an Expectation-Maximization (EM) algorithm that can iteratively infer the model parameters.

E-step:

$$\hat{e}_{i,j}^{x,y,z} = \frac{e_{i,j}^{x,y,t} \rho_z \phi_i^{x,z} \phi_j^{y,z}}{\sum_{c=1}^k \rho_c \phi_i^{x,c} \phi_j^{y,c} + \frac{\rho_0}{2} (\phi_i^{x,0} \phi_j^{y,0} + \phi_i^x \phi_j^{y,0})} \quad (7.3)$$

$$\hat{e}_{i \rightarrow j}^{x,y,0} = \frac{e_{i,j}^{x,y,t} \rho_0 \phi_i^{x,0} \phi_j^{y,0}}{2 \sum_{c=1}^k \rho_c \phi_i^{x,c} \phi_j^{y,c} + \rho_0 (\phi_i^{x,0} \phi_j^{y,0} + \phi_i^x \phi_j^{y,0})} \quad (7.4)$$

M-step:

$$\rho_z = \sum_{i,j,x,y} \hat{e}_{i,j}^{x,y,z}, \quad \theta_{x,y} = \frac{\sum_{i,j} e_{i,j}^{x,y,t}}{\sum_{i,j,x,y} e_{i,j}^{x,y,t}} \quad (7.5)$$

$$\phi_i^{x,z} = \frac{\sum_{j,y} \hat{e}_{i,j}^{x,y,z}}{\sum_{u,j,y} \hat{e}_{u,j}^{x,y,z}}, \quad \phi_i^{x,0} = \frac{\sum_{j,y} \hat{e}_{i \rightarrow j}^{x,y,0}}{\sum_{u,j,y} \hat{e}_{u \rightarrow j}^{x,y,0}} \quad (7.6)$$

We update \hat{e}, ϕ, ρ in each iteration ($\theta_{x,y}$ is a constant). In the E-step, we perform the clustering by estimating \hat{e} . In the M-step, we estimate the ranking distribution ϕ . Like other EM algorithms, the solution converges to a local maximum and the result may vary with different initializations. The EM algorithm can be run multiple times with random initializations to find the solution with the best likelihood.

The subnetwork for topic z is naturally extracted from the estimated \hat{e} (expected link weight attributed to each topic). For efficiency purposes, we remove links whose weight is less than 1, and then filter out all resulting isolated nodes. We can then recursively apply the same generative model to the constructed subnetworks until the desired hierarchy is constructed.

Learning link type weights

The generative model described above does not differentiate between the importance of different link types. However, we may wish to discover topics that are biased towards certain types of links, and the bias may vary at different levels of the hierarchy. For example, in the computer science domain, the links between venues and other entities may be more important indicators than other link types in the top level of the hierarchy; however, these same links may be less useful for discovering subareas in the lower levels (e.g., authors working in different subareas may publish in the same venue).

We therefore extend our model to capture the importance of different link types. We introduce a *link type weight* $\alpha_{x,y} > 0$ for each link type (x,y) . We use these weights to scale a link's observed weight up or down, so that a unit-weight link of type (x,y) in the original network will have a *scaled* weight $\alpha_{x,y}$. Thus, a link of type (x,y) is valued more when $\alpha_{x,y} > 1$, less when $0 < \alpha_{x,y} < 1$, and becomes negligible as $\alpha_{x,y}$ approaches 0.

When the link type weights $\alpha_{x,y}$ are specified for our model, the EM inference algorithm is unchanged, with the exception that all the $e_{i,j}^{x,y,t}$ should be replaced by $\alpha_{x,y} e_{i,j}^{x,y,t}$. When all $\alpha_{x,y}$'s are equal, the weight-learning model reduces to the basic model. Most of the time, the weights of the link types will not be specified explicitly by users, and must therefore be learned from the data.

We first note an important property of our model, justifying our previous claim that link weights need not be integers.

Property 3 (Scale-invariant) *The EM solution is invariant to a constant scaleup of all the link weights.*

Due to the scale-invariant property of the link weights, we can assume that *w.l.o.g.*, the product of all the non-zero link weights remains invariant before and after scaling:

$$\prod_{e_{i,j}^{x,y,t} > 0} e_{i,j}^{x,y,t} = \prod_{e_{i,j}^{x,y,t} > 0} \alpha_{x,y} e_{i,j}^{x,y,t} \quad (7.7)$$

which reduces to $\prod_{x,y} \alpha_{x,y}^{n_{x,y}} = 1$, where $n_{x,y} = |E_{x,y}^t|$ is the number of non-zero links with type (x, y) . With this constraint, we maximize the likelihood $p(\{e_{i,j}^{x,y,t}\} | \theta, \rho, \phi, \alpha)$:

$$\max \prod_{v_i^x, v_j^y} \frac{(\theta_{x,y} s_{i,j}^{x,y,t})^{\alpha_{x,y} e_{i,j}^{x,y,t}} \exp(-\theta_{x,y} s_{i,j}^{x,y,t})}{(\alpha_{x,y} e_{i,j}^{x,y,t})!} \quad (7.8)$$

$$s.t. \prod_{x,y} \alpha_{x,y}^{n_{x,y}} = 1, \alpha_{x,y} > 0 \quad (7.9)$$

With Stirling's approximation $n! \sim \left(\frac{n}{e}\right)^n \sqrt{2\pi n}$, we transform the log likelihood:

$$\max \sum_{v_i^x, v_j^y} \left(\alpha_{x,y} e_{i,j}^{x,y,t} \log(\theta_{x,y} s_{i,j}^{x,y,t}) - \theta_{x,y} s_{i,j}^{x,y,t} \right) \quad (7.10)$$

$$- \alpha_{x,y} e_{i,j}^{x,y,t} \left[\log(\alpha_{x,y} e_{i,j}^{x,y,t}) - 1 \right] - \frac{1}{2} \log(\alpha_{x,y} e_{i,j}^{x,y,t})$$

$$s.t. \sum_{x,y} n_{x,y} \log \alpha_{x,y} = 0 \quad (7.11)$$

Using the Langrange multiplier method, we can find the optimal value for α when the other parameters are fixed:

$$\alpha_{x,y} = \frac{\left[\prod_{x,y} \left(\frac{1}{n_{x,y}} \sum_{i,j} e_{i,j}^{x,y,t} \log \frac{e_{i,j}^{x,y,t}}{\theta_{x,y} s_{i,j}^{x,y,t}} \right)^{n_{x,y}} \right]^{\frac{1}{\sum_{x,y} n_{x,y}}}}{\frac{1}{n_{x,y}} \sum_{i,j} e_{i,j}^{x,y,t} \log \frac{e_{i,j}^{x,y,t}}{\theta_{x,y} s_{i,j}^{x,y,t}}} \quad (7.12)$$

With some transformation of the denominator:

$$\begin{aligned} \beta_{x,y} &= n_{x,y} \sum_{i,j} e_{i,j}^{x,y,t} \log \frac{e_{i,j}^{x,y,t}}{\theta_{x,y} s_{i,j}^{x,y,t}} \\ &= \frac{\sum_{i,j} e_{i,j}^{x,y,t}}{n_{x,y}} \sum_{i,j} \frac{e_{i,j}^{x,y,t}}{\sum_{i,j} e_{i,j}^{x,y,t}} \log \frac{e_{i,j}^{x,y,t} / \sum_{i,j} e_{i,j}^{x,y,t}}{\theta_{x,y} s_{i,j}^{x,y,t} / \sum_{i,j} e_{i,j}^{x,y,t}} \end{aligned} \quad (7.13)$$

we can see more clearly that the link type weight is negatively correlated with two factors: the average link weight and the KL-divergence of the expected link weight distribution to the observed link weight distribution. The first factor is used to balance the scale of link weights of different types (e.g., a type-1 link always has X times greater weight than a

type-2 link). The second factor measures the importance of a link type in the model. The more the prediction diverges from the observation, the worse the quality of a link type.

So we have the following iterative algorithm for optimizing the joint likelihood:

1. Initialize all the parameters.
2. Fixing α , update ρ, θ, ϕ using EM equations (7.3)-(7.6), with all the $e_{i,j}^{x,y,t}$ replaced by $\alpha_{x,y} e_{i,j}^{x,y,t}$.
3. Fixing ρ, θ, ϕ , update α using Eq. (7.12).
4. Repeat steps 2) and 3) until the likelihood converges.

In each iteration, the time complexity is $O(\sum_{x,y} n_{x,y})$, i.e., linear to the total number of non-zero links. The likelihood is guaranteed to converge to a local optimum. Once again, a random initialization strategy can be employed to choose a solution with the best local optimum.

7.2.2 Topical Pattern Mining and Ranking

Having discovered the topics using our generative model, we can now identify the most representative topical patterns for each topic. This is done in two stages: topical pattern mining and ranking the mined patterns. These stages are nearly identical to those described in the original CATHY framework, but we present them here for completeness.

Pattern mining in each topic

A pattern P^x of type x is a set of type- x nodes: $P^x = \{v_i^x\}$. For example, a pattern of a ‘term’ type is a set of unigrams that make up a phrase, such as $\{support, vector, machine\}$ (or ‘support vector machine’ for simpler notation). A more general definition of a pattern can involve mixed node types within one pattern, but is beyond the scope of this paper.

A pattern P that is regarded to be representative for a topic t must first and foremost be frequent in the topic. The frequency of a pattern $f(P)$ is the number of documents (or other meaningful information chunks) that contain all the nodes in the pattern (or the number of star objects that are linked to all the nodes). The pattern must also have sufficiently high *topical frequency* in topic t .

Definition 7 (Topical Frequency) *The topical frequency $f_t(P)$ of a pattern is the number of times the pattern is attributed to topic t . For the root node o , $f_o(P) = f(P)$. For each topic node with subtopics C^t , $f_t(P) = \sum_{z \in C^t} f_z(P)$ (i.e., topical frequency is the sum of sub-topical frequencies.)*

We estimate the topical frequency of a pattern based on two assumptions: i) For a type- x topic- t pattern of length n , each of the n nodes is generated with the distribution $\phi^{x,t}$, and ii) the total number of topic- t patterns is proportional

to ρ_t .

$$f_t(P^x) = f_{Par(t)}(P^x) \frac{\rho_t \prod_{v_i^x \in P^x} \phi_i^{x,t}}{\sum_{z \in C^{Par(t)}} \rho_z \prod_{v_i^x \in P^x} \phi_i^{x,z}} \quad (7.14)$$

Both ϕ and ρ are learned from the generative model as described in Section 7.2.1.

To extract topical frequent patterns, all frequent patterns can first be mined using a pattern mining algorithm such as FP-growth [27], and then filtered given some minimal topical frequency threshold *minsup*.

Pattern ranking in each topic

The same four criteria for judging the quality of a pattern are used:

- **Frequency** – A representative pattern for a topic should have sufficiently high topical frequency.
- **Purity** – A pattern is pure in a topic if it is only frequent in this topic and not frequent in other topics. *Example: ‘query processing’ is more pure than ‘query’ in the Databases topic.*
- **Phraseness** – A group of entities should be combined together as a pattern (a ‘phrase’) if they co-occur significantly more often than the expected co-occurrence frequency given the chances of occurring independently. *Example: ‘active learning’ is a better pattern than ‘learning classification’ in the Machine Learning topic.*
- **Completeness** – A pattern is not complete if it rarely occurs without the presence of a longer pattern. *Example: ‘support vector machines’ is a complete pattern, whereas ‘vector machines’ is not because ‘vector machines’ is almost always accompanied by ‘support’ in occurrence.*

The pattern ranking function should take these criteria into consideration. The ranking function must also be able to directly compare patterns of mixed lengths, such as ‘classification,’ ‘decision trees,’ and ‘support vector machines.’

Let N_t be the number of documents that contain at least one frequent topic- t pattern, T a subset of $C^{Par(t)}$ that contains t , and N_T the number of documents that contain at least one frequent topic- z pattern for some topic $z \in T$.

We use the following ranking function that satisfies all these requirements:

$$r^t(P) = \begin{cases} 0, & \text{if } \exists P' \supseteq P, f_t(P') \geq \gamma f_t(P) \\ p(P|t) \left(\log \frac{p(P|t)}{\max_T p(P|T)} + \omega \log \frac{p(P|t)}{p_{indep}(P|t)} \right) & \text{o.w.} \end{cases} \quad (7.15)$$

where $p(P|t) = \frac{f_t(P)}{N_t}$ is the occurrence probability of a pattern P , measuring frequency; $p_{indep}(P|t) = \prod_{v \in P} \frac{f_t(v)}{N_t}$ is the probability of independently seeing every node in pattern P , measuring purity; and $p(P|T) = \frac{\sum_{t \in T} f_t(P)}{N_T}$ is the probability of phrase P conditioned on a mixture T of t and other sibling topics, measuring phraseness. Incomplete patterns are filtered if there exists a superpattern P' that has sufficiently high topical frequency compared to P . $\gamma \in [0, 1]$ is a parameter that controls the strictness of the completeness criterion, where a larger value of γ deems more

phrases to be complete. Complete phrases are ranked according to a combination of the other three criteria. Frequency plays the most important role. The weight between purity and phraseness is controlled by a parameter $\omega \in [0, +\infty)$, with larger values of ω biasing the ranking more heavily towards phraseness.

7.3 Experiments

We evaluate the performance of our proposed method on two datasets (see Table 7.4 for summary statistics of the constructed networks):

- **DBLP.** We collected 33,313 recently published computer science papers from DBLP³. We constructed a heterogeneous network with three node types: term (from paper title), author and venue, and 5 link types: term-term, term-author, term-venue, author-author and author-venue.⁴
- **NEWS.** We crawled 43,168 news articles on 16 top stories from Google News,⁵ and ran an information extraction algorithm [37] to extract entities. We constructed a heterogeneous network with three node types: term (from article title), person and location, and 6 link types: term-term, term-person, term-location, person-person, person-location and location-location.

Our recursive framework relies on two key steps: subtopic discovery and topical pattern mining. The major contribution of this paper is the subtopic discovery step. Hence, our evaluation is twofold: i) we evaluate the efficacy of subtopic discovery given a topic and its associated heterogeneous network; and ii) we perform several ‘intruder detection’ tasks to evaluate the quality of the constructed hierarchy based on human judgment.

7.3.1 Efficacy of Subtopic Discovery

We first present a set of experiments designed to evaluate just the subtopic discovery step (Step 2 in Section 7.2).

Evaluation Measure. We extend the pointwise mutual information (PMI) metric in order to measure the quality of our multi-typed topics. The metric of pointwise mutual information PMI has been proposed in [52] as a way of measuring the semantic coherence of topics. It is generally preferred over other quantitative metrics such as perplexity or the likelihood of held-out data [64]. In order to measure the quality of our multi-typed topics, we extend the definition of PMI as follows:

³We chose papers published in 20 conferences related to the areas of Artificial Intelligence, Databases, Data Mining, Information Retrieval, Machine Learning, and Natural Language Processing from <http://www.dblp.org/>

⁴As a paper is always published in exactly one venue, there can naturally be no venue-venue links.

⁵The 16 topics chosen were: Bill Clinton, Boston Marathon, Earthquake, Egypt, Gaza, Iran, Israel, Joe Biden, Microsoft, Mitt Romney, Nuclear power, Steve Jobs, Sudan, Syria, Unemployment, US Crime.

Table 7.2: Heterogeneous pointwise mutual information in DBLP (20 Conferences and Database area)

DBLP (Database Area)	T-T	T-A	A-A	T-V	A-V	Overall
TopK	-0.5228	-0.1069	0.4545	0.0348	-0.3650	-0.0761
NetClus	-0.3962	0.0479	0.4337	0.0368	-0.2857	0.0260
CATHYHIN (equal weight)	0.0561	0.4799	0.6496	0.0722	-0.0033	0.3994
CATHYHIN (norm weight)	-0.1514	0.3816	0.6971	0.0408	0.2464	0.3196
CATHYHIN (learn weight)	0.3027	0.6435	0.5574	0.1165	0.1805	0.5205
DBLP (20 Conferences)	T-T	T-A	A-A	T-V	A-V	Overall
TopK	-0.4825	-0.0204	0.5466	-1.0051	-0.4208	-0.0903
NetClus	-0.1995	0.5186	0.5404	0.2851	1.2659	0.4045
CATHYHIN (equal weight)	0.2936	0.8812	0.6595	0.5191	1.0466	0.6949
CATHYHIN (norm weight)	0.1825	0.8674	0.9476	0.7472	1.3307	0.7601
CATHYHIN (learn weight)	0.4964	1.0618	0.7161	1.1283	1.7511	0.9168

For each topic, PMI calculates the average relatedness of each pair of the words ranked at top-K:

$$PMI(\mathbf{w}, \mathbf{w}) = \frac{2}{K(K-1)} \sum_{1 \leq i < j \leq K} \log \frac{p(w_i, w_j)}{p(w_i)p(w_j)} \quad (7.16)$$

where $PMI \in [-\infty, \infty]$, and \mathbf{w} are the top K most probable words of the topic. $PMI = 0$ implies that these words are independent; $PMI > 0$ (< 0) implies they are overall positively (negatively) correlated.

However, our multi-typed topic contains not only words, but also other types of entities. So we define *heterogeneous* pointwise mutual information as:

$$HPMI(\mathbf{v}^x, \mathbf{v}^y) = \begin{cases} \frac{2}{K(K-1)} \sum_{1 \leq i < j \leq K} \log \frac{p(v_i^x, v_j^y)}{p(v_i^x)p(v_j^y)} & x = y \\ \frac{1}{K^2} \sum_{1 \leq i, j \leq K} \log \frac{p(v_i^x, v_j^y)}{p(v_i^x)p(v_j^y)} & x \neq y \end{cases} \quad (7.17)$$

where \mathbf{v}^x are the top K most probable type- x nodes in the given topic. When $x = y$, HPMI reduces to PMI. The HPMI-score for every link type (x, y) is calculated and averaged to obtain an overall score. We set $K = 20$ for all node types.⁶

Methods for Comparison:

- **CATHYHIN (equal weight)** – The weight for every link type is set to be 1.
- **CATHYHIN (learn weight)** – The weight of each link type is learned, as described in Section 7.2.1. No parameters need hand tuning.
- **CATHYHIN (norm weight)** – The weight of each link type is explicitly set as: $\alpha_{x,y} = \frac{1}{\sum_{i,j} e_{i,j}^{x,y}}$. This is a

⁶The one exception is venues, as there are only 20 venues in the DBLP dataset, so we set $K = 3$ in this case.

Table 7.3: Heterogeneous pointwise mutual information in NEWS (16 topics collection and 4 topics subset)

NEWS (4 topics subset)	T-T	T-P	P-P	T-L	P-L	L-L	Overall
TopK	-0.2479	0.1671	0.0716	0.0787	0.2483	0.3632	0.1317
NetClus	0.1279	0.3835	0.2909	0.3240	0.4728	0.4271	0.3575
CATHYHIN (equal weight)	1.0471	0.7917	0.4902	0.8506	0.6821	0.6586	0.7610
CATHYHIN (norm weight)	0.7975	0.8825	0.5553	0.8682	0.8077	0.7346	0.8023
CATHYHIN (learn weight)	0.9935	0.9354	0.5142	0.9784	0.7389	0.7645	0.8434

NEWS (16 topics)	T-T	T-P	P-P	T-L	P-L	L-L	Overall
TopK	-1.7060	-0.8663	-0.8462	-1.0238	-0.5665	-0.4578	-0.8783
NetClus	-0.3847	0.0943	0.0313	-0.1114	0.1291	0.1376	-0.0274
CATHYHIN (equal weight)	0.7804	1.0170	0.8393	0.8354	0.9467	0.6382	0.8749
CATHYHIN (norm weight)	0.8579	1.1143	0.9086	0.8530	0.9624	0.7143	0.9284
CATHYHIN (learn weight)	0.9234	1.1109	0.7966	0.9731	0.9718	0.6965	0.9500

heuristic normalization which forces the total weight of the links for each link type to be equal.

- **NetClus** – The current state-of-the-art clustering and ranking method for heterogeneous networks. We use the implementation in Deng *et al.* [17]. The smoothing parameter λ_S is tuned by a grid search in $[0, 1]$. Note that the link type weight learning method for CATHYHIN does not apply to NetClus because NetClus is not a single unified model.
- **TopK** – Select the top K nodes from each type to form a pseudo topic. This method serves as a baseline value for the proposed HPMI metric.

Experiment Setup. We discover the subtopics of four datasets:

- DBLP (20 conferences) – Aforementioned DBLP dataset.
- DBLP (database area) – A subset of the DBLP dataset consisting only of papers published in 5 Database conferences. By using this dataset, which roughly represents a subtopic of the full DBLP dataset, we analyze the quality of discovered subtopics in a lower level of the hierarchy.
- NEWS (16 topics) – Aforementioned NEWS dataset.
- NEWS (4 topic subset) – A subset of the NEWS dataset limited to 4 topics, which center around different types of entities: Bill Clinton, Boston Marathon, Earthquake, Egypt.

Experiment Results. All the methods finish in 1.5 hours for these datasets. As seen in Tables 7.2 and 7.3, our generative model consistently posts a higher HPMI score than NetClus (and TopK) across all links types in every dataset. Although NetClus HPMI values are better than the TopK baseline, the improvement of our best performing method - CATHYHIN (learn weight) - over the TopK baseline are better than the improvement posted by NetClus by

factors ranging from 2 to 5.8. Even the improvement over the TopK baseline of CATHYHIN (equal weight), which considers uniform link type weights, is better than the improvement posted by NetClus by factors ranging from 1.6 to 4.6.

CATHYHIN with learned link type weights consistently yields the highest overall HPMI scores, although CATHYHIN with normalized link type weights sometimes shows a slightly higher score for particular link types (e.g., Author-Author for both DBLP datasets, and Person-Person for both NEWS datasets). CATHYHIN (norm weight) assigns a high weight to a link type whose total link weights were low in the originally constructed network, pushing the discovered subtopics to be more dependent on that link type. Normalizing the link type weights does improve CATHYHIN performance in many cases, as compared to using uniform link type weights. However, this heuristic determines the link type weight based solely on their link density. It can severely deteriorate the coherence of dense but valuable link types, such as Term-Term in both DBLP datasets, and rely too heavily on sparse but uninformative entities, such as Venues in the Database subtopic of the DBLP dataset.

We may conclude from these experiments that CATHYHIN’s unified generative model consistently outperforms the state-of-the-art heterogeneous network analysis technique NetClus. In order to generate coherent, multi-typed topics at each level of a topical hierarchy, it is important to learn the optimal weights of different entity types, which depends on the link type density, the granularity of the topic to be partitioned, and the specific domain.

Table 7.4: # Links in our datasets

<i>DBLP</i> (# Nodes)	Term (6,998)	Author (12,886)	Venue (20)
Term	693,132	900,201	104,577
Author	–	156,255	99,249

<i>NEWS</i> (# Nodes)	Term (13,129)	Person (4,555)	Location (3,845)
Term	686,007	386,565	506,526
Person	–	53,094	129,945
Location	–	–	85,047

7.3.2 Topical Hierarchy Quality

Our second set of evaluations assesses the ability of our method to construct a hierarchy of multi-typed topics that human judgement deems to be high quality. We generate and analyze multi-typed topical hierarchies using the DBLP dataset (20 conferences) and the NEWS dataset (16 topics collection).

Experiment Setup. As in the evaluation of the original CATHY framework, we adapt the same two tasks from Chang et al. [8], who were the first to explore human evaluation of topic models. Each task involves a set of questions asking

Table 7.5: Results of Intruder Detection tasks (% correct intruders identified)

	DBLP				NEWS			
	Phrase	Venue	Author	Topic	Phrase	Location	Person	Topic
CATHYHIN	0.83	0.83	1.0	1.0	0.65	0.70	0.80	0.90
CATHYHIN₁	0.64	–	–	0.92	0.40	0.55	0.50	0.70
CATHY	0.72	–	–	0.92	0.58	–	–	0.65
CATHY₁	0.61	–	–	0.92	0.23	–	–	0.50
CATHY_{heur_HIN}	–	0.78	0.94	0.92	–	0.65	0.45	0.70
NetClus_{pattern}	0.33	0.78	0.89	0.58	0.23	0.20	0.55	0.45
NetClus_{pattern_1}	0.53	–	–	0.58	0.20	0.45	0.30	0.40
NetClus	0.19	0.78	0.83	0.83	0.15	0.35	0.25	0.45

humans to discover the ‘intruder’ object from several options. Three annotators manually completed each task, and their evaluations scores were pooled.

The first task is Phrase Intrusion, which evaluates how well the hierarchies are able to separate phrases in different topics. Each question consists of X ($X = 5$ in our experiments) phrases; $X - 1$ of them are randomly chosen from the top phrases of the same topic and the remaining phrase is randomly chosen from a sibling topic. The second task is Entity Intrusion, a variation that evaluates how well the hierarchies are able to separate entities present in the dataset in different topics. For each entity type, each question consists of X entity patterns; $X - 1$ of them are randomly chosen from the top patterns of the same topic and the remaining entity pattern is randomly chosen from a sibling topic. This task is constructed for each entity type in each dataset (Author and Venue in DBLP; Person and Location in NEWS). The third task is Topic Intrusion, which tests the quality of the parent-child relationships in the generated hierarchies. Each question consists of a parent topic t and X candidate child topics. $X - 1$ of the child topics are actual children of t in the generated hierarchy, and the remaining child topic is not. Each topic is represented by its top 5 ranked patterns of each type - e.g., for the NEWS dataset, the top 5 phrases, people, and locations are shown for each topic.

For each question, human annotators select the intruder phrase, entity, or subtopic. If they are unable to make a choice, or choose incorrectly, the question is marked as a failure. The instructions remained the same as those used in the original CATHY studies, and complete set of instructions for the tasks can be found in Appendix C (the instructions for Entity Intrusion are identical to those for Phrase Intrusion.)

Methods for Comparison:

- **CATHYHIN** – As defined in Section 7.2
- **CATHYHIN₁** – The pattern length of text and every entity type is restricted to 1.
- **CATHY** – As defined in Chapter 5, the hierarchy is constructed only from textual information.
- **CATHY₁** – The phrase length is restricted to 1.

- **CATHY_{heuristic.HIN}** – The method presented in the previous chapter, which uses a heuristic entity ranking method based on the textual hierarchy generated by CATHY, and the original links in the network. Since neither CATHY nor CATHY₁ provides topical ranks for entities, we use this method to have a comparison for the Entity Intrusion task.
- **NetClus_{pattern}** – NetClus is used for subtopic discovery, followed by the topical mining and ranking method of CATHYHIN, as described in Section 7.2.2 (this can also be thought of CATHYHIN, where Step 2 is replaced by NetClus).
- **NetClus_{pattern.1}** – Equivalent to NetClus_{pattern} with the pattern length of text and every entity type restricted to 1.
- **NetClus** – As defined in [63].

The pattern mining and ranking parameters for both CATHY and CATHYHIN are set to be $minsup = 5, \omega = \gamma = 0.5$. The optimal smoothing parameter for NetClus is $\lambda_S = 0.3$ and 0.7 in DBLP and NEWS respectively.

Table 7.5 displays the results of the intruder detection tasks. For the Entity Intrusion task on the DBLP dataset, we restricted the entity pattern length to 1 in order to generate meaningful questions. This renders the methods CATHYHIN₁ and NetClus_{pattern.1} equivalent to CATHYHIN and NetClus_{pattern} respectively, so we omit the former methods from reporting.

Experiment Results. The Phrase Intrusion task performs much better when phrases are used rather than unigrams, for both CATHYHIN and CATHY, on both datasets. The NEWS dataset exhibits a stronger preference for phrases, as opposed to the DBLP dataset, which may be due to the fact that the terms in the NEWS dataset are more likely to be noisy and uninformative outside of their context, whereas the DBLP terms are more technical and therefore easier to interpret. This characteristic may also help explain why the performance of every method on DBLP data is consistently higher than on NEWS data. However, neither phrase mining and ranking nor unigram ranking can make up for poor performance during the topic discovery step, as seen in the three NetClus variations. Therefore, both phrase representation and high quality topics are necessary for good topic interpretability.

For the Entity Intrusion task, all of the relevant methods show comparable performance in identifying Author and Venue intruders in the DBLP dataset (though CATHYHIN is still consistently the highest). Since the DBLP dataset is well structured, and the entity links are highly trustworthy, identifying entities by topic is likely easier. However, the entities in the NEWS dataset were automatically discovered from the data, and the link data is therefore noisy and imperfect. CATHYHIN is the most effective in identifying both Location and Person intruders. Once again, both better topic discovery and improved pattern representations are responsible for CATHYHIN’s good results, and simply enhancing the pattern representations, whether for CATHY or NetClus, cannot achieve competitive performance.

Table 7.6: The ‘Egypt’ topic and the least sensible subtopic, as generated by three methods (only Phrases and Locations are shown)

CATHYHIN	CATHY _{heuristic.HIN}	NetClus _{pattern}
{egypt; egypt; death toll; morsi} / {Egypt; Egypt Cairo; Egypt Israel; Egypt Gaza}	{egypt; egypt; morsi; egypt imf loan; egypt president} / {Egypt; Cairo; Tahrir Square; Port Said}	{bill clinton; power nuclear; rate unemployment; south sudan} / {Egypt Cairo; Egypt Coptic; Israel Jerusalem; Libya Egypt}
↓	↓	↓
{death toll; egyptian; sexual harassment; egypt soccer} / {Egypt Cairo; Egypt Gaza; Egypt Israel}	{supreme leader; army general sex; court; supreme court} / {US; Sudan; Iran; Washington}	{egypt; coptic pope; egypt christians; obama romney; romney campaign} / {Egypt Cairo; Egypt Coptic; Israel Jerusalem; Egypt}

CATHYHIN performs very well in the Topic Intrusion task on both datasets. Similar to the Phrase Intrusion task, both CATHYHIN and CATHY yield equally good or better result when phrases and entity patterns are mined, rather than just terms and single entities. The fact that CATHYHIN always outperforms CATHY demonstrates that utilizing entity link information is indeed helpful for improving topical hierarchy quality. As a worst-case study, Table 7.6 illustrates three representations of the topic ‘Egypt’ (one of the 16 top stories in NEWS dataset), each with its least comprehensible subtopic. The locations found within the CATHYHIN subtopic are sensible. However, CATHY_{heuristic.HIN} first constructs phrase-represented topics from text, and then uses entity link information to rank entities in each topic. Thus the entities are not assured to fit well into the constructed topic, and indeed, the CATHY_{heuristic.HIN} subtopic’s locations are not reasonable given the parent topic. Finally, NetClus_{pattern} conflates ‘Egypt’ with several other topics, and the pattern representations can do little to improve the topic interpretability.

In all three intruder detection tasks on both datasets, CATHYHIN consistently outperforms all other methods, showing that an integrated heterogeneous model consistently produces a more robust hierarchy which is more easily interpreted by human judgement.

Chapter 8

Conclusions

The outcome of any serious research can only be to make two questions grow where only one grew before.

(Thorstein Veblen)

8.1 Conclusions of the Dissertation

In this dissertation, we introduce KERT (Keyphrase Extraction and Ranking by Topic), a framework for topical keyphrase generation and ranking on collections of short texts. By altering the steps in the traditional methods of unsupervised keyphrase extraction, KERT was able to directly compare phrases of different lengths, resulting in a natural integrated ranking of mixed-length keyphrases. The effectiveness of KERT was demonstrated on two real world short document collections, yielding over 50% improvement over a baseline method according to human judgement and over 20% improvement according to mutual information.

We next adapted KERT for collections of longer text, outside of the scientific domain. We demonstrated the robustness of the KERT framework by easily altering the first step of the framework to use a topical assignment method which was better suited to a different dataset. We also showed the effectiveness of KERT on medium-length text as well as short, requiring only a small additional step of breaking text into windows after the topical assignment step is completed. The modular nature of KERT allows us to keep documents intact when performing the topic assignment step, ensuring that the relationships between unigrams within the same document are reflected by the model, information which would be lost if the documents had to be separated into windows from the very beginning. On the other hand, the simple and efficient pattern mining technique used in KERT yields the same result as using time- and information-intensive language-based processes such as chunking in order to discover good quality phrases. A new, task-centric evaluation of algorithm performance was also presented, effectively demonstrating the quality of the KERT framework in assisting with important and subjective classification-style tasks. Finally, we show that KERT's runtime is efficient compared to other phrase-generating approaches.

The next step was to move beyond a flat set of topics and introduce CATHY (Constructing a Topical HierarchY), a phrase-centric framework for topical hierarchy generation via recursive clustering and ranking. The process of topic

assignment underwent a serious transformation in order to be able to successfully generate topics where each topic is represented by a ranked list of topical phrases, such that a child topic is a subset of its parent topic. It is very difficult to use the output of a topic model over a dataset to split that dataset into subsets and proceed to discover the subtopics present in those subsets. We therefore created a graph-based algorithm which transformed the text into a term co-occurrence graph, and the topic assignment problem into one of link clustering. However, in keeping with our human-centric focus, we still ensured that every topic and subtopic could be represented as a high-quality list of mixed-length phrases. Both qualitative and quantitative evaluations against several hierarchy-generating methods show that CATHY is able to generate a much higher quality topical hierarchy from a collection of short texts.

The following chapter explored a further application of the CATHY framework to mine entity roles in topical communities, such as the role of an author in a research community, or the contribution of a user to a social network community organized around similar interests. In particular, we demonstrate our method on a bibliographic dataset, which we use to discover the roles of authors and publication venues in the context of the hierarchical topical communities. Finally, the penultimate chapter united the aims of chapters 5 and 6 by introducing CATHY HIN (Constructing a Topical HierarchY from a Heterogeneous Information Network), which is both able to construct a hierarchy and incorporate non-textual information during the construction. Using entity link information (e.g., terms are related via being used by the same author) improves the quality of the phrases in the constructed hierarchy, such that the topics generated by CATHY HIN are found to be slight better than those generated by CATHY.

8.2 Future Research Needs

There will never stop being a need for algorithms which generate results that are interpretable to humans. There will also likely never be a perfect way to evaluate the quality of these results. This dissertation has barely touched the surface of the problem of evaluating results aimed at human judgement, and there is a great need for more research in this direction. In particular, task-centric evaluation is a promising direction to explore, since the goal of these approaches is to improve the quality of human tasks of browsing and analyzing large collections of topical information. Datasets which naturally lend themselves to well-defined tasks, such as the Goodreads dataset and the task of characterizing the genre of a book, will hopefully become more widely and easily available.

This dissertation also leaves unsolved the natural next step for CATHY of working on longer texts. KERT employed a relatively simple and successful approach of breaking text into windows following the initial step of topic assignment (an approach which in itself merits further careful study, since we hypothesize that the ideal breaking technique might vary for text collections from different domains.) However, the term co-occurrence graph constructed by CATHY would be too noisy if used directly on a long document, since it would connect terms whose semantic dis-

tance is too great. On the other hand, simply breaking the long text into short windows prior to constructing the term co-occurrence graph loses information that does connect terms which are present in the same document. A formalized approach to adapting CATHY (and CATHY HIN) to collections that have longer text components is undoubtedly necessary.

All of the frameworks presented in this dissertation would benefit from working with high quality entity extraction algorithms. Identifying and extracting entities from text is a complex and widely-studied area. A good entity extraction algorithm would allow CATHY and CATHY HIN to work with better quality networking with text components, because the links between documents and the entities present in them would be found to be more trustworthy. Similarly, entity disambiguation algorithms (ones which discover that ‘POTUS’ and ‘Barack Obama’ refer to the same individual - at least, at the time of the writing of this dissertation) would improve the quality of the network datasets for CATHY and CATHY HIN. However, as seen in the sample results shown in Chapter 4, KERT could also greatly benefit from identifying entity information in some datasets, such as the names of authors or literary characters in the Goodreads dataset, differentiating them from actual phrases found in the text.

Overall, we hope that this dissertation has in some small way contributed to the understanding of generating topical phrases from document collections, such as book descriptions, or from networks with text and entity components, such as bibliographic data. We also hope that by keeping a human-centric orientation in mind while developing the presented techniques, we will encourage future research that will continue to help people with the many difficult, subjective, and confusing information tasks that will continue to face humanity.

Appendix A

Instructions for KERT User Study on DBLP Dataset

The following are the instructions of the user study presented in Chapter 3, which was distributed to participants as an Excel file.

Computer Science Topical Keyphrase Survey - Phrase Quality

Thank you for your interest! This survey is part of a research project studying the quality of topical keyphrases as generated by various algorithms. It is being conducted by Marina Danilevsky (danilev1@illinois.edu) and Chi Wang (chiwang1@illinois.edu), graduate students in the Data Mining group under Dr. Jiawei Han, in the Department of Computer Science at UIUC. Please feel free to contact us if you have any questions.

If you volunteer to participate and complete this survey, you will receive \$5!

This survey will present 5 collections of keyphrases in 5 tabs, where each keyphrase collection represents one of the following topics in the area of computer science: (1) Machine Learning, (2) Databases, (3) Data Mining, and (4) Information Retrieval. For each topic, you will be asked to evaluate the list of keyphrases on a scale of 1-5 based on the quality of the keyphrase.

A high quality keyphrase will exhibit all 4 of the following characteristics:

- Coverage: A high quality topical keyphrase should be representative of the topic, and provide good coverage of lots of work done in that topic. For example, “information retrieval” has better coverage than “cross-language information retrieval” in the Information Retrieval topic, because “information retrieval” is more general and more frequent in that topic, whereas “cross-language information retrieval” is more specific and less frequent in that topic.
- Purity: A high quality keyphrase should be strongly associated with only that topic, and not associated with the other topics. For example, “query processing” is more pure than “query” in the Database topic, since “query” is more often found in the other topics such as Information Retrieval, whereas “query processing” is more specific to the Database topic.
- Phraseness: A high quality keyphrase should be a phrase that makes sense in that topic, made up of unigrams

which frequently co-occur in that topic. For example, “learning classification” is not a very good phrase in the Machine Learning topic, because although “learning” and “classification” both occur often in that topic, they do not co-occur often. On the other hand, “active learning” is a very good phrase in the Machine Learning topic.

- **Completeness:** A high quality keyphrase should be a complete phrase, rather than an incomplete subset of a longer phrase. For example, “vector machines” is not a complete phrase in the Machine Learning topic, because it is almost always encountered as a subset of the phrase “support vector machines,” which is a complete phrase.

Phrases which exhibit all of these characteristics for the given topic should be ranked higher. Phrases which exhibit only some of these characteristics should be ranked lower.

For each topic, the keyphrases have been generated using multiple approaches, and are presented in completely random order. You are therefore asked to carefully evaluate each keyphrase independently. Please give every phrase a score! However do not spend more than a few seconds on each question.

This part of the survey consists of 4 tabs, labeled 3.1 - 3.4. On each tab, you will be asked to evaluate the quality of keyphrases for one of the topics. The topic will be specified at the top of the page. Please proceed through the tabs in order.

When you have completed this survey, please save this file to record your responses to this survey, and email this file to either Marina (danilev1@illinois.edu) or Chi (chiwang1@illinois.edu). We will then contact you to provide you with your reward of \$5.

Appendix B

Instructions for KERT User Studies on Goodreads Dataset

The following are the instructions of the two user studies presented in Chapter 4, which were administered using the Amazon Mechanical Turk Platform. A HIT for these studies consisted of 9 sets of topical phrases (one set per genre, per algorithm) and either a genre name, such as ‘Humor’, for the Genre Characterization study, or a book description for the Book Characterization study. Each user received \$0.30 per HIT as remuneration.

Instructions for the Genre Characterization Study

You will be asked to read several genre descriptions. You will then evaluate how well each genre description characterizes a given genre. Each genre description is a set of words or phrases, describing a broad class of books (a genre). Each genre description has been generated by a particular computer algorithm. We want to understand which of these algorithms are better at generating genre descriptions.

A Good (rating a 5) Genre Description is not too narrow (could only describe 1 or 2 books), and not too general (could describe any book)

Read through each genre description. Rate each genre description on how well it characterizes the genre from 1 (worst) to 5 (best). You can rate more than one genre description as being good, and more than one as being bad. Choose the best rating for each description, regardless of any other rating you put down.

Acceptance Criteria: To be completely transparent, the following criteria will be used to determine which HITs are approved and rejected:

- All questions must be answered.
- ALL descriptions should NOT be the same score (many responses can have the same score, but if all responses are the same the HIT will be rejected)

Instructions for the Book Characterization Study

You will be asked to read a book description and several genre descriptions. You will then evaluate how well each genre description characterizes this book. Each genre description is a set of words or phrases, describing a broad

class of books (a genre). Each genre description has been generated by a particular computer algorithm. We want to understand which of these algorithms are better at generating genre descriptions. You need NOT have read this book to complete this HIT.

A Good (rating a 5) Genre Description...

- Is not too narrow (could only describe 1 or 2 books), and not too general (could describe any book)
- Contains at least a couple phrases that are directly related to the presented book
- A majority of phrases should be similar to the presented book, though the phrases may not be strongly related
- Does not have many phrases that are completely unrelated to the presented book

Read through the book title, author(s) and description Rate each genre description on how well it characterizes the book from 1 (worst) to 5 (best) You can rate more than one genre description as being good, and more than one as being bad. Choose the best rating for each description, regardless of any other rating you put down.

Acceptance Criteria: To be completely transparent, the following criteria will be used to determine which HITs are approved and rejected:

- All questions must be answered.
- ALL descriptions should NOT be the same score (many responses can have the same score, but if all responses are the same the HIT will be rejected)

Appendix C

Instructions for CATHY User Study on DBLP Dataset

The following are the instructions of the two user studies presented in Chapter 5, which were distributed to participants as an Excel file.

Instructions for Computer Science Topical Keyphrase Survey

Thank you for your interest! This survey is part of a research project studying the quality of topical keyphrases as generated by various algorithms. It is being conducted by Marina Danilevsky (danilev1@illinois.edu) and Chi Wang (chiwang1@illinois.edu), graduate students in the Data Mining group under Dr. Jiawei Han, in the Department of Computer Science at UIUC. Please feel free to contact us if you have any questions.

If you volunteer to participate and complete this survey, you will receive \$15!

A keyphrase is a phrase which is a good label, or representation for a particular topic. This survey will present keyphrases from Computer Science topics. The survey is divided into 2 sections. Each question in Section 1 will present a set of phrases and ask you to select which phrase is most unlike the other phrases. Each question in Section 2 will present several sets of phrases and ask you to basically select which phrase set is most unlike the other phrase sets.

Please proceed through this survey by entering your responses for each question, moving through the tabs in order from left to right. Each section will begin with a tab explaining the instructions for answer the questions in that section in detail. Please read the detailed instructions for each section carefully! Please periodically save this file to not lose your work.

When you have completed this survey, please save this file to record your responses to this survey, and email this file to either Marina (danilev1@illinois.edu) or Chi (chiwang1@illinois.edu). We will then contact you to provide you with your reward of \$15.

Computer Science Topical Keyphrase Survey - Section 1 - Phrase Intrusion

Each question in this section will present a set of phrases and ask you to select which phrase is most unlike the other phrases - in other words, which phrase is the intruder?

For example, for the question:

- {black dog, orange cat, red dog, small ring}

The correct answer would be 4, because “small ring” is the phrase intruder, the phrase which is most unlike the others.

The phrase intrusion questions have been generated using multiple approaches, and are presented in completely random order. You are therefore asked to carefully evaluate each phrase intrusion questions independently. Please enter a response for each question! However do not spend more than a few seconds on each question. If you have really tried, but still feel like you cannot choose a clear answer for a question, please enter 0 as your response.

Computer Science Topical Keyphrase Survey - Section 2 - Topic Intrusion

Each question in this section will present a set of phrases that represents a parent topic as well as several other sets of phrases that represent potential child topics. You will be asked to select which topic is the least likely to be a child of the parent topic - in other words, which child topic is the intruder?

For example, for the question:

- Parent topic: {house pets, colorful animals }
 1. Child topic: {black dog, orange cat }
 2. Child topic: {blue fish, lizard }
 3. Child topic: {large table, small ring }
 4. Child topic: {bird, brown hamster }

The correct answer would be 3, because “small ring” and “large table” are not likely children of the parent topic.

The topic intrusion questions have been generated using multiple approaches, and are presented in completely random order. You are therefore asked to carefully evaluate each topic intrusion question independently. Please enter a response for each question! However do not spend more than a few seconds on each question. If you have really tried, but still feel like you cannot choose a clear answer for a question, please enter 0 as your response.

References

- [1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases, VLDB '94*, page 487499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.
- [2] B. Ball, B. Karrer, and M. E. J. Newman. Efficient and principled method for detecting communities in networks. *Physical Review E*, 84(3):036103, 2011.
- [3] K. Barker and N. Cornacchia. Using noun phrase heads to extract document keyphrases. In *Proceedings of the 13th Biennial Conference of the Canadian Society on Computational Studies of Intelligence: Advances in Artificial Intelligence, AI '00*, 2000.
- [4] S. Bird, E. Klein, and E. Loper. *Natural Language Processing with Python*. O'Reilly Media, Inc., 1st edition, 2009.
- [5] D. M. Blei and M. I. Jordan. Modeling annotated data. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval, SIGIR '03*, page 127134, New York, NY, USA, 2003. ACM.
- [6] D. M. Blei and J. D. Lafferty. Visualizing topics with multi-word expressions. *arXiv preprint arXiv:0907.1013*, 2009.
- [7] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3, 2003.
- [8] J. Chang, J. Boyd-Graber, C. Wang, S. Gerrish, and D. M. Blei. Reading tea leaves: How humans interpret topic models. In *Proceedings of the 2009 Conference on Advances in Neural Information Processing Systems, NIPS '09*, 2009.
- [9] D. H. Chau, C. Nachenberg, J. Wilhelm, A. Wright, and C. Faloutsos. Large scale graph mining and inference for malware detection. In *Proceedings of the 11th SIAM International Conference on Data Mining, SDM '11*, 2011.
- [10] X. Chen, M. Zhou, and L. Carin. The contextual focused topic model. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12*, 2012.
- [11] B.-H. Chou and E. Suzuki. Discovering community-oriented roles of nodes in a social network. In *Proceedings of the 12th international conference on Data warehousing and knowledge discovery, DaWaK'10*, 2010.
- [12] S.-L. Chuang and L.-F. Chien. A practical web-based approach to generating topic hierarchy for text segments. In *Proceedings of the 13th ACM International Conference on Information and Knowledge Management, CIKM '04*, 2004.
- [13] K. W. Church and P. Hanks. Word association norms, mutual information, and lexicography. In *Proceedings of the 27th annual meeting on Association for Computational Linguistics, ACL '89*, page 7683, Stroudsburg, PA, USA, 1989. Association for Computational Linguistics.

- [14] A. Clauset, C. Moore, and M. E. J. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453, May 2008.
- [15] J. Cohen. Weighted kappa: nominal scale agreement with provision for scaled disagreement or partial credit. *Psychological Bulletin*, 70(4):213220, 1968.
- [16] G. Costa and R. Ortale. A bayesian hierarchical approach for exploratory analysis of communities and roles in social networks. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '12, 2012.
- [17] H. Deng, J. Han, B. Zhao, Y. Yu, and C. X. Lin. Probabilistic topic models with biased propagation on heterogeneous information networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, 2011.
- [18] L. Di Caro, K. S. Candan, and M. L. Sapino. Using tagflake for condensing navigable tag hierarchies from tag clouds. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, 2008.
- [19] J. Eisenstein, A. Ahmed, and E. P. Xing. Sparse additive generative models of text. In *Proceedings of the 28th Annual International Conference on Machine Learning*, ICML '11, 2011.
- [20] M. G. Everett and S. P. Borgatti. Regular equivalence - general-theory. *Journal of Mathematical Sociology*, 19(1):2952, 1994.
- [21] S. Fortunato. Community detection in graphs. *Physics Reports*, 486:3-5, 2010.
- [22] S. Gauch, J. Chaffee, and A. Pretschner. Ontology-based personalized search and browsing. *Web Intelligence and Agent Systems*, 1(3/4):219–234, 2003.
- [23] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-6(6):721–741, Nov. 1984.
- [24] D. M. B. T. L. Griffiths and M. I. J. J. B. Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. In *Proceedings of the 2004 Conference on Advances in Neural Information Processing Systems*, volume 16 of *NIPS '04*, page 17, 2004.
- [25] T. L. Griffiths, M. Steyvers, D. M. Blei, and J. B. Tenenbaum. Integrating topics and syntax. In *Advances in Neural Information Processing Systems 17*, page 537544. MIT Press, 2005.
- [26] M. Grineva, M. Grinev, and D. Lizorkin. Extracting key terms from noisy and multitheme documents. In *Proceedings of the 18th international conference on World wide web*, WWW '09, 2009.
- [27] J. Han, J. Pei, Y. Yin, and R. Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining and Knowledge Discovery*, 8(1):5387, Jan. 2004.
- [28] K. Henderson, B. Gallagher, T. Eliassi-Rad, H. Tong, S. Basu, L. Akoglu, D. Koutra, C. Faloutsos, and L. Li. RolX: structural role extraction & mining in large graphs. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, 2012.
- [29] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42(1-2):177196, Jan. 2001.
- [30] R. Jin, V. E. Lee, and H. Hong. Axiomatic ranking of network role similarity. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, 2011.
- [31] X. Jin, S. Spangler, R. Ma, and J. Han. Topic initiator detection on the world wide web. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, 2010.
- [32] K. Jrvelin and J. Kekkonen. Cumulated gain-based evaluation of IR techniques. *ACM Transaction on Information Systems*, 20(4):422446, Oct. 2002.

- [33] H. Kim, Y. Sun, J. Hockenmaier, and J. Han. ETM: entity topic models for mining documents associated with entities. In *Proceedings of the Twelfth IEEE International Conference on Data Mining, ICDM '12*, 2012.
- [34] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
- [35] A. Lancichinetti, S. Fortunato, and J. Kertsz. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11(3):033015, 2009.
- [36] D. Lawrie and W. B. Croft. Discovering and comparing topic hierarchies. In *Proceedings of 2000 RIAO Conference*, 2000.
- [37] Q. Li, H. Ji, and L. Huang. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL '13*, 2013.
- [38] W. Li and A. McCallum. Pachinko allocation: DAG-structured mixture models of topic correlations. In *Proceedings of the 23rd Annual International Conference on Machine Learning, ICML '06*, 2006.
- [39] R. V. Lindsey, W. P. Headden, III, and M. J. Stipicevic. A phrase-discovering topic model using hierarchical pitman-yor processes. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL '12*, 2012.
- [40] X. Liu, Y. Song, S. Liu, and H. Wang. Automatic taxonomy construction from keywords. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12*, 2012.
- [41] Y. Liu, A. Niculescu-Mizil, and W. Gryc. Topic-link LDA: joint models of topic and author community. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09*, 2009.
- [42] Z. Liu, W. Huang, Y. Zheng, and M. Sun. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP '10*, 2010.
- [43] Z. Liu, P. Li, Y. Zheng, and M. Sun. Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP '09*, 2009.
- [44] C. D. Manning and H. Schtze. *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA, USA, 1999.
- [45] A. McCallum, X. Wang, and A. Corrada-Emmanuel. Topic and role discovery in social networks with experiments on enron and academic email. *Journal of Artificial Intelligence Research (JAIR)*, 30:249–272, 2007.
- [46] Q. Mei, D. Cai, D. Zhang, and C. Zhai. Topic modeling with network regularization. In *Proceedings of the 17th International Conference on World Wide Web, WWW '08*, page 101110, New York, NY, USA, 2008. ACM.
- [47] R. Mihalcea and D. Radev. *Graph-based natural language processing and information retrieval*. Cambridge University Press, 2011.
- [48] R. Mihalcea and P. Tarau. TextRank: bringing order into texts. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, EMNLP '04*, Barcelona, Spain, 2004.
- [49] D. Mimno, W. Li, and A. McCallum. Mixtures of hierarchical topics with pachinko allocation. In *Proceedings of the 24th Annual International Conference on Machine Learning, ICML '07*, 2007.
- [50] T. P. Minka. *Estimating a Dirichlet distribution*. <http://research.microsoft.com/en-us/um/people/minka/papers/dirichlet/minka-dirichlet.pdf>, 2000.
- [51] R. Navigli, P. Velardi, and S. Faralli. A graph-based algorithm for inducing lexical taxonomies from scratch. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence, IJCAI '11*, 2011.
- [52] D. Newman, J. H. Lau, K. Grieser, and T. Baldwin. Automatic evaluation of topic coherence. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, NAACL-HLT*, 2010.

- [53] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Proceedings of the 2001 Conference on Advances in Neural Information Processing Systems, NIPS '01*, 2001.
- [54] T. D. Nguyen and M.-Y. Kan. Keyphrase extraction in scientific publications. In *Proceedings of the 10th international conference on Asian digital libraries: looking back 10 years and forging new frontiers, ICADL'07*, page 317326, Berlin, Heidelberg, 2007. Springer-Verlag.
- [55] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled LDA: a supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP '09*, page 248256, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.
- [56] I. Sato and H. Nakagawa. Topic models with power-law using pitman-yor process. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '10*, 2010.
- [57] S. Schaeffer. Graph clustering. *Computer Science Review*, 1(1), 2007.
- [58] G. Schwarz. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.
- [59] J. Scripps, P.-N. Tan, and A.-H. Esfahanian. Exploration of link structure and community-based node roles in network analysis. In *Proceedings of the Seventh IEEE International Conference on Data Mining, ICDM '07*, 2007.
- [60] R. Snow, D. Jurafsky, and A. Y. Ng. Learning syntactic patterns for automatic hypernym discovery. In *Proceedings of the 2004 Conference on Advances in Neural Information Processing Systems, NIPS '04*, 2004.
- [61] M. Steyvers, P. Smyth, M. Rosen-Zvi, and T. Griffiths. Probabilistic author-topic models for information discovery. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '04*, page 306315, New York, NY, USA, 2004. ACM.
- [62] Y. Sun, J. Han, J. Gao, and Y. Yu. iTopicModel: information network-integrated topic modeling. In *Proceedings of the Ninth IEEE International Conference on Data Mining, ICDM '09*, 2009.
- [63] Y. Sun, Y. Yu, and J. Han. Ranking-based clustering of heterogeneous information networks with star network schema. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '09*, 2009.
- [64] J. Tang, M. Zhang, and Q. Mei. One theme in all views: modeling consensus topics in multiple contexts. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '13*, 2013.
- [65] T. Tomokiyo and M. Hurst. A language model approach to keyphrase extraction. In *Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment - Volume 18, MWE '03*, 2003.
- [66] P. D. Turney. Learning algorithms for keyphrase extraction. *Information Retrieval*, 2(4):303336, May 2000.
- [67] C. Wang, J. Han, Y. Jia, J. Tang, D. Zhang, Y. Yu, and J. Guo. Mining advisor-advisee relationships from research publication networks. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '10*, 2010.
- [68] X. Wang, A. McCallum, and X. Wei. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *Proceedings of the Seventh IEEE International Conference on Data Mining, ICDM '07*, page 697702, Washington, DC, USA, 2007.
- [69] W. Wong, W. Liu, and M. Bennamoun. Ontology learning from text: A look back and into the future. *ACM Computing Surveys (CSUR)*, 44(4):20, 2012.
- [70] N. Yuruk, M. Mete, X. Xu, and T. A. J. Schweiger. AHSCAN: agglomerative hierarchical structural clustering algorithm for networks. In *Proceedings of the 2009 International Conference on Advances in Social Network Analysis and Mining, ASONAM '09*, 2009.

- [71] E. Zavitsanos, G. Paliouras, G. A. Vouros, and S. Petridis. Discovering subsumption hierarchies of ontology concepts from text corpora. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence, WI '07*, 2007.
- [72] W. X. Zhao, J. Jiang, J. He, Y. Song, P. Achananuparp, E.-P. Lim, and X. Li. Topical keyphrase extraction from twitter. In *Proceedings of the 49th Annual Meeting of the Association for Comp. Ling.: Human Language Technologies Vol 1, HLT '11*, 2011.
- [73] D. Zhou, E. Manavoglu, J. Li, C. L. Giles, and H. Zha. Probabilistic models for discovering e-communities. In *Proceedings of the 15th international conference on World Wide Web, WWW '06*, 2006.
- [74] W. Zhou, H. Jin, and Y. Liu. Community discovery and profiling with social messages. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12*, 2012.