The Pennsylvania State University

The Graduate School

Department of Computer Science and Engineering

CREATING A SYNTACTIC DOCUMENT ONTOLOGY

A Thesis in

Computer Science and Engineering

by

Hui Han

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

December 2004

We approve the thesis of Hui Han.

Date of Signature

C. Lee Giles
Professor of Information Sciences and Technology
Professor of Computer Science and Engineering
Thesis Adviser
Chair of Committee

Hongyuan Zha
Associate Professor of Computer Science and Engineering

James Z. Wang
Assistant Professor of Information Sciences and Technology
Assistant Professor of Computer Science and Engineering

Jia Li
Assistant Professor of Department of Statistics
Assistant Professor of Computer Science and Engineering

Raj Acharya
Professor of Computer Science and Engineering
Head of the Department of Computer Science and Engineering

Abstract

An ontology is "a formal explicit specification of a shared conceptualization" [42]. Ontology has been widely studied in recent knowledge representation research, as shown by an increasing number of domain-specific ontologies. With the prevalence of digital libraries, academic documents have become an important part of the information on the web. Document ontologies are gaining increasing importance to the interoperability of heterogeneous digital libraries and the reuse of knowledge embedded in academic documents.

Document ontologies have been constructed from two aspects: the semantic structure and the syntactic structure of documents. The semantic structure specifies what the document is about, i.e. the content of the document; the syntactic structure refers to document structures such as title, author, affiliation, keywords, and citation links between documents.

The following three aspects are critical to creating a domain specific ontology: (1) semi-automatically creating a domain specific ontology, where techniques such as information extraction and data mining can be exploited; (2) maintaining an unambiguous specification of concepts or relationships; and (3) establishing inference rules to allow knowledge reasoning.

This thesis focuses on investigating the first two aspects, as shown by the following three types of work.

- First, we proposed a syntactic document ontology based on the DAML (DARPA Agent Markup Language) ontology library [1] to model the academic documents.

- Second, we developed a Support-Vector-Machines(SVM)-based classification method, for automatic document attributes (metadata) extraction from the header parts of documents and the bibliographic fields. Our method of metadata extraction from document headers achieved better results than using hidden Markov Model (HMMs) on the CMU datasets [90]. We also developed a novel method of parsing individual author names from the line of multiple authors.

- Third, we investigated both supervised and unsupervised learning methods for name entity disambiguation in author citations. We developed two supervised learning methods, one based on a hierarchical naive Bayes model , the other based on the Support Vector Machines. We also developed two unsupervised learning methods, one based on a hierarchical naive Bayes mixture model, the other based on a K-way spectral clustering method with QR decomposition. These methods are applied to 14 name datasets that we constructed based on the publication lists collected from authors' homepages, and the DBLP computer science bibliography. The K-way spectral clustering method with QR decomposition achieved best results, compared to the K-means clustering algorithm and the hierarchical naive Bayes mixture model. The hierarchical-naive-Bayes-model-based method achieved better name disambiguation accuracies than the SVM-based classification method when using only coauthor information. The main reasons are that our hierarchical

[1] http://www.daml.org/ontologies/

naive Bayes model captures the author patterns that are not easily incorporated into feature vector space model that is used by the SVM-based classifcation or the K-way spectral clustering methods. These author patterns are the prior probability of an author, the probability that an author writes a paper alone, and the probabilities that an author writes a future paper with previously unseen coauthors. SVM-based classification method achieved slightly better results than the hierarhical-naive-Bayes-model-based method when using paper title words, publication venue title words, or the combination of all types of citation features.

Table of Contents

List of Tables

List of Figures

Acknowledgments

I am most grateful to my thesis advisor, Professor C. Lee Giles, for the guidance, patience, and encouragement he has given me during my time here at The Pennsylvania State University. I am especially indebted for the financial support that he has provided to me over the years. I am appreciative to Professor Hongyuan Zha, for inspiration and enlightening discussions on a wide variety of topics. I am thankful to Professor James Z. Wang and Professor Jia Li, for their insightful commentary on my work. I wish to thank Dr. Mark Stefik for his valuable comments on our name disambiguation work during my visit to the Palo Alto Research Center.

Chapter 1

Introduction

1.1 Research Background of Ontology

1.1.1 Ontology and ontologies

The word "Ontology" (the first "O" is capitalized) is originated from the Greek philosophers Aristotle and Plato, and was used to describe the existence of being. According to Guarino [43], Ontology refers to a philosophical discipline, while an ontology (the first "o" is small case) specifies the conceptualization in a specific domain. Gruber [42] defines an ontology as the following:

> *An ontology is a formal explicit specification of a shared conceptualization. The conceptualization refers to an abstract model of some phenomenon in the world which identifies the relevant concepts of that phenomenon. Explicit means that the type of concepts used and the constraints on their use are explicitly defined. Formal refers to the fact that the ontology should be machine processible, i.e., the machine should be able to interpret the information provided unambiguously. Shared reflects the idea that an ontology captures consensual knowledge, that is, it is not restricted to some individual, but accepted by a group.*

In a short sentence, an ontology specifies a shared conceptualization. A pragmatic specification of a conceptualization contains two parts: a shared vocabulary and a specification (characterization) of the intended meaning of the vocabulary [43]. An ontology composed of just a vocabulary is of little use, because the arbitrary meanings of a term increases computational complexity that makes agents difficult to share understanding through only the vocabulary. An ontology excludes the unintended meaning of a term to constrain the interpretation of the term.

The goal of an ontology is to facilitate knowledge share. Agents or parties that communicate using a ontology must commit to the ontology by agreeing on

> *using a vocabulary (i.e., ask queries and make assertions) in a way that is consistent (but not complete) with respect to the theory specified by an ontology [42].*

For example, a person understands the text written by another person, because we share a common understanding of the vocabulary used in the text (e.g., English vocabulary). In another word, we share an ontology of the language.

An ontology in computer science accentuates the computational aspects, and is defined [1] as:

> *the attempt to formulate an exhaustive and rigorous conceptual schema within a given domain, a typically hierarchical data structure containing all the relevant entities and their relationships and rules (theorems, regulations) within that domain.*

[1] http://en.wikipedia.org/wiki/Ontology

This thesis uses the computer science definition of an ontology, and defines the ontology as consisting of three components: categories (or classes) of the objects in a domain, relationships among the categories and the objects, and inference rules for discovery of the implicit knowledge. There are three types of relationships that are commonly used in ontologies: taxonomic (generality) relationship, membership (association) relationship, and part-whole (aggregation) relationship. It is hard and may not be necessary to pre-define a central ontology covering all the knowledge in the world. Practically, domain-specific ontologies are developed in a distributed manner [57], and are integrated by using ontology mapping and matching techniques.

1.1.2 Ontologies and Knowledge Bases

Artificial Intelligence (AI) studies ascribing knowledge to machines so that machines could solve complex problems in a human-like way [43, 79]. In tradition, knowledge bases in AI contain human expertise of solving specific tasks, which is task dependent. A group of agents that address different tasks have to communicate in different ways. Previous research [21, 43] propose that knowledge engineering should model the world, not the thoughts of an expert, so that the knowledge ascribed to an intelligent agent is less dependent on a particular way of solving the problem. According to their propositions, the knowledge base contains the objective facts, in stead of experts' expertise of problem solving. An ontology is just such a knowledge base that models the reality and is thus task independent. An ontology provides easier way of interaction among agents that solve different tasks. Agents that communicate using an ontology need to communicate in a consistent manner, but do not need to have a complete knowledge of what

the ontology models [42]. Ontolingua [42] is a system designed for portable ontologies at the Knowledge Systems Lab of Stanford University. This system translates an ontology written in a standard language (such as KIF[38]) to different implemented ontology representation systems, for example, Loom[70], Epikit[39], Algernon[23], and pure KIF. KIF (Knowledge Interchange Format) is a language designed at the Knowledge Systems Lab of Stanford University, with the goal of interchanging knowledge among disparate implemented representation systems.

1.1.3 Web Ontologies

The World Wide Web has been widely recognized for its function of facilitating information share by providing a man-machine interacting mechanism [2]. The Web is written in natural language, thus human understandable and machine processible. However, the web is not machine understandable. With the expectation for knowledge share, researchers forsee the next generation of web as the semantic web, which is designed for machine understanding. The following is a dream of web researchers about the semantic web[13].

[2] http://www.w3c.it/talks/ck2003/slide18-0.htm

"The entertainment system was belting out the Beatles "We Can Work It Out" when the phone rang. When Pete answered, his phone turned the sound down by sending a message to all other local devices that had a volume control. "

...

His sister, Lucy, was on the line from the doctor?s office At the doctor?s office, Lucy instructed her Semantic Web agent through her handheld Web browser. The agent promptly retrieved information about ...?

The semantic web depends on the structured and formal information and information access methods to eventually enable semantic indexing and retrieval. Ontologies are needed for the semantic web to function, because ontologies bring structure to information,such as attributes (descriptive data), classes, sub classes, and relations among entities. Such information structure conveys semantics [13]. Inference rules in ontologies allow knowledge reasoning and help to make the above dream of automation come true. The OWL Web Ontology Language [3] is W3C recommended ontology language, and is designed to formally define the terminology used on web, and thus to facilitate machine interpretability. OWL is developed based on the other two web ontology languages, SHOE (Simple HTML Ontology Extensions)[47], and DAML (DARPA Agent Markup Language) and OIL (Ontology Inference Layer) [4]. These web ontology languages were designed for web page authors to annotate their web pages with formal knowledge representation semantics.

[3] http://www.w3.org/2001/sw/WebOnt/

[4] http://www.daml.org/

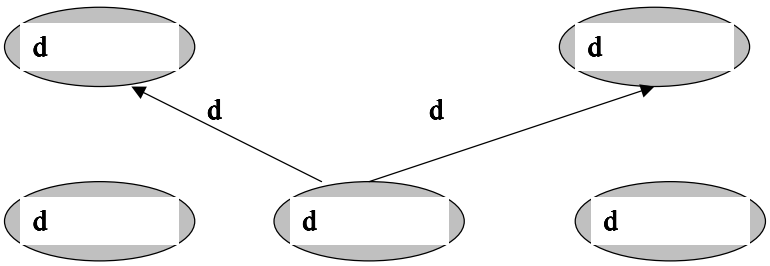


Fig. 1.1. The first level view of a collection of documents.

1.1.4 Document Ontologies

1.1.4.1 Semantic Document Ontology and Syntactic Document Ontology

The "document" in this research refers to academic documents (research papers). With the prevalence of digital libraries, academic documents have become an important part of the information on the Web. How to effectively retrieve relevant information from the sea of digital documents is an important research topic. To effectively exploit the information from documents, we need a classification of documents, relationships among documents, and inference rules to discover hidden information from documents. An ontology is a good tool for these purposes, because an ontology specifies the domain conceptualization usually using three components: classification, relationships and inference rules.

Document ontologies have been constructed from two aspects: the semantic structure and the syntactic structure of documents, and therefore are classified into **semantic document ontologies** and **syntactic document ontologies**.

The semantic document ontology specifies what the document is about, i.e., the ontology of content. For example, the scholarly ontologies (ScholOnto) project [78, 92, 66] models researchers' claim on their work, and relations with other literatures such as supports, challenges, related problems, etc. Srivastava et al. [95] capture the concepts expressed in a document by analyzing the statistical relationships among document terms using LSI (Latent Semantic Indexing). Desmontils et al. extract the taxonomies (lightweight ontology) from web pages to build a structured index of a web site[28]. Automatically constructing semantic document ontologies require a computer to understand the content of the documents written in natural language, which is a challenging task currently.

The syntactic document ontology specifies the syntactic structure of documents. All documents are organized following a set of rules, which we call the syntax of documents. For example, syntactically, a document can have a title, author name(s), author affiliations and addresses, an abstract, a bibliography and a set of sections. It is feasible for a computer to process the syntactic document structure accurately. Document ontologies defined by SHOE (Simple HTML Ontology Extensions) [47] and DAML (DARPA Agent Markup Language) [4] are constructed based on the syntactic structure of documents. The document metadata such as Dublin core [106] provides a simple description on the same document structure.

Ontologies structure information [47]. Document syntactic structure reveals document semantics in certain degree. We view a collection of documents in two levels. At first level (shown by Figure 1.1), we regard a document as an unstructured sequence of words. Simple relationships of references exist among documents. This level is what

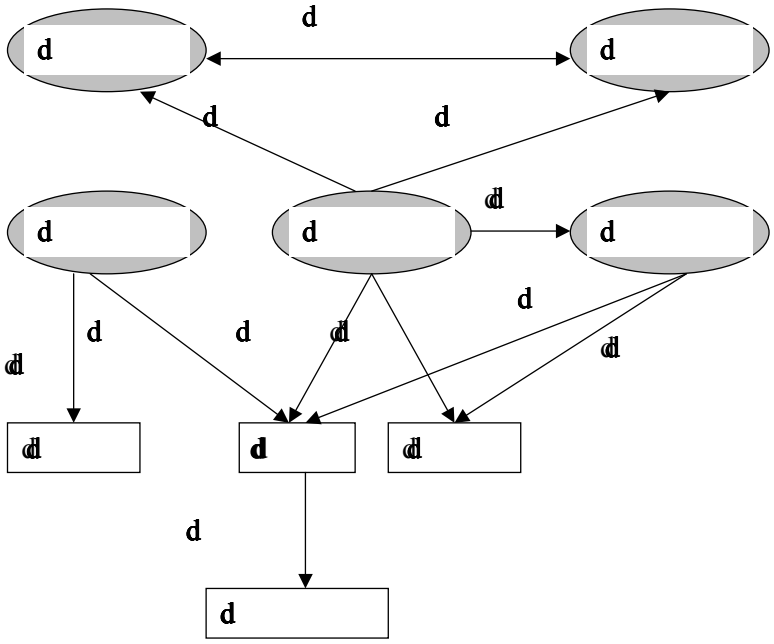


Fig. 1.2. The second level view of a collection of documents.

current research work focuses on. At this level, we have limited information about documents. The second level of view (shown by Figure 1.2) is an ontological view of a collection of documents. Each document has an internal structure, represented by a set of document attributes such as title, author, abstract, bibliography and a set of sections. Each document attribute reveals a unique aspect of the documents and helps with information display, identification, search and classification. For example, the citation links help to understand the information flow within the literature collections; the coauthor relationship indicates different research communities in different areas [55]. Different types of relationships exist among documents. Besides reference relationships, we can model if two documents are the same, related in research topics, or similar in sentence

level of writing. Inference rules are defined based on the classification and relationships, so that implicit information existing among documents can be discovered. For example, the change of publication numbers and publication venues in an institution over a time period may indicate the research development in that institution. The ontological view of documents reveals rich information among a collection of documents.

1.1.4.2 Syntactic Document Ontologies, Information Retrieval and Digital Library

Different from the classical way of indexing by keywords, syntactic document ontologies allow indexing by lexical concepts and entities [71, 103, 57]. Previous work [52, 5] shows that using domain specific semantic knowledge improves document retrieval and classification performance significantly. Keyword matching has limitations in retrieving related information, mainly because of word sense ambiguity. Documents that contain matched query words may be incorrectly retrieved when the matched words of the documents have different meanings from the query words. Relevant information expressed by the synonyms of the query keywords may be missed. For example, when retrieving the homepage(s) of a person by feeding the person's name to a general search engine, the search engine may return other web pages that contain the query name, e.g., a web page of the query person's talk slides. Syntactic document ontologies allow semantic match, that is, retrieving information that matches user query semantically. Ontologies explicitly model the category of each webpage, thus allowing advanced semantic query

with restrictions (e.g., document types). Compared to keyword matching, using syntactic document ontologies can improve the accuracy of document retrieval, and easily address the homepage finding task mentioned above.

Document attributes that are modeled by ontologies explicitly provide multiple search indexes to documents. Previous work [40, 65, 107, 14] uses limited numbers of the above indexes for different retrieval tasks. Combining multiple document attributes may improve performance of retrieving information from documents.

Syntactic document ontologies are important to the interoperability of heterogeneous digital libraries. Modeling documents that are from heterogeneous digital libraries using the same ontology can help provide uniform retrieval interface to end users. The document syntactic structure gives documents an innate canonical form. "Canonical form" means the simplest and most basic form to represent a document unambiguously [5]. The canonical form of documents helps to identify and gather duplicates of documents that are in different formats.

An open formal knowledge representation system that encodes the extracted information is critical to the reuse of resources [57]. Syntactic document ontologies allow reuse of the knowledge embedded in documents. For example, syntactic document ontologies can help to maintain an extensive knowledge base that contains name entities of general importance (e.g., researchers, organizations).

1.2 Contributions in This Thesis

The contributions in this thesis are summarized as follows:

[5] http://c2.com/cgi/wiki?CanonicalForm

- Created a syntactic document ontology based on the DAML[6] ontology library.
- Developed a Support-Vector-Machines(SVM)-based classification method for automatic document attributes (metadata) extraction from the header part of the document [45].
- Developed a word clustering method based on domain databases and word typographic properties [44].
- Developed a method of parsing individual authors from a line of multiple authors based on SVM classification.
- Applied the hidden Markov models for bibliographic field extraction.
- Developed both supervised and unsupervised learning methods for name disambiguation in author citations. The supervised learning methods [46] includes a hierarchical-naive-Bayes-model-based approach, and a SVM-based approach. The unsupervised learning methods includes a hierarchical-naive-Bayes-mixture-model-based approach, and a K-way-spectral-clustering-based approach.

1.3 Organization of The Thesis

The thesis is structured in three parts. The first part (Chapter2) presents the specification of our syntactic document ontology. The second part (Chapter 3) presents our methods of automatic document attributes extraction. The third part (Chapter 4) presents our methods of name entity disambiguation in author citations.

[6] http://www.daml.org/ontologies/

- **A Syntactic Document Ontology (Chapter 2)**. This chapter presents the specification of a syntactic document ontology created based on the DAML ontology library. The ontology specifies the necessary components to model academic documents, proposes the concepts of canonical documents and author names.

- **Automatic Document Attributes Extraction (Chapter 3)**. This chapter presents a Support-Vector-Machines(SVM)-based classification method for automatic document attributes extraction (e.g., title, authors, author affiliations, and publication year) from the header part of the documents. This chapter also presents the work of bibliographic fields extraction using hidden Markov models, and the word clustering method based on domain databases and word orthographic properties.

- **Name Entity Disambiguation in Author Citations(Chapter 4)**. This chapter presents both supervised and unsupervised learning methods of name entity disambiguation in author citations. The supervised learning methods include one based on a hierarchical naive Bayes model, and the other based on the Support Vector Machines. The unsupervised learning methods include one based on a hierarchical naive Bayes mixture model, and the other based on a spectral clustering method.

Chapter 2

A Syntactic Document Ontology

An ontology is composed of three components: categories, relationships, and inference rules. A document ontology is an ontology on the classification of documents, the relations among documents, and the inference rules on the documents and their relationships. A document ontology can be either semantic or syntactic. The semantic ontology specifies what a document is about, i.e., the content of the document. The syntactic ontology, however, specifies the syntactical structures of a document. For example, syntactically, a document has the title, authors, the author affiliations and addresses, the publication date, and the publication venue, etc. The syntactic ontology does not involve the understanding of the semantic content of the document, which makes it suitable for automatic machine processing. In the following section, we discuss the components of a syntactic document ontology.

2.1 Document

A document in this research refers to the academic documents (research papers). An academic document usually has a set of attributes, such as title, author, publication venue, publication year, etc. Each attribute can be either literal (e.g., publication date) or an object with its own attributes (e.g., the author with information of affiliation, address and SSN).

We define the following attributes for an academic document in our syntactic document ontology. Some attributes are imported from other ontologies, mainly from the SHOE ontology and the DAML ontology. Table 2.1 describes the imported ontologies "person", "organization", and "bibliography". We use the prefix such as"daml:doc" to mark the ontology we import; prefix "base" refers to the base ontology we import from http://www.cs.umd.edu/projects/plus/SHOE/onts/base.html. Besides, we define new document attributes as marked by bold fonts below, such as "figure", "equation", "acknowledgment", "citationContext", etc, as shown in Table 2.2. Each instance is unambiguously identified by a "uri(u niform resource identifier)".

- The following are the name spaces of the ontologies we import:
  - daml:doc – http://www.cs.umd.edu/projects/plus/DAML/onts/docmnt1.0.daml
  - daml:dc – http://orlando.drc.com/daml/ontology/DC/current/
  - daml:person – http://orlando.drc.com/daml/ontology/Person/current
  - daml:organization – http://orlando.drc.com/daml/Ontology/Organization/3.1/ Organization-ont.daml
  - daml:bibliography – http://www.cs.yale.edu/ dvm/daml/bib-ont.daml
- The attributes of the syntactic document ontology:
  - daml:doc:title
  - daml:doc:subject
  - **author(document, daml:person)**
  - **authorOrg(document, daml:organization)**

  - daml:doc:publishDate
  - **publisher(document, daml:organization)**
  - daml:doc:volume
  - **pageNum(document, base.NUM)**
  - daml:bibliography
  - **figure**
  - **equation**
  - **acknowledgment**
  - **citationContext**
  - daml:doc:containedIn
  - daml:dc:source

2.2 Category

We focus on the following categories of academic documents in the ISA hierarchy, which are defined by the SHOE ontology.

- BookArticle
  - ConferencePaper
  - JournalArticle
  - WorkshopPaper
- TechnicalReport

daml:person	daml:organization	daml:bibliography
alias	organizationName	author
displayName	organizationAlias	booktitle
firstName	hasType	editor
gender	hasMember	institution
hasAddress	hasAddress	journal
hasEmail	hasTelephoneNumber	note
hasTelephoneNumber	hasEmail	number
identficationNumber	partOf	pages
lastName	subOrganization	publisher
middleName		title
nickname		volume
ssn		year
title		

Table 2.1. The attributes of the imported ontologies "person", "organization" and "bibliography".

figure		equation	Acknowledgment	citationContext	sentence
category	attribute	attribute	attribute	attribute	attribute
table	caption	variable	daml person	sentence	text
figure	header		daml:organization	document	
	variable		projectNO		

Table 2.2. Self defined document ontologies

- Thesis
  - DoctoralThesis
  - MastersThesis

2.3 Relationship

The most explicit relationships among documents are "reference" and "isReferencedBy" as shown below. It is necessary to map a reference to a document in order to establish such relationships.

- reference(SHOE:document, SHOE:document)
- IsReferencedBy(SHOE:document, SHOE:document)

We also observe the implicit and useful relationships among documents such as "similiarTo" and "related" [40]. Besides using text similarity, an alternative way exploits the document attributes to determine the document similarity or relatedness. For example, two documents written by the same author are likely to be similar or related in topic. Since there may be different criteria to measure the similarity and relatedness, the criteria needs to be specified explicitly in the relationship. Another necessary relationship we propose is "sameAs", which links documents with the same canonical form. This helps identify duplicate documents collected in different digital libraries, or the same document in different forms. We specify such relationships below without going into implementation details.

- similarTo(document, document, criteria)
- related(document, document, criteria)
- sameAs(document, document)

In addition, we define the coAuthor relationship to help construct the coAuthor relationship network. The coAuthor relationship reveals the social network [55], and

may facilitate interesting discoveries of researcher collaboration groups or collaboration patterns.

- coAuthor(person, person, document)

2.3.1 Inference Rules

Inference rules are not defined here.

2.4 Canonical Document and Canonical Name

2.4.1 The Canonical Document

Because of the variety in the formats and storage sources, and the typographical errors, the duplicates of a same document may appear in different forms in multiple data collections. Such phenomenon may cause significant inconvenience in scientific data collection, database integration and management.

As the *unambiguous* specification of a concept is a fundamental property of the ontology, the concept of *canonical document* is a necessary component of a document ontology. A related research issue is how to determine the canonical form of the documents. One possible solution borrows the idea of "authorized name" from the digital library practice, as introduced in next section. This solution, however, usually needs significant amount of manual effort on defining the canonical form of a named entity. Another solution could use combinations of the canonical document attributes to determine the canonical form of a document, such as the canonical author, the canonical title,

the canonical publication date, the canonical acknowledgment, the canonical affiliation, the canonical figure, and the canonical equation.

Establishing the canonical form of a document helps to manage the duplicates of a document that are distributed in different digital libraries. The canonical forms of a document and a citation also helps establish a mapping between the document and its citation format. Such mapping facilitates populating the "reference", "IsReferencedBy", and "sameAs" relationships as specified in Chapter 2.3

2.4.2 The Canonical Name

We define a canonical name as the minimal name that is invariant and complete for name entity disambiguation. A canonical name may have more than just the name of the individual as constituents. An example of a canonical name would be a name that has all the characteristics of a name entity including abbreviations and AKA's.

Proposing the canonical name is to solve the confusion caused by ambiguous names. For example, an author may appear in different publications in various name forms, and different authors may share the identical name. Current research work on canonical names focuses on two aspects. One aspect is determining the consistent name representation (canonical form) of a named entity [41]. The other aspect is determining if the names from different information resources refer to a specific named entity [104, 30, 46]. This thesis focuses on the second aspect of the work, and proposes various machine learning methods that automatically disambiguate names in author citations.

Chapter 3

Automatic Document Attributes Extraction

3.1 Introduction and Related Work

Document attributes are also called document metadata, which describes the syntactic structure of a document. Douments here refer to research papers. Most of the directly indexable information (e.g., authors' names, affiliations, addresses, and the title of the paper) are gathered in the header of a research paper. The header [90] consists of all the words from the beginning of the paper up to either the first section, usually the introduction, or to the end of the first page, whichever occurs first. Document syntactic structure is also reflected in bibliographic fields.

Metadata is critical for the interoperability of Digital Libraries (DL) [72, 81], in that metadata facilitates the discovery of content stored in distributed archives [27, 69]. The digital library CITIDEL (Computing and Information Technology Interactive Digital Educational Library), part of NSDL (National Science Digital Library), uses OAI-PMH (Open Archive Initiatives Protocols for Metadata Harvesting) to harvest metadata from all applicable repositories and provides integrated access and links across related collections [58]. Support for the Dublin Core (DC) metadata standard [105] is a requirement for the OAI-PMH compliant archives, while other metadata formats optionally can be transmitted.

However, providing metadata is the responsibility of each data provider with the quality of the metadata a significant problem. Many data providers [56, 18] have had significant harvesting problems with XML syntax and encoding issues, even leading to unavailability of service [69]. In fact, some digital libraries have no metadata to harvest (some search engines have little or no metadata), or metadata that is not OAI compliant, e.g., CiteSeer [63]. Non-compliant metadata must be either automatically wrapped to work with the OAI protocol, or manually encoded. Building tools for automatic document metadata extraction and representation will therefore significantly improve the amount of metadata available, the quality of metadata extracted, and the efficiency and speed of the metadata extraction process.

Methods used for automatic metadata extraction include regular expressions, rule-based parsers, and machine learning. In general machine learning methods are robust and adaptable and, theoretically, can be used on any document set. Generating labeled training data is the rather expensive price that has to be paid for supervised learning systems. Although regular expressions and rule-based systems do not require any training and are straightforward to implement, their dependence on the application domain and the need for an expert to set the rules or regular expressions causes these methods to have limited use. Machine learning techniques for information extraction include symbolic learning, inductive logic programming, grammar induction, Support Vector Machines, hidden Markov models, and statistical methods. Hidden Markov models (HMMs) are the widely used generative learning method for representing and extracting information from sequential data. However, HMMs are based on the assumption that features of

the model they represent are not independent from each other. Thus, HMMs have difficulty exploiting regularities of a semi-structured real system. Maximum entropy based Markov models [73] and conditional random fields [62] have been introduced to deal with the problem of independent features.

This chapter discusses two machine learning methods, the Support Vector Machines (SVM), for automatic metadata extraction from the document header part, and the hidden Markov Models (HMM), for automatic bibliographic field extraction. The SVM-based extraction method is the focus of this chapter.

Recent work by Chieu [20] suggests that the information extraction task can also be addressed as a classification problem. Encouraged by their success in handling high dimensional feature spaces for classification problems [51, 34], we investigated Support Vector Machines (SVMs) for metadata extraction from the document headers. Related work of using SVMs includes Kudoh et al. using the SVM method for chunk identification, Mcnamee et al. using a SVM for named entity extraction [76, 61, 98], and Pasula et al. using relational probability models to solve identity uncertainty problems [82].

The SVM-based metadata extraction algorithm combines line classification and chunk identitication, and has the function of extracting individual names from a list of authors. This algorithm includes a new feature extraction method and an iterative line classification process using contextual information. In the experimental results section we illustrate the dominance of the introduced SVM-based metadata extraction algorithm over the well-known HMM based systems [90] for metadata extraction from document headers.

The remainder of the chapter is organized as follows: section 3.2 describes the problem and dataset; section 3.3 presents our metadata extraction method, together with the cross validation results on 500 training headers; section 3.4 presents the experiment result of our metadata extraction algorithm on the test dataset; section 3.5 presents the method and experimental results of bibliographic field extraction using hidden Markov models; section 3.6 concludes and discusses future work.

3.2 Problem Definition and Dataset

The Dublin Core has been widely used as a metadata standard and defines 15 elements for resource description: Title, Creator, Subject, Description, Contributor, Publisher, Date, Type, Format, Identifier, Source, Relation, References, Is Referenced By, Language, Rights and Coverage. However, this is only a basic set of metadata elements and is used by OAI-PMH for "minimal" interoperability. Extending document metadata through information on both authors (such as affiliation, address, and email), and documents (such as publication number and thesis type), would provide greater representation power. It would also help in building unified services for heterogeneous digital libraries, while at the same time enabling sophisticated querying of the databases and facilitating construction of the semantic web [12]. Seymore et al. defined 15 different tags for the document header [90] to populate the Cora search engine [74], 4 of which are the same as those in the Dublin Core. Two of the remaining tags, introduction and end of page, are functional rather than informative, indicating the end of the header. Leaving out the functional tags, we adopt their format as extended metatags for research papers. We further propose to define affiliation as part of the address, instead of an exclusive tag.

Extended Metatag	**DC Element**	**Explanation**
Title	Title	Title of the paper
Author	Creator	The name(s) of the author(s) of the document
Affiliation		Author's affiliation
Address		Author's address
Note		Phrases about acknowledgment, copyright, notices, and citations
Email		Author's email address
Date		Publication date
Abstract Introduction	Description	An account of the content Introduction part in the paper
Phone		Author's phone number
Keyword	Subject	The topic of the content of the document
Web		URL of Author's webpage of the document
Degree		Language associated with thesis degree
Pubnum		Publication number of the document
Page		The end of the page

Table 3.1. Extended metatags and their mapping to Dublin Core metadata elements

Table 3.1 is a short explanation of the extended metatags and the mapping to Dublin Core metadata elements. These metatags mark the document attributes from the header part of documents.

Figure 3.1 is an example of meta-tagged document header. Document metadata(attributes) extraction can also be viewed as labeling the text with the corresponding metatags. Each metatag corresponds to a class. Lines 22 and 25 are multi-class lines containing chunks of information from multiple classes. We define a chunk of information as consecutive words that belong to the same class. Line 22 and 25 contain the chunks of

1:<title> Stochastic Interaction and Linear Logic +L+ </title>
2:<author> Patrick D. Lincoln John C. Mitchell Andre Scedrov +L+ </author>
3: <abstract> Abstract +L+
4:We present stochastic interactive semantics for prepositional linear +L+
...
22:<email> jcm@cs.stanford.edu </email> <web> http://theory.stanford.edu/people/jcm/home.html </web> <affiliation> Department of Computer Science, Stanford University, </affiliation> <address> Stanford, CA 94305.</address> <note> Supported in part +L+
23:by an NSF PYI Award, matching funds from Digital Equipment Corporation, the Pow-ell Foundation, and Xerox Corporation; and the Wallace F. and Lucille M. Davis Faculty +L+
24:Scholarship. +L+ </note>
25:<email> andre@cis.upenn.edu </email> <web>http://www.cis.upenn.edu/∼andre </web> <affiliation> Department of Mathematics, University of Pennsylvania, </affiliation> <address> Philadelphia, PA 19104-6395. </address> <note> Partially supported by +L+
26:NSF Grants CCR-91-02753 and CCR-94-00907 and by ONR Grant N00014-92-J-1916. Sce-drov is an American Mathematical Society Centennial Research Fellow. +L+ </note>

Fig. 3.1. A labeled document header and metadata. Each line starts with the line number.

5 classes: email, web, affiliation, address, and note. All other lines contain information belonging to one class only and are therefore called single-class lines.

We use the labeled dataset provided by Seymore et al. [90] to test our method of metadata extraction. The dataset contains 935 headers of computer science research papers, with 500 of those belonging to the training set and the remaining 435 headers belonging to the test set. The training set includes a total of 10025 lines and 23557 word tokens whereas there are 8904 lines and 20308 word tokens in the test set. These headers are text files converted from the pdf and ps files. Each line ends with a carriage return and the line break marks $+L+$ are provided by the dataset for identification.

The document headers are semi-structured. We observe that among total 10025 lines from 500 training headers, the majority (9775 lines, 97.51%) are single-class lines and only 250 (2.49%) lines are multi-class lines. Even after removing the abstract section which is mostly single-class lines, multi-class lines still account for only 4.98% of all lines. Classifying each line into one or more classes thus appears to be more efficient for meta-tagging than classifying each word. Table 3.2 lists the class distributions of the lines from the 500 training headers.

The predicted tags for previous and next lines are also good indicators of the class(es) to which a line belongs. For instance, an abstract has consecutive lines uninterrupted by lines of other classes, and title lines usually come before author lines. Making use of such contextual information among lines increased the line classification performance in our experiments introduced below.

We propose a third algorithm for processing the lines predicted to contain chunks of information from multiple classes. Since each chunk has consecutive words, we consider extracting metadata from the multi-class lines as the problem of seeking the optimal chunk boundaries. Recognition of individual author names within multi-author lines can also be considered as the problem of seeking the right chunk boundary, in this case between the author names. For example, does the line "Chungki Lee James E. Burns" refer to two authors "Chungki Lee" and "James E. Burns," two authors "Chungki Lee James" and "E. Burns," or one author "Chungki Lee James E. Burns"?

Based on the structural patterns of the document headers, we decompose the metadata extraction problem into two sub-problems – (1) line classification and (2) chunk identification of multi-class and multi-author lines. Accurate line classification

Class No.	Class Name	Number of Lines	Percentage
1	Title	832	8.3%
2	Author	724	7.2%
3	Affiliation	1065	10.6%
4	Address	629	6.3%
5	Note	526	5.2%
6	Email	336	3.4%
7	Date	182	1.8%
8	Abstract	5007	50.0%
9	Introduction	326	3.3%
10	Phone	61	0.6%
11	Keyword	142	1.4%
12	Web	38	0.4%
13	Degree	169	1.7%
14	Pubnum	116	1.1%
15	Page	166	1.7%

Table 3.2. Class distribution among 10025 total lines from 500 training header

is a critical step, since it directly affects the performance of the chunk identification module.

3.3 Metadata Extraction Algorithm

This section describes two important aspects of our work, SVM classification and feature extraction. The metadata extraction algorithm is discussed in detail, together with the corresponding ten-fold cross-validation results on the 500 training headers. Performance is evaluated using accuracy, precision, recall, and F measure.

3.3.1 Support Vector Machine Classification

Support Vector Machine is well known for its generalization performance and ability in handling high dimensional data. Consider a two class classification problem. Let $\{(x_1, y_1), \ldots, (x_N, y_N)\}$ be a two-class training dataset, with x_i a training feature

vector and their labels $y_i \in$ {-1, +1}. The SVM attempts to find an optimal separating hyperplane to maximally separate two classes of training samples. The corresponding decision function is called a classifier. The kernel function of an SVM is written as $K(x_a, x_b)$ and it can be an inner product, Gaussian, polynomial, or any other function that obeys Mercer's condition [102, 24].

In the simplest case, where $K(\vec{x}_a, \vec{x}_b) = \vec{x}_a \cdot \vec{x}_b$ and the training data is linearly separable, computing an SVM for the data corresponds to minimizing $\|\vec{w}\|$ such that

$$y_i(\vec{w} \cdot \vec{x}_i - w_0) - 1 \geq 0, \forall i$$

Thus SVM yields the lowest complexity linear classifier that correctly classifies all data. The solution for $\vec{w}$ in the linear case is

$$\vec{w} = \sum_{i=1}^{N} y_i \lambda_i \vec{x}_i$$

The formalism behind SVMs has been generalized to accommodate nonlinear kernel functions and slack variables for mis-classifications. We write the output of a nonlinear SVM as:

$$\begin{aligned} f(x, \vec{\lambda}) &= \sum_{i=1}^{N} y_i \vec{\lambda} \Phi(\vec{x}_i) \cdot \Phi(x) + \lambda_0 \\ &= \sum_{i=1}^{N} y_i \vec{\lambda} K(\vec{x}_i, \vec{x}_j) + \lambda_0 \end{aligned} \tag{3.1}$$

Thus, $K(\cdot)$ is a dot product in a nonlinear feature space Φ. The objective function for Equation 3.1 is:

$$E(\vec{\lambda}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \lambda_i \lambda_j K(\vec{x}_i, \vec{x}_j) - \sum_{i=1}^{N} \lambda_i, \tag{3.2}$$

subject to the box constraint $\forall_i, 0 \leq \lambda_i \leq C$ and the linear constraint $\sum_i y_i \lambda_i = 0$. C is a user-defined constant that represents a balance between the model complexity and the approximation error; in the Lagrangian, C is multiplied by the sum of the magnitude of the slack variables used for absorbing miss-classifications. When Equation 3.2 is minimal, Equation 3.1 will have a classification margin that is maximized for the training set [36]. In our experiments we use a Gaussian kernel function of the form

$$K(x_i, x_j) = \exp(\frac{-\|x_i - x_j\|^2}{\sigma^2}) \tag{3.3}$$

Our experiments are based on the software SVM^{light} [50]. We set the parameter gamma (-g), the spread of the Gaussian kernel as 0.1, and all other parameters set by SVM^{light}. We extend the SVM to multi-class classifiers in the "One class versus all others" approach, i.e., one class is positive and the remaining classes are negative.

3.3.2 Feature Extraction

Most of the previous work on information extraction uses word-specific feature representations [90, 61, 98]. Recent research on the topic suggests that line-specific features could also be useful [73].

We make use of both **word** and **line**-specific features to represent our data. Each line is represented by a set of word and line-specific features.

We design a rule-based, context-dependent word clustering method explained below for word-specific feature generation, with the rules extracted from various domain databases and text orthographic properties of words (e.g. capitalization) [89]. Word clustering methods group similar words and replace the original words by their cluster labels as the features for text classfication tasks. Distributional clustering methods have shown significant dimensionality reduction and accuracy improvement in text classification [8, 94, 29]. While distributional clustering needs to use labeled training data, our rule-based method relies on the prior knowledge embedded in domain databases.

We collect the following databases to gather apriori knowledge of the domain:

- Standard on-line dictionary of Linux system
- Bob Baldwin's collection of 8441 first names and 19613 last names
- Chinese last names
- USA state names and Canada province names
- USA city names
- Country names from the World Fact Book [3], and
- Month names and their abbreviations

We also construct domain databases, i.e., word lists from training data for classes: affiliation, address, degree, pubnum, note, abstract, keyword, introduction, and phone. Words and bigrams that appear frequently in the lines of each class are selected to enter

these word lists. Frequency thresholding is used to define the list size [110]. The abstract class word list contains one word "abstract" and the affiliation class list contains words shown in Table 3.3.

We then cluster words and bigrams based on their membership in the domain databases and their text orthographic properties. Words and bigrams in the same cluster are represented by a common feature (cluster label), which we call the word-specific feature. For example, an author line *"Chungki Lee James E. Burns"* is represented as *"Cap1NonDictWord: :NameWord: :NameWord: :SingleCap: :NameWord:"*, after word clustering. The original author words "James," "E.," "Burns" has little discriminatory power for classifying the line as class "Author". However, the line of features after conversion :NameWord: :SingleCap: :NameWord: is a good indicator of its true class.

The complete list of features used in the experiments is displayed in Table 3.4. Converting words into features follows a specific-to-general priority order. For example, a word appearing in both the name dictionary and the word dictionary is converted to :NameWord:. It is difficult to assign classes when the overlapping features are word lists of classes "degree", "pubnum", "note", and "affiliation". To avoid information loss, we encode such overlapping features using a vector of four digits. For example, vector `1001` means that the word is contained in both "degree" and "affiliation" domain word lists.

Word-specific feature extraction reduces the dimensionality of the feature space and helps with the sparse data problem often encountered with text. Such word clustering shows significant improvement in our experiments of classifying lines [44]. A reason is that the word cluster statistics give a more robust estimate than the original sparse word statistics [8, 94].

The feature vector of a line is constructed based on the word-specific features and the line-specific features of the line. The weight of a word-specific feature in the line feature vector is the number of times this feature appears in the line.

The following is the list of line-specific features used. In particular, feature ClinePos is found to be very important in correct classification of title lines.

CsenLen Number of tokens [1] a line contains.

ClinePos The position of a line, i.e., line number.

CDictWordNumPer The percentage of dictionary words among all tokens of a line.

CNonDictWordNumPer The percentage of non-dictionary words among all tokens of a line.

CCap1DictWordNumPer The percentage of dictionary words with first letter capitalized among all tokens of a line.

CCap1NonDictWordNumPer The percentage of non-dictionary words with first letter capitalized among all tokens of a line.

CdigitNumPer The percentage of numbers among all tokens of a line.

Line-specific features include features that represent percentages of different types of class-specific words in a line. "CaffiNumPer" refers to the percentage of the affiliation words among all tokens of the line. "CaddrNumPer", "CdateNumPer", "CdegreeNumPer", "CphoneNumPer", "CpubNumPer", "CnoteNumPer", and "CpageNumPer" are the features that represent percentages of address words, date words, degree words,

[1] A token can be a word or a number

DF Value	Word	DF Value	Word
325	University	37	Laboratory
221	Department	34	Technology
111	Univ	33	Dept
77	Institute	27	Systems
47	Research	26	School
39	Sciences	26	Center

Table 3.3. Affiliation class word list

phone words, publication number words, note words, and page number words among all tokens in a line respectively. The weight of a line-specific feature is its percentage value.

Our experiments show that SVM doesn't handle well the case when different features have very different ranges of values. For example, the feature "CsenLen" could have a weight of 40, while the line-specific feature **CdictWordNumPer** weight is over the range [0, 1]. Features with large scale may dominate the features with small weight. Therefore, we normalize the weight of a feature by dividing the maximal weight of the feature in a line, which is also call the $||X||_\infty$ normalization. Such normalization increases classification performace as shown in the next section. Another way of normalizing features of different scales could be normalizing each individual feature vector to unit (Euclidean) length. This method helps to abstract from different document lengths.

3.3.3 Line Classification Algorithms

The following is a two-step algorithm for classifying text lines into a single class or multiple classes. The two components are a independent line classification followed by a contextual line classification.

Feature	Explanation
:email:	using regular expression match
:url:	using regular expression match
:singleCap:	a capital letter like M or M.
:postcode:	such as PA, MI
:abstract:	abstract
:keyword:	key word, key words, keyword, keywords
:intro:	introduction
:phone:	tel, fax, telephone
:month:	a word in the month list
:prep:	at, in, of
:degree:	a word or bigram in the degree domain word list
:pubnum:	a word or bigram in the publication number domain word list
:notenum:	a word or bigram in the note domain word list
:affi:	a word or bigram in the affiliation domain word list
:addr:	a word or bigram in the address domain word list
:city:	a word or bigram in the city name list
:state:	a word or bigram in the state name list
:country:	a word or bigram in the country name list
:NameWord:	a word in one of the 3 name lists
:Cap1DictWord:	a dictionary word with first letter capitalized
:DictWord:	small case dictionary word
:NonDictWord:	small case non dictionary word
:Dig[3]:	a number of three digits
The word-specific feature considers text orthographic properties, e.g., BU-cs-93 is converted to :CapWord2-LowerWord2-Digs2:	

Table 3.4. Word-specific feature set

3.3.3.1 Independent Line Classification

At the first step, feature vectors are generated based on the feature extraction methods described in the previous section. After removing features that appear in less than 3 lines, we get feature vectors of 1100 dimensions on average for ten-fold cross

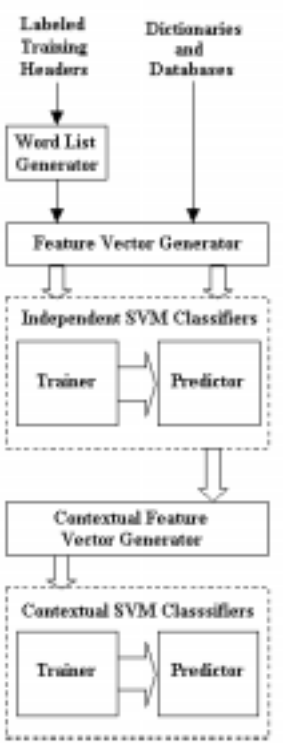


Fig. 3.2. Overview of the line classification training module.

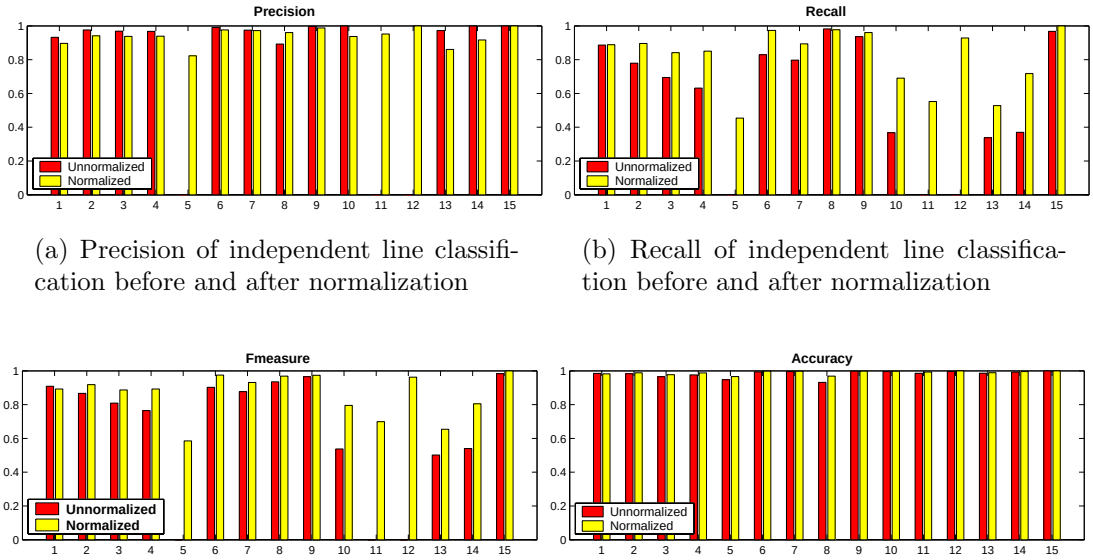
validation. A feature vector is labeled as class C if the corresponding line contains words belonging to class C. Training feature vector set for class C is generated by collecting all the feature vectors with label C as positive samples and all the rest as negative; the same procedure applies to all classes. Note that a feature vector could have multiple labels and thus can belong to multiple training feature vector sets. 15 classifiers are then trained on the 15 labeled feature vector sets. Test lines are classified into one or more classes if their feature vectors are scored positive by the corresponding classifier. This

process is called independent line classification (also shown in Figure 3.2), since each line is classified independently.

Table 3.5 lists the ten-fold cross-validation results on the training dataset for the independent line classification algorithm. Figure 3.3 shows the Precision, Recall, Accuracy and F measure of independent line classification before and after normalization using ten-fold cross-validation on 500 training headers. The effect of normalization is a significant improvement in performance. Normalization is especially important in identifying the rare classes, such as class 5 (note), 11 (keywords), and 12 (web). Consider class 5 "note" as an example, the positive note samples occupy 5.3% (53 out of 1001.5 averaged for each fold of ten-fold cross validation) of all test samples. Without normalization, the note classifier classifies all testing samples into non-"note" classes. Thus, the recall for class 5 "note" is zero and the precision is infinite. Normalization appears to increase the importance of features in the class "note", which then enhances "note" samples for the "note" classifier.

3.3.3.2 Iterative Contextual Line Classification

The second step makes use of the sequential information among lines as discussed in section 3.2 to improve the classification of each line. We encode the class labels of N lines before and after the current line L as binary features and concatenate them to the feature vector of line L that is formed in the independent line classification. A contextual line classifier for each metatag is then trained based on these labeled feature vectors with additional contextual information. Line feature vectors for testing are extended the same way. Their neighbor lines' class labels are those predicted by the independent



(a) Precision of independent line classification before and after normalization

(b) Recall of independent line classification before and after normalization

(c) F Measure of independent line classification before and after normalization

(d) Accuracy of independent line classification before and after normalization

Fig. 3.3. A set of performance of independent line classification before and after normalization

line classifier. Test lines are then reclassified into one or more classes by the contextual line classifiers. This contextual line classification is repeated such that in each iteration, the feature vector of each line is extended by incorporating the neighbor lines' class label information predicted in the previous iteration. The procedure converges when the percentage of lines with new class labels is lower than a threshold. The threshold value is set to 0.7% in our experiments, and N is chosen to be 5. Ramshaw et al. show the positive effect of a similar iterative algorithm on transformation-based learning for rule-selection [88].

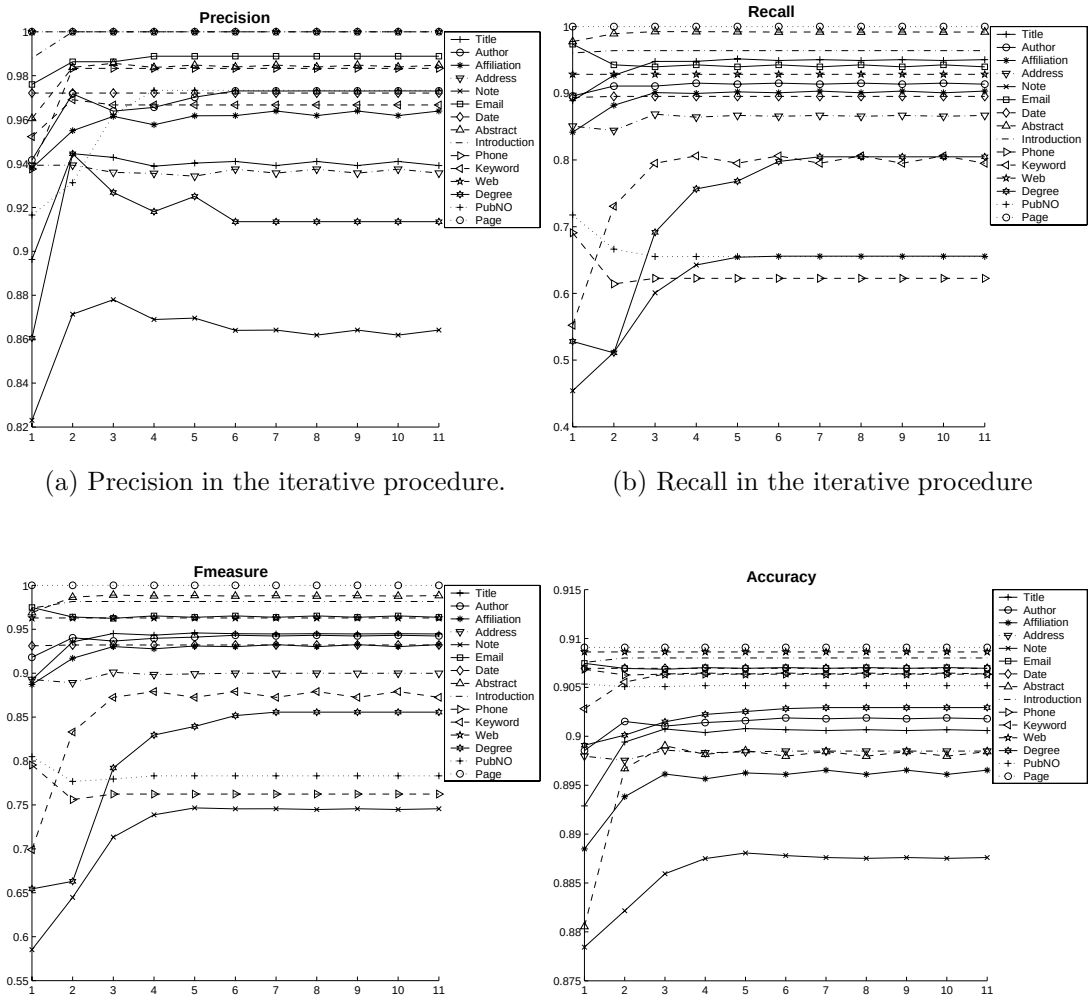
We call lines that are classified to be negative by all of the 15 classifiers as orphan lines. In our ten-fold cross validation on the training dataset, on average, 55.4 out of 923 lines (6.0%) in each fold were orphan lines. These lines may correspond to the points in the feature space that are close to the separating hyperplane or are within the margins. These points are easily disturbed by noise. We hypothesize that the separating hyperplane closest to the point may indicate the true class of the corresponding line. We experimented with three methods to calculate the closest hyperplane of the orphan lines and the best one achieved 44.64% accuracy based on ten-fold cross validation on training data. We define V_i as the classification score of the orphan line for classifier i; and Neg_i as the mean of the negative classification score for the classifier i on other test feature vectors. The best performing method for orphan line classification considers the line's relative distance to the hyperplane. That is, an orphan line is assigned to class i if $i = argmin(V_i/Neg_i)$.

The contextual information we use for line classification is encoded by the binary features P_{ij} if the previous ith closest line belongs to class j and N_{ij} if the next ith closest line belongs to class j, with $i \in \{1, .., 5\}$ and $j \in \{1, .., 15\}$.

We found that choosing P_{ij} and N_{ij} to be {1, 0.5}, instead of {1, 0} achieves better line classification performance, based on the experiment on the training dataset. This is because the line feature values are already normalized into the range [0, 1]. Choosing the midpoint of this range as the weight for up to 150 (=15∗10) contextual features is a type of normalization and is found to be more effective.

Figure 3.4 shows the performance evaluated by Precision, Recall, F measure and Accuracy of line classification in each round of the iterative contextual line classification.

As expected, the performance is stabilized within the first 10 iterations. Figure 3.4 also shows that the first two rounds are responsible for most of the performance improvement. This behavior suggests that two iteration steps can be used instead of waiting for absolute convergence. Table 3.6 lists the results achieved for each of the 15 classes when the iterative procedure converges. The small sample sizes of the class – degree, note, phone, keyword, and publication number – as shown in Table 3.6 may account for their poor classification performance. Seymore et al. report the same phenomenon on the class – degree, note and publication number – using HMM model [90].



(a) Precision in the iterative procedure.

(b) Recall in the iterative procedure

(c) F measure in the iterative procedure

(d) Accuracy in the iterative procedure

Fig. 3.4. Performance (Precision, Recall, F measure and Accuracy) in each round of the iterative procedure. X axis refers to each round in the iterative procedure

Class Name	Precision	Recall	F measure	Accuracy
Title	89.6%	88.9%	89.3%	98.2%
Author	94.2%	89.6%	91.8%	98.8%
Affiliation	93.8%	84.2%	88.7%	97.7%
Address	93.9%	85.1%	89.3%	98.8%
Note	82.3%	45.4%	58.5%	96.6%
Email	97.6%	97.4%	97.5%	99.8%
Date	97.2%	89.4%	93.1%	99.8%
Abstract	96.1%	97.7%	96.9%	96.9%
Introduction	98.8%	96.0%	97.4%	99.8%
Phone	93.8%	69.1%	79.5%	99.8%
Keyword	95.2%	55.2%	69.9%	99.3%
Web	100%	92.8%	96.3%	99.9%
Degree	86.0%	52.8%	65.4%	98.9%
Pubnum	91.7%	71.8%	80.5%	99.6%
Page	100.0%	100.0%	100.0%	100.0%

Table 3.5. Independent line classification performance.

Class Name	Precision	Recall	F measure (Increase)	Accuracy
Title	93.9	95.0	94.5(5.2)	99.1
Author	97.3	91.4	94.2(2.4)	99.2
Affiliation	96.4	90.3	93.3(4.5)	98.6
Address	93.6	86.7	90.0(0.71)	98.8
Note	86.4	65.6	74.6(16.0)	97.6
Email	98.9	94.0	96.4(-1.1)	99.8
Date	97.2	89.5	93.2(0.1)	99.8
Abstract	98.5	99.2	98.8(1.9)	98.8
Introduction	100.0	96.4	98.2(0.8)	99.9
Phone	98.3	62.3	76.2(-3.3)	99.7
Keyword	96.7	79.5	87.2(17.3)	99.7
Web	100.0	92.8	96.3(0.0)	99.9
Degree	91.4	80.5	85.6(20.1)	99.3
Pubnum	97.3	65.5	78.3(-2.2)	99.6
Page	100.0	100.0	100.0(0.0)	100.0

Table 3.6. Performance (%) of contextual line classification iteration algorithm when converges and the F measure increase than that of the independent line classification

N-Class	Number of Lines	Percentage
2	212	84.8%
3	33	13.2%
4	4	1.6%
5	1	0.4%

Table 3.7. The distribution of the multi-class lines in 500 training headers

3.3.4 Extract Metadata from Multi-Class Lines

After classifying each line into one or more classes, we now extract metadata from each multi-class line based on the predicted class labels for this line. As discussed the metadata extraction task from multi-class lines is turned into the chunk identification task. Chunk identification of an N-class line is analogous to finding $N-1$ chunk boundaries in the line. Punctuation marks and spaces between words are candidate chunk boundaries.

Table 3.7 shows that 86% of the multi-class lines in training data are two-class lines. We search for the optimal chunk boundary which yields the maximum difference between the two chunks. Independent line classifiers are applied to calculate the difference between chunks.

Every punctuation mark and space can be a candidate chunk boundary for two-class lines. We consider only punctuation marks as candidates if two or more punctuation marks are used in the line; otherwise we try each punctuation mark and space. Assuming that each class has only one chunk in the line, two-class chunk identification is to find the optimal chunk boundary.

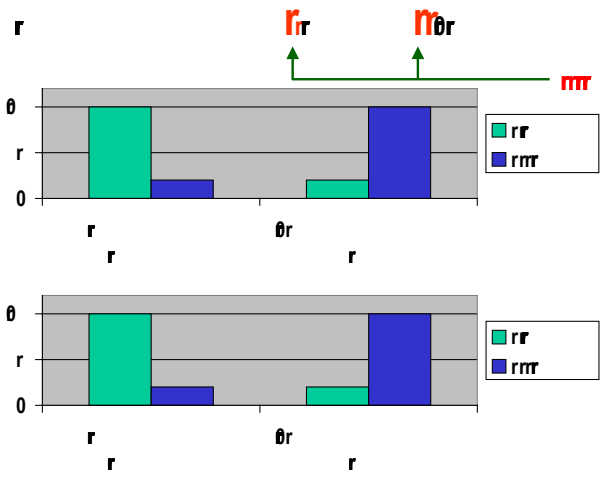


Fig. 3.5. The idea of the two-class chunk identification algorithm.

Figure 3.5 shows the idea of our two-class chunk identification algorithm. *"The Ohio State University, Columbus, OH 43210-1277"* is an example of two-class line of affiliation and address. Each comma is a candidate chunk boundary. We call the affiliation classifier as classifier 1 and the address classifier as classifier 2. The classifiers we use here are the SVM line classifiers trained by single-class lines of the training dataset. We consider each chunk as a short line. The first comma separates the line into two chunks: the first chunk is *"The Ohio State University"*, and the second chunk is *"Columbus, OH 43210-1277"*. The affiliation classifier achieved the classification score about 10 on the first chunk, while the address classifier achieved much low classification score on classifying the first chunk. In contrast, the affiliation classifier achieves much lower classification score than the address classifier on classifing the second chunk. The second comma separates the line into two chunks: the first chunk is *"The Ohio State University,*

Columbus", and the second chunk is *"OH 43210-1277"*. Both affiliation classifier and address classifier achieved lower classification score on classifying both chunks of the line. In comparison, the first comma marks the maximal difference between two chunks and is chosen as the optimal chunk boundary. The following algorithm implements this idea of two-class chunk identification.

Definitions:

P_1 the classification score of chunk P by classifier 1;

P_2 the classification score of chunk P by classifier 2;

N_1 the classification score of chunk N by classifier 1;

N_2 the classification score of chunk N by classifier 2;

$P_{12} = P_1$ - P_2; $N_{21} = N_2$ - N_1;

$PN_1 = P_1$ - N_1; $PN_2 = P_2$ - N_2;

We choose the optimal chunk boundary as the punctuation mark or space yielding the maximal $P_{12} * N_{21}$. Chunk P is classified into class 1 if $PN_1 > 0$, and (1) $PN_1 * PN_2 < 0$ or (2) $PN_1 * PN_2 > 0$ and $\|PN_1\| = max(\|PN_1\|, \|PN_2\|)$, class 2 otherwise.

This two-class chunk identification algorithm results in an accuracy of 75.5% (160 out of 212 two-class lines from training samples). Accuracy here is defined as the percentage of the lines whose chunk boundaries are correctly predicted versus the total number of two-class lines. (This is the lower boundary of the accuracy.)

Many N-class ($N > 2$) chunk identification tasks may be simplified to two-class chunk identification tasks. For instance, using the positions of email and URL in the line,

we may simplify the three-class chunk identification tasks as two-class chunk identification tasks. The position of the email address in the following three-class line "*International Computer Science Institute, Berkeley, CA 94704. email: aberer@icsi.berkeley.edu. Supported by Schweizerische Gesellschaft zur Forderung der Informatik und ihrer Anwendungen*" is a natural chunk boundary between the other two classes.

We are exploring more general multi-class chunk identification techniques.

3.3.5 Recognize Authors in the Multi-Author Lines

We consider the author lines with less than 4 words as *single-author lines* and the author lines with 4 or more words as *multi-author lines*. We further define a multi-author line where the authors are separated by spaces only as *space-separated multi-author line*. Similarly, a multi-author line where the authors are separated by punctuation marks is defined as *punctuation-separated multi-author line*.

We extract a total of 326 multi-author lines from the training dataset as the dataset for our experiment on recognizing authors from the multi-author lines. Among the 326 multi-author lines, 227(69.6%) lines are punctuation-separated and 99(30.4%) are space-separated. Based on the different characteristics punctuation-separated multi-author lines and space-separated multi-author lines possess, we choose the following different strategies for either case.

3.3.5.1 Chunk Identification of Punctuation-Separated Multi-Author Lines

As we discussed before, to recognize each name from a multi-author line is to identify chunk boundaries between author names. It is obvious that the spaces and

punctuation marks between words are the candidate chunk boundaries. The problem now becomes classifying each space or punctuation mark as chunk boundary or not. We consider only punctuation marks in the line as candidate chunk boundaries if there are two or more punctuation marks in the line; otherwise, we examine each space and punctuation mark as the candidate chunk boundary. The dictionary word "and" is considered as a punctuation mark. Spaces next to a punctuation mark are ignored.

We design a feature vector for each space and punctuation mark using both the raw features of the punctuation mark itself such as "," or "&", and the contextual features listed in Table 3.8. We also convert each word of the line into a 5-tuple $< FN, LN, L, FC, D >$. Each element of the 5-tuple is defined as follows.

FN**:** 1 if the word is in the first name list, 0 otherwise.

LN**:** 1 if the word is in the last name list, 0 otherwise.

L**:** 1, 2 or 0, indicates the word is of one letter, two letters, or more than two letters, respectively.

FC**:** 1 if the word is capitalized, 0 otherwise.

D**:** 1 if the word is a dictionary word, 0 otherwise.

We use the attributes defined in the above tuple to represent the 8th contextual feature in Table 3.8. The motivation is that if the closest word to a punctuation mark appears only on the first name list, or only on the last name list, it helps to classify if this punctuation mark is the right chunk boundary. For example, if "Leonidas Fegaras, David Maier" satisfies this pattern "[10010(First name)] [01011(Last name)], [10011(First name)] [00010(Last name)]", it will be reasonable to classify the comma as the right chunk

No.	Feature
1	The number of total punctuation marks of the same kind in the line
2	The position of this punctuation mark
3	The number of words before this punctuation mark
4	The number of words after this punctuation mark
5	The number of words between the previous and the current punctuation mark
6	The number of words between the current and the next punctuation mark
7	The ratio of the number of words before and after this punctuation mark
8	The previous and next 5 words in converted feature representation

Table 3.8. Contextual features for each candidate chunk boundary in punctuation-separated multi-author line

boundary. However, the big overlap between the first name list and the last name list makes such feature representation of each word ineffective.

We find from the stepwise feature selection that the dominating features in classifying chunk boundary are the punctuation marks themselves. Therefore in implementation, we design simple heuristic rules to make use of the punctuation marks to extract each name from the punctuation-separated multi-author line.

Table 3.9 lists the chunk identification performance on punctuation-separated multi-author lines. The evaluation is based on the percentage of punctuation marks that are classified correctly.

3.3.5.2 Chunk Identification for Space-Separated Multi-Author Lines

Unlike punctuation-separated lines, space-separated multi-author lines do not have any explicit information for chunk boundary recognition. The valid patterns for

Accuracy	Precision	Recall	F measure
93.31	82.38	96.65	88.95

Table 3.9. Chunk boundary identification performance of punctuation-separated multi-author lines

author names are the source of information in this case. [Mary(Full Name)] [Y.(Name Initial)], for instance, cannot be a valid name.

The algorithm for extracting names from space-separated multi-author lines has the following four steps.

- Generate all candidate ways of name separation for the space-separated multi-author lines based on the valid patterns of names that we define in Table 3.10.
- Design the feature vector for each candidate way of name separation. We manually label each candidate way of name separation as 1 or -1 by justifying each name from the web.
- Train a SVM name separation classifier from the labeled training samples.
- If the test space-separated multi-author line has only one candidate way of name separation, it is the predicted way of name separation. Otherwise, classify each of its candidate ways of name separation, and pick the one with the highest classification score as the correct way of name separation.

For example, based on the valid name patterns, "Idit Keidar Danny Dolev" has only one way of separation, which contains two authors "Idit Keidar" and "Danny Dolev". For another example, the author line "Alan Fekete David Gupta Victor Luchangco

Pattern Class	**Patterns**
1	$(F\|F^-)F$, $(F\|F^-)(F\|F^-)F$ $(F\|F^-)(F\|F^-)(F\|F^-)F$ e.g., Yu-Chee Tseng
2	$(F\|F^-)IF$, $(F\|F^-)IIF$, $(F\|F^-)IIIF$ e.g., Dhabaleswar K. Panda
3	IF, IIF e.g., C. L. Giles
4	$I(F\|F^-)F$
5	$(F\|F^-)ssF$ e.g., Th.P. van der Weide

Table 3.10. The valid patterns of a name. "F"- Full Name; "F^-" - Full Name with hyphen, e.g., Jon-hey; "I" - Name Initial; "s" - lower case word

Classification Score	Class label	Candidate name sequences
1.6398636	1	Alan Fekete ⋄ David Gupta ⋄ Victor Luchangco ⋄ Nancy Lynch ⋄ Alex Shvartsman
0.8996393	-1	Alan Fekete ⋄ David Gupta ⋄ Victor Luchangco Nancy ⋄ Lynch Alex Shvartsman
0.0061073704	-1	Alan Fekete ⋄ David Gupta Victor ⋄ Luchangco Nancy ⋄ Lynch Alex Shvartsman

Table 3.11. An example of candidate name sequences

Nancy Lynch Alex Shvartsman" has three candidate ways of name separations. Figure 3.11) shows the three name separations where names are separated by ⋄. The "1" and "-1" in front of each name separation marks the sequence as a positive sample or a negative sample. The numerical value at the beginning of each separation is the classification score. The first separation achieves the highest score and is predicted correctly.

The feature vector designed for each name separation is based on the following features. Let us assume L is a line that contains M names, n_1, n_2 through n_M. For name n_i $(1 \leq i \leq M)$ that has N_k words, we define the following five features.

$Form_{i,j}$ the form of the j^{th} word of n_i, $Form_{i,j} \in \{F, F^{-}, I, s, o\}$. "$o$" - others.

$Pos_{i,j}$ the position of the j^{th} word of n_i in the line.

$FN_{i,j}$ is equal to 1 if the j^{th} word of n_i is only in the first name list, 0 otherwise.

$LN_{i,j}$ is equal to 1 if the j^{th} word of n_i is only in the last name list, 0 otherwise.

$NonDic_{i,j}$ is equal to 1 if the j^{th} word of n_i is a non-dictionary word, 0 otherwise.

The feature $Form_{i,j}$ has non-numerical values such as "F", "I" or "s". We enumerate each of these name patterns and assign these values as the weights of the corresponding features.

We generated all the candidate name separations expanded from the 99 space-separated name separations as the name separation dataset. We achieve a classification accuracy of 90.9% for ten-fold cross validation on this dataset. The accuracy is the precentage of correctly predicted author lines among all 99 author lines.

Using SVM-based classification to classify name separations helps to find the implicit regularities that could have been missed by the manual inspection. A regularity discovered from the training data is that hyphenated names such as Jon-hey are not likely to be the last name.

$$Precision = \frac{A}{A+C} \quad Recall = \frac{A}{A+B}$$

$$Accuracy = \frac{A+D}{A+B+C+D}$$

$$Fmeasure = \frac{2Precision*Recall}{Precision+Recall}$$

3.4 Experimental Results

Performance is evaluated by precision, recall, F measure, and accuracy as described below.

Overall evaluation: The overall word classification accuracy for the header is the percentage of the header words that are tagged with the words' true labels.

Class-specific evaluation: We define A as the number of true positive samples predicted as positive, B as the number of true positive samples predicted as negative, C as the number of true negative samples predicted as positive and D as the number of true negative samples predicted as negative. A sample refers to a line in the line classification task and refers to a word when evaluating the final metadata extraction performance.

We apply the metadata extraction method with the parameters that are chosen from ten-fold cross-validation experiments on 500 training headers to 435 test headers. Our method achieves an overall accuracy of 92.9%, better than 90.1% reported by Seymore et al. Table 3.12 compares our method with the HMM method of multi-state L+D model from Seymore et al. on the classification performance for each class, except two functional classes "introduction" and "end of page". We were unable to obtain the

Class	HMM(A)	SVM(A)	SVM(P)	SVM(R)
Title	98.3	98.9	94.1	99.1
Author	93.2	99.3	96.1	98.4
Affiliation	89.4	98.1	92.2	95.4
Address	84.1	99.1	94.9	94.5
Note	84.6	95.5	88.9	75.5
Email	86.9	99.6	90.8	92.7
Date	93.0	99.7	84.0	97.5
Abstract	98.4	97.5	91.1	96.6
Phone	94.9	99.9	93.8	91.0
Keyword	98.5	99.2	96.9	81.5
Web	41.7	99.9	79.5	96.9
Degree	81.2	99.5	80.5	62.2
Pubnum	64.2	99.9	92.2	86.3

Table 3.12. Comparison on the performance(%) of metadata extraction using HMM and SVM evaluated based on words. A - Accuracy; P - Precision and R - Recall

class-specific accuracy method used by Seymore et al. at the time we submit this paper. Therefore, we also list class-specific precision and recall for more effective evaluation.

We present below another example ofdocument header with its true labels (Figure 3.6) and the labels (Figure 3.7) predicted by our metadata extraction algorithm. We also present the labels (Figure 3.8) our algorithm predicted for the example of document header shown in Figure 3.1 in section 3.3. The bold fonts indicate the predicted labels that are different from the true labels. Both examples show the good performance of our algorithm on labeling the single-class lines, and recognizing the individual authors from the multi-author lines. Line 6 in Figure 3.7 and line 22 in Figure 3.8 also show the good performance of our two-class chunk identification algorithm. The only difference between our algorithm's predictions and the original labels is line 7. Although we count this as a false prediction (in our evaluation), the original label "note" for this line can be argued itself. The line contains two email addresses. Therefore it could be labeled as email

1:<title> THE CORAL USER MANUAL +L+
2:A Tutorial Introduction to CORAL +L+ </title>
3:<author> Raghu Ramakrishnan Praveen Seshadri Divesh Srivastava +L+ </author>
4:<author> S. Sudarshan +L+ </author>
5:<affiliation> Computer Sciences Department, +L+
6:University of Wisconsin-Madison,</affiliation><address> WI 53706, U.S.A. +L+ </address>
7:<note>The authors' e-mail addresses are fraghu,divesh, praveeng@cs.wisc.edu; sudarsha@research.att.com.+L+</note>

Fig. 3.6. A document header with the true labels.

just as well. This kind of uncertainty of labels is rare, though. Figure 3.8 shows the direct impact the line classification has on the chunk identification performance. Wrongly classifying the five-class line 22 in Figure 3.8 as the four-class line causes the further incorrect chunk identification. Wrongly classifying the five-class line 25 as a single class line "note" also disables the further chunk identification algorithm. A reason that line 25 is wrongly classified as single-class line is that our contextual line classification algorithm in Section 3.3.3.2 over weighs the contextual information of the "note" text from line 22 to line 26.

3.5 Bibliographic Field Identification Using Hidden Markov Models

We apply hidden Markov models to bibliographic field extraction, as suggested by Seymore et al. [90].

A hidden Markov model is composed of five parameters: a finite set of states $Q = \{q_1, q_2, ..., q_L\}$; a finite set of observations $\Sigma = \{\sigma_1, \sigma_2, ..., \sigma_K\}$; state transition probabilities $P(q_{k-1} \rightarrow q_k)$; observation emission probabilities for each state $P(q_k \uparrow \sigma_i)$

1: chunk(1) - <title> - THE CORAL USER MANUAL
2: chunk(1) - <title> - A Tutorial Introduction to CORAL
3: chunk(1) - <author> - Raghu Ramakrishnan
chunk(2) - <author> - Praveen Seshadri
chunk(3) - <author> - Divesh Srivastava
4: chunk(1) - <author> - S. Sudarshan
5: chunk(1) - <affiliation> - Computer Sciences Department,
6: chunk(1) - <affiliation> - University of Wisconsin-Madison
chunk(2) - <address> - WI 53706, U.S.A.
*7: chunk(1) - **<email>** - The authors' e-mail addresses are fraghu,divesh,praveeng@cs.wisc.edu; sudarsha@research.att.com.*

Fig. 3.7. The labeles predicted by the SVM metadata extraction algorithm on the document header shown in Figure 3.6.

and an initial state distribution. Transition probabilities and emission probabilities are the two main parameters of HMM models.

HMMs for bibliographies may be constructed as follows: each state corresponds to a bibliographic class such as "author" or "title"; each word is an observation, and each state emits words following a class-specific multinomial distribution [74]. Extracting the bibliographic fields from unseen references using HMMs reveals the most likely state sequence for the observation, as shown by Equation 3.4. The most likely state sequence is computed by the Viterbi algorithm [87], where x is the observation sequence and M is the HMM model.

$$E(P(x|M)) = \underset{q_1,\ldots,q_l \in Q^l}{argmax} \prod_{k=1}^{l+1} P(q_{k-1} \rightarrow q_k) P(q_k \uparrow \sigma_k), \tag{3.4}$$

The HMM structure, i.e. the number of states and allowed transitions between states, can be constructed in two ways. One is an apriori manual construction; the other

1:chunk(1) - <*title*> *- Stochastic Interaction and Linear Logic*
2:chunk(1) - <*author*> *- Patrick D. Lincoln*
chunk(2) - <*author*> *- John C. Mitchell*
chunk(3) - <*author*> *- Andre Scedrov*
3:chunk(1) - <*abstract*> *- Abstract*
4:chunk(1) - <*abstract*> *- We present stochastic interactive semantics for propositional linear*
...
22:chunk(1) - **<note>** *- jcm@cs.stanford.edu*
chunk(2) - <*web*> *- http://theory.stanford.edu/people/jcm/home.html)*
chunk(3) - <*affiliation*> *- Department of Computer Science, Stanford University*
chunk(4) - **<address>** *- Stanford, CA 94305.* **Supported in part**
23:chunk(1) - <*note*> *- by an NSF PYI Award , matching funds from Digital Equipment Corporation, the Pow-ell Foundation, and Xerox Corporation; and the Wallace F. and Lucille M. Dav is Faculty*
24:chunk(1) - <*note*> *- Scholarship.*
25:chunk(1) - **<note>** *- andre@cis.upenn.edu*
http://www.cis.upenn.edu/∼andre Department of Mathematics, University of Pennsylvania, Philadelphia, PA 19104-6395. **Partially supported by**
26:chunk(1) - <*note*> *- NSF Grants CCR-91-02 753 and CCR-94-00907 and by ONR Grant N0001 4-92-J-1916. Scedrov is an American Mathematical Society Centennial Research Fellow.*

Fig. 3.8. The labeles predicted by the SVM metadata extraction algorithm on the document header shown in Figure 3.1.

is automatic construction from labeled training data, using various merging techniques. Seymore et al. use "neighboring merge", "V merge" and Bayesian model merging techniques to learn the structure; for details please see [90, 96].

Transition probabilities are learned from the sequences of class labels in the training data. Class-specific emission distributions are estimated from the word frequencies in the training data. A good emission distribution estimation needs a sufficient amount of observations and a good data representation.

We conduct experiments on the bibliographic dataset [90] of 500 labeled references. The dataset is randomized before being split into 250 training samples and 250

	Author	Title	Date	Editor	Institution	Journal	Location	Note	Pages	Publisher	Tech	Booktitle	Volume
before	92.2	90.6	89.4	54.1	69.8	71.3	66.9	43.5	71.6	71.4	40.0	87.1	73.3
after	97.9	91.0	96.8	78.6	71.3	78.8	71.6	48.0	95.3	69.7	66.8	93.1	89.6

Table 3.13. F measure of bibliographic field words tagging before and after cluster feature representation.

test samples. Table 3.13 shows the average accuracies of 10 experiment trials of our bibliographic fields extraction algorithm before and after using word specific features. Using word specific features outperforms using original words as features for bibliographic fields extraction.

3.6 Discussion and Future Work

This chapter describes two machine learning methods for metadata extraction. One is a classification-based method using Support Vector Machines (SVM) for metadata extraction from the header part of a document. The other applies the hidden Markov Model (HMM) to bibliographic field extraction. The SVM-based method used for document header metadata extraction achieves nominally better results than hidden Markov Model based methods. This occurs because we use apriori information of the structural pattern of the data, feature extraction based on domain specific databases, an appropriate normalization technique, and an iterative correction procedure. In addition, the method we propose for extracting individual names from a list of author names has good performance. Our results indicate a promising classification-based method for information extraction.

There are some aspects of the SVM-based method that could still be improved. The line classification performance limits the further multi-class line chunk identification performance as shown in Figure 3.8. We would like to add the functionality to correct the errors caused by the line classification algorithm. Some chunks such as an integrated name may be broken into two lines occasionally. In this case, the multi-class chunk identification algorithm may make the incorrect decision. We would like to combine some of the consecutive lines of the same class to decrease the corresponding errors. Currently we assume that each line has only one chunk for a class. This is not appropriate even though it is rare for a class to have multiple chunks of the same class in one line. It is worthwhile to explore more general multi-class chunk identification techniques.

In addition to extracting the taggable metadata from the header part of the research papers, and the bibliographic fields, it is worthwile to develop a robust and accurate wrapper to extract equations and figures from document texts.

Chapter 4

Name Disambiguation in Author Citations

4.1 Name Ambiguities in Author Citations

Due to name variation, identical names, name misspellings, inconsistent inclusion of initials, pseudonyms, and marriage, we observe two types of name ambiguities in research papers or bibliographies (citations). The first type is that an author has multiple name labels. For example, the author "David S. Johnson" may appear in multiple publications under different name abbreviations such as "David Johnson", "D. Johnson", or "D. S. Johnson", or a misspelled name such as "Davad Johnson". Another example is "Michelle Q Wang" who has a different name after marriage: Michelle Q Wang-Baldonado or Michelle QW Baldonado. The second type is that multiple authors may share the same name label. For example, "D. Johnson" may refer to "David B. Johnson" from Rice University, "David S. Johnson" from AT&T research lab, or "David E. Johnson" from Utah University (assuming the authors still have these affiliations). Table 4.1 shows another example of name ambiguity in author citations.

Name ambiguity can affect the quality of scientific data gathering, can decrease the performance of information retrieval and web search, and can cause the incorrect identification of and credit attribution to authors. For example, identical names cause the ambiguity of the "author page" in the web DBLP (Digital Bibliography & Library

Project) [1]. The author page of "Yu Chen" in the DBLP contains citations from three different authors with the same name: Yu Chen from University of California, Los Angeles; Yu Chen from Microsoft Beijing; Yu Chen as the senior professor from Renmin University of China. Such name ambiguity causes the incorrect identification of authors. For example, the author page of "Jia Li" in the DBLP refers to the "Jia Li" from the Department of Statistics at the Pennsylvania State University. However, the "Home Page" link in her author page directs to the professor with the identical name in the Department of Mathematical Sciences at the University of Alabama in Huntsville. We observe from CiteSeer [40] the incorrect attribution to the authors due to similar ambiguity. "D. Johnson" is the most cited author in Computer Science according to CiteSeer's statistics in May 2003 (http://citeseer.nj.nec.com/mostcited.html). However, the citation number that "D. Johnson" obtained in CiteSeer's statistics is actually the sum of several different authors such as "David B. Johnson", "David S. Johnson", and even "Joel T. Johnson".

4.2 Related Work

Name ambiguity is a special case of the general problem of *identity uncertainty*, where objects are not labeled with unique identifiers [82]. Previous research has addressed the identity uncertainty problem using different methods, such as record linkage [35], duplicate record detection and elimination [16, 64, 77], merge/purge [48], data association [11], database hardening [22], citation matching [75, 75], name matching [15, 99, 17], address matching [26], and name authority control in library cataloging

[1] http://WWW.Informatik.Uni-Trier.DE/~ley/db/index.html

J. E. Smith	**Author citations**
Computer Science	Rapid Profiling via Stratified Sampling, S. Sastry, R. Bodik, **J. E. Smith**, 28th Int. Symposium on Computer Architecture, 2001.
	Relational Profiling: Enabling Thread-Level Parallelism in Virtual Machines, Timothy Heil and **J. E. Smith**, 33rd Int. Symp. on Microarchitecture, 2000.
	Concurrent Garbage Collection using Hardware Assisted Profiling, Timothy Heil and **J. E. Smith**, International Symposium on Memory Management, 2000.
Management Science	**Smith, James E.**, "Moment Methods for Decision Analysis", Management Science 39 (1993).
	Smith, James E., "Generalized Chebychev Inequalities: Theory and Applications in Decision Analysis", Operations Research 43 (1995).
	Smith, James E., Samuel Holtzman and James E. Matheson, "Structuring Conditional Relationships in Influence Diagrams", Operations Research 41 (1993).
Hydrology	Henry E.J. and **Smith J.E.** 2002. The Effect of SurfaceActive Solutes on Water Flow and Contaminant transport in Variably Saturated Porous Media with Capillary Fringe Effects. Journal of Contaminant Hydrology.
	Henry E.J., **Smith J.E.**, and Warrick A.W. 2002. Two-Dimensional Modeling of Flow and Transport in the Vadose Zone with Surfactant-Induced Flow. WATER RESOURCES RESEARCH.
	Smith, J.E. and Zhang F.Z. 2001. Determining Effective Interfacial Tension and Predicting Finger Spacing for DNAPL Penetration into Water-Saturated Porous Media. Journal of Contaminant Hydrology.

Table 4.1. Partial citation clusters of three canonical authors of the same name label "J. E. Smith".

practice [104, 30, 41]. At the concept level these methods include word sense disambiguation [101, 60].

Name authority control and name matching are the work most similar to ours. Name authority control aims to find the authoritative form of names, i.e., the unambiguous reference to an individual [30]. Getty's ULAN (Union List of Artist's Names) [1] and the Library of Congress name authority file [2] are good examples of such authorized

names. Name authority control usually provides a set of rules and standardized terms for consistent name representation (e.g., the form of the name to be used). A "canonical name"[46] includes an authorized name as a special case. Although much work in name authority control relies on manual analysis [41], recent research [30, 46] considers automated systems. Such automated systems use supervised learning methods, relying much on a priori knowledge of ambiguous name entities or name word lists. For example, DiLauro et al. [104, 30] propose a semi-automatic algorithm to disambiguate composers and artists in the Levy music collection. However, their algorithm largely depends on a priori knowledge of ambiguous name entities or name word lists (e.g., the Library of Congress name authority file).

Name matching [15, 17, 22, 99] usually identifies a name entity with different name labels from duplicate records of different syntactic formats. For example, "Bart Selman" and "B. Selman" are ambiguous name labels of the same person who authored the work cited as "Critical behavior in satisfiability" [22]. Name matching does not focus on the case of different name entities that have identical name labels. Our method disambiguates names from different records (citations) authored by the same name entity, and addresses both types of name ambiguities previously mentioned. Our method works in conjunction with name matching that usually uses string-based comparison to induce the correct name entities from names with misspellings and abbreviations.

4.3 Supervised and Unsupervised Learning Methods for Name Disambiguation

Given a set of citations that have the ambiguous (e.g. identical) name label, how do we disambiguate authors if the name label refers to a single author, or different authors with ambiguous names? This thesis studies both supervised and unsupervised learning methods to disambiguate authors in citations.

The supervised learning methods consider each canonical author name as a class, and identify the correct author class for each citation. We developed two supervised learning methods: one based on a hierarchical naive Bayes probability model, and the other based on Support Vector Machines.

The supervised learning methods need labeled data. However, the authors' previous citations or identification information are not necessarily available to train the classifiers. With unsupervised learning methods, we do not need labeled data for training. The name disambiguation problem can be formulated as partitioning collections of citations into clusters, with each cluster containing only citations authored by the same author, thus disambiguating authorship in citations to induce author name identities. We develop two unsupervised learning methods: one based on a hierarchical naive Bayes mixture model, and the other based on a K-way spectral clustering method.

All methods use three attributes of the citations associated with each canonical name entry in the training citations: coauthor names, paper titles, and publication venue titles. "Publication venue title" refers to the title of any the publication sources, such

as proceedings or journals. Author names in citations are represented by the first name initial and last name, the minimal name information seen in citations.

4.4 Experimental Datasets

To study and compare all the above methods, we collect the following two types of citations in different ways.

4.4.1 Type I Data: 2 Web Collected Name Datasets

The first type of data is publication lists collected from the web, mostly from researchers' homepages. This type of data contains two datasets, one is 15 different "J Anderson"s who have 229 citations in total, shown in Table 4.2, the other is 11 different "J Smith"s [2] who have 339 citations in total. Both "J Anderson" and "J Smith" are ambiguous names in the database of our EbizSearch system - a CiteSeer like search engine specializing in the E-Business area [84]. We query "Google" using name information such as "J Anderson", or the full name information available in our EbizSearch databases such as "James Anderson", and the keyword "publications". We manually check the returned links, identify the canonical authors who have the same first name initial and last name as the query name, and collect their publication web pages to construct our datasets.

[2] available upon request

J Anderson (id)	Affiliation	Research Area	Size
James Nicholas Anderson (1)	UK Edinburgh	Communication interface research	8
James E. Anderson (2)	Boston College	Economics	14
James A. Anderson (3)	Brown Univ.	Neural network	3
James B. Anderson (4)	Penn. State Univ.	Chemistry	6
James B. Anderson (5)	Univ. of Toronto	Biology	21
James B. Anderson (6)	Univ. of Florida	Entomology	17
James H. Anderson (7)	U. of North Carolina at Chapel Hill	Computer processors	54
James H. Anderson (8)	Stanford Univ.	Robot	4
James D. Anderson (9)	Univ. of Toronto	Dentistry	5
James P. Anderson (10)	N/A	Computer Security	3
James M. Anderson (11)	N/A	Pathology	5
James Anderson (12)	UK	Robot vision and philosophy	19
James W. Anderson (13)	Univ. of KY	Medicine	10
Jim Anderson (14)	Univ. of Southampton	Mathematician	20
Jim V. Anderson (15)	Virginia Tech Univ.	Plant pathology	40

Table 4.2. The citation dataset of 15 "J Anderson"s. Column 1, 2 & 3 shows the available "Identification information" of a "J Anderson", e.g., the full name of each "J Anderson", his or her affiliation and research area. "Size" lists the number of citations for each canonical author. For space limitation, we do not list here the web sites where we download the citations.

4.4.2 Type II Data: 14 Name Datasets Constructed Based on DBLP Computer Science Bibliography

The second type of citations are mainly downloaded from the DBLP Computer Science bibliography which contains more than 400,000 citation records with parsed citation attributes in the XML format. We concatenate the three attributes in each citation as a string, and then cluster citations with author names of the same first name initial and the same last name. We sort the formed citation clusters by the number of name variations contained. Top ranked ambiguous names are popular names from Asia, such as "J. Lee", "S. Lee", "Y. Chen" and "C. Chen". Besides these four name datasets, we also use other 10 large sets of ambiguous names from the DBLP bibliography as

Name	≥ 2		≥ 3		≥ 4		≥ 5		≥ 6		≥ 7		≥ 8		≥ 9		≥ 10	
	N	**C**	**N**	**C**	**N**	**C**	**N**	**C**	**N**	**C**	**N**	**C**	**N**	**C**	**N**	**C**	**N**	**C**
A. Gupta	26	577	22	569	18	557	17	553	17	553	17	553	14	532	12	516	11	507
A. Kumar	14	244	11	238	9	232	7	224	7	224	6	218	6	218	5	210	5	210
C. Chen	61	800	50	778	40	748	35	728	29	698	27	686	25	672	22	648	20	630
D. Johnson	15	368	11	360	10	357	9	353	8	348	7	342	6	335	6	335	6	335
J. Lee	100	1417	91	1399	58	1300	55	1288	46	1243	44	1231	40	1203	38	1187	38	1187
J. Martin	16	112	13	106	11	100	6	80	6	80	5	74	5	74	4	66	4	66
J. Robinson	12	171	10	167	8	161	8	161	7	156	7	156	7	156	6	148	6	148
J. Smith	31	927	25	917	21	905	19	897	17	887	16	881	15	874	14	866	12	848
K. Tanaka	10	280	9	278	8	275	8	275	6	275	6	265	5	265	5	258	5	258
M. Brown	13	153	13	153	10	144	8	136	7	131	7	131	7	131	5	115	5	115
M. Jones	13	259	12	257	11	254	10	259	9	245	9	245	9	245	6	221	6	221
M. Miller	12	412	10	408	7	399	7	399	5	389	5	389	5	389	5	389	5	389
S. Lee	86	1458	74	1439	56	1385	45	1341	40	1316	38	1304	36	1290	36	1290	36	1290
Y. Chen	71	1264	61	1244	48	1205	42	1181	36	1151	30	1115	27	1094	25	1078	22	1051

Table 4.3. The 14 DBLP name datasets varied by size. "$\geq i$" means that the dataset contains authors who have at least i citations. In each size variation of the dataset, the column "N" lists the number of authors each name label corresponds to. For example, the dataset that contains "J. Lee" of at least 2 citations has 100 different "J. Lee", such as "Jaejin Lee", "Jon Lee", etc. The column "C" lists the total number of citations in the corresponding dataset.

shown in Table 4.3. Each name dataset has more than 10 name variations. Moreover, we enrich the datasets with publication lists downloaded from author homepages that are found when we label the canonical author names (Next subsection describes how we label the data). The goal is to provide each canonical author name with the maximal amount of available citation information.

The DBLP datasets seem to be more challenging than the web collected datasets, because of the following reasons:

- Since most authors in the DBLP datasets come from the Computer Science community, different researchers are likely to have overlapping research interests, and

publish papers in the same research area. Databases and networks are the two most popular research areas in our datasets. The common paper or publication venue title keywords shared by different authors are in fact "ambiguous" information, which makes the classification harder..

- The number of canonical names to be disambiguated in each name dataset is high. For example, "J Lee" dataset contains 100 canonical names; "S Lee" dataset contains 86 canonical names; "Y Chen" dataset contains 71 canonical names and "C Chen" contains 61 canonical names.

- The complete publication venue title information we obtain does not cover and replace all the publication venue title abbreviations in the datasets. This may under-exploit the publication venue information.

4.4.3 Data Processing

4.4.3.1 Labeling

For evaluation purpose, we manually label the canonical name entities and associated citations from both the web collected datasets and the DBLP datasets. Citations listed in an author's publication home page are considered as being written by the same author. Authors with the same name and same affiliation, or same email address are considered to be the same. Authors of the same name that also have the same co-author names (in a complete name format) are very likely the same author. Citations that have the same name label, and are about the same topic are likely to be written by the same author. We also sent emails to some authors to confirm their authorship of citations.

The citations for which we had insufficient information to be judged were eliminated. Moreover, we populate the datasets with publication lists downloaded from the available home page URLs of authors in the datasets. Duplicate citations are detected and removed using CiteSeer's citation matching algorithm [40].

4.4.3.2 Data Preprocessing

We preprocess all the citations in the following manner.

- Parse each citation to obtain three types of attributes: coauthor name(s), paper title and publication venue title. We parse the web collected citations by using regular expression matching to extract the features. To minimize the mistakes in the citation parsing procedure, we manually correct the parsing results. Citation attributes can also be extracted by methods such as regular expression matching, rule-based system [19], hidden Markov models [90, 93, 97], or Support Vector Machines [45]. The citations from the DBLP datasets are already in the XML format with parsed attributes.

- Simplify all the author names in the citations to first name initial and last name. For example, "Yong-Jik Kim" is simplified to "Y Kim". A reason for the simplification is that the first name initial and last name format is popular in bibliographic records. Since more name information usually helps name entity disambiguation, insufficient name information from simplified name format would be good for evaluating our algorithms. Moreover, the simplified name format may avoid some cases of name misspellings. Third, the simplified name format helps to construct the

ambiguous name datasets, because there are usually more canonical names that share the identical first name initial and last name than the canonical names that share the complete name.

- Stem the words of paper titles and publication venue titles using Krovetz's stemmer [59], and remove the stop words such as "a", "the", etc.

- Replace the conference or publication venue title abbreviations by their full names for more information. The full names of the conference or publication venue titles are obtained from the DBLP websites [3].

In the experiments for supervised learning methods, each name dataset is randomly split, with half of them used for training, and the other half used for testing. For example, the "J Anderson" dataset contains 117 citations for training and 112 citations for testing; the "J Smith" dataset contains 172 training citations and 166 testing citations. A citation database is then constructed for each name dataset, based on the parsed and pre-processed training citations. For example, the citation database of "J Anderson" contains 15 canonical name entries for 15 different "J Anderson"s, with each name entry associated with available identity information, such as full name, affiliation, research area, as well as authored citations.

4.4.4 Evaluation Method

We evaluate experimental results based on the confusion matrix, where $A[i, j]$ represents the number of "Author i" predicted as "Author j" in matrix A. $A[i, i]$ represents

[3] http://www.informatik.uni-trier.de/~ley/db/conf/indexa.html and http://www.informatik.uni-trier.de/~ley/db/journal/index.html

the number of correctly predicted names for "Author j". We define the disambiguation accuracy as the sum of diagonal elements divided by the total number of elements in the matrix.

4.5 Supervised Name Entity Disambiguation

4.5.1 Generative Model vs. Discriminative Model

As mentioned in the previous section, supervised learning approaches consider each canonical author name as a class, and classify each citation into its author class. We study two machine learning approaches for name disambiguation, one based on a generative model and the other based on a discriminative model.

Generative models model the generation of the data. These models are built from the positive training data. They usually provide good insight of the nature of the data, and can create new instances of the data. Also, the generative models facilitate easy incorporation of domain knowledge [9]. Due to insufficient information about the data, however, it may not be easy to find a model that fits the data,

Discriminative models (such as Support Vector Machines) distinguish between the positive data and negative data. They do not explain how data is generated. A discriminative model is trained using both positive and negative data. However, such models usually treat each citation (or document) as a "bag-of-words", which makes it difficult to incorporate domain knowledge.

This section proposes a generative model, the hierarchical naive Bayes model, for name disambiguation. Our experimental results show that this model captures author

writing patterns, such as how an author coauthors with other authors. In comparison, we also experiment with a Support-Vector-Machine(SVM)-based model for name disambiguation. The hierarchical-naive-Bayes-model-based approach classifies a citation to an author based on the probabilities, while the SVM-based classification uses a distance measure [98]. In addition, a probability model allows us to systematically combine different models [49], and is easily extensible to more information; the vector space representation of citations in classification approaches usually needs to tune weights for different attributes [15, 99].

Given a full citation with the query name implicitly omitted, the name disambiguation is to predict the most likely canonical name from the training citations. For example,"[J. Anderson], S. Baruah, K. Jeffay. Parallel Switching in Connection-Oriented Networks. IEEE Real-Time Systems Symposium 1999: 200-209" is a test citation. "J. Anderson" is the omitted query name. The hierarchical-naive-Bayes-model-based approach estimates the author-specific probabilities, such as the prior probability of each author, and his/her probabilities of coauthoring with coauthors, using certain keywords in the title of the paper, and publishing papers in certain places, as described in detail in next section. Given a new citation and its query author name, name disambiguation is to search the training citations and choose the canonical name entry with the highest posterior probability of producing this citation. The SVM-based approach considers each author as a class, and classifies a new citation to the closest author class. With the SVM-based approach, we represent each citation in a vector space; each coauthor name and keyword in paper/publication venue title is a feature of the vector.

4.5.2 The Hierarchical Naive Bayes Model

We assume that each author's citation data is generated by the hierarchical naive Bayes model. Based on this assumption, we use his/her past citations as the training data to estimate the model parameters. We then use the Bayes rule to calculate the probability that each name entry X_i (where $i \in [1, N]$ and N is the total number of candidate name entries in the citation database) would have generated the input citation.

4.5.2.1 Target Function

Given an input test citation C_m with the omission of the query author, the target function is to find a name entry X_i in the citation database with the maximal posterior probability of producing the citation C, i.e.,

$$argmax_i P(X_i|C_m) \tag{4.1}$$

Using the Bayes rule, the problem becomes finding

$$max_i \frac{P(C_m|X_i)P(X_i)}{P(C_m)} \tag{4.2}$$

where $P(X_i)$ denotes the prior probability of X_i authoring papers, and is estimated from the training data as the proportion of the citations of X_i among all the citations. The prior is useful to incorporate the knowledge, such that a prolific author can have large $P(X_i)$. $P(C_m)$ denotes the probability of the citation C_m and is omitted since it does

not depend on X_i. Then Function 4.2 becomes

$$max_i P(C_m|X_i)P(X_i) \tag{4.3}$$

4.5.2.2 Model Hierarchy

We assume that coauthors, paper titles, and publication venue titles are independent citation attributes. Therefore, we decompose $P(C_m|X_i)$ in Function 4.3 as

$$P(C_m|X_i) = \prod_j P(A_j|X_i) = P(A_1|X_i)P(A_2|X_i)P(A_3|X_i) \tag{4.4}$$

where A_j denotes the different type of attribute; that is, A_1 - the coauthor names; A_2 - the paper title; A_3 - the publication venue title.

We build a hierarchical naive Bayes model to estimate the conditional probability of each type of attribute A_j given a canonical author X_i. The model tries to capture the following author patterns that we hypothesize. (1) Different authors X_i have different probabilities of writing papers alone, or writing papers with previously seen or unseen coauthors. (2) Each author X_i has his/her own list of previously seen coauthors, and a unique probability distribution on these previously seen coauthors to write papers with. Similarly, authors have the following patterns of keyword usage. (1) Different authors have different probabilities of using previously used (seen) or unused (unseen) keywords. (2) Each author has his/her own list of previously used keywords, and a unique probability distribution on these previously used keywords of paper title or publication venue title.

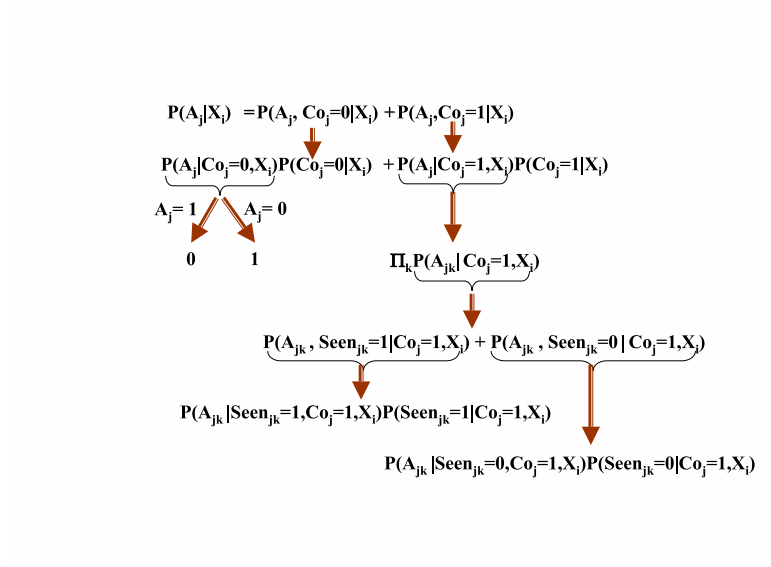


Fig. 4.1. A hierarchical naive Bayes model.

Figure 4.1 shows an example of estimating the conditional probability $P(A_1|X_i)$ that an author writes a paper with coauthors. $A_1 = (A_{11}, A_{12}, \cdots, A_{1k}, \cdots, A_{1K(1)})$ and $K(1)$ is the total number of coauthors of X_i. "Co_1" stands for "Has coauthor"; "$Seen_1$" stands for "Previously seen coauthor"; "A_{1k}" stands for the k^{th} coauthor of X_i in a citation. Equations (4.5), (4.6), (4.7), (4.8), (4.9), and (4.10) explain Figure 4.1 in more detail. Equations (4.5) and (4.6) show that an author may write papers alone or write papers with coauthors. Equation (4.8) shows that an author may write papers with previously seen or unseen coauthors, conditioned on that the author writes papers with coauthors.

$$P(A_1|X_i) = \begin{cases} P(A_1|Co_1 = 0, X_i) * P(Co_1 = 0|X_i) \\ \text{(if } A_1 \text{ writes paper alone)} \\ P(A_1|Co_1 = 1, X_i) * P(Co_1 = 1|X_i) \\ \text{(if } A_1 \text{ writes paper with coauthors)} \end{cases} \tag{4.5}$$

$$P(A_1|Co_1 = 0, X_i) = \begin{cases} 1 & \text{if } A_1 \text{ is empty} \\ 0 & \text{if } A_1 \text{ is not empty} \end{cases} \tag{4.6}$$

$$P(A_1|Co_1 = 1, X_i) = \prod_k P(A_{1k}|Co_1 = 1, X_i) \tag{4.7}$$

$$\begin{aligned} P(A_{1k}|Co_1 = 1, X_i) &= P(A_{1k}, Seen_{1k} = 1|Co_1 = 1, X_i) + P(A_{1k}, Seen_{1k} = 0|Co_1 = 1, X_i) \\ &= P(A_{1k}|Seen_{1k} = 1, Co_1 = 1, X_i) * P(Seen_{1k} = 1|Co_1 = 1, X_i) + \\ &= P(A_{1k}|Seen_{1k} = 0, Co_1 = 1, X_i) * P(Seen_{1k} = 0|Co_1 = 1, X_i) \end{aligned} \tag{4.8}$$

We assume that different elements in an attribute type are conditionally independent from each other. An attribute element refers to an individual coauthor, an individual word in the paper title, or an individual word in the publication venue title. Equation (4.7) shows that the every individual coauthor, A_{1k} ($k \in \{0, \cdots, K(1)\}$), is independent from each other. Equation (4.9) shows that the every individual word of paper title (or publication venue title), A_{jk} ($k \in \{0, \cdots, K(j)\}$ and $j \in \{2, 3\}$), is independent from each other. The assumptions used in this model, the conditional independence

$$
\begin{aligned}
P(A_j|X_i) &= P(A_j, Co_j = 1|X_i) \\
&= P(A_j|Co_j = 1, X_i)P(Co_j = 1|X_i) \\
&= \prod_k P(A_{jk}|Co_j = 1, X_i) \qquad (4.9)
\end{aligned}
$$

$$
\begin{aligned}
P(A_{jk}|Co_j = 1, X_i) &= P(A_{jk}, Seen_{jk} = 1|Co_j = 1, X_i) + P(A_{jk}, Seen_{jk} = 0|Co_j = 1, X_i) \\
&= P(A_{jk}|Seen_{jk} = 1, Co_j = 1, X_i)P(Seen_{jk} = 1|Co_j = 1, X_i) + \\
&\quad P(A_{jk}|Seen_{jk} = 0, Co_j = 1, X_i) * P(Seen_{jk} = 0|Co_j = 1, X_i) \qquad (4.10)
\end{aligned}
$$

of attributes and the conditional independence of attribute elements, may not hold for real-world data, since there exist cases such as multiple coauthors always appearing together. However, empirical evidence shows that naive Bayes often performs well in spite of such violation. Previous research shows that the violation of the word independence assumption sometimes may affect slightly the classification accuracy [37, 32].

Similarly, we have Equations (4.9) and (4.10) to model $P(A_2|X_i)$ and $P(A_3|X_i)$. Equation (4.10) is different from Equation (4.8) in that a paper title or a publication venue title must have keywords, that is, $P(Co_j = 1|X_i) = 1$ and $P(A_j, Co_j = 0|X_i) = 0$, where $j \in \{2, 3\}$. To avoid overflow, we use log probabilities in our implementation.

4.5.2.3 Model Parameters Estimation

This subsection describes estimation of the conditional probabilities that are decomposed from $P(A_1|X_i)$ from the training citations. The probability estimation is the maximum likelihood estimation for parameters of multinomial distributions. The pseudo

count 1 is added in parameter estimation to avoid zero probability in the estimation results. Parameter estimations for $P(A_2|X_i)$ and $P(A_3|X_i)$ are similar to the estimation of $P(A_1|X_i)$.

- $P(Co_1 = 0|X_i)$ - the probability of X_i writing a future paper alone conditioned on the event of X_i, estimated as the proportion of the papers that X_i authors alone among all the papers of X_i.

- $P(Co_1 = 1|X_i)$ - the probability of X_i writing a future paper with coauthors conditioned on the event of X_i. $P(Co_1 = 1|X_i) = 1 - P(Co_1 = 0|X_i)$.

- $P(Seen_{1k} = 1|Co_1 = 1, X_i)$ - the probability of X_i writing a future paper with previously seen coauthors conditioned on the event that X_i writes a future paper with coauthors. We regard the authors coauthoring a paper with X_i at least twice in the training citations as the **"seen coauthors"**; the other coauthors coauthoring a paper with X_i only once in the training citations is considered as the **"unseen coauthors"**. Therefore, we estimate $P(Seen_{1k} = 1|Co_1 = 1, X_i)$ as the proportion of the number of times that X_i coauthors with "seen coauthors" among the total number of times that X_i coauthors with any coauthor. Note that if X_i has n coauthors in a training citation C_m, we count that X_i coauthors n times in citation C_m.

- $P(Seen_{1k} = 0|Co_1 = 1, X_i)$ - the probability of X_i writing a future paper with "unseen coauthors" conditioned on the event that X_i writes a paper with coauthors. This probability and $P(Seen_{1k} = 1|Co_1 = 1, X_i)$ do not depend on k. $P(Seen_{1k} = 0|Co_1 = 1, X_i) = 1 - P(Seen_{1k} = 1|Co_1 = 1, X_i)$

- $P(A_{1k}|Seen_{1k} = 1, Co_1 = 1, X_i)$ - the probability of X_i writing a future paper with a particular coauthor A_{1k} conditioned on the event that X_i writes a paper with previously seen coauthors. We estimate it as the proportion of the number of times that X_i coauthors with A_{1k} among the total number of times X_i coauthors with any coauthor.

- $P(A_{1k}|Seen_{1k} = 0, Co_1 = 1, X_i)$ - the probability of X_i writing a future paper with a particular coauthor A_{1k} conditioned on the event that X_i writes a paper with unseen coauthors. Considering all the names in the training citations as the population and assuming that X_i has equal probability to coauthor with an unseen author, we estimate $P(A_{1k}|Seen_{1k} = 0, Co_1 = 1, X_i)$ as 1 divided by the total number of author (or coauthor) names in the training citations minus the number of coauthors of X_i. However, the small citation size may underestimate the population of new coauthors that X_i will coauthor with in the real-world. This may in turn underestimates the probability of an author coauthoring with previously seen coauthors. In this case a larger population size is needed.

4.5.2.4 Computational Complexity

Suppose a citation database consists of N canonical authors, where each author has an average of M training citations, and each citation has an average of K attribute elements. The computational complexity for training (estimating the probabilities) the above model is $O(MNK)$; the computational complexity for the query step using coauthor information alone is $O(NK)$ for each query citation. This complexity indicates the scalability of our algorithm to real-world applications.

4.5.3 Support-Vector-Machines-based approach

This approach considers each canonical author as a class, and trains a classifier for each canonical author class. Given a full citation with the omission of the query name, the goal of name disambiguation is to classify this citation to the closest author class. Each citation is represented by a feature vector, with each coauthor name and keyword in the paper/publication venue title as a feature and its frequency in the citation as the feature weight. We use the $\|X\|_\infty$ to normalize the weight of features with different ranges of values, which was shown to improve the classification performance [45].

We choose Support Vector Machines [102, 24] as classifiers because of their good generalization performance and ability in handling high dimensional data. All experiments use SVM^{light} [51].

4.5.3.1 Feature Ranking

The SVM is designed for a two class classification problem, as explained in Chapter 3.3.1. The linear kernel of SVM classification can also be used to rank features, as explained in the following. Let $\{(\vec{\mathbf{x_1}}, y_1), ..., (\vec{\mathbf{x_N}}, y_N)\}$ be a two-class training dataset, with $\vec{\mathbf{x_i}}$ a training feature vector and their labels $y_i \in$ {-1, +1}. The SVM attempts to find an optimal separating hyperplane to maximally separate two classes of training data. The corresponding decision function is called a classifier. In the case where the training data is linearly separable, computing an SVM for the data corresponds to minimizing $\|\vec{\mathbf{w}}\|$ such that

$$y_i(\vec{\mathbf{w}} \cdot \vec{\mathbf{x}}_i + w_0) - 1 \geq 0, \forall i \tag{4.11}$$

The linear decision function is

$$f(\vec{\mathbf{x}}) = sgn\{(\vec{\mathbf{w}} \cdot \vec{\mathbf{x}}) + w_0\} = sgn\{\sum_{i}^{n} \alpha_i^* y_i (\vec{\mathbf{x}}_i \cdot \vec{\mathbf{x}}) + w_0^*\} \tag{4.12}$$

If $f(\vec{\mathbf{x}}) > 0$, the data $\vec{\mathbf{x}}$ belongs to class 1; otherwise, $\vec{\mathbf{x}}$ belongs to class 2. The absolute value of $f(\vec{\mathbf{x}})$ indicates the distance of $\vec{\mathbf{x}}$ from the other class. In the final decision function $f(\vec{\mathbf{x}})$, the training samples with non zero coefficients α_i^* lie closest to the hyperplane, and are called support vectors. As Equation 4.12 shows, $f(\vec{\mathbf{x}})$ is a weighted sum of all features, plus a constant term as the threshold. n is the number of support vectors. Zhang et al. [114] propose to rank the features according to their contribution in separating the differences between two classes. We formalize such a contribution of a feature by Expression 4.13, where x_{ij} is the weight of the feature j in support vector i. We use such ranking of features to analyze the classification performance by SVMs (Chapter 4.5.4.3).

$$\sum_{i}^{n} \alpha_i * y_i x_{ij} \tag{4.13}$$

We extend SVMs to multi-class classification using the "One class versus all others" approach, i.e., one class is positive and the remaining classes are negative.

4.5.4 Experiments

4.5.4.1 Experiment Design

The experiments are conducted on the two types of data as described in Chapter 4.4. With each approach, we conduct 10 experiments with randomly split dataset for each experiment. For each experiment, we explore multiple schemes based on different combinations of the utilized citation attributes. The goal is to study the contributions of different citation attributes on name disambiguation. Both approaches use three schemes which use alone one citation attribute, and at least one of two "Hybrid" schemes which combine aspects of all three attributes. In the hierarchical-naive-Bayes-model-based approach, "Hybrid I" computes the equal joint probability of different attributes. In the SVM-based approach, "Hybrid I" combines different attributes in the same feature space. The "Hybrid II" scheme is specific to the hierarchical naive Bayes model and uses the coauthor attribute alone when a coauthor relationship exists between a coauthor in the test citation and a candidate name entry in the citation database; otherwise, "Hybrid II" uses the equal joint probability of all the three attributes. Flexibility of manipulating attributes is an advantage of using a probability model. The absence of a particular attribute can be handled by omitting the corresponding probabilities. "Hybrid II" is motivated by the experimental observation that with the "J Anderson" dataset, adding title words decreases the number of disambiguated names when using only the co-author information. We observe that the coauthor information is valuable for name disambiguation, and design the "Hybrid II" scheme to preserve the names disambiguated by using coauthor information alone.

Scheme	Coauthor		Paper title		Publication Venue title		Hybrid I		Hybrid II
Approach	Bayes	SVM	Bayes	SVM	Bayes	SVM	Bayes	SVM	Bayes
Mean	71.3%	64.4%	77.9%	82.9%	72.1%	74.4%	91.3%	95.6%	93.5%
StdDev	2.1%	3.8%	3.3%	1.9%	2.1%	3.0%	1.6%	1.7%	1.8%
P Value	1.38E-05		0.003		0.012		0.0003		

Table 4.4. The mean and the standard deviation (StdDev) of the 10 name disambiguation accuracy trials on the "J Anderson" dataset, with both the hierarchical-naive-Bayes-based approach(Bayes) and the SVM-based approach(SVM); and the statistical significance (two tail P value) of the performance difference by the two approaches.

Scheme	Coauthor		Paper title		Publication Venue title		Hybrid I		Hybrid II
Approach	Bayes	SVM	Bayes	SVM	Bayes	SVM	Bayes	SVM	Bayes
Mean	75.2%	60.0%	82.3%	84.2%	76.3%	78.4%	92.9%	94.5%	93.0%
StdDev	3.0 %	2.9%	3.5%	1.7%	2.2%	2.3%	2.0%	1.3%	2.1%
P Value	1.2E-09		0.074		0.035		0.031		

Table 4.5. The mean and the standard deviation (StdDev) of the 10 name disambiguation accuracy trials on the "J Smith" dataset, with both the hierarchical naive Bayes approach(Bayes) and the SVM approach(SVM); and the statistical significance (two tail P value) of the performance difference by the two approaches.

We evaluate the experiment performance by "accuracy", and define the "accuracy" as the percentage of the query names correctly predicted. The next section shows experiment results and analysis on all the 16 name datasets.

4.5.4.2 Experimental Results on the Web Collected Name Datasets

Table 4.4 shows the mean and the standard deviation (StdDev) of the 10 name disambiguation accuracy trials on the "J Anderson" name dataset, using both approaches. Table 4.5 shows the similar trials on the "J Smith" name dataset. The experiment results on these two name datasets are similar, most likely due to the two name datasets having similar probability distributions, since most citations in both datasets are derived from

Scheme(Accuracy)	Seen		Unseen		Alone	
	Correct	Wrong	Correct	Wrong	Correct	Wrong
Coauthor71(63.4%)	64(100%)	0	3(20%)	12(80%)	4(12.1%)	29(87.9%)
Paper title words(76.8%)	86(83.5%)	17(16.5%)	0(%)	9(100%)	N/A	N/A
Publication Venue title words(72.3%)	79(83.2%)	16(16.8%)	2(11.8%)	15(88.2%)	N/A	N/A
Hybrid I(90.2%)	60(93.8%)	4(6.2%)	14(93.3%)	1(6.7%)	27(81.8%)	6(18.2%)
Hybrid II(93.8%)	64(100%)	0(0%)	14(93.3%)	1(6.7%)	27(81.8%)	6(18.2%)

Table 4.6. The name disambiguation performance on the "J Anderson" dataset, using five schemes of the attributes in the hierarchical naive Bayes approach. The first column is the scheme used and the associated overall accuracy. The other columns show the distribution (number and relative percentage under each category) of correct and incorrect name disambiguation in three categories "Seen", "Unseen" and "Alone" respectively. For the 4th and 5th row of the table, "Seen" means that the true name entity uses a subgroup of the paper/publication venue title words in the training data; "Unseen" means otherwise. For the other rows of the table, "Seen" means the existence of a previous coauthorship between the true name entity and at least one given coauthor in the test citation; "Unseen" means no existence of previous coauthorship between the true name entity and any coauthor in the test citation; "Alone" means the query citation has only a single author (the query author).

labeled homepages. We analyze the experiment results in detail as follows.

(1) Different attributes have different contributions for name disambiguation

Consider the "J Anderson" dataset as an example. Table 4.4 shows that using paper title words alone achieves higher average accuracy (77.9%, 82.9%) than using either coauthor (71.3%, 64.4%) or publication venue information alone (72.1%, 74.4%) with both approaches. Table 4.6 shows in detail one experiment using the naive Bayes approach; all other 9 experiments show similar results. We observe that authors in this dataset have higher probabilities of reusing title words than collaborating with previously seen coauthors. Table 4.6 shows an example of the probability distribution of each attribute.

For example, Row 4 in Column 2&3 (with header "Seen") shows that 92.0% ((86+17) out of 112) test citations reuse the words in paper titles; Row 5 in Column 2&3 shows that 84.8% ((79+16) out of 112) test citations reuse words in publication venue titles; and Row 3 in Column 2&3 shows that only 57.1% (64 out of 112) test citations have the previously seen coauthor relationship.

The above probability distribution indicates that authors in this dataset tend to use the same words for multiple papers, probably because multiple papers are about the same project. And the authors in some research areas such as Biology or Plant pathology tend to have a few places they prefer to submit papers. For example, J. Anderson 15 (Jim V. Anderson; J. Anderson 15 refers to the 15th table entry) publishes 37.5% (15 out of 40) of his papers in the same publication venue "Plant physiology". Such consistent information contained in the publication venue title helps name entity disambiguation more than the paper title words, especially when the name entities to be disambiguated have diverse research areas.

(2) The hierarchical-naive-Bayes-model-based approach better captures the coauthoring patterns of an author than the SVM-based approach

Table 4.4 and 4.5 show that the hierarchical-naive-Bayes-model-based approach (71.3%, 75.2% average accuracy) outperforms the SVM-based approach (64.4%, 60.0% average accuracy) when using coauthor information alone in both datasets. The reason is that the SVM-based approach is incapable of handling the cases when the test citation contains no coauthor seen in the training set. However, the hierarchical-naive-Bayes-model-based approach reasonably captures the probabilities of an author coauthoring with both previously seen and unseen coauthors. The prior of the author helps to disambiguate the single

author of a citation. For example, Table 4.6 also shows that using coauthor information alone disambiguates correctly 64 (100%) out of 64 query names with coauthors having previously seen coauthorship with the true name, 3 (20%) out of 15 query names with coauthors having no previously seen coauthorship with the true name, and 4 (12.1%) out of 33 query names authoring alone.

(3) "Hybrid II" performs best (93.5% average accuracy) among all five schemes using the hierarchical-naive-Bayes-model-based approach

The hierarchical naive Bayes probability model has the flexibility of manipulating citation attributes, and enables the easy construction of two hybrid schemes. Both "Hybrid I" and "Hybrid II" perform better than using each citation attribute alone in all experiments."Hybrid II" performs best (93.5% average accuracy) among all five schemes. The hybrid schemes perform better than using each citation attribute alone because three attributes together provide additional information. For example, using coauthor information alone, the system has limitations of predicting correctly the cases where no given coauthor in the test citation has a seen coauthorship with the given true name, or the query name has no coauthors. Comparing row 6 &7 with row 3 in column 4 & 6 of Table 4.6 shows that both hybrid schemes predict $34(= (14-3)+(27-4))$ extra citations, i.e., $34/(3+4) = 485.7\%$ extra citations correctly than using coauthor information alone for the cases when no previously seen coauthor relationship exists.

Row 6 in Column 2 of Table 4.6 shows that "Hybrid I" has less disambiguated names than using coauthor information alone (Row 3 in Column 2), where the previously seen coauthorship exists. This suggests that incorporating the title words information from papers and publication venues may add noise, and thus decreases the number of

correctly disambiguated names, from 64 to 60 in this case. This motivates our "Hybrid II" model, which preserves the disambiguation results obtained by using coauthor information alone when the previously seen coauthorship exists. Table 4.4 shows that "Hybrid II" improves the name disambiguation accuracy in "J Anderson" dataset from 91.3% to 93.5% on average.

(4) The SVM-based approach slightly outperforms the hierarchical-naive-Bayes-model-based approach overall

Except in the case of using coauthor information alone, the SVM-based approach slightly outperforms the hierarchical-naive-Bayes-model-based approach in both name datasets. The better performance by SVM-based approach is statistically significant, except in the case of using paper title words alone. One reason may lie in the nature of the two approaches. While the hierarchical-naive-Bayes-model-based approach models an author's writing patterns only based on the citations of this author, the SVM-based approach look at the citations of all authors and maximize the distinction between an author class and other author classes. Therefore, SVM-based approach can capture and highly rank the features unique to a class, while the hierarchical-naive-Bayes-model-based approach ranks the same features unique or not unique to an author class, assuming an author has the same distribution of these features.

For example, "*[James E. Smith], Kevin F. McCardle. Options in the Real World: Some Lessons Learned in Evaluating Oil and Gas Investments. Operations Research.*" is a test citation of "J Smith 2". The paper title keywords "evaluate" and "option" are unique to "J Smith 2" and are seen in training citations. However, "Hybrid I" of the hierarchial-naive-Bayes-model-based approach predicts "J Smith 2" as the second

Feature	Ranking(score) in SVM		Probability estimated by Bayes	
	J Smith 2	J Smith 5	J Smith 2	J Smith 5
evaluate	3 (0.18)	846 (-0.01)	0.72%	0.09%
option	11 (0.09)	888 (-0.01)	0.36%	0.09%
research	69 (0.01)	130 (0.03)	0.16%	4.33%

Table 4.7. The rankings (ranking scores) of three features by SVMs in author class "J Smith 2" and "J Smith 5", and the probabilities "J Smith 2" and "J Smith 5" use these three features , as estimated by the naive Bayes Model.

most likely author, and wrongly predicts "J Smith 5" as the most likely author due to his higher prior and higher probability of using the publication venue title keyword "Research". The "Hybrid I" of the SVM-based approach highly ranks features that are unique to "J Smith 2", and correctly classifies this citation to "J Smith 2". Table 4.7 shows the features ranked by SVM-based approach, and the probabilities "J Smith 2" and "J Smith 5" generate these features as estimated by the hierarchical naive Bayes probability model. For example, the keyword "evaluate" and "option" are respectively ranked as the 3rd and 11th most important feature of "J Smith 2", by the ranking algorithm described in Section 4.5.3.1.

4.5.4.3 Experimental Results on the DBLP Name Datasets

The 14 DBLP name datasets obtain similar disambiguation results to the "J Anderson" and "J Smith" datasets. However, we also observe the difference, and analyze the experiment results shown in Table 4.8 as follows.

Scheme	Coauthor		Paper title		Venue title		Hybrid I		Hybrid II
Approach	**Bayes**	**SVM**	**Bayes**	**SVM**	**Bayes**	**SVM**	**Bayes**	**SVM**	**Bayes**
A Gupta	84.4(2.5)	82.5(2.2)	73.5(2.2)	74.4(3.0)	55.1(1.7)	54.9(2.8)	88.3(2.6)	88.4(1.9)	91.2(2.7)
A Kumar	69.9(2.1)	55.5(5.1)	73.3(2.2)	78.2(3.2)	67.0(2.0)	63.7(3.9)	83.7(2.5)	85.8(3.2)	86.5(2.6)
C Chen	72.2(2.2)	64.6(1.3)	66.7(2.0)	64.8(2.2)	41.1(1.2)	43.3(2.1)	81.0(2.4)	83.6(1.4)	82.2(2.5)
D Johnson	77.3(2.3)	58.8(2.9)	78.6(2.4)	78.5(2.7)	59.8(1.8)	61.8(1.7)	86.2(2.6)	85.7(3.0)	83.5(2.5)
J Lee	72.4(2.2)	69.2(0.9)	69.0(2.1)	68.4(1.2)	46.3(1.4)	46.2(2.4)	84.9(2.5)	85.9(1.0)	83.5(2.5)
J Martin	69.1(2.1)	58.9(2.7)	54.4(1.6)	63.9(7.5)	58.7(1.8)	62.4(3.7)	84.1(2.5)	89.1(2.8)	82.0(2.5)
J Robinson	71.0(2.1)	61.2(4.0)	76.3(2.3)	77.3(3.7)	65.1(2.0)	66.5(3.3)	90.4(2.7)	90.0(2.1)	89.6(2.7)
J Smith	71.6(2.1)	63.3(1.6)	77.6(2.3)	78.5(1.4)	72.0(2.2)	72.4(1.4)	89.6(2.7)	89.5(1.2)	88.3(2.6)
K Tanaka	87.4(2.6)	78.8(3.1)	81.1(2.4)	87.0(2.3)	73.2(2.2)	71.8(2.4)	92.6(2.8)	95.0(1.1)	92.9(2.8)
M Brown	73.6(2.2)	66.3(3.9)	81.5(2.4)	83.6(3.3)	64.4(1.9)	62.3(6.8)	89.3(2.7)	90.1(3.7)	89.3(2.7)
M Jones	72.3(2.2)	56.7(2.9)	68.0(2.0)	71.0(4.1)	71.7(2.1)	71.8(3.8)	86.9(2.6)	89.6(1.8)	86.7(2.6)
M Miller	93.0(2.8)	87.1(2.1)	85.1(2.6)	89.3(1.2)	81.9(2.5)	79.8(2.2)	95.9(2.9)	97.5(0.7)	95.8(2.9)
S Lee	70.4(2.1)	69.8(1.9)	67.7(2.0)	67.2(2.3)	51.0(1.5)	52.5(1.2)	82.9(2.5)	82.6(1.4)	74.5(2.2)
Y Chen	79.0(2.4)	73.6(1.4)	69.4(2.1)	68.8(1.4)	52.2(1.6)	54.4(1.3)	86.0(2.6)	87.5(1.1)	86.9(2.6)
Mean	**76.0**	**67.6**	**73.0**	**75.1**	**61.4**	**61.7**	**87.3**	**88.6**	**86.6**
StdDev	**7.4**	**9.8**	**7.9**	**8.1**	**11.5**	**10.5**	**4.0**	**4.0**	**5.3**
P Value	**3.73E-05**		**0.03**		**0.58**		**0.01**		

Table 4.8. The name disambiguation accuracy (%), mean and standard deviation on 14 DBLP datasets of different names, using multiple schemes of attributes with both the hierarchical naive Bayes approach(Bayes) and the SVM approach(SVM); and the statistical significance (two tail P value) of the performance difference by the two approaches.

- Using coauthor information alone and using publication venue title alone achieves best and worst average accuracies over the 14 DBLP datasets, respectively. Therefore, the coauthor information seems to be robust to all datasets, including the web collected datasets. Publication venue title abbreviations seem to constrain the information of research areas to be revealed. Similar to the web collected datasets, the hybrid models perform better than using one type of attribute alone.

- Conforming to the previous observation, the hierarchical-naive-Bayes-model-based approach significantly outperforms the SVM-based approach when using coauthor information alone. This shows that the hierarchical-naive-Bayes-model-based approach well captures the coauthoring patterns of an author, especially the patterns

	PCseen	**PCunseen**	**PN**	**PKunseen**	**PJunseen**
J Anderson	30.0%	42.6%	27.4%	66.6%	53.3%
J Smith	29.7%	45.1%	25.2%	57.5%	46.8%
DBLP	57.3%	33.4%	9.3%	57.2%	50.7%

Table 4.9. The average conditional citation attribute probability distribution of an author X_i from the "J Anderson" dataset, "J Smith" dataset, and the DBLP datasets. (The probability estimation is shown in Section 4.5.2). PCunseen: the probability of X_i writing a future paper with previously unseen coauthors; PCseen: the probability of X_i writing a future paper with previously seen coauthors; PN: the probability of X_i writing a future paper alone; PKunseen: the probability of X_i using unseen words for a future paper title; PJunseen: the probability of X_i publishing a future paper in a publication venue (or proceeding) with different title words from previous publication venue titles.

of writing papers alone and writing a future paper with previous unseen coauthors. However, SVM-based approach outperform the hierarchical-naive-Bayes-model-based approach in other cases.

- The "Hybrid II" does not improve "Hybrid I" in DBLP datasets. One possible reason is that simplifying each name as first name initial and last name introduces name ambiguity. This is popular in the DBLP datasets. For example, different names "Sung Jin Kim " and "Seon-Kyu Kim" are simplified to the same name label "S Kim". Using coauthor information alone in this case does not always achieve accurate name disambiguation. More name information may help solve the problem. The other possible reason is that paper titles may not always add noise information.

4.5.5 Conclusions

This section describes two supervised learning approaches to disambiguate name entities in citations. The hierarchical-naive-Bayes-model-based approach determines the author with the highest posterior probability of writing the paper of a citation; the SVM-based approach classifies a test citation to the closest author class. Both approaches use three types of citation attributes: coauthor names, paper titles, and publication venue titles. Coauthor names appear to be the most robust attribute for name disambiguation; using coauthor information alone performs consistently well in all the 16 datasets. Using joint information of all utilized citation attributes outperforms using a type of citation attribute alone. Both approaches, using combinations of all utilized citation attributes, achieve around 90.0% accuracies in disambiguating all name datasets.

Both approaches have advantages. While the SVM-based approach highly ranks the features specific to an author class, the hierarchical naive Bayes probability model captures information beyond the seen features, e.g., the unseen coauthors and keywords, and the prior of an author. The hierarchial naive Bayes model especially well captures the coauthoring patterns of an author. The flexibility of manipulating different attributes is the advantage of such a probability model. To achieve similar effect to "Hybrid II" model in the hierarchical-naive-Bayes-model-based approach, the feature vector model usually needs to experimentally tune the weights for different attributes based on performance on training or validation datasets. Both approaches allow "non-existence call" if the

confidence of the prediction, i.e., the probability in the hierarchial-naive-Bayes-model-based approach, or the distance in the SVM classification, is too low. In this case the algorithms recommend a new name entity to the database.

Further improvements can be obtained, i.e. semantic word clustering on paper titles and publication venue titles [111]. A researcher usually has a research area or areas that do not change over a period of time, and his/her paper or submitted publication venue titles are closely related to his/her research topic. However, the paper and publication venue title words are sparse, and an author may not reuse a certain group of title words with high probabilities. Therefore, it is reasonable to cluster "similar" title words into research fields and model the probabilities that this researcher uses similar words in the paper title. Such a word cluster reduces feature sparseness, and has more robust probability estimates by averaging statistics for similar words [8]. Existing word clustering methods we can apply include methods based on the Word Net [10], distributional word clustering [8, 83, 25, 29], bipartite word clustering [113], and committee-based word clustering [68]. Research keywords classification schemes such as the ACM classification may also help to map different title words into a research category.

Both approaches can also be applied to the author disambiguation in the context of documents. More attributes can be used, such as the author's affiliation. Words and bigrams from the paper abstracts may also provide useful information for disambiguation. To address real-world problems, we would take wrong spelling and all other author name problems into account, to find the canonical name of an author. We would also like to disambiguate similar corporate names appeared in academic and publishing worlds, for example, the "Loyola" college.

Generating the labeled training data is the rather expensive price that has to be paid for supervised learning systems. However, we may save some labor by collecting training citations from manually organized databases, such as the DBLP Computer Science Bibliography, or collecting publication lists from researchers' home pages.

4.6 An Unsupervised Learning Method: Hierarchical Naive Bayes Mixture Model

As the information of canonical names corresponding to an ambiguous name label is not necessarily available, the supervised learning methods are limited in practical use. In this section, we study an unsupervised learning method, a hierarchical naive Bayes mixture model, which does not need labeled data for name disambiguation. The name disambiguation problem is accordingly formulated as partitioning collections of citations into clusters, with each cluster containing only citations authored by the same author, thus disambiguating authorship in citations to induce author name identities. As the choice of K can be an independent research issue, and this research focuses on the study of the performance of the clustering algorithms, we set K as the number of canonical names an ambiguous name label corresponds to in the dataset.

Clustering methods appear to be a natural solution for disambiguation problems. In the task of word sense disambiguation, a sense is often seen to correspond to a cluster, and instances of words with the same sense are expected to be part of the same cluster [83, 25, 67, 31, 68, 108]. This section introduces a clustering method based on the hierarchial-naive-Bayes-model-based approach introduced in section 4.5. The k-means algorithm is used for comparison.

4.6.0.1 The Mixture Model

We assume that a citation C_m is generated by a mixture of K components (canonical authors). As the choice of K can be an independent research issue, and this research focuses on the study of the hierarchical naive Bayes mixture model, we set K as the number of canonical names an ambiguous name label corresponds to in the labeled dataset. Equation (4.14) shows that the probability of citation C_m is equal to the weighted sum of C_m's probability for each canonical author X_i alone. $P(X_i)$ is the weight, or prior probability for each canonical author X_i.

$$P(C_m) = \sum_{i=1}^{K}(P(X_i) * P(C_m|X_i)) \tag{4.14}$$

Each of the K canonical authors is modeled by a hierarchical naive Bayes model as described in next section. We use the Expectation-Maximization (EM) algorithm to estimate the mixture model parameters, as described in the next section, with the target function of maximizing the likelihood of the citation dataset, i.e.,

$$\max(\sum_{m}(P(C_m))) \tag{4.15}$$

After the model parameters are estimated, we assign each citation C_m to the canonical author that maximizes $P(X_i|C_m)$. According to Bayes rule, each citation C_m is assigned to the canonical author that has the maximal probability of producing C_m, that is, $max\ (P(C_m|X_i) * P(X_i))$.

4.6.1 The Expectation-Maximization Algorithm

Step1. Initialization. Randomize and equally assign N citations (N is the total number of citations in the dataset) into K clusters. Estimate the following probabilities: the prior probability of each of the K components, $P(k)$ ($k \in \{1, \cdots, K\}$); the hierarchical conditional probabilities as shown in Figure 4.1 (e.g., $P(Co_j = 1|k)$, $P(Co_j = 0|k)$, $P(Seen_{jk} = 1|Co_j = 1, k)$, $P(Seen_{jk} = 0|Co_j = 1, k)$, $P(A_{jk}|Seen_{jk} = 1, Co_j = 1, k)$, $P(A_{jk}|Seen_{jk} = 0, Co_j = 1, k)$); and $P(C_m|k)$.

$$P(k) = \frac{1}{K} \tag{4.16}$$

Step2. E-step. Reassign all citations to each cluster according to the posterior probability of each cluster producing the citation C_m.

$$P(k|C_m) = \frac{P(C_m|k) * P(k)}{\sum_k (P(C_m|k) * P(k))} \tag{4.17}$$

Step3. M-step. Compute $P(k)$, hierarchical conditional probabilities (e.g., $P(Co_j = 1|k)$, $P(Co_j = 0|k)$, $P(Seen_{jk} = 1|Co_j = 1, k)$, $P(Seen_{jk} = 0|Co_j = 1, k)$, $P(A_{jk}|Seen_{jk} = 1, Co_j = 1, k)$, $P(A_{jk}|Seen_{jk} = 0, Co_j = 1, k)$), and $P(C_m|k)$. N is the total number of citations in the dataset.

$$P(k) = \frac{\sum_m (P(k|C_m))}{N} \tag{4.18}$$

Step4. If the algorithm converges ($||\sum_m(P(C_m)) - \sum'_m(P(C_m))|| < 0.1$), classify each citation C_m to the component(cluster) k that maximizes $P(k|C_m)$. Otherwise, continue step2 and step3.

$$P(C_m) = \sum_k P(C_m|k) * P(k) \tag{4.19}$$

4.6.2 The K-means Algorithm

To help studying the performance of our algorithms on name disambiguation, we choose the K means algorithm as the baseline algorithm for comparison. In K means algorithm, each citation is represented by a feature vector, with each coauthor name and each keyword of the paper title and the publication venue title as a feature of the vector. The weight of each feature is chosen as its "tf.idf" value. The distance measure used in K means algorithm is the Euclidean distance between the citation feature vector to be assigned and cluster centroid feature vector.

The K means algorithm is briefly described in the below.

Step 1. Initialization. Randomize and equally assign N citations (N is the total number of citations in the dataset) into K clusters.

Step 2. Calculate the centroid feature vector of each cluster, m_k, by averaging the feature vectors of all citations in the cluster. N_k is the number of citations in cluster k. $\vec{C_m}$ refers to the feature vector of a citation C_i.

$$\vec{m_k} = \frac{\sum_{i=1}^{N_k}(\vec{C_m})}{N_k} \tag{4.20}$$

Step 3. Assign each citation to its closest cluster, measured by the Euclidean distance. p is the dimension of a feature vector. C_{ij} refers to the value of the feature j in the vector of citation C_i.

$$k = argmin_k||\vec{C_m} - \vec{m_k}|| \tag{4.21}$$

$$||\vec{C_m} - \vec{m_k}|| = \sum_{j=1}^{p}(C_{ij} - m_{kj})^2 \tag{4.22}$$

Step 4. Step 2 and 3 are iterated until the algorithm converges (when the cluster assignment does not change).

4.6.3 Cluster Semantically Similar Words

The hierarchical naive Bayes model takes each word of the paper title or the publication venue title as a feature. These title words contain the information such as the research field, keywords in the research direction, and the preference of title word usage of an author X_i. However, the paper and publication venue title words are sparse, and an author may not reuse a certain group of words with high probabilities. Therefore, it is reasonable to cluster the semantically similar words and model the probability that an author uses the similar words for his/her paper title. In another word, we cluster the paper title words and publication venue title words, and then replace each title word by its cluster label, which we call "feature transformation", before applying the hierarchical-naive-Bayes-model-based name disambiguation approach.

Our experiments choose the CBC (Clustering By Committee) word clustering algorithm by Pantel and Lin [68], because of its good performance on sparse feature

space as demonstrated in Pantel and Lin's paper. The CBC algorithm uses the following word similarity measure.

The CBC algorithm represents each word w by a feature vector. Each context (neighboring) word of w is a feature, represented by c; the value of the feature c is the pointwise mutual information between w and c, defined as:

$$mi_{w,c} = \frac{\frac{F_c(w)}{N}}{\frac{\sum_i(F_i(w))}{N} * \frac{\sum_j(F_c(j))}{N}} \tag{4.23}$$

$F_c(w)$ is the frequency count of a word w occurring in the context c. $N = \sum_i \sum_j F_i(j)$ is the total frequency counts of all words and their contexts. The mutual information is multiplied by the following discount factor to alleviate the bias the mutual information has towards infrequent words/features. The similarity between two words are the cosine coefficient of their mutual information vectors.

$$\frac{F_c(w)}{F_c(w)+1} * \frac{min(\sum_i F_i(w), \sum_j F_c(j))}{min(\sum_i F_i(w), \sum_j F_c(j)) + 1} \tag{4.24}$$

4.6.4 Experiments

We apply both K means algorithm and the hierarchical naive Bayes mixture model to both types of datasets. Tables 4.10, 4.11 show the average and the best results of 10 times experiments on both types of datasets respectively. Figure 4.2 shows the histogram of the best results achieved from 10 times experiments by both methods. The hierarchical naive Bayes mixture model is shown to outperform the K means algorithm on all datasets. Although both algorithms are prone to local minima, the mixture model

appears to be a better fit for the problem of name disambiguation in author citations than the K means algorithm. The main reason is that our hierarchical naive Bayes model captures the author patterns that are not easily incorporated into feature vector space model that is used by K-means algorithm. These author patterns are the prior probability of an author, the probability that an author writes papers alone, the probabilities that an author writes a future paper with previously unseen coauthors, and the probabilities that an author writes a future paper using previously unused keywords.

We applied the CBC word clustering algorithm to clustering paper title words and publication venue title words. We then made "feature transformation" to titles by replacing each title word of a citation by its cluster label. We applied the hierarchical-naive-Bayes-mixture-model-based method to these citations that are "feature transformed". Tables 4.10 and 4.11 list the name disambiguation results on the DBLP datasets before and after the feature transformation. The word clustering algorithm is shown to improve the name disambiguation results on datasets of "A Kumar", "D Johnson", "J Smith", "K Tanaka", and "Y Chen". However, word clustering does not improve the name disambiguation results on all the datasets. A possible reason is that the word clustering gathers information and also loses information. Choice of the size of a cluster affects the balance of information gain and information lose. Large size cluster seems to gather more information than smaller size cluster. However, large size cluster can lose more information than the smaller size cluster.

	Average		Best	
	K means	**Mixture model**	**K means**	**Mixture model**
A. Gupta	29.7%	47.6%	37.1%	56.2%
A. Kumar	43.0%	46.3%	55.6%	60.0%
C. Chen	24.6%	38.2%	34.9%	45.4%
D. Johnson	41.2%	44.5%	56.8%	55.8%
J. Lee	19.0%	49.5%	24.9%	53.3%
J. Martin	39.2%	65.6%	46.7%	80.3%
J. Robinson	28.7%	57.1%	38.7%	72.3%
J. Smith	34.6%	61.7%	48.4%	68.2%
K. Tanaka	50.1%	59.5%	70.8%	66.3%
M. Brown	40.3%	66.0%	53.6%	78.3%
M. Jones	36.8%	65.7%	46.1%	76.5%
M. Miller	50.6%	59.8%	66.7%	68.9%
S. Lee	20.4%	46.4%	23.8%	51.7%
Y. Chen	28.8%	49.0%	46.1%	51.8%
Mean	**34.8%**	**54.1%**	**46.4%**	**63.2%**
StdDev	**10.0%**	**9.2%**	**13.9%**	**11.2%**
P Value	**5.41E-06**		**0.00047**	

Table 4.10. The name disambiguation accuracies(%) on 14 DBLP name datasets achieved by both methods. "Mixture model" refers to our hierarchical naive Bayes mixture model. "Avg" means average results. "StdDev" means standard deviation. "P value" is the two tail value result from T-test.

	Average		Best	
	K means	**Mixture model**	**K means**	**Mixture model**
J. Anderson	30.0%	57.6%	41.0%	65.6%
J. Smith	31.2%	59.8%	48.5%	65.4%
Mean	**30.6%**	**58.7%**	**44.8%**	**65.5%**
StdDev	**0.85%**	**1.56%**	**5.30%**	**0.14%**
P Value	**0.011**		**0.117**	

Table 4.11. The accuracies(%) on disambiguating 2 web collected name datasets "J Anderson" and "J Smith". "Mixture model" refers to our hierarchical naive Bayes mixture model. "Avg" means average results. "StdDev" means standard deviation. "P value" is the two tail value result from T-test.

4.7 An Unsupervised Learning Method:

K-way Spectral Clustering with QR Decomposition

4.7.1 Introduction

K-means, naive Bayes and Gaussian mixture model are widely used clustering methods. However, these methods are prone to local minima, and initial data partitions

	Average		Best	
	Original citations	**Word-clustered citations**	**Original citations**	**Word-clustered citations**
A. Gupta	47.6%	46.8%	56.2%	54.0%
A. Kumar	46.3%	48.1%	60.0%	56.7%
C. Chen	38.2%	41.2%	45.4%	45.2%
D. Johnson	44.5%	45.6%	55.8%	60.9%
J. Lee	49.5%	45.3%	53.3%	48.6%
J. Martin	65.6%	66.4%	80.3%	77.3%
J. Robinson	57.1%	57.6%	72.3%	66.2%
J. Smith	61.7%	62.7%	68.2%	71.5%
K. Tanaka	59.5%	61.6%	66.3%	73.3%
M. Brown	66.0%	65.5%	78.3%	80.0%
M. Jones	65.7%	62.9%	76.5%	70.1%
M. Miller	59.8%	59.5%	68.9%	63.2%
S. Lee	46.4%	39.5%	51.7%	42.1%
Y. Chen	49.0%	49.2%	51.8%	52.3%
Mean	**54.1%**	**53.7%**	**63.2%**	**61.5%**
StdDev	**9.2%**	**9.5%**	**11.2%**	**12.1%**
P Value	**0.62**		**0.21**	

Table 4.12. The accuracies(%) on disambiguating 14 name datasets from DBLP, using the hierarchical-naive-Bayes-model-based method, when the datasets use original words (Original citations), and when the datasets have title words clustered (Word-clustered citations).

can seriously impact the clustering results [109]. Spectral clustering methods use eigen-decomposition techniques and find an approximation of the global optimal solution in terms of defined criteria function [109, 112]. We propose using K-way spectral clustering [112], a graph model that has been successfully applied to data mining and cluster analysis, for name disambiguation in citations as described in detail in next section. Our experiments show better results than both K-means and the hierarchical naive Bayes mixture.

The contribution in the research reported in this section is the selection of features for name disambiguation and a novel application of a K-way spectral clustering method to name disambiguation in author citations. Through extensive experimentation, we gain

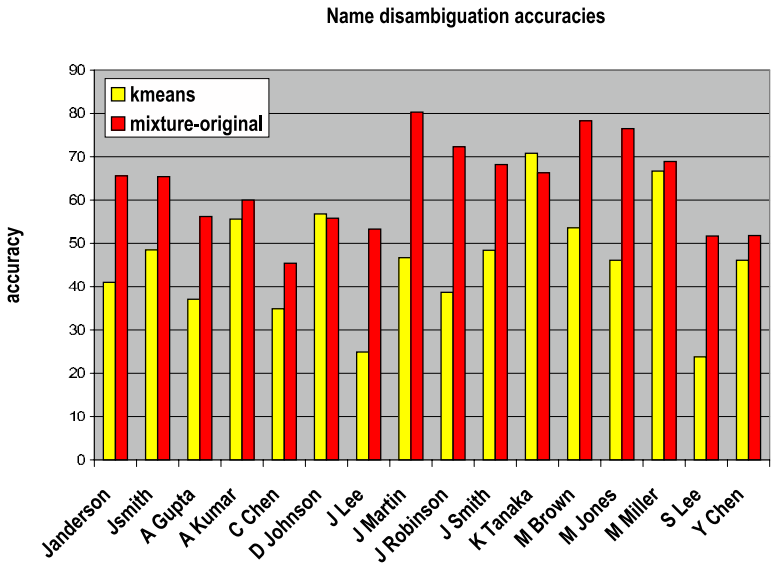


Fig. 4.2. The best name disambiguation accuracies of 10 times experiments on 16 name datasets by the naive Bayes mixture model and the K means algorithm.

insights in the factors that affect the name disambiguation performance, and propose possible solutions for disambiguation performance improvement.

4.7.2 K-way Spectral Clustering with QR Decomposition

Spectral clustering methods compute eigenvalues and eigenvectors of a Laplacian matrix (or singular values and singular vectors of certain matrix) related to the given graph, and construct data clusters based on such spectral information [33, 53, 80, 86, 91]. Recent research on theoretical understanding of spectral methods found that important algebraic structures in general exist in the eigenvectors and in the singular vector matrices for data clusters [6, 112]. In particular, Zha et. al [112] found that minimizing a sum-of-square cost function can be reformulated as a trace maximization problem associated

with the Gram matrix of the data vectors. They show that a partial eigen decomposition of the Gram matrix obtains the *global* optimal solutions for a relaxed version of the trace maximization problem. Accordingly, the cluster assignment for each data vector can be found by computing a pivoted QR decomposition of the eigenvector matrix. The K-way spectral clustering with QR decomposition is shown in their experiments to outperform the K-means algorithm [112].

Next we describe the spectral clustering method for experiments to cluster citations of the same name label but different authors. We model each citation as a node in an undirected graph. Each edge (i, j) in the graph is assigned a weight that reflects the similarity between two citations i and j. The name disambiguation problem for author citations is defined as a partition of the graph so that citations that are more similar to each other, e.g. authored by the same author, belong to the same cluster.

4.7.2.1 Citation Matrix and Feature Design

We observe that an author's citations usually reveal his or her identification information, such as the author's research area, and his or her individual patterns of co-authoring. We use three types of citation attributes to design features for name disambiguation: co-author names, paper titles, and publication venue titles. A feature is a component of a citation attribute, e.g., a co-author name or a pre-processed word in the title of a paper or publication venue. It should be noted that our technique can also be extended to use other information, e.g., the affiliations and addresses of authors.

We construct citation vectors for each name dataset. With m features in the name dataset, each citation can be represented as a m-dimensional vector, i.e., $M =$

$(\alpha_1, \cdots, \alpha_m)$. If the ith feature in the dataset appears in citation M, α_i is the feature i's weight. Otherwise, $\alpha_i = 0$. We study two types of feature weight assignment, the usual "TFIDF"; and the normalized "TF" ("NTF"), where

$$ntf(i,d) = \frac{freq(i,d)}{max(freq(i,d))} \tag{4.25}$$

$freq(i,d)$ refers to the term frequency of feature i in a citation d. $max(freq(i,d))$ refers to the maximal term frequency of feature i in any citation d. With the "NTF" (normalized "TF") scheme, the weights of features with different ranges of values are normalized. The "NTF" scheme has been shown to improve the classification performance [45]. Using completely unsupervised learning methods, we do not have training data to learn the weights for different type of features. However, we propose combining supervised learning methods in our future work for automatic feature weight assignment. The Gram matrix of the citation vectors represents the pairwise cosine similarities between citations. We apply the K way spectral clustering algorithm to the Gram matrix as described in the following two subsections.

4.7.2.2 Spectral Relaxation

Given a set of m-dimensional citation vectors α_i, $i = 1, \cdots, n$, we form the m-by-n citation matrix $A = [\alpha_1, \cdots, \alpha_n]$. A partition Π of the citation vectors can be written in the following form

$$AE = [A_1, \cdots, A_k], A_i = [{\alpha_1}^{(i)}, \cdots, {\alpha_{s_i}}^{(i)}], \tag{4.26}$$

where E is a permutation matrix, and A_i is m-by-s_i, i.e., the ith cluster contains the citation vectors in A_i. For a given partition Π in Equation 4.26, the associated sum-of-squares cost function is defined as

$$ss(\Pi) = \sum_{i=1}^{k} \sum_{s=1}^{s_i} ||\alpha_s^{(i)} - m_i||^2, m_i = \frac{\sum_{s=1}^{s_i} \alpha_s^{(i)}}{s_i}, \tag{4.27}$$

i.e., m_i is the mean vector of the citation vectors in cluster i. It was shown in [112] that the minimization of the above sum-of-square cost function can be formulated as a relaxed maximization problem

$$max \quad [trace(X^T A^T AX)], \tag{4.28}$$

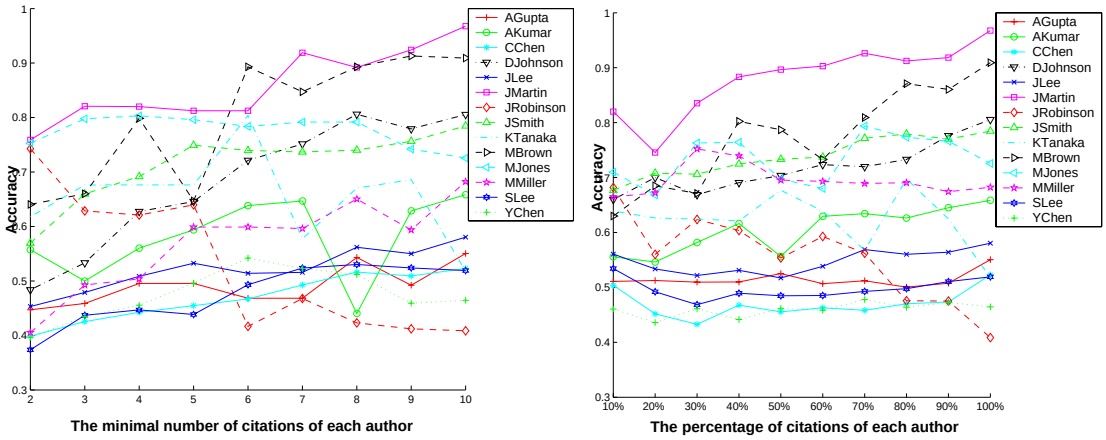
where $X^T X = I_k$ and X can be an arbitrary orthonormal matrix. It turns out that the above trace maximization problem has a closed-form solution.

Theorem. *(Ky Fan) Let H be a symmetric matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$, and the corresponding eigenvectors $U = [u_1, \cdots, u_n]$. Then*

$$\lambda_1 + \cdots + \lambda_k = \max_{X^T X = I_k} trace(X^T HX). \tag{4.29}$$

Moreover, the optimal X^* is given by $X^* = [u_1, \cdots, u_k]Q$ with Q an arbitrary orthogonal matrix.

It follows from the above theorem that we need to compute the largest k eigenvectors of the Gram matrix $A^T A$. Let X_k be the n-by-k matrix consisting of the largest eigenvectors of $A^T A$. Each row of X_k corresponds to a citation vector, and the above



(a) (b)

Fig. 4.3. Name disambiguation accuracies that change with the variation of the dataset size. X axis in Figure (a) and (b) shows the two different types of dataset size variations. Y axis represents name disambiguation accuracy using the "TFIDF" feature weighting. Lines of different colors and shapes represent different name datasets.

process can be considered as transforming the original citation vectors in a m-dimensional space to new citation vectors in the k-dimensional space. However, the goal here is not to reconstruct the citation matrix using a low-rank approximation but rather to capture its cluster structure, as shown in the next subsection.

4.7.2.3 Cluster Assignment Using Pivoted QR Decomposition

Assume that the best partition of the citation vectors in A that minimizes $ss(\Pi)$ is given by $A = [A_1, \cdots, A_k]$, where each sub matrix A_i corresponds to a cluster. The

Gram matrix of A can be written as

$$A^T A = \begin{pmatrix} A_1^T A_1 & 0 & \cdot & 0 \\ 0 & A_2^T A_2 & \cdot & 0 \\ 0 & 0 & \cdot & A_k^T A_k \end{pmatrix} + E \equiv B + E. \tag{4.30}$$

When the overlap among clusters represented by the sub matrices A_i is small, the norm of E will be small compared with the block diagonal matrix B in the above equation. Let the largest eigenvector of $A_i^T A_i$ be y_i, and

$$A_i^T A_i y_i = u_i y_i, ||y_i|| = 1, i = 1, \cdots, k, \tag{4.31}$$

then the columns of the matrix

$$Y_k = \begin{matrix} s_1 \\ s_2 \\ \vdots \\ s_k \end{matrix} \begin{pmatrix} y_1 & & & \\ & y_2 & & \\ & & \ddots & \\ & & & y_k \end{pmatrix} \tag{4.32}$$

span an invariant subspace of B. Let the eigenvalues and eigenvectors of $A^T A$ be $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n, A^T A x_i = \lambda_i x_i, i = 1, \cdots, n$. After some manipulation, it can be shown that

$$X_k^T \equiv [x_1, \cdots, x_k] = Y_k V + O(||E||), \tag{4.33}$$

where V is an k-by-k orthogonal matrix. Ignoring the $O(||E||)$ term, we see that

$$X_k^T = [\underbrace{y_{11}v_1, \cdots, y_{1s_1}v_1}_{\text{cluster 1}}, \cdots, \underbrace{y_{k1}v_k, \cdots, y_{ks_k}v_k}_{\text{cluster k}}], \tag{4.34}$$

where $y_i^T = [y_{i1}, \cdots, y_{is_i}]$, and $V^T = [v_1, \cdots, v_k]$. A key observation is that all v_i are orthogonal to each other. Once we have selected a v_i, we can jump to other clusters by looking at the orthogonal complement of v_i. Also notice that $||y_i|| = 1$, so the elements of y_i can not be all small. A robust implementation of the above idea can be obtained as follows: we pick a column of X_k^T which has the largest norm, say, it belongs to cluster i; we then orthogonalized the rest of the columns of X_k^T against this column. For the columns belonging to cluster i the residual vector will have small norm, and for the other columns the residual vectors will tend to be not small. We then pick another vector with the largest residual norm, and orthogonalized the other residual vectors against this residual vector. The process can be carried out k steps, and it turns out to be exactly QR decomposition with column pivoting applied to X_k^T, i.e., we find a permutation matrix P such that

$$X_k^T P = QR = Q[R_{11}, R_{12}], \tag{4.35}$$

where Q is a k-by-k orthogonal matrix, and R_{11} is a k-by-k upper triangular matrix. We then compute the matrix

$$\hat{R} = R_{11}^{-1}[R_{11}, R_{12}]P^T = [I_k, R_{11}^{-1}R_{12}]P^T. \quad (4.36)$$

The cluster membership of each citation vector is determined by the row index of the largest element in absolute value of the corresponding column of $\hat{R}$.

4.7.3 Experiments

4.7.3.1 Experiment Design

For each name dataset, we vary the size of the datasets in two different ways. The first selects the authors associated with at least a minimal number of citations (as shown by the columns of Table 4.3 in section 4). The second randomly selects a percentage (from 10% through 100%, with step size of 10%) of the citations of each author from the dataset containing authors that have at least 10 citations. We compare the disambiguation accuracy achieved in each size variation of the datasets to study the effect of dataset size on name disambiguation. In each size variation of the dataset, we applied the K-way spectral clustering algorithm and we compare two schemes of feature weighting: the "TFIDF" and "NTF" schemes. We also study the contribution of each citation attribute on name disambiguation, by using co-author names alone, paper title words alone, and publication venue title words alone, respectively. Then we investigate the effect of the amount of name information on disambiguation, by representing the first name with first name initial and first three characters of the first name, respectively. As the choice of number of clusters could be an important yet separate research issue, and

is not the focus of our current work, we pre-defined the number of clusters as labeled. That is, if there are N correct clusters, the dataset is clustered into N clusters.

4.7.3.2 Experimental Results on the DBLP Name Datasets

Effect of Dataset Size on Name Disambiguation Disambiguation accuracy, as shown in Figure 4.3, changes with the two types of dataset size variations, as described in section 4.7.3.1. For each dataset from the second type of size variation, we report the average accuracy of 10 times experiments, where in each experiment we randomly select a certain percentage of the citations of each author. The results show that the increase of author citations generally improves the disambiguation performance. For example, the accuracy of disambiguating "J. Martin" increases from 82% to 96.8% when we increase the percentage of citations of each " J. Martin" from 10% to 100%. However, results on the "J. Robinson" dataset show the opposite trend. We observe that two "J. Robinson"s with the largest number of citations both publish papers on the topic of "databases". These two "J. Robinson"s are always clustered together. It appears that the increase in citations in this case introduces errors and decreases the disambiguation accuracy. To resolve this, we probably need more features, such as author's affiliations for successful disambiguation. Overall, the experiments on disambiguating "M. Jones", "D. Johnson", "M. Brown" and "M. Miller" achieved higher accuracies than on other names, such as the four popular Asian names, "J. Lee", "S. Lee", "C. Chen", and "Y. Chen". Table 4.13 shows the detailed results on each name dataset with the second type of dataset size variation.

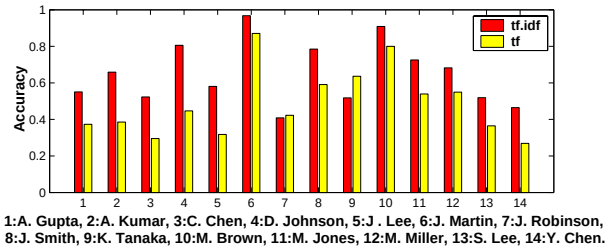


Fig. 4.4. Name disambiguation accuracies (Y axis) when using two feature weighting schemes. X axis represents 14 name datasets. For each name dataset, the left bar represents the usual "TFIDF" feature weighting and the right bar represents the "NTF" feature weighting.

Name	**10%**	**20%**	**30%**	**40%**	**50%**	**60%**	**70%**	**80%**	**90%**	**100%**
A. Gupta	51.1	51.3	51.0	51.0	52.5	50.7	51.2	50.1	50.8	53.9
A. Kumar	55.6	54.6	58.2	61.6	55.7	63.0	63.4	62.6	64.5	64.3
C. Chen	50.4	45.2	43.3	46.8	45.6	46.3	45.8	47.0	47.3	50.6
D. Johnson	66.0	70.0	66.9	69.1	70.4	72.4	72.0	73.3	77.6	79.1
J. Lee	56.1	53.4	52.2	53.1	51.7	53.8	56.9	56.0	56.4	56.2
J. Martin	82.0	74.5	73.6	88.3	89.7	90.3	92.6	91.1	91.8	96.8
J. Robinson	68.2	56.0	62.4	60.4	55.4	59.3	56.2	47.6	47.5	39.2
J. Smith	67.7	70.8	70.6	72.5	73.3	73.7	77.2	77.9	77.0	77.4
K. Tanaka	63.9	62.7	62.4	62.1	67.7	63.9	56.7	69.8	62.5	50.8
M. Brown	63.0	68.5	67.1	80.2	78.7	73.3	81.0	87.1	86.0	87.0
M. Jones	71.0	66.8	76.3	76.4	70.0	68.0	79.4	77.3	76.6	70.6
M. Miller	66.5	67.2	75.2	74.0	69.5	69.3	68.9	69.1	67.4	67.4
S. Lee	53.4	49.2	46.9	48.9	48.5	48.5	49.3	49.8	51.1	50.4
Y. Chen	46.1	43.6	46.1	44.2	46.2	45.9	47.8	46.4	47.3	45.5
Mean	61.5	59.6	60.9	63.5	62.5	62.7	64.2	64.7	64.6	63.5
Std	9.4	9.9	11.0	13.2	13.0	12.4	14.0	14.9	14.7	16.8

Table 4.13. Name disambiguation accuracies (%) that change with the dataset size variation of the 14 DBLP name datasets. i% means that i% citations of each author is randomly selected. "Std" means standard deviation.

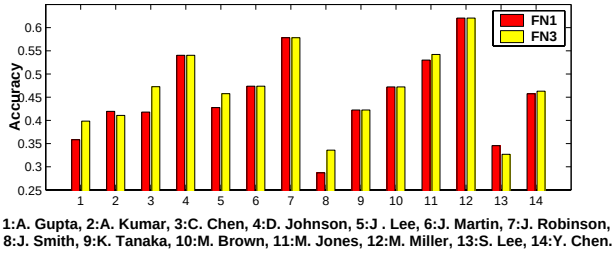


Fig. 4.5. Name disambiguation accuracies (Y axis) when using different amount of first name information. The X axis represents 14 name datasets. For each name dataset, the left bar (**FN1**) represents the result of using the first name initial; the right bar (**FN3**) represents the result of using first three characters of the first name.

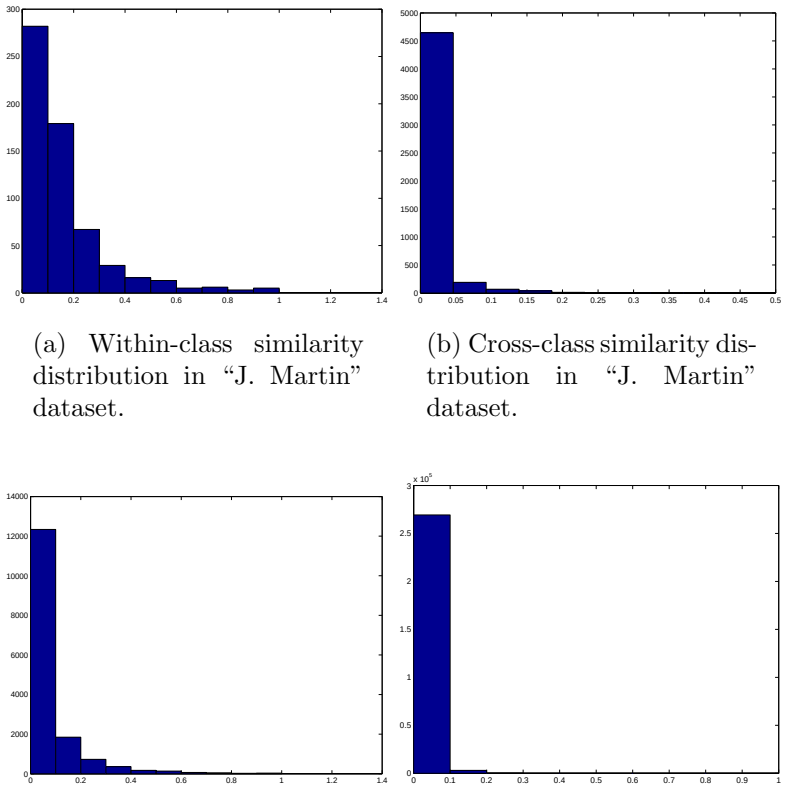
"TFIDF" vs. "NTF" In each size variation of the 14 DBLP name datasets, we compare two feature weighting schemes: "TFIDF" and the "NTF". Experimental results show that "TFIDF" performs better than "NTF" in general. Figure 4.4 shows an example on the 14 DBLP name datasets containing authors that have at least 10 citations. Experiments on other size variations of the datasets show similar results. "TFIDF" outperforms "NTF" because of the nature of the weighting schemes. "TFIDF" considers not only the frequency of a feature in a citation but also the distribution of a feature in all the citations of a name dataset. "NTF" considers only the feature frequency in one citation, which is limited by the fact that there seems to be very few words that are repeated in a single citation. As such, the "TFIDF" scheme better captures features specific to an author than "NTF" does. This indicates that a good feature weighting is important to the performance of name disambiguation. Improvements may be achieved using better feature weighting techniques such as Log Entropy [7].

Effect of Amount of Name Information on Disambiguation Simplifying each name with the first name initial and last name introduces name ambiguity. For example, the names "Sung Jin Kim" and "Seon-Kyu Kim" are simplified to the same name label "S. Kim". To investigate this effect, we did another set of experiments, representing the first name by its first three characters. We observe that most names from the DBLP database have complete first name information, while web collected publication lists contain many names that are in the format of first name initial and last name. Such inconsistent name formats cause one author to be represented by two different features and introduces name ambiguity. Therefore, we only report experimental results on all citations collected from the DBLP database Bibliography in which we vary the representation of first names. The citation vectors are constructed with only co-author names, and we do not consider the cases when an author has no co-authors. Figure 4.5 shows the results on the datasets that contain all author citations. Representing the first name by its first three characters improves disambiguation accuracy for most names, e.g. "A. Gupta", "C. Chen", "J. Lee", "J. Smith", "M. Jones" and "Y. Chen". We observe that many different co-authors in these datasets have the same name label in the simplified format of first name initial and last name, e.g. 18.7% different co-authors in the "C. Chen" dataset, 29.5% co-authors in the "J. Lee" dataset, and 12% co-authors in the "Y. Chen" dataset. Therefore, adding additional name information may decrease name ambiguity, and improve the disambiguation accuracy. However, we notice the classification accuracy drops on the specific name datasets "A. Kumar" and "S. Lee" when representing the first name by its first three characters. Two reasons may explain why. The first is that DBLP citations still have inconsistent name representations for

Name	Coauthor 1	Coauthor 2	PTitle	Venue title
A. Gupta	37.9%	39.8%	47.7%	24.7%
A. Kumar	25.7%	34.0%	61.0%	45.2%
C. Chen	33.3%	37.3%	43.7%	23.7%
D. Johnson	31.9%	41.2%	53.4%	50.0%
J. Lee	38.8%	45.1%	38.1%	19.6%
J. Martin	37.9%	62.5%	50.0%	65.2%
J. Robinson	41.2%	53.0%	43.2%	37.2%
J. Smith	46.7%	58.4%	44.0%	24.7%
K. Tanaka	49.6%	54.5%	68.6%	46.5%
M. Brown	50.4%	57.4%	61.7%	36.5%
M. Jones	43.9%	61.8%	50.2%	33.5%
M. Miller	52.4%	53.7%	52.4%	53.0%
S. Lee	34.3%	36.1%	37.7%	30.4%
Y. Chen	37.3%	43.1%	31.2%	19.8%
Mean	40.1%	48.4%	48.8%	36.4%
Std	7.7%	10.0%	10.3%	13.9%

Table 4.14. Name disambiguation accuracies using co-author information alone, paper title words alone (PTitle) and publication venue title words (Venue title)alone. "Std" - Standard Deviation. Column "Coauthor 1" considers the names that do not have co-authors as being incorrectly disambiguated; "Coauthor 2" does not consider the cases where names have no co-authors.

the same author, for example, the two formats "W. Tsai" and "Wen Tsai" for the same author. Simplifying names in the first name initial and last name format, however, represents the above two names as the same. The second reason is name misspellings. For example, "Kohji Zettsu" is misspelled in some citations as "Koji Zettsu". Representing the first name by its first three or more characters keeps such name misspelling, and incorrectly recognizes the above two name expressions as different. This indicates that combining techniques on duplicate string detection [15, 99, 17, 85] may improve the name disambiguation performance.



(a) Within-class similarity distribution in "J. Martin" dataset.

(b) Cross-class similarity distribution in "J. Martin" dataset.

(c) Within-class similarity distribution in "C. Chen" dataset.

(d) Cross-class similarity distribution in "C. Chen" dataset.

Fig. 4.6. The histogram of within-class and cross-class similarity distribution in "J. Martin" and "C. Chen" datasets. X axis represents the similarity value. Y axis represents the number of citation pairs from the same class (within-class) or from different classes (cross-class) that have the corresponding similarity value.

Coauthor Name vs. Paper Title vs. Publication Venue Title We achieved different disambiguation accuracies using each citation attribute alone. Table 4.14 shows an example on the 14 DBLP name datasets containing authors who have at least 10 citations. Experiments on other size variations of the datasets show similar results. Because

the names without co-authors can not be disambiguated by using co-author names alone, we evaluate the performance using two methods. The first method considers the names that do not have co-authors as being incorrectly disambiguated, as shown in column "Coauthor 1". The second method, shown in column "Coauthor 2", does not consider the cases when authors have no co-authors. "Coauthor 2" in Table 4.14 shows that using co-author information alone outperforms using only paper title words or publication venue title words in most name datasets. We hypothesized that the publication venue title information is more stable than paper title information, because an author may not reuse certain keywords for paper titles, and paper titles usually contain sparse information. Some paper titles, for example, "Where am I?", give little information about the author's research topic. Surprisingly, Table 4.14 shows that using publication venue title words alone generally performs worse than using paper title words alone. The possible reasons are the following. First, the publication venue title information is not always available in the datasets, or is parsed wrong. Second, "Ph.D. Dissertation", as parsed as the publication venue title, does not reveal the author's research area. Third, different publication venue titles may share the same abbreviation. For example, "IJCS" can refer to "International Journal of Comparative Sociology", or "International Journal of Communication Systems". Simply mapping the publication venue title abbreviation to the entry of a publication venue title full name database may introduce misleading information. It would be helpful if we could disambiguate publication venue title abbreviations by the context information such as the topic of the paper. Fourth, the full publication venue title information we obtain does not cover all the publication venue

title abbreviations in the datasets. This may under-exploit the publication venue information. Fifth, since most authors from DBLP datasets are from the Computer Science community, different authors are very likely to have the same or similar research area and publish papers in the same place. In this case, the publication venue title information is not discriminative. According to the above different contributions made by different citation attributes, we can automatically tune different weights for different attributes for improvement, as shown in previous work [15, 99, 17].

Effect of Author Research Area Diversity on Disambiguation We observe that many authors from the DBLP datasets have close research areas. For example, over 25% of authors in each name dataset of all author citations publish papers about "networks". For example, 36.1% (31 out of 86) "S. Lee" and 39.4% (28 out of 71) "Y. Chen" publish papers about "networks". "Databases" is another popular research topic. 24.0% (24 out of 100) of "J. Lee", 29.0% (9 out o 31) "J Smith", and 33.3% (4 out of 12) of "J. Robinson" publish papers about "databases". Correspondingly, many authors share words of the same word stem such as "network", "database", "comput" and "system". Different authors also publish papers in the same publication venues. Such common words from publication venue titles can be considered as "ambiguous information", and make accurate clustering hard. It is even more challenging to distinguish two authors of the same name label who co-author the same paper, as shown by the following example, "Chien-Chang Chen, Chaur-Chin Chen. Filtering methods for texture discrimination. Pattern Recognition Letters. 1999."

We consider each author as a class, and plot the within-class and the cross-class similarity distributions using the histogram for each name dataset. The ideal case is that the within-class similarity is distributed around "1" and the cross-class similarity is distributed around "0". Figure 4.6 shows that the difference between the within-class and cross-class similarity distribution for "C. Chen" is less than that of the "J. Martin" dataset. This explains why the disambiguation accuracy on "C. Chen" is worse than that on "J. Martin". A possible solution for improvement can be a set of features that enlarge the differences between citations of different authors.

4.7.3.3 Experimental Results on the Web Collected Name Datasets

Given DBLP's narrowness of topical coverage, name disambiguation on DBLP databases seems to be more challenging than it might be using other citation databases. The task of name disambiguation appears to be even more difficult when the nationalities that occur most frequently have a relatively small set of family names. To see results from other domains where the population of possible names may be larger, we have also conducted another two sets of experiments on the second type of citations, i.e. 11 "J Smith" of 338 citations and 15 "J Anderson" of 229 citations. We achieved 82.3% and 71.3% accuracy respectively on these two datasets using spectral clustering, better than both the K means algorithm and the hierarchical naive Bayes mixture model. with "TFIDF" feature weighting schema. This also shows that higher disambiguation accuracy can be achieved using only the three citation attributes when ambiguous authors have more diverse research areas.

4.7.4 Conclusions and Discussion

We study the name disambiguation in author citations using a K-way spectral clustering method with QR decomposition for cluster assignment. We also study several factors that may affect the disambiguation performance, such as feature weight assignment, dataset size, the amount of name information, and the author research area diversity. The spectral method outperforms k-means algorithm and the hierarchical naive Bayes mixture model.

The experimental results also show that as expected, the more features used (coauthor names, paper and publication venue title words) in author classification, the better the classification accuracy. We achieved 61.5% to 63.5% average accuracy on 14 DBLP name datasets with a variety of sizes. The highest accuracy 96.8% is achieved on the "J. Martin" dataset containing "J. Martin"s that have at least 10 citations. The disambiguation accuracies on "S. Lee", "J. Lee", "Y. Chen" and "C. Chen" are lower than on other names. The possible reason is that these datasets contain more ambiguous authors than other datasets, and many authors from these datasets have close research areas. Experiments on disambiguating 11 "J. Smith"s and 15 "J Anderson"s (citations are publication lists collected mainly from authors' home pages) show 84.3% and 71.2% accuracy respectively.

Further improvements could be obtained through semantic word clustering on paper titles and publication venue titles [111]. We observe that a researcher usually has a research area or areas that do not change over a period of time, and his/her paper or submitted publication venue titles are closely related to his/her research topic.

However, the paper and publication venue title words are sparse, and an author may not reuse a certain group of title words. Moreover, our current work does not recognize the similarity between words such as "Neurocomputing" and "NeuroScience". Therefore, it is reasonable to cluster "similar" title words into research fields, and use a new set of features that summarize similar words. Such a word cluster reduces feature sparseness, and usually has more robust probability estimates by averaging statistics for similar words [8]. Existing word clustering methods we can apply include methods based on the Word Net [10], distributional word clustering [8, 83, 25, 29], bipartite word clustering [113], committee-based word clustering [68], and other word similarity measures [100, 54] . Research keywords classification schemes such as the ACM classification may also help to map different title words into a research category.

In our hand-labeling of the datasets, we used extra information such as affiliations, email addresses, resumes, home pages, and some human judgment. Therefore, in order to improve the name disambiguation performance, we most likely need more features as those that are used in our hand-labeling than the three citation attributes that we currently use. Words and bigrams from the paper abstracts may also provide useful information for disambiguation. We would also like to address the issue of automatically choosing the number of name clusters.

We wish to combine both unsupervised and supervised learning methods, to build a practical reinforced name disambiguation system in the future. We would like to add string-based components [15, 99] for better representation of author names and for finding the canonical name of an author. We would also like to disambiguate similar corporate names appearing in academic and publishing world, such as "Loyola College." It may

also be useful to extend our name disambiguation systems in digital documents to other applications, especially in the academic, patent, medical records, or genealogy fields.

Chapter 5

Conclusions

With the prevalence of digital libraries, academic documents have become an important part of the information on the Web. How to effectively retrieve relevant information from the sea of digital documents is an important research topic. Current research work on text classification and clustering underexploits the important information embedded in a collection of documents. For example, a document is viewed as an unstructured sequence of words. Only simple reference relationships among documents are used. An ontology is a good tool to explicitly model the rich information among a collection of document, because an ontology specifies a domain conceptualization through document attributes, a classification of documents, relationships among documents, and inference rules that can help discover hidden information from documents. A document ontology enhances interoperability among heterogeneous digital libraries and helps to reuse knowledge that is embedded in documents. While a computer is hard to understand the content of documents written in natural language, it is feasible for a computer to process the syntactic structure of the documents. This thesis presents the research work toward creating and populating a syntactic document ontology.

5.1 Summary of the Thesis Work

The following summarizes the research work of this thesis toward creating and populating a syntactic document ontology.

- Chapter 2 presents the conceptual specification of a syntactic document ontology in aspects of document attributes, document categories and relationships among documents. We also propose the concepts of canonical document and canonical name.

- To make the created syntactic document ontology useful, it is important to populate the syntactic document ontology automatically. Chapter 3 describes a Support-Vector-Machines(SVM)-based classification method for automatically extracting document attributes (the title, author name, author affiliation and address, abstract, publication venue, publication date, etc.) from header parts of documents and bibliographic fields. Our method of document attributes extraction from document headers achieved better results than using hidden Markov Model (HMMs) on the CMU datasets [90]. This algorithm includes a novel method of parsing individual author names from the line of multiple authors, and a word clustering method based on domain databases and word orthographic properties.

- Name ambiguity is a popularly seen problem in documents and citations. While ontologies require the unambiguous specification of a concept, we investigate both supervised and unsupervised learning methods for name entity disambiguation in author citations. The supervised learning approaches consider each canonical author name as a class, and classify each citation into its author class. We developed

two supervised learning methods, one based on a hierarchical naive Bayes model, the other based on the Support Vector Machines. The unsupervised learning approaches partition collections of citations into clusters, with each cluster containing only citations authored by the same author, thus disambiguating authorship in citations to induce author name identities. We developed two unsupervised learning methods, one based on a hierarchical naive Bayes mixture model, the other based on the spectral clustering algorithm [112]. These methods are applied to 16 name datasets we constructed based on the publication lists collected from author's homepages, and the DBLP computer science bibliography. The K-way spectral clustering method with QR decomposition achieved best results, compared to the K-means clustering algorithm and the hierarchical naive Bayes mixture model. The hierarchical-naive-Bayes-model-based method achieved better results than the SVM-based classification method when using only coauthor information. The main reasons are that our hierarchical naive Bayes model captures the author patterns that are not easily incorporated into feature vector space model that is used by SVM-based classifcation or K-way spectral clustering methods. These author patterns are the prior probability of an author, the probability that an author writes papers alone, and the probabilities that an author writes a future paper with previously unseen coauthors. The SVM-based classification method achieved slightly better results than the hierarhical-naive-Bayes-model-based method when using paper title words, publication venue title words, and combination of all types of citation features.

5.2 Discussion and Future Work

- To put the syntactic document ontology into practical use, it is neccessary to implement the ontology using ontology lanaguages. It would be an interesting research work to see the applications of the syntactic document ontology to document classification work.

- The SVM-based classification techniques for automatic document attributes extraction can be further improved, especially the chunk identification approaches. The method can also be generalized to web information extraction.

- The name disambiguation work in this thesis crosses over a couple of research themes: the tension between symbolic and statistical methods for modeling and learning, and the tension between theoretical and practical goals. To find the best solution for disambiguating author names in citations, widely-ranging techniques need to be tried, and reasons for the variations of a person's name need to be enumerated. Prior probablities about different types of data in databases may help judge which type of data can be used as evidence in hypothetical reasonsing about whether multiple names correspond to the same person. For example, the likelyhood that an author's papers cite self versus someone else with a similar name; the likelyhood that two authors with similar names are at same institution. More information can be gathered automatically to augment the sources of information that is used in this research. For example, check the publication lists in author's homepages; analyze citations in author's own papers; check citations in review papers, which may provide more consistent information about authors; estimate

age or graduation dates, which may prune the paper that is too old to be written by an author; use the institutional affiliations of an author and the authors cited by this author, because students of a professor may tend to cite other students in the same community. Rich symbolic reprentation of names may be considered, for example, the inclusion or exclusion of name initials or complete names or spelling variations.

References

[1] Getty's ULAN (Union List of Artist's Names). http://www.getty.edu/research/conducting_research/vocabularies/ ulan/.

[2] The Library of Congress name authority file. http://www.loc.gov/marc/authority/index.html.

[3] *The World Factbook 2001.* Central Intelligence Agency, 2001.

[4] A. Ankolekar, M. Burstein, J. Hobbs, O. Lassila, D. Martin, S. Mcllraith, S. Narayanan, M. Paolucci, T. Payne, K. Sycara, and H. Zeng. Daml-s: Semantic markup for web services. In *Proceedings of the International Semantic Web Working Symposium (SWWS)*, 2001.

[5] Anupriya Ankolekar, Young-Woo Seo, and Katia Sycara. Investigating semantic knowledge for text learning. In *Proceedings of ACM SIGIR Workshop on Semantic Web*, 2003.

[6] Yossi Azar, Amos Fiat, Anna R. Karlin, Frank McSherry, and Jared Saia. Spectral analysis of data. In *Proceedings of 33rd ACM Symposium on Theory of Computing*, pages 619–626, 2001.

[7] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. *Modern Information Retrieval.* ACM Press / Addison-Wesley, 1999.

[8] L. Douglas Baker and Andrew K. McCallum. Distributional clustering of words for text classification. In W. Bruce Croft, Alistair Moffat, Cornelis J. van Rijsbergen, Ross Wilkinson, and Justin Zobel, editors, *Proceedings of the 21st ACM International Conference on Research and Development in Information Retrieval*, pages 96–103, 1998.

[9] A. Banerjee, I. Dhillon, J. Ghosh, and S. Sra. Generative model-based clustering of directional data. In *Proceedings of The 9th ACM SIGKDD Conference on Knowledge Discovery and Data Mining(KDD)*, pages 19–28, 2003.

[10] S. Banerjee and T. Pedersen. An adapted lesk algorithm for word sense disambiguation using wordnet. In *Proceedings of the 3rd International Conference on Intelligent Text Processing and Computational Linguistics (CICLING)*, 2002.

[11] Yaakov Bar-Shalom and Thomas E. Fortmann. *Tracking and Data Association*. Academic Press, 1988.

[12] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 2001.

[13] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, 2001.

[14] M. Berry, Z. Drmac, and E. Jessup. Matrices, vector spaces and information retrieval. *Journal of Society for Industrial and Applied Mathematics*, 41(2):335–362, 1999.

[15] Mikhail Bilenko, Raymond Mooney, William Cohen, Pradeep Ravikumar, and Stephen Fienberg. Adaptive name matching in information integration. *IEEE Intelligent Systems*, 18(5):16–23, 2003.

[16] Dina Bitton and David J. DeWitt. Duplicate record elimination in large data files. *ACM Transactions on Database Systems*, 8(2):255–265, 1983.

[17] L K Branting. Name-matching algorithms for legal case-management systems. *Journal of Information, Law and Technology (JILT)*, 1, 2002.

[18] Tim Brody. Celestial aggregator service. http://celestial.eprints.org/, 2002.

[19] Mary Elaine Califf and Raymond J. Mooney. Relational learning of pattern-match rules for information extraction. In *Proceedings of the 16th National Conference on Artificial Intelligence*, pages 328–334, 1999.

[20] Hai Leong Chieu and Hwee Tou Ng. A maximum entropy approach to information extraction from semi-structured and free text. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI 2002).*, pages 786–791, 2002.

[21] William J. Clancey. The knowledge level reinterpreted: Modeling socio-technical systems. pages 33–50, 1993.

[22] William W. Cohen, Henry A. Kautz, and David A. McAllester. Hardening soft information sources. In *Proceedings of the 6th International Conference on Knowledge Discovery and Data Mining*, pages 255–259, 2000.

[23] J. M. Crawford and B. Kuipers. Towards a theory of access-limited logic for knowledge representation. In *Proceedings of of the 1st International Conference on Principles of Knowledge Representation and Reasoning*, pages 67–78. Morgan Kaufmann, 1989.

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

[25] Ido Dagan, Fernando C. N. Pereira, and Lillian Lee. Similarity-based estimation of word cooccurrence probabilities. In *Meeting of the Association for Computational Linguistics*, pages 272–278, 1994.

[26] L. Daniel and J. Slezak. Street talk: the word on address-matching. *Business Geographics*, pages 26–33, 1995.

[27] H.V. de Sompel and C. Lagoze. The open archives initiative protocol for metadata harvesting, January 2001.

[28] E. Desmontils and C. Jacquin. Indexing a web site with a terminology oriented ontology. In *Proceedings of SWWS'01, The first Semantic Web Working Symposium, Stanford University, California, USA, July 30 - August 1, 2001*, pages 549–565, 2001.

[29] Inderjit Dhillon, S. Manella, and R. Kumar. A divisive information-theoretic feature clustering for text classification. *Journal of Machine Learning Research(JMLR)*, 3:1265–1287, 2003.

[30] Tim DiLauro, G. Sayeed Choudhury, Mark Patton, James W. Warner, and Elizabeth W. Brown. Automated name authority control and enhanced searching in the levy collection. *D-Lib Magazine*, 7(4), 2001.

[31] William B. Dolan. Word sense ambiguation: Clustering related senses. *Proceedings of the 15th International Conference on Computational Linguistics*, pages 712–716, 1994.

[32] Pedro Domingos and Michael J. Pazzani. Beyond independence: Conditions for the optimality of the simple bayesian classifier. In *International Conference on Machine Learning*, pages 105–112, 1996.

[33] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering in large graphs and matrices. In *SODA: ACM-SIAM Symposium on Discrete Algorithms*, pages 291–299, 1999.

[34] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of the 17th International Conference on Information and Knowledge Management*, pages 148–155, November 1998.

[35] I. P. Fellegi and A. B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64:1183–1210, 1969.

[36] Gary Flake, Eric Glover, Steve Lawrence, and C. Lee Giles. Extracting query modifications from nonlinear SVMs. In *Proceedings of the International World Wide Web Conference*, pages 317–324, May 7-11 2002.

[37] J. Friedman. On bias, variance, 0/1-loss, and the curse-of-dimensionality. *Journal of Data Mining and Knowledge Discovery*, 1, 1997.

[38] M. Genesereth and R. Fikes. Knowledge interchange format, version 3.0 reference manual. Technical Report Logic-92-1, Computer Science Department, Stanford University, 1992.

[39] M. R. Genesereth. *The Epikit manual.* Epistmemics Inc, 1992.

[40] C. Lee Giles, Kurt Bollacker, and Steve Lawrence. CiteSeer: An automatic citation indexing system. In *Proceedings of the 3rd ACM Conference on Digital Libraries*, pages 89–98, 1998.

[41] Peter Gillman. National name authority file: Report to the national council on archives. Technical Report British Library Research and Innovation Report 91, The British Library Board, 1998.

[42] T. R. Gruber. A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2):199–220, 1993.

[43] Nicola Guarino. Formal ontology, conceptual analysis and knowledge representation. *Special issue: the role of formal ontology in the information technology*, 43(5-6):625–640, 1995.

[44] Hui Han, C. Lee Giles Eren Manavoglu, and Hongyuan Zha. Rule-based word clustering for document classification. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003)*, pages 445–446, 2003.

[45] Hui Han, C. Lee Giles, Eren Manavoglu, Hongyuan Zha, Zhenyue Zhang, and Edward A. Fox. Automatic document metadata extraction using support vector machines. In *Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital libraries*, pages 37–48, 2003.

[46] Hui Han, C. Lee Giles, Hongyuan Zha, Cheng Li, and Kostas Tsioutsiouliklis. Two supervised learning approaches for name disambiguation in author citations. In *Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital libraries*, 2004.

[47] Jeff Heflin, James Hendler, and Sean Luke. SHOE: A knowledge representation language for internet applications. Technical Report CS-TR-4078, Dept. of Computer Science, University of Maryland at College Park, 1999.

[48] Mauricio A. Hernandez and Salvatore J. Stolfo. Real-world data is dirty: Data cleansing and the merge/purge problem. *Data Mining and Knowledge Discovery*, 2(1):9–37, 1998.

[49] Thomas Hofmann. Probabilistic latent semantic analysis. In *Proceedings of Uncertainty in Artificial Intelligence, UAI'99*, 1999.

[50] Thorsten Joachims. Making large-scale support vector machine learning practical. *Advances in Kernel Methods: Support Vector Machines*, 1998.

[51] Thorsten Joachims. A statistical learning model of text classification with support vector machines. In *Proceedings of SIGIR-01, 24th ACM International Conference on Research and Development in Information Retrieval*, pages 128–136, 2001.

[52] Kalervo Jrvelin and Jaana Keklinen. Ir evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 41–48, 2000.

[53] Ravi Kannan, Santosh Vempala, and Adrian Vetta. On clusterings: Good, bad and spectral. In *Proceedings of the 41st Foundations of Computer Science*, pages 367–380, 2000.

[54] J. Karlgren and M. Sahlgren. From words to understanding. *In Kanerva et al. (eds.) Foundations of Real World Intelligence. CSLI publications*, pages 294–308, 2001.

[55] Henry A. Kautz, Bart Selman, and Mehul A. Shah. The hidden web. *AI Magazine*, 18(2):27–36, 1997.

[56] Alan Kent. Oai harvester crawling status. *http://www.mds.rmit.edu.au/ ajk/oai/interop/summary.htm*, 2001.

[57] Atanas Kiryakov, Borislav Popov, Damyan Ognyanoff, Dimitar Manov, Angel Kirilov, and Miroslav Goranov. Semantic annotation, indexing, and retrieval. In *Proceeding of the 2nd International Semantic Web Conference (ISWC2003)*, 2003.

[58] Deborah Knox. Citidel: making resources available. In *Proceedings of the 7th Annual Conference on Innovation and Technology in Computer Science Education*, pages 225–225, 2002.

[59] Robert Krovetz. Viewing Morphology as an Inference Process,. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 191–203, 1993.

[60] Robert Krovetz and W Bruce Croft. Word sense disambiguation using machine-readable dictionaries. In *Proceedings of the 12th Annual ACMSIGIR Conference*, pages 127–136, 1989.

[61] Taku Kudoh and Yuji Matsumoto. Use of support vector learning for chunk identification. In *Proceedings of the 4th Conference on Computational Natural Language Learning*, pages 142–144, 2000.

[62] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289, 2001.

[63] Steve Lawrence, C. Lee Giles, and Kurt Bollacker. Digital libraries and Autonomous Citation Indexing. *IEEE Computer*, 32(6):67–71, 1999.

[64] Mong-Li Lee, Tok Wang Ling, and Wai Lup Low. Intelliclean: a knowledge-based intelligent data cleaner. In *In 6th International Conference on Knowledge Discovery and Data Mining*, pages 290–294, 2000.

[65] David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In W. Bruce Croft and Cornelis J. van Rijsbergen, editors, *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, pages 3–12, Dublin, IE, 1994. Springer Verlag, Heidelberg, DE.

[66] Gangmin Li, Victoria Uren, Enrico Motta, Simon Buckingham Shum, and John Domingue. Claimaker: Weaving a semantic web of research papers. In *Proceedings of International Semantic Web Conference*, 2002.

[67] H. Li and N. Abe. Word clustering and disambiguation based on co-occurrence data. In *Proceedings of the 17th Internation Conference on Computational Linguistics*, pages 749–755, 1998.

[68] Dekang Lin and Patrick Pantel. Concept discovery from text. In *Proceedings of Conference on Computational Linguistics*, pages 577–583, 2002.

[69] Xiaoming Liu. Federating heterogeneous digital libraries by metadata harvesting. *Ph.D. Dissertation Old Dominion University*, December 2002.

[70] R. MacGregor. Loom users manual. Technical Report ISI/WP-22, USC/Information Sciences Inst, 1992.

[71] Kavi Mahesh, Jacquelynn Kud, and Paul Dixon. Oracle at trec8: A lexical approach. In *Proceedings of the eighth Text Retrieval Conference(TREC-8)*, 1999.

[72] Catherine C. Marshall. Making metadata: A study of metadata creation for a mixed physical-digital collection. In *Proceedings of the 3rd ACM International Conference on Digital Libraries*, pages 162–171, June 1998.

[73] Andrew McCallum, Dayne Freitag, and Fernando Pereira. Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of the 17th International Conference on Machine Learning*, pages 591–598, 2000.

[74] Andrew McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the contruction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000.

[75] Andrew McCallum, Kamal Nigam, and Lyle H. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *Knowledge Discovery and Data Mining*, pages 169–178, 2000.

[76] Paul Mcnamee and James Mayfield. Entity extraction without language-specific resources. In *Proceedings of CoNLL-2002*, pages 183–186, 2002.

[77] Alvaro E. Monge and Charles Elkan. An efficient domain-independent algorithm for detecting approximately duplicate database records. In *Research Issues on Data Mining and Knowledge Discovery*, pages 23–29, 1997.

[78] Enrico Motta, Simon Buckingham, and John Domingue. Ontology-driven document enrichment: Principles and case studies. In *Proceedings of KAW'99: 12th Banff Knowledge Acquisition Workshop*, 1999.

[79] A. Newell. The knowledge level. *Artificial Intelligence*, 18(1), 1982.

[80] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Proceedings of Advances in Neural Information Processing Systems*, pages 849–856, 2001.

[81] Andreas Paepcke, Chen-Chuan K. Chang, Hector Garcia-Molina, and Terry Winograd. Interoperability for digital libraries worldwide. *Communications of the ACM*, 41(4):33–42, 1998.

[82] Hanna Pasula, Bhaskara Marthi, Brian Milch, Stuart Russell, and Ilya Shpitser. Identity uncertainty and citation matching. In *Proceedings of Neural Information Processing Systems: Natural and Synthetic 15*, 2002.

[83] Fernando C. N. Pereira, Naftali Tishby, and Lillian Lee. Distributional clustering of english words. In *Meeting of the Association for Computational Linguistics*, pages 183–190, 1993.

[84] Yves Petinot, Pradeep B. Teregowda, Hui Han, C. Lee Giles, Steve Lawrence, Arvind Rangaswamy, and Nirmal Pal. ebizsearch: An oai-compliant digital library for ebusiness. In *Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital libraries*, pages 199–209, 2003.

[85] Ari Pirkola, Jarmo Toivonen, Heikki Keskustalo, Kari Visala, and Kalervo Jarvelin. Fuzzy translation of cross-lingual spelling variants. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 345–352, 2003.

[86] A. Pothen, H. D. Simon, and K.-P. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and Application*, 11:430–452, 1990.

[87] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 77(2):257–286, 1989.

[88] Lance Ramshaw and Mitch Marcus. Text chunking using transformation-based learning. In *Proceedings of the 3rd Workshop on Very Large Corpora*, pages 82–94, 1995.

[89] Patrick Schone and Daniel Jurafsky. Knowlege-free induction ofinflectional morphologies. In *Proceedings of the North American chapter of the Association for Computational Linguistics (NAACL)*, 2001.

[90] Kristie Seymore, Andrew McCallum, and Roni Rosenfeld. Learning hidden Markov model structure for information extraction. In *Proceedings of AAAI 99 Workshop on Machine Learning for Information Extraction*, 1999.

[91] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[92] Simon Buckingham Shum, Enrico Motta, and John Domingue. Scholonto: an ontology-based digital library server for research documents and discourse. *Internatinal Journal on Digital Libraries*, 3(3):237–248, 2000.

[93] M. Skounakis, M. Craven, and S. Ray. Hierarchical hidden markov models for information extraction. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, 2003.

[94] N. Slonim and N. Tishby. The power of word clusters for text classication. In *Proceedings of the 23rd European Colloquium on Information Retrieval Research (ECIR)*, 2001.

[95] Sadanand Srivastava, James Gil de Lamadrid, and Chakravarthi S. Velvadapu. Document ontology: A statistical approach. In *Proceedings of International Conference Advances in Infrastructure for e-Business, e-Education, e-Science, and e-Medicine on the Internet*, 2002.

[96] Andreas Stolcke. *Bayesian Learning of Probabilistic Language Models*. PhD thesis, Dept. of Electrical Engineering and Computer Science, University of California at Berkeley, 1994.

[97] Atsuhiro Takasu. Bibliographic attribute extraction from erroneous references based on a statistical model. In *Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital libraries*, pages 49–60, 2003.

[98] Koichi Takeuchi and Nigel Collier. Use of support vector machines in extended named entity, 2002.

[99] Sheila Tejada, Craig Knoblock, and Steven Minton. Learning domain-independent string transformation weights for high accuracy object identification. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 350–359, 2002.

[100] Egidio L. Terra and Charles L. A. Clarke. Frequency estimates for statistical word similarity measures. In Marti Hearst and Mari Ostendorf, editors, *HLT-NAACL 2003: Main Proceedings*, pages 244–251, 2003.

[101] H. R. Turtle and W. B. Croft. Uncertainty in information retrieval systems. *Uncertainty Management in Information Systems*, pages 189–224, 1996.

[102] Vladimir Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

[103] E. M. Voorhees. Using wordnet for text retrieval. In C. Fellbaum, editor, *In WordNet: An Electronic Lexical Database*, pages 285–303, 1998.

[104] J. W. Warner and E. W. Brown. Automated name authority control. In *Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital libraries (JCDL01)*, 2001.

[105] S. Weibel. The dublin core: A simple content description format for electronic resources. *NFAIS Newsletter*, 40(7):117–119, 1999.

[106] S. Weibel, J. Kunze, C. Lagoze, and M. Wolf. Dublin core metadata for resource discovery. *RFC 2413, The Internet Society*, 1998.

[107] H. D. White and K. W. McCain. Visualizing a discipline: an author co-citation analysis of information science 1972-1995. *Journal of the American Society for Information Science*, 49(4):327–355, 1998.

[108] Dominic Widdows and Beate Dorow. A graph model for unsupervised lexical acquisition. In *19th International Conference on Computational Linguistics*, pages 1093–1099, Taipei, Taiwan, August 2002.

[109] Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th ACM International Conference on Research and Development in Information Retrieval (SIGIR03)*, pages 267–273, 2003.

[110] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, pages 412–420, 1997.

[111] Y. Y. Yao, S.K.M. Wong, and L. S. Wang. A non-numeric approach to uncertain reasoning. *International Journal of General Systems*, 23(4):343–359, 1995.

[112] H. Zha, C. Ding, M. Gu, X. He, and H. Simon. Spectral relaxation for k-means clustering. In *Neural Information Processing Systems (NIPS 2001)*, pages 1057–1064, 2001.

[113] Hongyuan Zha, Xiaofeng He, Chris Ding, Ming Gu, and Horst Simon. Bipartite graph partitioning and data clustering. In *Proceedings of ACM CIKM 2001, the 10th International Conference on Information and Knowledge Management*, pages 25–32, 2001.

[114] Xuegong Zhang and Wing H. Wong. Recursive sample classification and gene selection based on svm: method and software description. In *Technical Report, Department of Biostatistics, Harvard School of Public Health*, 2001.

Vita

Hui Han enrolled in the Ph. D. program in computer science at the Pennsylvania State University in 1999 fall. Since 1999 she has been employed in the Department of Computer Science and Engineering or the School of Information Sciences and Technologies at the Pennsylvania State University as a teaching/research assistant. Hui Han is a student member of the Association for Computing Machinery.