DOCTOR OF PHILOSOPHY

A STUDY ON NAME DISAMBIGUATION USING WEB DIRECTORIES

（ウェブディレクトリを用いた人名の曖昧性解消に関する研究）

Supervisor **Professor Jun ADACHI**

THE UNIVERSITY OF TOKYO

Graduate School of Information Science and Technology

Name **Quang Minh VU**

Date **December 17th, 2007**

Tặng Bố Mẹ Kính Yêu của Con!

To my parents, for their loves, devotions and constant supports

Contents

List of Figures

List of Tables

List of Abbreviations

WWW	World Wide Web
DARPA	Defense Advanced Research Projects Agency
VSM	Vector Space Model
NER	Named Entity Recognition
WSD	Word Sense Disambiguation
IR	Information Retrieval
SKB	Similarity via Knowledge Base
LDA	Latent Dirichlet Allocation
EM	Expectation Maximization
POS tagging	Part-of-Speech tagging
SVD	Singular Value Decomposition
tf	term frequency
df	document frequency
idf	inverse document frequency

Acknowledgements

This dissertation is a milestone in my academy record. It summarizes the achievements of my three year research at the Adachi Laboratory, Graduate School of Information Science and Technology, The University of Tokyo. I would like begin the dissertation by expressing my gratitude to the people to whom I own much thankfulness for their kind supervises and supports. Without their helps, this research could not be accomplished.

First of all, I would like to thank my supervisor professors: Professor Jun Adachi and Professor Atsuhiro Takasu. They have supervised me very kindly to find research directions and to do research successfully. It is very happy for me to receive their supervisions and they have influenced my research methodology. I have inherited from them much academic knowledge and many research experiences. Professor Adachi has taught me his experiences not only about the field of computer science, but also about many fields of natural sciences in general. From him I have learned the relationship between the computer science and other fields of study and the impacts that the computer science can contribute to the society. Professor Takasu has taught me his deep knowledge and experiences about computer science. Especially, I have learned from him many knowledges about machine learning, data mining, and information retrieval. He has suggested me many advices about my approaches during my research.

During my study at the Adachi Laboratory, I have good chances to enjoy the research life with many seniors and colleagues: Dr. Tomonari Masada, Dr. Hiromi Wakaki, Mr. Akira Uemura, Mr. Kouhei Suzuki, Mr. Junya Aizawa, Mr. Hisashi Kurasawa, Mr. Takumi Tsujishita, Mr. Masaharu Tatsumi, and Mr. Kenji Hirohata. We have enjoyed many cheers and happiness together. I specially thank Dr. Tomonari Masada for his assistances and advices to my research. He supported me very much when I was in my

beginning steps to natural language processing. He also helped me at the beginning of my research.

To study in Japan is a wonderful advantage to me. I would like to take this opportunity to thank to the Minitry of Education, Culture, Sports, Science and Technology of Japan[1] and the Honjo International Scholarship Foundation[2] for their finance supports to my ten years of studies in Japan. Upon receiving their kind supports, I have chances to work and study in the outstanding academic environments: Kyoto University and The University of Tokyo. My studies in Japan earn me the knowledge in computer science as well as the knowledge about Japanese society. I own many Japanese friends and neighborhoods for their warm friendships and their hearty kindness.

My doctor degree is a fruit of efforts from my family, my parents and my sister as well. They have been together with me in advantages, successes as well as in difficulties and challenges. I would like to thank them for their loves and supports during my life and my studies from the elementary school upto now and beyond to the future. They have been devoting their powers, ardour and spirits to my growths and my progresses. They have been expecting to my advancements and sharing with me my successes and achievements more than any other people.

[1]日本政府文部科学省 in Japanese, http://www.mext.go.jp/
[2]財団法人本庄国際奨学財団 in Japanese, http://www.hisf.or.jp/

Chapter 1

Introduction

The World Wide Web (WWW)[9, 10, 23] has been becoming the largest database ever seen in the history. It is built upon the global network which connects all people arround the world together. Resources in the WWW are created by millions of people. At any moment, many new documents are published which contain latest information from many places around the world.

In order to benefit the WWW resources well, it is required to engineer web data to bring information with high value to end users. Search engines play a role of a front gate to access the WWW. Users can retrieve information by querying the search engines. Search engines first preprocessed resources in the WWW and stored them in index databases. Then, upon receiving users' requests, search engines can quickly search the index databases and return result documents to users.

This Chapter is devoted to describe the overview of our research. We present the research problem that we have targeted, the achievements that we have obtained, and the contributions by our research. First, we present the status of this current WWW, its essential features as well as its developing trends in future. Then, we present the problem definition in our research. Our research target is to improve the performance and the convenience of search engines so that users can get in hand information of their needs more quickly and more precisely. Especially, we focus on personal name queries which search for information related to people. Next, we present the overview of our approach that try to re-rank result documents to bring documents which match to information needs of users toward the top. Finally, we summarize the main achievements

and contributions by our research.

1.1 The World Wide Web as a Huge Reservoir of Information

1.1.1 The New Generation Internet

The internet has been becoming a vital infrastructure for communication in the world[85]. The internet history began from the late of 1960s when the Defense Advanced Research Projects Agency (DARPA) of the United States of America built an intranet computer network for the purpose of military research. During the 1970s, the uses of intranet networks expanded into institutes and universities. In the 1980s, there appeared efforts to connect distant computer networks together to create a globally connected network. Network cables and optical fibers were set up to connect continents and communication protocols were designed to support transmissions and receptions of data. From the 1990s, internet providers established and they broadened the internet to reach organizations, companies, and households, etc.

The global internet is constructed upon a giant and vigorous network infrastructure. It has exploited latest advancements in semiconductor technologies and hardware technologies. Network cables and data transmission technologies have utilized semiconductor technologies for robust and high speed data transmissions. Terminal computers like end user machines and application servers together with network devices like routers, switches also benefited latest hardware technologies for reliable and high speed data processing. Recent improvements in the semiconductor and hardware technologies have improved functions and utilities of electronic devices. They have also reduced sizes and power consumptions of devices. Therefore, the ranges of machines that join the internet now extend to pocket computers, mobile phones, and home appliances, etc. This quickly increasing number of terminal machines makes the presence of internet come to every corner in daily life.

The above advancements play as the key hardware for the internet. Upon this huge fundamental hardware infrastructure, many kinds of services are created and deployed. Emails, hyper text pages are among the first generation of applications. Up to now, many new kinds of applications join the internet like chat, video conference, file sharing,

and e-commerce, etc. They can be regarded as softwares of the internet that bring to end-users new value-added services and utilities. The WWW may be the application which is the most proximity to people. Access to the WWW is almost free and retrieval of data is instant. Almost all the internet citizens use the WWW everyday for their livings, their works, and their leisure, etc. The internet together with the WWW have really become the largest computer architecture in terms of hardware scales, software complexities and the number of end-users.

1.1.2 The World Wide Web as a Heterogeneous Environment of Data

The appearance of WWW has substantially changed the method of exchanging information. It has reduced the cost of publishing information, it has shortened the time to access to information, and it has lowered the barrier between the information publishers and the information receivers. Before the WWW era, the major methods to publishing information were books, newspapers, radio news, television programs, etc. The expensive costs of these methods limited the number of publishers and authors who could create and distribute information. To print a book or to produce a television programs, it cost months and years of preparation and publishment before the information can reach the receivers. Therefore, among the numbers of people who wanted to publish information, only a part of them were able to publish. However, in the current WWW era, everything have been changed. The cost to use the WWW is so cheap that everyone can publish information for almost free. Organizations like companies, universities operate their own servers to advertise their information. Newspapers, companies, and publishers produce news, articles, and books in the WWW besides the conventional paper printing books and newspapers. Besides organizations, any individual can also rent a hard disk space in a server and set up his/her own homepage.

The amount of information in the WWW has been increasing everyday and the WWW has become a heterogeneous environment of database[7, 94]. Since the publishers of web information grow rapidly in number, the WWW database is mixed from many different kinds of documents. Web documents are from different kinds of sources, so their topics are various and their writing styles are not uniform. Formats of documents vary from official and formal formats in documents created by organizations to

unofficial and informal formats in documents created by individuals. Organizations' documents usually contain high reliable news while individual people's documents may contain low reliable news. Representation techniques of documents are also various. Before, most of the web pages are static in contents but today, new web pages are often in dynamically changing contents. Web pages that can interact with user and provide real time information are increasing.

The WWW can be regarded as the mirror of the real world[83, 100]. Changes in the real world are reflected in the virtual WWW. Newspapers always update news from around the world at every second. People express their comments, their opinions, and their reviews in their homepages and blogs. People who have the same interests, hobbies come together to form communities and social networks. Internet citizens now live in the two worlds at the same time: one is the real world, another is the virtual WWW world. They can live, work and enjoy in the virtual world. Internet games, e-commercial services are new kinds of web applications that are providing users new values and benefits.

1.2 Search for Useful Information from the World Wide Web

1.2.1 How to Retrieve Useful Information

The enormous amount of data in the WWW is an advantage to users when they want to acquire knowledge. Users can access to the information at any time, they can manipulate data freely, they have many chances to earn new knowledge and to create new value to data. However, to utilize the WWW in an effective way is a big problem. Since the access to the data is the same for anyone, the people who exploit the WWW the most effectively are the winners in the WWW era.

Search engines are now playing as an important tool to help users to access to information[19, 48]. Before the appearance of search engines, people have to know the url addresses of the web pages in order to get the information. However, users can only remember and can input a very little number of url addresses. Prior to search engines, some web directory services try to categorize web pages into topics. Topics are then organized in hierarchy structures to help users navigate through the directories easily.

Although web directories can help users to find the topics of interests by their hierarchy topic structures, the organizing of web directories requires much human works. Since the web database is changing rapidly, it is almost impossible for web directories to keep up with that changing and to organize web directories in an appropriate structure that can reflect the contents in the WWW well.

It is the appearance of search engines that causes revolution in the method to access to information. Search engines try to capture all contents that exist in the WWW. They first crawl the WWW by following hyperlinks among web pages. Then, texts in the crawled pages are indexed to capture the page contents and store them in the indexing database. Search engines also analyze hyperlinks between web pages[56, 86] to rank web pages in the order of importances, qualities and reliabilities. With the help of search engines, users can look for valuable information much more easily. Users' information needs are represented in queries that are sent to search engines. The search engines look up in their indexing database and retrieve documents which are close to users' queries in less than a second. The search engines are now serving as important gateways to access to the WWW.

1.2.2 Information of Entities in the World Wide Web

The information about people and entities is contained in a significant amount of pages from the whole WWW database. Since the WWW is a mirror space of the real world, it is natural that people and entities in the real world become topics for documents created in the WWW. Entities that appear in the WWW documents are countries, organizations, companies, universities, people, products, etc. These entities interact with each other in the real world and create news in the virtual WWW. News and facts about entities are scattered in the WWW. They appear in different resources like newspapers, shopping web sites, company homepages, user homepages, etc. Therefore, the needs to track the same entity appearing in different pages are important when users want to collect useful information about an entity. When tracking for the same entity in different pages, there raise two problems. The first problem is the hyponymy problem where the same name is used to mention different entities. The second problem is the synonymy problem where different names are used to mention the same entity.

In order to find useful information about an entity effectively, both these two problems must be solved. Among the two problems, the hyponymy problem could be solved by analyzing of documents and similarity measurement of contexts relevant to entities in documents. On the other hand, the second one is much more difficult since it is difficult to enumerate candidates for synonymy names of an entity.

1.2.3 Name Disambiguation in Searching for People in the World Wide Web

Among the searching needs for entities, the searching need for people in the WWW is an increasing need. While searching for a person, users often use the name of that person as a query to the search engine. The search engine uses the query to look up the indexing database and to match documents that contain the query terms. However, because of the name hyponymy problem, several people may have the same name. As the result, documents matched with the query often contain information about different people.

We consider an example as follows. Suppose that a user wants to search for information about Mr. Jim Clark, the founder of the Netscape company. He/she sends the query "Jim Clark" to the Google search engine [1]. He/she looks at the results and is confused because there are different Jim Clarks there. Besides Mr. Jim Clark at the Netscape, there are Jim Clark the motor racer and many other Jim Clarks. Since the Jim Clark of interests is mixed with other Jim Clarks, he/she has to filter to get useful information manually. Search engines will be more friendly and more convenient if they can filter the person of interest automatically.

We try to develop an algorithm that can help users to disambiguate ambiguous personal names. The name disambiguation problem is close to the word sense disambiguation[50] which has been researched before. The word sense disambiguation is used in applications like machine translation, text summarization, information extraction, etc. In the word sense disambiguation tasks, senses of ambiguous words are usually known in advanced and can be defined in dictionaries. However, for the case of personal name ambiguity, it is impossible to predict and define senses of names. Because of this sub-

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

stantial difference between the two problems, name disambiguation problem requires a new tackling approach in order to achieve good disambiguation performances.

1.3 Our Contributions

1.3.1 Problem Statements

We define the name disambiguation problem as follows. When a user wants to search to information about a person, he/she first inputs the name of that person in the query form and sends to a search engine. The search engine uses the personal name query to look up for matched documents and returns the results to user. Ambiguous names in the results can be disambiguated by one of these two procedures: the clustering procedure or the re-ranking procedure. In the clustering procedure [30, 76, 91, 109], the search engine try to cluster result documents so that each cluster is correspondent to exact one person. This clustering procedure is also used in word sense disambiguation problem. In the word sense disambiguation problem, if we can define candidate senses, we will know the number of clusters in the truth result and the clustering algorithm will perform better when the number of clusters is known. However, for the case of personal name disambiguation, to know the number of people is impossible. Also, when searching for a person, users are interested in only one person. Therefore, all the clusters but the cluster corresponding to that person are meaningless to users. From this point of view, we adopt the re-ranking procedure [107, 117] to disambiguate personal names. The re-ranking procedure interacts with users during search operations. Users first select a document that refers to the person they are looking for. The search engine receives this feedback and tries to re-rank result documents so that documents which are close in contents to the selected document will go to the top. We evaluate the performance of the re-ranking algorithm based on how many useful documents it can bring to the top. If all the documents relevant to the person of interests, the algorithm is graded the perfect point of 100%.

1.3.2 Overview of Our Approach

In our research, we use additional information besides the information in name ambiguity documents [110–115]. Basically, when using extra information to support the disambiguation process, we can hope to improve the performance of the system. In fact, our approach is analogous to the way that human disambiguates ambiguous names. In order to easy understand our approach, we begin with an analyzing how human disambiguates personal names. Then, we present our approach where we teach computer to imitate the method by human to disambiguate personal names.

How human disambiguates ambiguous names?

Let us consider a case that a user try to disambiguate a computer scientist from other people of other professions. Suppose that the user has in his/her hand two documents about the computer scientist. In the first document, he/she reads terms like "computer", "programming", "information". In the second document, he/she reads terms like "software", "algorithm". While reading these documents, he/she thinks about the computer topic and it is natural that he/she concludes that both these two documents refer to the same person, and this person is likely to be a computer scientist. If we notice the human's disambiguation process in this example, we can learn that human uses background knowledge and relations of information between documents to differentiate people. Infact, the user already has had a background about the computer topic before reading the two documents. And while reading the two documents, he/she thinks of the topic computer and he/she may conclude that the two documents refer to the same people because they are both about the computer topic. As we can see from this example, the background knowledge plays an important role in the human's disambiguation process. This background knowledge is the knowledge that human accumulates everyday from educations, from books, newspapers, and from daily experiences, etc. The broader knowledge that human owns, the more ability that human can disambiguate people.

How the computer can imitate the human?

If computers can imitate the way of human to disambiguate ambiguous names,

then computers can improve the disambiguation performance. We should equip some kinds of additional information to computers and teach them to exploit that supportive resource. Actually, additional resources have been used in some previous researches. For example, in supervised training approaches, training data is a kind of additional resource; in named entity recognition approaches, extraction rules also play the role of external helps. In our research, we try to equip the computer some knowledge so that it can use the knowledge to like human. This knowledge is different from the training data in supervised learning algorithm since its source is different from the source of web documents being disambiguated. The knowledge is general information which we can get from different places in the WWW. In this way, the collecting of knowledge for computer is similar to the knowledge accumulation of human.

How to equip a knowledge for the computer

We choose to provide a knowledge base for the computer by collecting several sets of documents on several topics. We use the knowledge base in the process of calculating document similarities. We call our method as "**Similarity via Knowledge Base (SKB)**". Actually, we can use any document set that satisfies our requirements, that is any document set that contains documents being categorized into topics. We think that some web directories like the Dmoz directories[2], the Google directories[3], and the Yahoo directories[4] are suitable for our purpose. These directories are created by human and they contain documents categorized into topics. Since we can use them as they are, we can reduce the cost of preparing knowledge base.

1.4 Organization of the Dissertation

This dissertation is organized as follows. In Chapter 2, we present previous researches that are related to our research. These researches are about disambiguation of word senses and disambiguation of personal names in some application domains. We discuss the ability to extend previous approaches to our research problem as well as their limitations when using them. In Chapter 3, we introduce the use of knowledge base to improve

[2]http://www.dmoz.org
[3]http://directory.google.com
[4]http://dir.yahoo.com

the vector space model, a conventional scheme for document similarity measurements. We point out the limitations of the vector space model to the name ambiguous problem web documents and explain how we use information from the knowledge base to measure document similarities more effectively. In Chapter 4, we further improve the use of knowledge base by applying the latent Dirichlet allocation method to the knowledge base. The latent Dirichlet allocation method can extract topics contained in documents in the knowledge base. For each topic, we extract important terms that are strongly related to the topic. Extracted topics are used to modify documents and measure document similarities. Then, in Chapter 5, we summarize our experiments to verify the effectiveness of our approaches. Our experiments used name ambiguity web documents from the Google search engine[5] and web directories from the Dmoz directories[6], the Google directories[7], and the Yahoo directories[8]. Next, in Chapter 6, we explain the demo system that we developed to illustrate how our approach worked and show what kind of information from the knowledge base it exploited. Finally, in Chapter 7, we conclude our research. We review our contributions in this research. We discuss the advantages and the disadvantages in our approach. We also discuss how our approach can be extended to the other kinds of applications.

[5] http://www.google.com
[6] http://www.dmoz.org
[7] http://directory.google.com
[8] http://dir.yahoo.com

Chapter 2

Related Researches

In this Chapter, we study the researches which are related to our approach. To look from a broad point of view, our research target of personal name disambiguation relates with the research of word sense disambiguation. To look from a narrow point of view, our research relates to some other researches about personal name disambiguation. However, while our research focuses on personal names in web documents, some other related researches focus on personal names in other domains like: the newspaper domain, the scientific publication domain. In the web document domain, there are some researches about documents relevant to famous people but our research try to disambiguate documents related to general people, which may be famous people or ordinary.

We review some of the previous approaches to the problems in word sense disambiguation and the problems in personal name disambiguation. These two classes of problems have some common characteristics and some approaches can work with both the two classes. We present previous approaches in each class in sequence and discuss the interchangeability of an approach in one class to the other class. For the word sense disambiguation problem, we present some approaches that use context similarities, supervised learning, unsupervised learning, and selectional preferences. For the personal name disambiguation problem, we present some approaches that use the vector space model, the named entity recognition, the keyword extraction, and the social network utilization. In the presentation of each approach, we will discuss the possibility of extending that approach to our problem of personal name disambiguation in the web

domain.

2.1 Word Sense Disambiguations

2.1.1 Problem Definition

The problem of word sense disambiguation has been researched for a long time [31, 37, 62, 81, 93, 106]. This problem appears in some applications like machine translation and text summarization. In these problems, the system needs to understand the correct meaning and function of ambigous words. There are two kinds of tasks in the word sense disambiguation. The first kind is to understand words' meanings. Some words have the same spell but they have different meanings. For example, the term "bank" may mean the place where people can deposit and withdraw money. It may also mean an area beside a river. This kind of word ambiguity is the ambiguity about word meanings or word definitions. The second kind of tasks is to understand the words' functions. For example, the word "record" may function as a noun or a verb. In the sentence "I bought a new record at the bookshop", "record" functions as a noun. However, in the sentence "I record the television program", "record" functions as a verb.

Since the two kinds of tasks in the word sense disambiguation have different characteristics, the approaches for each kind of tasks is different from the other. In the first kind of task, where the meanings of ambiguous words are different, the main approach is to differentiate the contexts related to different senses of words. Different senses of words often appear in different contexts, so words surrounding the ambiguous words are used to recognize the different contexts. Some previous approaches try recognize different contexts to disambiguate word senses. We present some supervised approaches and an unsupervised approach from this class of approaches. In the second kind of tasks in word sense disambiguation, where the functions of words are the objects to be disambiguated, there is a different approach. We present a selectional preference approach that tries to select the correct function of word by using the function of words surrounding an ambiguous word.

2.1.2 Supervised Approaches

In the supervised approaches [21, 39, 61, 64, 77, 84], some kinds of additional information are used to learn the contexts which appear together with words' senses. Some examples of additional information are training corpuses, dictionaries, and thesauri, etc.

Training data approach

In the approach that uses training data[39], example uses of a word with different senses are prepared in the training data. In the training phase, the system learns the probabilities that words are generated given a sense. In the test phase, the system uses these probabilities to calculate the probability that a sense is attached with an ambiguous word. The details are as follows.

Denote K senses of a word w as $s_1, s_2, ..., s_K$. Given a context c where w appears, we try to calculate the probabilities $P(s_k|c)$ for every $k = 1, 2, ..., K$. Denote the bag of words for the context c where w appears as $\{v_1, v_2, ..., v_J\}$. The system calculates a sense for w using the sense with the maximum of probability.

$$
\begin{aligned}
s' &= \arg\max_{s_k} P(s_k|c) && (2.1)\\
P(s_k|c) &= \frac{P(c|s_k)P(s_k)}{P(c)} \\
&\propto P(c|s_k)P(s_k) \\
&\propto P(s_k)\prod_{j=1}^{J} P(v_j|s_k) \\
&\propto \log P(s_k) + \sum_{j=1}^{J} \log P(v_j|s_k) && (2.2)
\end{aligned}
$$

The probabilities $\log P(v_j|s_k)$ are calculated in advance using the training data.

Dictionary based approach

The dictionary base approach [78, 120] uses a dictionary that provides the system

definitions of different senses for an ambiguous word. For each sense s_k, the definition D_k of that sense in the dictionary provides a bag of word for that sense. The dictionary is also used to get definitions $D_{v_j}$for words v_j that appear in the context c. We denote the definition . The score for each candidate sense is calculated by the overlap between D_k and D_{v_j}. The sense with the maximum score is assigned to w.

$$\begin{aligned} \text{score}(s_k) &= \text{overlap}(D_k, \cup_{v_j} D_{v+j}) && (2.3) \\ s' &= \arg\max_{s_k} \text{score}(s_k) && (2.4) \end{aligned}$$

Thesaurus based approach

The thesaurus based approach [33] uses word categories in a thesaurus to disambiguate word sense. Assume that each sense s_k of w is associated with a category topic $t(s_k)$ and a word v_j in the context c is associated with a category topic $t(v_j)$. Then, the score for s_k is the repetition of $t(s_k)$ in $t(v_j)$.

$$\begin{aligned} \text{score}(s_k) &= \text{repetition}(t(s_k), \cup_{v_j} t(v_j)) && (2.5) \\ s' &= \arg\max_{s_k} \text{score}(s_k) && (2.6) \end{aligned}$$

2.1.3 Unsupervised Approaches

In the unsupervised approach [20], the disambiguation process is carried without any training data with tagged labels. The algorithm calculates the probability $P(v_j|s_k)$ using only the contexts where ambiguous senses of words appear. This is done by a expectation maximization (EM) algorithm on context data. The algorithm can be summarized as follows.

The algorithm assumes that contexts where senses of ambiguous words are generated from a model. The model is parameterized by the set of parameters $P(s_k), k = 1, 2, ..., K$ and $P(v_j|s_k), j = 1, 2, ..., J; k = 1, 2, ..., K$. The probability of generating a context c_i is calculated as.

$$
\begin{aligned}
P(c_i) &= \sum_{k=1}^{K} P(c_i, s_k) \\
&= \sum_{k=1}^{K} P(c_i|s_k)P(s_k) \\
&= \sum_{k=1}^{K} \prod_{v_j \in c_i} P(v_j|s_k)P(s_k) \qquad (2.7) \\
\log P(c_i) &= \log \sum_{k=1}^{K} P(c_i|s_k)P(s_k) \qquad (2.8)
\end{aligned}
$$

The system tries to find the optimal set of parameters $P(s_k)$ and $P(v_j|s_k)$ that maximize the probability of generating $c_i, i = 1, 2, ..., I$. Denote the set of parameters as μ, the set of contexts as C. The objective function to be maximized is as follows.

$$
\begin{aligned}
L(C|\mu) &= \log \prod_{i=1}^{I} P(c_i) \\
&= \log \prod_{i=1}^{I} P(c_i) \\
&= \sum_{i=1}^{I} \log \sum_{k=1}^{K} P(c_i|s_k)P(s_k) \qquad (2.9) \\
P(c_i|s_k) &= \prod_{v_j} P(v_j|s_k)^{\text{count}(v_j \text{ in } c_i)} \qquad (2.10)
\end{aligned}
$$

The optimization of $L(C|\mu)$ is carried out in the expectation maximization (EM) fashion as follows.

1. **Initialization**
   Assign random probabilities to parameters $P(s_k), k = 1, 2, ..., K$ and $P(v_j|s_k), j = 1, 2, ..., J; k = 1, 2, ..., K$.

2. **E-step**
   Use the Equa. (2.10) to calculate $P(c_i|s_k)$.

3. **M-step**
   In this step we try to choose a new set of parameters μ^{new} so that the value of Equa. (2.9) increases close to maximum. The mathematical derivation gives the following Equas. to update μ^{new}.

$$
\begin{aligned}
h_{ij} &= \frac{P(c_i, s_k)}{P(c_i)} \\
&= \frac{P(s_k)P(c_i|s_k)}{\sum_{k'=1}^{K} P(s_{k'})P(c_i|s_{k'})} && (2.11) \\
P(v_j|s_k) &= \frac{\sum_{i=1}^{I} \text{count}(v_j \text{ in } c_i) \cdot h_{ik}}{\sum_{j'} \sum_{i=1}^{I} \text{count}(v_{j'} \text{ in } c_i) \cdot h_{ik}} && (2.12) \\
P(s_k) &= \frac{\sum_{i=1}^{I} h_{ik}}{\sum_{k'=1}^{K} \sum_{i=1}^{I} h_{ik'}} && (2.13)
\end{aligned}
$$

After the calculation of parameters $P(s_k), k = 1, 2, ..., K$ and $P(v_j|s_k), j = 1, 2, ..., J; k = 1, 2, ..., K$, the disambiguation can be done in the same way as the training approach using Equa. (2.1) and Equa. (2.2).

2.1.4 Selectional Preference Approaches

The selectional preference approach [24, 74] is to disambiguate the function that a ambiguous word play in a context. In this approach, the sentences containing ambiguous words are parsed to get part-of-speech (POS) tags of words and predicate-argument relations among words [29, 49, 87]. Selectional preferences are a set of rules that regulate possible candidates of POS tag pairs for valid predicate-argument pairs. The system uses these selectional preference rules to disambiguate functions for ambiguous words.

2.2 Personal Name Disambiguations

In this Section, we study some previous approaches on the personal name disambiguation. The problems of disambiguation personal name have been studied in several application scenarios. Some typical applications are the disambiguation of author names in scientific publications, the disambiguation of people mentioned in newspapers, and the disambiguation of famous people mentioned in web pages. For different kinds of applications, different approaches have been proposed that exploited the specific features and characteristics of input information appearing in that application scenarios.

2.2.1 Context Similarity Measurement Approaches

The context similarity measurement approach disambiguates ambiguous names based on contexts of people. Words surrounding personal names are used to form contexts of people. This approach has been used in personal name disambiguation in newspaper documents [5]. In this research, the authors used an internal document co-reference system [6, 34, 70] to extract text relevant to a person in a document. Then, they used the vector space model (VSM) [4, 59, 92, 118] to measure similarities between articles.

Vector space model method

The vector space model [4] is the conventional method for measuring the similarity of two documents. In the vector space model, a document is represented by a feature vector formed from the weights of terms in the document. Here, we review the tf-idf term weighting scheme [4], which is a conventional approach often used in the vector space model. In the tf-idf term weighting scheme, term weights are calculated using the weight of term in relation with the document containing the term and the weight of term in relation with the document set.

Term frequency

If a term appears frequently in a document, then that term may be strongly related to the document concerned, so its weight should be proportional to its number of occurrences in the document. Therefore, a term frequency represents the weight of that term in relation with the documents containing the term. From the probabilistic point of view, the term frequency of term t is proportional to the probability of observing the

term in a document d.

$$tf(t,d) \sim P(t|d) \tag{2.14}$$

Inverse document frequency

The tf-idf weighting scheme also uses a term's occurrences in the document set to calculate term particularity. Intuitively, if a term appears frequently in many documents its particularity is less than terms that appear in fewer documents.

Denote a set of N documents as $S = \{d_1, d_2, ...d_N\}$, $df(t)$ is the number of documents in S containing t. According to Zipf's law [63, 71, 122], term particularity is proportional to $\log \frac{N}{df(t)}$.

According to [1], we can also derive the term particularity using the information theory as follows. Suppose that we are given a document from the document collection S. If we do not have any more information about the document, the document can be any document in S. Hence, when we see the document, we earn the entropy $-\log \frac{1}{N}$.

Now, let us assume that we know that the document contains term t. How much is the value of that information in terms of entropy? Since we know that the document contains term t, the document now is chosen from $df(t)$ documents. When we see the document we only earn the additional entropy $-\log \frac{1}{df(t)}$. Hence, the information that the document contains term t gives us an amount of entropy as follows.

$$\begin{aligned} idf(t) &= -\log \frac{1}{N} - (-\log \frac{1}{df(t)}) \\ &= \log \frac{N}{df(t)} \end{aligned} \tag{2.15}$$

We get the same equation as the derivation using the Zipf law.

Using the term frequency (tf) and the inverse document frequency (idf), a term's weight is then calculated as follows.

$$\begin{aligned} tf\_idf(t,d) &= tf(t,d) \cdot idf(t)) \\ &= tf(t,d) \cdot \log \frac{N}{df(t)} \end{aligned} \tag{2.16}$$

Measurement of similarities

Equa. (2.16) is used to calculate weights of terms and to create a document vector for each document. Then, we use inner product between pairs of vectors to measure the similarity between pairs of documents. These similarity values are used in the clustering algorithm to cluster documents into groups and to disambiguate ambiguous names.

2.2.2 Keyword Extraction Approaches

In order to build contexts of people well, some researches extracted important terms and phrases in documents [17, 88, 104]. Term frequencies and co-occurrence counts of terms are processed using statistical method to recognize important terms that are related to people.

In [88], they first created contexts for terms using term co-occurrence information. The meaning of a term t was represented by terms that co-occur with t in documents. Weights of co-occurring terms formed the context vector of t. Since the number of terms that co-occur may be large, the singular value decomposition (SVD) method[38] was used to reduce the dimension of context vectors. After building context vectors of terms, context vector of documents were created by summation of term context vectors. The method was called *second order context vectors* [95]. Then, similarities between documents were calculated using the log likelihood between context vectors. This statistical approach was effective when the number of documents were large. In the research, they experimented with famous people, such as the soccer players Ronaldo and David Beckham, and the former Prime Minister of Israel, Shimon Peres.

In [17], the authors used the C-value/NC-value [36] method to extract phrases in documents. The C-value/NC-value approach used POS tags of words and extraction rules to extract candidates of phrases in documents. Then candidate phrases were weighted and a threshold value was used to filter out noise phrases. After extracting phrases in documents, they sent these phrases as queries to search engines and used snippets of the resulting documents to build contexts of phrases. Contexts of people in documents were summation of contexts of phrases in documents. This method is expensive because it requires to query the search engine frequently to build contexts

for key phrases.

2.2.3 Named Entity Recognition Approaches

Profiles of people are useful information when disambiguating people. Several research groups have proposed several methods of extracting personal profiles related to people like birthday, birthplace, names of friends, names of relatives. Besides personal profiles, named of entities in documents are also useful. Entities such as organization names, city names, countries may have direct links with people and play as sharp separating criteria for the disambiguation.

In [69, 73, 97], they try to extract personal profiles like birthday, birthplace, occupation, etc, to build contexts of people. They prepared template patterns before hand, and used these patterns to recognize personal information. Named of entities were extracted using POS tags of words [18, 27, 28, 58, 60, 96, 121] and extraction rules[18]. Training data with sample of named entities being tagged were used to train the recognition algorithm and to optimize parameters of the algorithm. Then, the trained algorithm worked with new documents to extract named entities. Dictionaries like of lists of named entities, professions, research topics, etc also were useful to extract profiles of people. In [46], they used databases such as DBLP[1] and Amazon[2], to extract professions, research keywords, and authors' names.

2.2.4 Social Network Utilization Approaches

Social networks of people[55, 119] are also useful while disambiguating personal name as different people tend to have different human relationships. It has been used in the applications of disambiguating authors in scientific publications [72] and disambiguating people in a movie database [68].

In [68], they disambiguated personal name in movie databases. They used graph to represent relationships between movie actors. Names of actors were represented by vertices and edges were drawn between pairs of actors that played in the same movie. In [72], relationships between authors of publication papers were represented in graph

[1] http://dblp.uni-trier.de
[2] http://www.amazon.com

in the similar manner. Using the personal relation graph, they disambiguated vertices of ambiguous same by analyzing their neighbor vertices.

To deal with ordinary people, who are included in few relevant documents, [8] proposed a method to extract a group of people simultaneously. People in this group are related to one another so their relevant web page set has a greater number of pages; these pages may share the same topic and be connected. The authors proposed two methods of extracting a group of people: one uses link information in web pages and the other uses the Agglomerative Conglomerative Double Clustering (A/CDC) algorithm [98, 99] to group together web pages having the same topic. This method required to know the community of people in advance. It may not be practical since when searching for a person on the web, we may not know his or her social network in advance.

2.3 Discussions

In this Chapter, we have introduced some researches which are related with our research problem of disambiguating ambiguous names of people. We discuss in this section the common aspects and the different aspects between previous research problems and our research problem. By comparing with previous research problems, we can learn how to extend previous approaches to our problem, and how to improve them so that they can work with new problem on disambiguating ambiguous senses in the domain of personal names.

The word sense disambiguation is the foremost problem being researched in the series of sense disambiguation researches. Some methods for the word sense disambiguation have originated methods for other sense disambiguation problem. For example, the approach that uses bags of words to represent contexts of senses has been adopted to the name disambiguation problem. However, the method of selectional preference that uses rules to regulate interactions between words is difficult to extend to the name disambiguation problem. It is worth to notice that the problem definition for name disambiguation problem should be different from the problem definition for word sense disambiguation. There are at least two different points between the two problems as follows. The first different point is that, in the word sense disambiguation, senses for a word can be predict in advance and in some applications, there is a definition for each

sense. This difference makes the training approach which works well with the word sense disambiguation problem to be unsuitable to the problem of name sense disambiguation. This is because the sense of a name may be an arbitrary person, training data for that person is unavailable. The second different point is in the application output in the word sense disambiguation and the personal name disambiguation. In the conventional word sense disambiguation, the output requires all senses of words to be tagged. Similar to this output target, some researches on personal name disambiguation also try to label all appearances of names. However, in the search application, where users are interested in only one person, labeling all name appearances may serve more than user requirements. If the labeling is corrected for all name appearances, the results satisfy the user requirement. But if we can not label all appearances correctly, it is better when we focus on names of the person that users are interested in and label these names. In order to match user requirements more closely, the re-ranking of documents so that the person of interests goes to the top may be more appropriate than the clustering of people into clusters. Since the system does not know for whom that users are searching, it needs a feedback from users to understand the target person. Then, upon the feedback, the system will try to re-rank documents so that documents mention the target person go to the top.

We presented some previous researches that tried to disambiguate personal names. They focused on personal names in some kinds of applications but only few of them targeted the personal names in the document space like the WWW. These approaches, including approaches that targeted in personal name in web documents are limited when applying to the general cases of personal names in web documents. The approach in [5] that used the vector space model is limited for web documents since web documents mentioning the same person may be about different events related to that person and therefore they may have different topics. For examples, documents mentioning a computer scientist may be web pages about his/her research interests, web pages about the course that he/she teaches, web pages of conferences that he/she presents his/her research results, etc. These documents have the same general topic about computer, but their specific topics may differ to each other. On the other hand, the approaches in [17, 88] that extract keywords, key phrases related to people are difficult when the person in concern is mentioned in only few pages. In such a case, the few number of

documents prevents the statistical algorithms to perform well. The approaches [46, 69] that try to extract personal profiles and names of entities related to are limited in the case of web documents since web documents contain noise information which is a disadvantage for named entity recognition algorithms. The approaches [8, 11, 22, 67, 68, 72] that utilize information from social networks of people can not be applied to general cases of personal names since we do not always know social networks of all people.

Since the previous approaches have limitations when disambiguating general cases of personal names in web documents, we should improve them to fit them with the new problem. We should also think of new ideas to exploit new features that newly appear in our problem and propose new approaches for the general cases of personal names in web documents. We will discuss the details of our approaches in Chapter 3 and Chapter 4.

Chapter 3

Using Web Directories to Improve Context Extractions for Name Disambiguations

In the Chapter 2, we have studied some researches about word sense disambiguation and personal name disambiguation. These researches are related to our research of name disambiguation in the web at some points. However, the differences in characteristics between web documents and other kinds of documents being used in previous researches prevent the directly applying of previous approaches to our problem.

In this Chapter, we present our approach to disambiguate personal names in web documents. The key point in our approach is that we use web directories as additional information to facilitate the disambiguation tasks. Naturally, when we use additional information, we can improve the disambiguation performance. Other kinds of additional information have been used in previous approaches like taxonomies of word, manually tagged training data, etc. However, the use of a neutral and raw data like web directories is hardly seen in previous approaches. Our use of web directories only requires a cheap cost and a short preparation time. Therefore, we hope that our approach can be deployed in many applications. In the coming Sections, we present the details about how we exploit information from web directories, the algorithms and formulae we use to deal with web directories and search result documents to disambiguate personal names.

3.1 Difficulties in Processing of Web Documents

The WWW space is a heterogeneous environment that hosts a variety of end-users and application services. Since the information publishers are of different kinds, web documents are very various. They are documents of organizations, companies or individual. They are formal documents as well as informal documents. They have different topics, writing styles, formats. Therefore, data in the web is very noisy and it is difficult to build an algorithm that can treat well any kind of web data.

Information of people in web documents is very different from information of people in other kinds of documents like: news articles, academic publications. In documents like news articles or academic publications, the entire of document is related to the concerned person. However, in web documents, only a part of document may contain the useful information and it may be mixed with noise data.

When working with general web documents, the previous approaches such as the vector space model approach, the named entity recognition approach, the keyword extraction approach have some limitations as follows.

The vector space model based on the tf-idf weighting scheme measures the similarity of two documents by using the inner product of document feature vectors. It works well when the two documents concern the same topic. When two documents concern the same topic, they have many common terms, so the inner product is large. Although the tf-idf weighting scheme works well with documents on the same topic, it may not work well with documents relevant to the same person, as they have very few terms in common. There are two reasons for this. First, documents relating to the same person need not to be about the same topic. Rather, they may have slightly different specific topics under the same general topic; therefore, common terms between documents are rare. Second, because documents on the web contain noisy information, only text surrounding a person's name seems to be relevant to that person, not the whole document. This further reduces the number of common terms.

In the named entity recognition approach, the key point is that it learns the patterns of named entities and recognizes named entities when they match the patterns. Basically, the learn process requires training data with examples of named entities being tagged. There are two limitations of this learning approach. The first limitation

is that it requires much training text data which much be prepared by human. Since web documents are huge in amount, it is hardly to prepare tagged training data. The second limitation is that data in the web changes very dynamically. Therefore, it is almost impossible to create and maintain the pattern database so that the database can keep up to date with new patterns.

In the keyword extraction approach, the system has to count the frequencies of words and phrases to find out important words and important phrases. When a person is mentioned in a number of pages in the web, this statistical approach is effective. However, an arbitrary person in the web may have only few documents. For this kind of person, statistical approach is ineffective because the frequencies of terms and phrases are few.

3.2 Introduction of Knowledge Bases

As we summarized in the previous section, the previous approaches face some difficulties when working with web data. In order to work well with web data, an approach should be able to deal with data having different characteristics; it should only require a little cost for preparation of training data.

In order to cope with name disambiguation in web documents, we should use a kind of extra information to enrich contexts in web documents. Naturally, if we have additional resources beside the web documents and if we find an effective algorithm to utilize those resources, we can hope to improve the disambiguation performance.

Even the human, when disambiguating personal name, we use our knowledge to recognize important information of people like professions, specializations, hobbies, etc. Let us consider an example when we want to find out for a computer scientist and to disambiguate his name from other names. In the result set for his name query, we may read documents that contain terms related to computer, software, information technology, etc. Since we have our knowledge about computer, we recognize the topics related to computer in the documents and we can disambiguate the computer scientist from the other people. As we can see from this example, if we do not know about computer, we have no way to recognize the computer topics in result documents and we could not find out the person of our interests.

It is worth for computer to learn the way human disambiguate personal names and to take a similar approach. The computer should own a kind of knowledge in order to disambiguate personal names effectively. In fact, the use of additional information can be seen in some previous approaches. For example, in the some researches on word sense disambiguation, the approach of using training data to learn the contexts of word senses is an approach that uses additional information. Word taxonomies and dictionaries which are used in some researches can be also regarded as additional information. The named entity recognition approach also uses additional information. It uses training data that contains examples of named entities to learn for matching patterns and to optimize the matching algorithms.

A knowledge base for computer

The purpose of preparing a knowledge base for computer is to help it to recognize important terms in web documents. In the word sense disambiguation problem, example uses of word senses are used for the training algorithm. However, this approach can not be applied to train senses of personal names. This is because we do not have a list of people that we want to disambiguate. Therefore, the preparation of senses for that people is impossible. Instead, we prepare general contexts that may appear in web documents and use these contexts as a role of additional information. We prepare several sets of documents, each set contains documents on the same topics and we utilize contexts in these document sets. We call such a collection a knowledge base, because it collects knowledge of several topics in several sets of documents. In our research, we use web directories in the role of a knowledge base. Below, we use "knowledge base" and "web directories" interchangeably to refer to a collection of documents on several topics and we use "a directory" to refer to a set of documents on the same topic. We name our method **"Similarity via Knowledge Base (SKB)"** to separate it from the vector space model based on the tf-idf term weighting scheme. Hereafter, we refer to the vector space model based on the tf-idf term weighting scheme as the traditional vector space model, or the VSM for its abbreviation.

Utilization of information from web directories

The roles of web directories is to help the extraction of contexts in web documents.

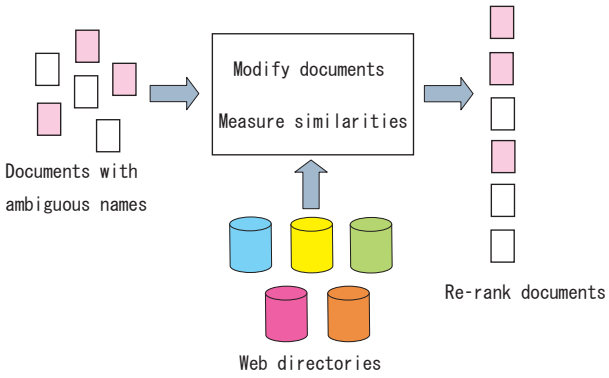


Figure 3.1: Overview of our approach

We compare web documents with contexts contained in web directories to recognize important information and use contexts in web directories to extract and to enrich important information related to people. The overview of our approach is shown in Fig. 3.1. The outline of utilization of web directories is as follows Web directories are first preprocessed to refine the topics and contexts. Then, these topics and contexts are used to modify web documents to amplify weights of terms related to important contexts. Document similarities are calculated using modified documents. We will explain the details of calculation in the next Sections.

3.3 Cooperation of Knowledge Bases and the Vector Space Model

3.3.1 Preprocessing of Web Directories

We first process web directories to calculate their feature vectors. A web directory is a collection of documents which are close in topics. We use the tf-idf scheme to calculate the document vector for each document. A feature vector of a web directory is calculated using document vectors of documents inside the web directory.

Denote D as a web directory and $\{d_1, d_2, ..., d_{|D|}\}$ as the set of documents in D. Denote $\vec{d_i} = (d_{i,1}, d_{i,2}, ..., d_{i,n})$ as the document vector of document d_i. We combine the

vectors $\vec{d_1}, \vec{d_2}, ..., \vec{d}_{|D|}$ to calculate the feature vector $\vec{D} = (D_1, D_2, ..., D_n)$ as follows.

$$D_k = \left(\frac{\sum_{i=1}^{|D|} d_{i,k}^{\alpha}}{|D|} \right)^{\frac{1}{\alpha}} \tag{3.1}$$

where α is a tuning parameter.
When $\alpha = 1$, we get the familiar arithmetic mean of d_i. When $\alpha = -1$, we get the harmony mean. When $\alpha \to \infty$ and $\alpha \to -\infty$, we get the values as follows.

$$\lim_{x \to \infty} D_k = \max_{i=1,...,|D|} d_{i,k} \tag{3.2}$$

$$\lim_{x \to -\infty} D_k = \min_{i=1,...,|D|} d_{i,k} \tag{3.3}$$

We try several values of $\alpha = 1, -1, 2, \infty$ and we get the good result of directory feature vectors when α is in around 1.

3.3.2 Modification of Term Weights in Documents

In a web document, text relevant to a person tends to be short because only a part of the document mentions the person and the web document may contain noise. Therefore, term weights calculated by Equa. (2.16) for keyword terms and for other terms differ only slightly.

Assume we have a directory whose topic is close to the document's topic. As the directory has abundant text, keywords related to the topic appear more frequently, so their term weights will be larger than the weights of other terms. It is reasonable to assume that keywords will appear on the web document as frequently as they appear in the directory if the relevant text on the web document increases in length. Therefore, we can use the large weights of keyword terms in the directory to amplify the small weights of keyword terms in the document [112].

Denote $S_{dir} = \{dir_1, dir_2, ...dir_K\}$ as a set of web directories on several topics, M as the total number of documents in S_{dir}, $tf(t, dir_i)$ as the number of times term t appears in the directory dir_i, $df(t, S_{dir})$ as the number of documents in S_{dir} containing t, and $length(dir_i)$ as the total number of word counts in the directory dir_i. We calculate term weights for the feature vector of directory dir_i as follows.

$$idf_{DIR1}(t, S_{dir}) = log\left(\frac{M}{df(t, S_{dir})}\right) \tag{3.4}$$

$$tf\_idf_{DIR1}(t, dir_i, S_{dir}) = \frac{tf(t, dir_i) \times idf_{DIR1}(t, S_{dir})}{length(dir_i)} \tag{3.5}$$

Because we have to compare feature vectors between directories in the next calculation step, we normalize term weights by dividing them by the lengths of the directories to facilitate comparison.

We modify the term weights in documents by taking the mean of the term weights calculated by Equas. (2.16) and (3.5). We have tested the arithmetic mean and the geometric mean, and the geometric mean gives the better result of the two, because, when taking the arithmetic mean, terms that do not appear in the document have weights larger than zero and the total weight of these terms dominates the total weight of terms that appear in the document. Following on from this experimental result, we use the geometric mean in our research.

The details of term weight modification can be formalized as follows.

$$\begin{aligned} tf\_idf_{SKB1}(t, doc, dir_i) &= \sqrt{tf\_idf(t, doc, S_{doc}) \times tf\_idf_{DIR1}(t, dir_i, S_{dir})} \\ &= \sqrt{tf(t, doc) idf(t, S_{doc}) \times \frac{tf(t, dir_i) idf_{DIR1}(t, S_{dir})}{length(dir_i)}} \end{aligned} \tag{3.6}$$

Our modification of term weights functions analogously to a signal frequency filter. A document can be regarded as an information source and the set of all terms can be regarded as a range of frequencies. A document feature vector corresponds to a power spectrum, where a term weight corresponds to the power at a certain frequency. A directory will amplify weights of terms close to the topic in the directory while dampening the weights of the other terms.

3.3.3 Modification of Term Weights in Directories

The *idf* factor in the tf-idf weighting scheme can be explained using the information entropy theory [47]. For example, in [1, 2], the author explained the $idf = log(\frac{N}{df(t)})$ factor for a term t as the information amount gained by t. Without the observation that

t appears in a document d, d can be any document from a collection of N documents. However, given the fact that d contains t and there are df documents in the collection containing t, d is now chosen from df documents. Therefore, the information gained by t is the difference between the two entropies: $log(\frac{1}{df}) - (log(\frac{1}{N}) = log(\frac{N}{df})$.

We modify term weight measurements in the directories as follows [114, 115]. The explanation by [1] assumed that contexts of documents in the collection were independent to each other. Therefore, term t was assumed to be related with $df(t)$ different contexts in $df(t)$ documents. However, for our directories, documents in the same directory are supposed not to be independent to each other; some documents may have common contexts. If a term t that appears frequently in a certain directory but appears less frequently in general, then it tends to be strongly related to the directory's topic. Although t may have a large value of df, the number of contexts related to t should be much lower than df since documents containing t seem to have common context. Therefore, its gain of information amount should be increased.

We define the normalized document frequency of a term in a directory and in all directories as follows.

$$\overline{df}(t, dir_i) = \frac{df(t, dir_i)}{M_i} \tag{3.7}$$

$$\overline{df}(t, S_{dir}) = \frac{df(t, S_{dir})}{M} \tag{3.8}$$

where $df(t, dir_i)$ is the number of documents in dir_i containing term t, and M_i is the number of documents in dir_i.

We assume that when terms have normalized frequencies in a certain directory that are much larger than their normalized frequencies in all directories, then those terms appear to be topic terms in the directory concerned. We propose the following equations that can appropriately increase idf weights for topic terms.

$$modifier(t, dir_i) = \begin{cases} \frac{\overline{df}(t,dir_i)}{\overline{df}(t,S_{dir})}, \text{ if } \frac{\overline{df}(t,dir_i)}{\overline{df}(t,S_{dir})} > r \\ 1 \quad \text{otherwise} \end{cases} \tag{3.9}$$

$$idf_{DIR2}(t, S_{dir}, dir_i) = log\left(\frac{M}{df(t, S_{dir})} \times modifier(t, dir_i)\right) \tag{3.10}$$

where r is a given threshold that we call the document frequency ratio threshold.

We combine the modification Equa. (3.10) of term weights in directories with the modification Equa. (3.6) of term weights in documents and obtain the following equations to measure term weights.

$$tf\_idf_{DIR2}(t, dir_i, S_{dir}) = \frac{tf(t,dir_i) \times idf_{DIR2}(t,S_{dir},dir_i)}{length(dir_i)} \tag{3.11}$$

$$\begin{aligned} tf\_idf_{SKB2}(t, doc, dir_i) &= \sqrt{tf\_idf(t, doc, S_{doc}) \times tf\_idf_{DIR2}(t, dir_i, S_{dir})} \\ &= \sqrt{tf(t, doc) idf(t, S_{doc}) \times \frac{tf(t,dir_i) idf_{DIR2}(t,S_{dir})}{length(dir_i)}} \end{aligned} \tag{3.12}$$

Our idea of using information from directory structure to modify term weights of topic terms has common points with the term weighting scheme using the term entropy with regard to an external directory structure in [57]. Both our approach and the approach in [57] utilize the term probabilities in specific directories and in general directories to appreciate weights of topic terms. In [57], training documents and test documents are from the same source and term weights for test documents are calculated using their entropies in training documents. However, in our approach, web directories and name ambiguous documents are from different sources, so we only use term weights in web directories to modify the term weight measurements in name ambiguous documents as above.

3.4 Personal Name Disambiguations

3.4.1 Measurement of Document Similarities

The measurement of document similarities is performed in two steps. First, we find directories that have topics close to that of the document. Then, we measure the document similarities using these selected directories. The details are as follows.

Find directories close in topic with the document

Because we do not know the documents' topics in advance, we have to guess their topics. For each document, we choose k directories in the knowledge base whose similarities to the document are the top k largest values. The similarity between a document d and a directory Dir is measured as follows.

$$SIM(doc, dir) = \sum_{t \in doc \cap dir} tf\_idf_{SKB}(t, doc, dir) \tag{3.13}$$

where $tf\_idf_{SKB}(t, doc, dir)$ is replaced by $tf\_idf_{SKB1}(t, doc, dir)$ or $tf\_idf_{SKB2}(t, doc, dir)$.

We call these top k directories of document doc the document's representative directories and denote this set of directories as $R(doc)$.

Measure document similarities

Denote a pair of documents as (doc_1, doc_2). For each directory dir_i in the union set $R(doc_1) \cup R(doc_2)$, we calculate the similarity between documents doc_1 and doc_2 via directory dir_i.

$$\begin{aligned} &SIM(doc_1, doc_2, dir_i) \\ &= \sum_{t \in doc_1 \cap doc_2} tf\_idf_{SKB}(t, doc_1, dir_i) \times tf\_idf_{SKB}(t, doc_2, dir_i) \end{aligned} \tag{3.14}$$

After calculating the similarities of doc_1, doc_2 via all representative directories, we take their sum as the similarity of the document pair (doc_1, doc_2).

$$SIM(doc_1, doc_2) = \sum_{dir_i \in R(doc_1) \cup R(doc_2)} SIM(doc_1, doc_2, dir_i) \tag{3.15}$$

3.4.2 Name Disambigutation by the Re-ranking of Documents

We use the re-ranking method to disambiguate ambiguous names. In the searching for people, users usually search for only one person. Therefore, by bringing documents which are relevant to the person of interest, we can help users to find useful documents more quickly. The re-ranking procedure is done by interactions with users. Users first select a document that refers to the person of interest. Upon receiving users' feedback, the system re-ranks all documents by the order of similarities to the selected document.

3.5 Conclusions

Disambiguation of people in web searches is an increasing requirement for the new trends in web search systems. We propose a new method that uses web directories as a knowledge base to improve the disambiguation performance. Using web directories, we propose two approaches to better measure term weights. In Chapter 5, we report the experiments of our approaches that uses several existing web directories to disambiguate documents of people on the web. The results showed a significant improvement with our system over the conventional methods: the vector space model method and the named entity recognition method. We also verified the robustness of our methods experimentally with different web directory structures and with different parameter values.

Chapter 4

Extractions of Topics in Web Directories for Improvements of Name Disambiguations

In the Chapter 3, we introduced our approach of using web directories as a role of an additional information when disambiguating ambiguous names. We utilized web directories by using term frequencies of topic terms in web directories to modify term weights in documents that contain ambiguous names. Our algorithm was an improvement of the conventional vector space model. We proposed calculating equations intuitively so that we could exploit information from web directories.

In this Chapter, we introduce the use of the latent Dirichlet allocation (LDA) method [16] to exploit information in web directories. The LDA method is a method that models the generation processes of documents. It bases on the Dirichlet distribution in statistical theory. Using the Dirichlet distribution, the LDA method assumes that each term in a document is generated in two steps. In the first step, a topic of a term is selected from a topic distribution which follows a Dirichlet distribution. In the second step, after the term's topic is decided, the term is selected from a word distribution corresponding to the selected topic, which also follows a Dirichlet distribution. By applying the Dirichlet distribution to the document generation processes, we are able to extract important topics in documents and to recognize important terms in the topics. We first introduce the Dirichlet distribution and the document generation processes in

the LDA method. Then, we introduce the applying of the LDA method to preprocess web directories and to disambiguate ambiguous names in web documents.

4.1 Introduction to Some Basic Distributions

The Dirichlet distribution is a generalization of the beta distribution. In order to understand the Dirichlet distribution well, we first introduce the beta distribution and then the Dirichlet distribution.

4.1.1 Beta Distribution

Suppose we have an unbalance coin that when we flip it, the probabilities of getting a head side and getting a tail side are not equal. If we flip the coin n times, what is the probability that we get k times of head side. Denote the probability of getting a head side as p, the probability of getting a tail side as $1-p$. The probability of getting head side k times is as follows.

$$P(k \text{ times of head side}) = \frac{n!}{k!(n-k)!}p^k(1-p)^{n-k} \tag{4.1}$$

Now, let us consider the reverse problem. If we flip the coin n times and get k times of head side, are we able to guess the character of the coin, that is can we guess the probability p that it produces head side. Actually, we can not know the exact value of p, but we can understand that at which value p tends to take. In the statistical language, we can calculate the distribution of p on the interval $(0, 1)$ as follows.

$$P(p = t|n, k) = \frac{P(n, k, p = t)}{P(n, k)} \tag{4.2}$$

$$\begin{aligned} P(n, k, p = t) &= P(n, k|p = t)P(t) \\ &= \frac{n!}{k!(n-k)!}t^k(1-t)^{n-k}P(t) \end{aligned} \tag{4.3}$$

Since $P(t)$ is uniform when we do not have any information about the coin and P(n,k) is the same for any t we get.

$$\begin{aligned} P(p=t|n,k) &\propto t^k(1-t)^{n-k} \\ &= \frac{1}{N}t^k(1-t)^{n-k} \end{aligned} \tag{4.4}$$

The normalization factor N is calculated as.

$$\begin{aligned} N &= \int_0^1 t^k(1-t)^{n-k}dt \\ &= \frac{k!(n-k)!}{(n+1)!} \end{aligned} \tag{4.5}$$

Therefore, we the arrive the distribution of p is as follows.

$$P(p=t|n,k) = \frac{(n+1)!}{k!(n-k)!}t^k(1-t)^{n-k} \tag{4.6}$$

If we denote $k=\alpha-1$ and $n-k=\beta-1$ and generalize Equa. (4.6) for the real values of α and β as follows.

$$P(p=t|\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}t^{\alpha-1}(1-t)^{\beta-1} \tag{4.7}$$

The distribution in Equa. (4.7) is called the beta distribution [35]. As we can see from the above derivation its meaning is that we can understand the distribution of the unknown coin if we observe the head side $\alpha-1$ times and the tail side $\beta-1$ times. A character about the expectation of beta distribution is as follows.

$$\begin{aligned} E(t) &= \int_0^1 P(p=t|\alpha,\beta)tdt \\ &= \int_0^1 \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}t^{\alpha-1}(1-t)^{\beta-1}tdt \\ &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\frac{\Gamma(\alpha+1)\Gamma(\beta)}{\Gamma(\alpha+\beta+1)} \\ &= \frac{\alpha}{\alpha+\beta} \end{aligned} \tag{4.8}$$

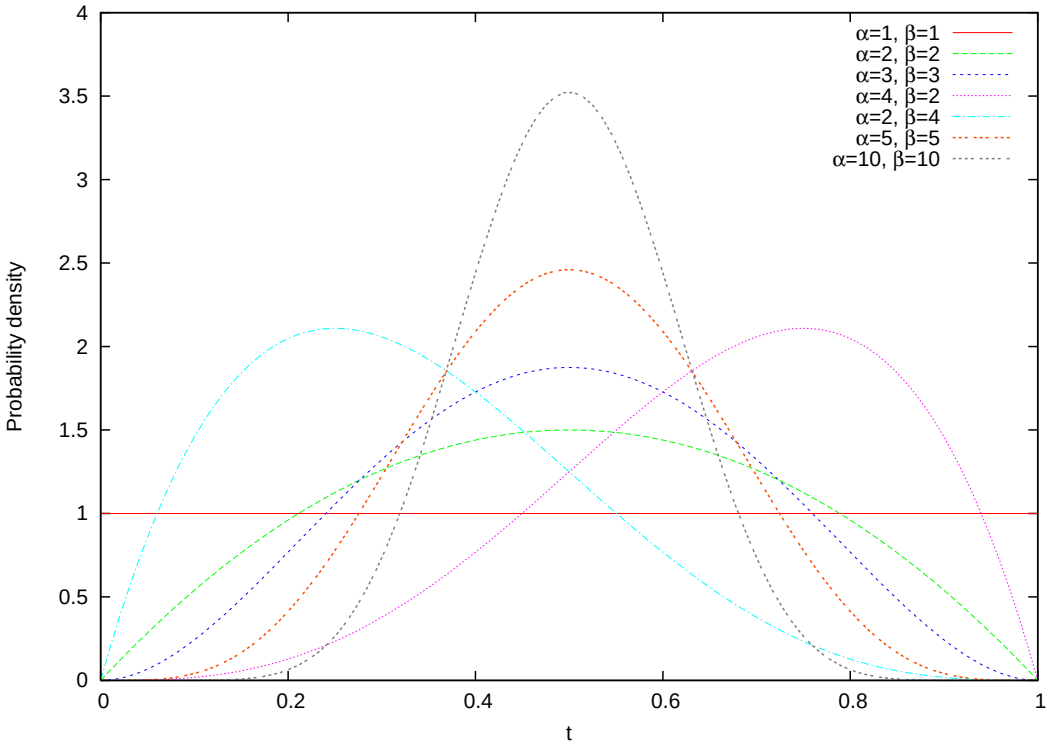


Figure 4.1: Density probability functions of the beta distributions

This result agrees with the fact that the modeling of the coin should match the observation of the coin.

Fig. 4.1 shows the density probability functions of the beta distributions with some pair values of α and β. As we can learn from this Fig., the beta distribution is symmetric when parameters α and β are equal. It reshapes towards 1 when α is larger than β, and it reshapes towards 0 when α is smaller than β.

4.1.2 Dirichlet Distribution

The Dirichlet distribution is a generalization of the beta distribution [35, 40]. Suppose we a coin of n sides and the parameters (α, β) are extended to $(\alpha_1, \alpha_2, ..., \alpha_n)$ corresponding to n sides of the coin. Then the distribution of the coin is as follows.

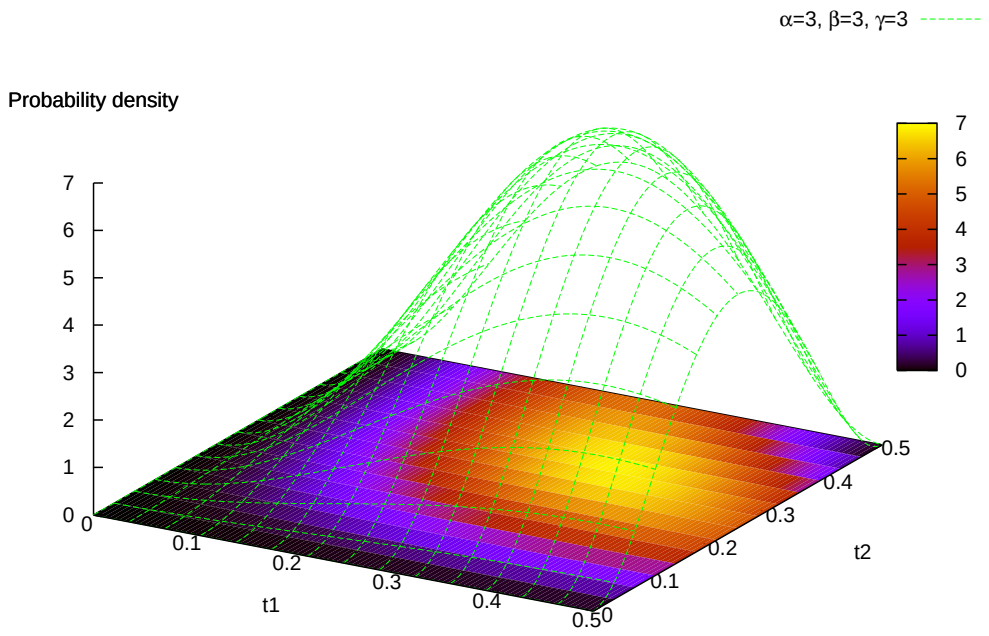


Figure 4.2: Density probability functions of the Dirichlet distribution ($\alpha = 3, \beta = 3, \gamma = 3$)

$$P(t_1, t_2, ..., t_n | \alpha_1, \alpha_2, ..., \alpha_n) = \frac{\Gamma(\alpha_1 + \alpha_2 + ... + \alpha_n)}{\Gamma(\alpha_1)\Gamma(\alpha_2)...\Gamma(\alpha_n)} t_1^{\alpha_1 - 1} t_2^{\alpha_2 - 1} ... t_n^{\alpha_n - 1} \tag{4.9}$$

Similar to the beta distribution, the expectation of t_i in the Dirichlet distribution is as follows.

$$E(t_i) = \frac{\alpha_i}{\alpha_1 + \alpha_2 + ... + \alpha_n} \text{ , for } i = 1, 2, ..., n \tag{4.10}$$

Figs. 4.2, 4.3, and 4.4 show some examples of the Dirichlet distribution of three variables.

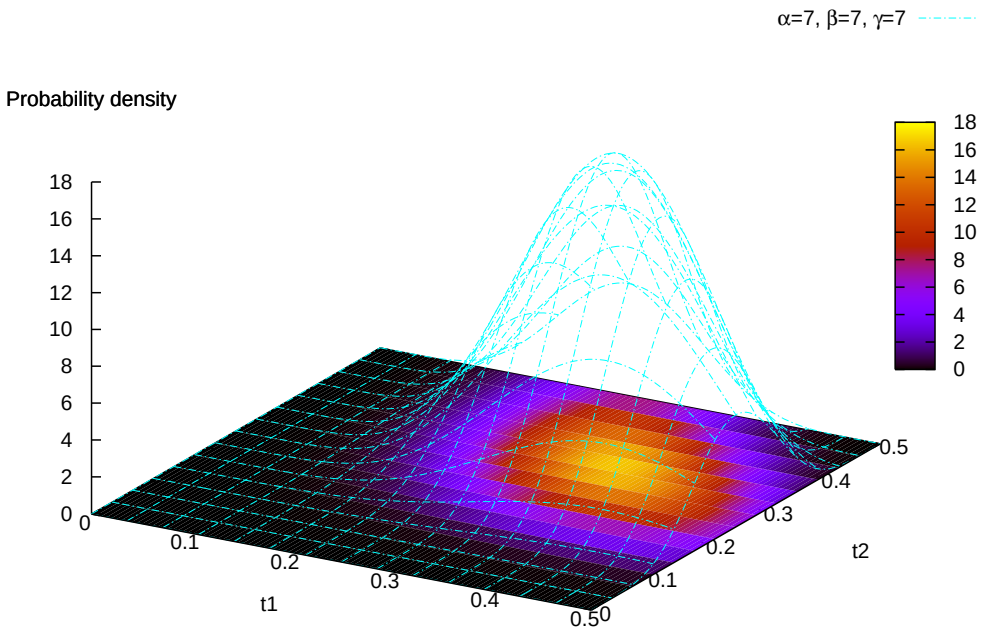


Figure 4.3: Density probability functions of the Dirichlet distribution ($\alpha = 7, \beta = 7, \gamma = 7$)

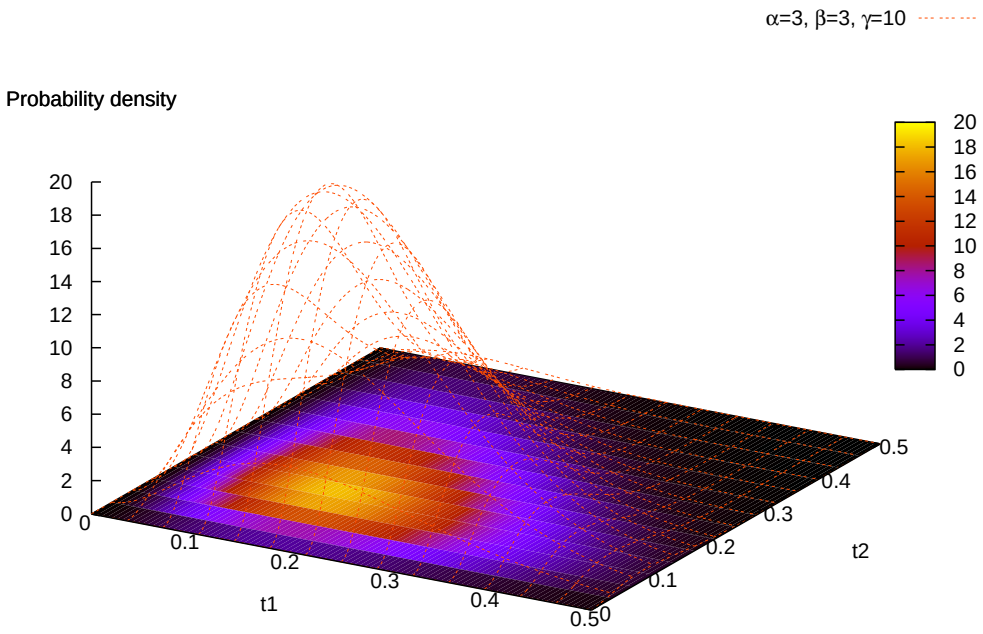


Figure 4.4: Density probability functions of the Dirichlet distribution ($\alpha = 3, \beta = 3, \gamma = 10$)

4.2 Latent Dirichlet Allocation Method

4.2.1 Document Generation Process

The latent Dirichlet allocation (LDA) method [16, 45] is an effective method to find out the relationship between topics and words. It uses the Dirichlet distribution to model the topic distributions for documents and the word distributions for topics. It has two main assumptions as follows. First, it assumes that all documents are mixtures from a set of topics. Given a set of documents, these topics are not defined explicitly but they exist prior to documents latently. Each document has its own distribution over topics. Second, the LDA assumes that topics in turn are mixtures of words, each topic has its own distribution over words.

Let D, T, and W to be the number of documents, the number of latent topics, and the number of different words, respectively. For a document d, its topic distribution is represented by a vector $\Theta_d = (\vartheta_{d,1}, \vartheta_{d,2}, ..., \vartheta_{d,T})$. Vector Θ_d satisfies the following condition.

$$\sum_{t=1}^{T} \vartheta_{d,t} = 1 \tag{4.11}$$

The vectors satisfying (4.11) form a $T-1$ simplex space. The topic distribution vector of the document d_i has the distribution density as follows.

$$P(\Theta_d|\vec{\alpha}) = \frac{\Gamma(\sum_{t=1}^{T} \alpha_t)}{\prod_{t=1}^{T} \Gamma(\alpha_t)} \prod_{t=1}^{T} \vartheta_t^{\alpha_t - 1} \tag{4.12}$$

where $\vec{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_T)$ and $\alpha_1, \alpha_2, ..., \alpha_T$ are hyper-parameters.

For a topic t, its word distribution is represented by a vector $\Phi_t = (\varphi_{t,1}, \varphi_{t,2}, ..., \varphi_{t,W})$. Vector Φ_t satisfies the following condition.

$$\sum_{w=1}^{W} \varphi_{t,w} = 1 \tag{4.13}$$

The vectors satisfying (4.13) form a $W-1$ simplex space and the distribution density in the $W-1$ simplex space is also assumed to follow a Dirichlet distribution as follows.

$$P(\Phi|\vec{\beta}) = \frac{\Gamma(\sum_{i=1}^{W} \beta_i)}{\prod_{i=1}^{W} \Gamma(\beta_i)} \varphi_1^{\beta_1 - 1} ... \varphi_W^{\beta_W - 1} \tag{4.14}$$

where $\vec{\beta} = (\beta_1, \beta_2, ..., \beta_W)$ and $\beta_1, \beta_2, ..., \beta_W$ are hyper-parameters.

4.2.2 Parameter Reference

In order to find out latent topics and distribution of topics for documents, the LDA has to inference the parameter vectors Θ, Φ for all documents and topics. It can be solved by the expectation maximization method [16] or the Gibbs sampling method [45]. Here, we describe the details about Gibbs sampling method, which we use for our problem.

In order to find optimal parameters, the Gibbs sampling method try to solve the problem of assigning topic IDs to words in documents. The topic ID assignment problem is equivalent to the problem of finding parameter vectors Θ, Φ. That is, once we know the topic IDs for words in the documents, we can calculate the parameter vectors Θ, Φ. Vice verse, once we know the parameter vectors Θ, Φ, we can assign topic IDs to words in the documents.

Denote $\overrightarrow{w} = (w_1, w_2, ..., w_L)$ as the vector composed by lining up all words in all documents. Denote t_i as the topic ID assigned to the word w_i and define $\overrightarrow{t} = (t_1, t_2, ..., t_L)$ as the topic vector. The procedure of the Gibbs sampling algorithm is as follows.

1. **Initial step**
   We assign each word w in each document an arbitrary topic ID.

2. **Update step**
   For each word w_i, we remove its old topic ID and try to assign a new topic ID t_i for w_i using current topic IDs of other words in all documents. Define $\overrightarrow{t_{-i}} = (t_1, t_2, ...t_{i-1}, t_{i+1}, ...t_L)$. Using the Dirichlet distribution, the posterior distribution of the topic ID t_i is as follows.

$$P(t_i = t|\overrightarrow{t_{-i}}, \overrightarrow{w}) \propto \frac{n^{(t)}_{-i,d} + \alpha_t}{[\sum_{t'=1}^{T} n^{(t')}_{-i,d} + \alpha_{t'}]} \frac{n^{(w)}_{-i,topic_t} + \beta_w}{\sum_{w'=1}^{W} [n^{(w')}_{-i,topic_t} + \beta_{w'}]} \tag{4.15}$$

   where $w = w_i$, d is the document contains w_i, $n^{(t)}_{-i,d}$ is the number of words in d to be assigned the topic t except w_i, and $n^{(w)}_{-i,topic_t}$ is the total number of times the word w is assigned the topic t except w_i.

3. Repeat the update step until convergence

The parameter vectors Θ, Φ can be derived from topic IDs of words as follows.

$$\vartheta_{d,t} = \frac{n_d^{(t)} + k\alpha}{\sum_{t'=1}^{T}[n_d^{(t')} + \alpha_{t'}]} \tag{4.16}$$

$$\varphi_{t,w} = \frac{n_{topic_t}^{(w)} + \beta_w}{\sum_{w'=1}^{W}[n_t^{(w')} + \beta_{w'}]} \tag{4.17}$$

where $n_d^{(t)}$ is the number of words in dir_d to be assigned the topic t and $n_t^{(w)}$ is the total number of times the word w is assigned the topic t.

4.3 Applying Latent Dirichlet Allocation Method to Web Directories

We propose to use the LDA method to utilize web directories more effectively. The outline of our approach is shown in Fig. 4.5

We use the LDA method to When applying the LDA method for web directories in our approach, since we know that documents in the same directory have the resemble topic distribution, we use the same topic distribution for all documents in the same directory. We parameterize topic distributions of directories and word distributions of topics as follows.

In the conventional LDA method[16], the distribution density of these vectors in the $T-1$ simplex space is assumed to follow the same Dirichlet distribution for all documents. However, since we know in advance that each directory has its own specific topic, we assume that documents in a directory are influenced mainly by the specific topic associated with that directory, while receiving small influences from topics of other directories. We model this assumption by using different hyper-parameters to build different Dirichlet distributions for different directories. A hyper-parameter vector $\vec{\alpha}^{(i)}$ for a directory i is set to have a large hyper-parameter $\alpha_i = k\alpha$ for its associated specific topic, while having small hyper-parameters $\alpha_j = \alpha$ for other topics $j \neq i$ as follows.

We call k the bias factor of directories.

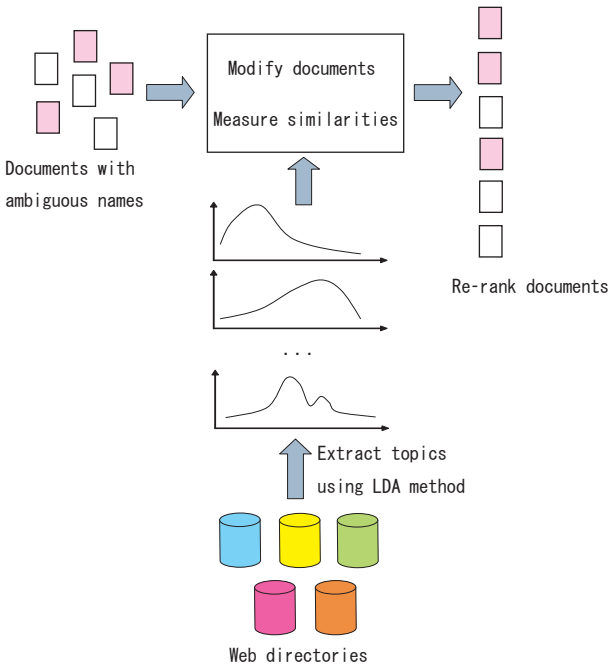


Figure 4.5: Overview of using LDA with web directories to disambiguate personal names

$$\vec{\alpha}^{(i)} = (\alpha, \alpha, ..., \alpha_i = k\alpha, ..., \alpha) \tag{4.18}$$

The use of the LDA method requires the selection of these parameters: the number of topics T, hyper-parameters $\alpha = (\alpha_1, \alpha_2, ..., \alpha_T)$ and $\beta = (\beta_1, \beta_2, ..., \beta_W)$. In our research, we assume that each directory forms its own specific topic so we select the number of topics to be the number of directories. We also assume that topic distribution for a directory is mainly influenced by the directory's own topic while receiving small influence from topics of other directories. In order to model this assumption, we use different hyper-parameters α for different directories. The hyper-parameter α for the directory i is set to have a large parameter $\alpha_i = k\alpha_0$ for its associated specific topic while small parameters $\alpha_j = \alpha_0$ for other topics $j \neq i$ as follows.

$$\alpha^{(i)} = (\alpha_0, \alpha_0, ..., \alpha_i = k\alpha_0, ..., \alpha_0) \tag{4.19}$$

We call k the bias factor of directories.

For each directory, we assume that all its documents follow the same topic distribution. That is, we introduce only one topic distribution vector for each directory.

$$\Theta_{dir} = (\vartheta_{dir,1}, \vartheta_{dir,2}, ..., \vartheta_{dir,T}) \tag{4.20}$$

Since we use biased hyper-parmeter, the initial step of the Gibbs sampling algorithm is modified so that terms in a directory tend to be assigned the directory's associated topic ID k times easier than other topic IDs. Also, using Dirichlet distribution with biased hyper-parameters for directories, we have the posterior distribution of topic ID t_i for a term w_i calculated in the update step as follows.

$$P(t_i = t|\overrightarrow{t_{-i}}, \overrightarrow{w}) \propto \begin{cases} \frac{n^{(t)}_{-i,dir_d}+k\alpha}{[\sum_{t'=1}^{T} n^{(t')}_{-i,dir_d}]+(k+T-1)\alpha} \frac{n^{(w)}_{-i,topic_t}+\beta_w}{\sum_{w'=1}^{W}[n^{(w')}_{-i,topic_t}+\beta_{w'}]}, \\ \qquad \text{if } t = dir_{ID} \\ \frac{n^{(t)}_{-i,dir_d}+\alpha}{[\sum_{t'=1}^{T} n^{(t')}_{-i,dir_d}]+(k+T-1)\alpha} \frac{n^{(w)}_{-i,topic_t}+\beta_w}{\sum_{w'=1}^{W}[n^{(w')}_{-i,topic_t}+\beta_{w'}]}, \\ \qquad \text{if } t \neq dir_{ID} \end{cases} \tag{4.21}$$

where dir_ID is the directory where the document containing w_i exists.

$$\vartheta_{dir,t} = \begin{cases} \frac{n^{(t)}_{dir_d}+k\alpha}{[\sum_{t'=1}^{T} n^{(t')}_{dir_d}]+(k+T-1)\alpha}, \\ \qquad \text{if } dir = t \\ \frac{n^{(t)}_{dir_d}+\alpha}{[\sum_{t'=1}^{T} n^{(t')}_{dir_d}]+(k+T-1)\alpha}, \\ \qquad \text{if } dir \neq t \end{cases} \tag{4.22}$$

$$\varphi_{t,w} = \frac{n^{(w)}_{topic_t} + \beta_w}{\sum_{w'=1}^{W}[n^{(w')}_{topic_t} + \beta_{w'}]} \tag{4.23}$$

4.4 Using Extracted Topics for Document Similarity Measurements

4.4.1 Topic feature vectors of new documents

We use the results of Gibbs sampling algorithm running over web directories to calculate topic feature vectors of words in new documents and topic feature vectors of new documents as follows.

Topic feature vectors of words

Using the results of Gibbs sampling algorithm, we derive the term-topic relationship in new documents as follows.

$$\begin{aligned} P(t_w = t|w) &= \frac{P(t,w)}{P(w)} \\ &= \frac{P(t)P(w|t)}{P(w)} \\ &\propto P(t)P(w|t) \\ &= P(t)\varphi_{t,w} \end{aligned} \tag{4.24}$$

Here, $P(w)$ in (4.24) is independent to t and $P(t)$ is approximate using the topic distribution in web directories. The topic feature vector for word w is defined as follows.

$$\overrightarrow{p_w} = (p_{w,1}, p_{w,2}, ..., p_{w,T}) \tag{4.25}$$

where $p_{w,t} = P(t_w = t|w)$.

Topic feature vectors of documents

A topic feature vector for a document is the weighted sum of words in the document. The importance weight for a word is considered as how much amount of information that word conveys. Given the word w was observed, we learn the topic distribution attached with w: $(p_{w,1}, p_{w,2}, ..., p_{w,T})$. If we have not observed w, the topic distribution is the same for all topics: $(\frac{1}{T}, \frac{1}{T}, ..., \frac{1}{T})$. Therefore, the information amount conveyed by

w is the different of information amounts between these two distributions.

$$\begin{aligned} & weight(w) \\ &= -T\frac{1}{T}log[\frac{1}{T}] \\ &\quad -\sum_{t=1}^{T} -P(t_w = t|w)log[P(t_w = t|w)] \\ &= log[T] + \sum_{t=1}^{T} P(t_w = t|w)log[P(t_w = t|w)] \end{aligned} \tag{4.26}$$

The topic feature vector for a new document d is calculated as follows.

$$\overrightarrow{\rho_d} = \sum_{w \in d} weight(w)\overrightarrow{p_w} \tag{4.27}$$

4.4.2 Modification of documents

We use a topic t extracted from web directories to modify a document d as follows. Denote $\overrightarrow{d} = (tf_1, tf_2, ..., tf_W)$ as the original document vector, where tf_w is the number of times the word w appears in d. Denote the modified document and its vector as d_t and $\overrightarrow{d_t} = (tf_1^{(t)}, tf_2^{(t)}, ..., tf_W^{(t)})$, respectively. We assume that terms in the modified document are generated by either the original document d or the topic t. The probability that the modified document d_t generates word w is derived as follows.

$$\begin{aligned} & P(d_t \text{ not generate } w) \\ &= P(d \text{ not generate } w)P(t \text{ not generate } w) \\ &= (1 - P(w|d))(1 - P(w|t)) \end{aligned} \tag{4.28}$$

Hence,

$$\begin{aligned} & P(w|d_t) \\ &= 1 - P(d_t \text{ not generate } w) \\ &= P(w|d) + P(w|t) - P(w|d)P(w|t) \end{aligned} \tag{4.29}$$

$$tf_w^{(t)} = P(w|d_t)length(d) \tag{4.30}$$

where $length(d)$ is the length of document d.

4.4.3 Document Similarity Using Modified Documents

The similarity of a document pair is calculated using modified documents as follows.

Denote (d_1, d_2) as a pair of document. Denote $\overrightarrow{\rho_1} = (\rho_{1,1}, \rho_{1,2}, ..., \rho_{1,T})$, $\overrightarrow{\rho_2} = (\rho_{2,1}, \rho_{2,2}, ..., \rho_{2,T})$ as documents' topic feature vectors.

Selection of representative topics

For each document $d_i (i = 1, 2)$, we select m topics that have the top m values of $\rho_{i,j}$. We call these topics as representative topics for the document and denote this set of topics as $R_i = \{t_{i,1}, t_{i,2}, ..., t_{i,m}\}$.

Document similarities via topics

Let $R = R_1 \cup R_2$. The similarity of (d_1, d_2) via a topic $t \in R$ is defined as.

$$Sim(d_1, d_2, t) = \rho_{1,t}\rho_{2,t}\overrightarrow{d_{1,t}}\overrightarrow{d_{2,t}} \tag{4.31}$$

where $\overrightarrow{d_{1,t}}$ and $\overrightarrow{d_{2,t}}$ are modified documents of d_1 and d_2 using the topic t. Next, the document similarity of (d_1, d_2) is defined as the summarization of document similarities via all representative topics in R.

$$Sim(d_1, d_2) = \sum_{t \in R} \rho_{1,t}\rho_{2,t}\overrightarrow{d_{1,t}}\overrightarrow{d_{2,t}} \tag{4.32}$$

4.5 Conclusions

Name disambiguation in searching for people in the web is becoming more crucial as the needs for retrieving useful information from the explosive WWW database have

been increasing. We have proposed a new method that are suitable to deal with web documents that have their own specific characteristics. Since web documents are noisy and relevant information to people is difficult to extract, it is required to use some kinds of additional information resources to complement and to enrich relevant information. In our research, we have used web directories in the role of an additional information resource. We have applied the LDA method, an effective method to extract latent topics in a set of documents, to preprocess web directories and to extract distributions of important terms in topics that are contained in documents in web directories. Then, the extracted latent topics are used to modify documents of people in search results so that we can differentiate important terms strongly related to contexts of people from general terms and evaluate them with more weights. Our experiment results in Chapter 5 show that the LDA method are effective to extract latent topics and the use of these latent topics have improved the name disambiguation performance. In future, we plan to cooperate our approach with other approaches using other kinds of additional resources like approach using dictionaries, approach using name entity recognition method, with the hope that different kinds of additional resources can complement to each other and they can together achieve the best performance in overall.

Chapter 5

Experiments

In this Chapter, we summarize the experiments in our research. The purposes of these experiments are to verify the effectiveness and to evaluate the performances of our approaches. Our approaches proposed to use web directories as an additional information sources. In order to verify the feasibility and the effectiveness by using web directories, we chose some general web directories that were well-known in the web. Selections of document sets in web directories were carried in an objective manner so that the results of our experiments could be interpreted that our approaches did not depend much on the variety of web directories and the using web directories was a key factor to the improvement. We also chose real documents in the web that contained ambiguous names. Documents were results from the Google searching engine for some ambiguous name queries.

We also carried the experiments with two baseline methods on the same data set. We selected the Vector Space Model approach and the Named Entity Recognition approach as the two baseline methods. These two approaches were used in some previous researches about name disambiguations.

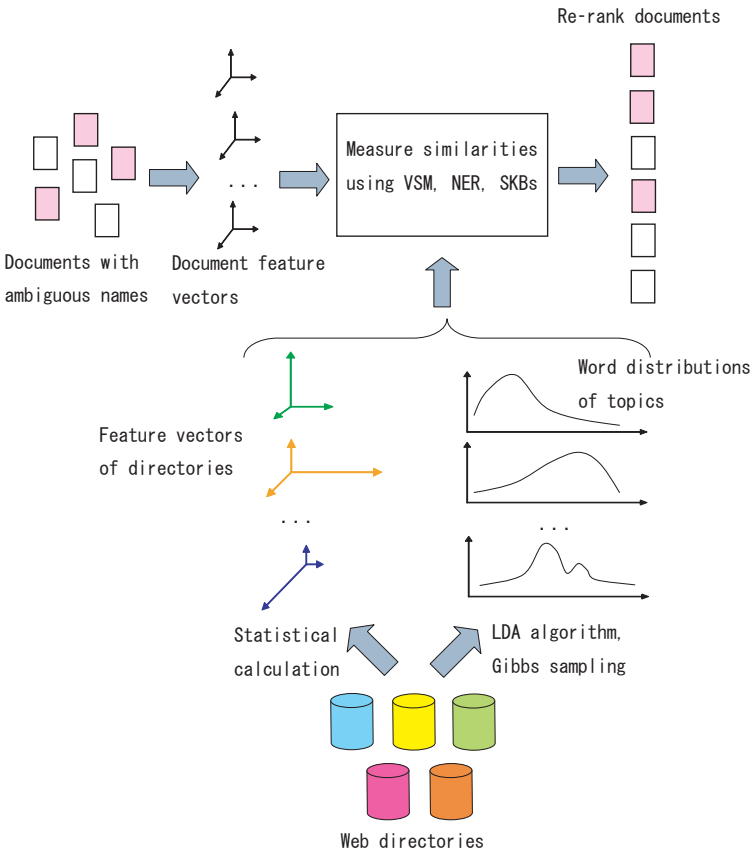


Figure 5.1: Experiment procedure

5.1 Experiment Method

5.1.1 Experiment Procedure

The flow chart of experiment procedure is shown in Fig. 5.1. The procedure consists of three main steps: the preprocessing step, the document similarity calculation step, and the name disambiguation step. The explanations for the three steps are as follows.

1. **Preprocessing**
   The input to the procedure was result documents from the search engine. These input documents were first preprocessed to convert data in the hypertext format

to the normal text format. The html tags in web pages were removed to get clean texts in documents. Next, we removed stop words and used the Porter stemming algorithm [1] to change words to their root forms. Among the remained words in a document, we chose n words before and n words after every personal name as relevant text of that person. Document texts were also processed to extract names of entities contained inside.

2. **Document similarity calculation**
   We calculated the similarities of all pairs preprocessed documents. This is the only step that has different calculation algorithms for different methods. The details of calculation algorithm for baseline approaches is described in Section 5.1.2. The details of calculation algorithm for our approaches is described in Section 5.3 and Section 5.4.

3. **Name disambiguation step**
   The algorithm in this step is the same for all methods. We used the re-ranking of documents[53, 107, 117] to disambiguate name and evaluated the re-ranking performances. Name disambiguation could be realize by clustering documents so that each cluster corresponds to exactly one person. However, we chose the re-ranking for evaluation because re-ranking is a practical use for name disambiguation in real web search applications. We discuss more details about the evaluation method in Section 5.1.3.

5.1.2 Baseline Methods

We compared our method with two conventional methods: VSM and named entity recognition (NER).

1. **Vector Space Model Method**
   In the VSM method, we removed stop words and stem words to their root form by using the Porter stemming algorithm. Then, among the remained words in a document, we chose n words before and n words after every personal name to

[1] http://tartarus.org/∼martin/PorterStemmer/

create the bag of word vector for the document. We used Equa. (2.16) to calculate the weight of these terms and built the feature vectors of documents. We took the inner products of document feature vectors for the similarities between document pairs.

2. **Named Entity Recognition method**
   In the NER method, we used the LingPipe software [2] to extract the entity names in the documents. Then, we used these names to construct feature vectors of the documents. The constituents of vectors were binary values (1 if a name appears in the document, 0 otherwise). We took the inner products of the document feature vectors for the similarities between documents.

5.1.3 Evaluation Metrics

In the traditional word sense disambiguation problem, the most often used evaluation method is to cluster documents and then evaluate the clustering results. This evaluation method is appropriate for word sense disambiguation problem since the definition of word senses may be defined in advanced. However, in the problem of name disambiguation, users are usually interested in only one person. Therefore, disambiguation by clustering may not be appropriate. Instead, we apply the method of re-ranking documents to disambiguate person of interests from other people. We evaluate the disambiguation performance by measuring the re-ranking performance [80, 108].

Average of performance over all documents in each set

We assumed that the user may choose any document doc_i in the result set, and evaluated the performance of the re-ranking result based on that document doc_i. We recorded the precision values at 11 recall points: 0%, 10%, 20%, ..., 90%, and 100% and denoted these as $P(doc_i, 0\%), P(doc_i, 10\%), P(doc_i, 20\%), ...,$
$P(doc_i, 90\%)$, and $P(doc_i, 100\%)$, respectively. We calculated the averaged precision values at these 11 recall points for all possible re-ranking sequences as follows.

$$P(D, k\%) = \frac{\sum_{doc \in D} P(doc, k\%)}{|D|} \tag{5.1}$$

[2] http://www.alias-i.com/lingpipe/

where D is a set of documents, $|D|$ is the number of documents in set D, and k $\in \{0, 10, 20, ..., 90, 100\}$.

Average of performance over all sets
We calculated the averaged precision over all sets

$$\overline{P}(k\%) = \frac{\sum_D P(D, k\%)}{N} \tag{5.2}$$

where N was the number of test sets.

Average of performance over all recall points
We also took the averaged value of these 11 averaged precision values.

$$\overline{P}_{overall} = \frac{\sum_{k=0,10,...,100} \overline{P}(k\%)}{11} \tag{5.3}$$

5.2 Data Sets

5.2.1 Data Sets of Web Directories

We chose three well-known directories to use in the experiments: the Dmoz directories, the Google directories and the Yahoo directories. For each directory, we selected document sets in objective fashion as follows. Document sets in the directory were organized in tree structure. First, we selected all document sets at the level two child nodes starting from the root node. Then, we removed document sets that have small number of documents. For each directory, we create two directory structures using thresholds of directory size to be 10 and 20, respectively. The details of six directory structures created are shown in Table 5.1.

5.2.2 Data Sets of People on the Web

We used 24 name queries to get documents from the Google search engine. These names were names of researchers specializing in research fields as shown in Table 5.2. For each name query, all documents of the researcher as well as documents of other people were used for experiments. We sent each name to the Google search engine and selected the

top 100 results. In each result set, there were documents relevant to the person who specialized in the field shown in the left column of Table 5.2 and documents relevant to other people. The documents for all these people were used in our experiments. We removed documents that were not html documents. For each name, the person bearing that name and specializing in the field shown in the left column of Table 5.2 was associated with from 10 to 50 documents, whereas other people bearing that name were associated with between one and 10 documents. Table 5.3 shows the number of people and the number of relevant documents. The first and third columns show the number of relevant documents, the second and fourth columns show the number of people who had that number of relevant documents.

We created two classes of document sets: the real namesake document sets and the pseudo namesake document sets. The details are as follows.

1. **Creation of real namesake document sets**
   The real namesake document sets were the 24 result sets as they were from the Google search engine.

2. **Creation of pseudo namesake document sets**
   In order to get a number of test data, we created name-ambiguous documents artificially as follows. We selected two result sets corresponding to the names of two people belonging to different research fields and mixed them together. Then, we replaced the personal names in the documents by the name X to create a set of documents of pseudo namesakes. In each mixed data set, there were two people with different professions, each with between 10 and 50 relevant documents.

Table 5.1: Number of directories and documents in directory structures

Directory name	Number of directories	Number of documents
Google10	214	6762
Google20	124	5318
Yahoo10	219	5979
Yahoo20	109	4524
Dmoz10	175	5701
Dmoz20	103	4551

Besides these two people, there were several other people with between one and 10 relevant documents. The mixing procedure was described in the following example. We selected the two result sets of queries "John D. Lafferty" and "Paul G. Hewitt". The two name queries were selected so that the two researchers bearing the two names had different research fields. This was to ensure that people in the mixed set have different professional careers. For example, the set mixed as above contained documents relevant to a computer scientist, a physicist and several other people. Personal name queries in these documents were replaced by a common name "X". We supposed these names are ambiguous and we tried to disambiguate them. This way of mixing documents yielded 216 test sets for the experiment data.

Although the pseudo namesake test sets were created by merging result sets of two different queries, they were not easier to disambiguate than the real namesake test sets because personal name in documents were removed. The pseudo namesake test sets were even more difficult than the real namesake test sets because they contained more number of documents and the documents referred to the more number of people.

5.3 Experiments on Using Web Directories to Extract Contexts of Document

5.3.1 Document Similarity Calculation

We used topics extracted from web directories to modify documents containing ambiguous personal name and calculated the document similarities according to the procedures

Table 5.2: List of 24 name queries

Field	Name
Computer science	Adachi Jun, Sakai Shuichi, Tom M. Mitchell Tanaka Katsumi, John D. Lafferty, Andrew McCallum
Physics	Paul G. Hewitt, Edwin F. Taylor, Paul W. Zitzewitz Frank Bridge, Kenneth W. Ford, Michael A. Dubson
Medicine	Scott Hammer, Thomas F. Patterson, Michele L. Pearson Henry F. Chambers, David C. Hooper, Lindsay E. Nicolle
History	John M. Roberts, David Reynolds, Thomas E. Woods Thomas A. Brady, William L. Cleveland, Peter Haugen

described in Section 4.4. Then, these similarity values were used to re-rank documents to disambiguate names.

5.3.2 Experiment Results

In this section, we compare the experimental results of our SKB methods and those of baseline methods VSM and NER with the documents of people as described in the Section 5.2.2. Furthermore, we also investigate the robustness of our SKB methods over changes in directory structures and varying parameters. We applied six directory structures described in Section 5.2.1 to our SKB methods and investigated performance. We also varied the window size parameter n, and the number of representative directories parameter k to verify the robustness of SKB methods. We experimented with the document frequency ratio threshold in Equa. (3.9), $r = 1, 2, 5, 10$, the window size parameter, $n = 10, 20, 30, ..., 90$, and 100, and the number of representative directories parameter, $k = 10, 20$, and 30.

The overall performance for each method

Figs. 5.2, 5.3, 5.4, 5.5, 5.6, 5.7 show the precision–recall graphs for the SKB methods using different directory structures and their comparisons with the baseline methods. Tables 5.4 and 5.5 show the comparison in terms of the averaged precision value P_{aver} between the baseline methods VSM, NER and our proposed methods SKB1 and SKB2. In this experiment, we set the window size $n = 50$ and the number of representative directories $k = 20$. We set the frequency document ratio threshold for SKB2 $r = 5$. As can be seen from these Tables, our SKB1 and SKB2 methods together with six different directory sets outperform the baseline methods VSM and NER.

Performance of SKB2 when varying the document frequency ratio thresh-

Table 5.3: Numbers of documents of people

Number of relevant documents	Number of people	Number of relevant documents	Number of people
1	942	$31 \sim 40$	3
$2 \sim 5$	33	$41 \sim 50$	4
$5 \sim 10$	8	$51 \sim 60$	6
$11 \sim 20$	5	$61 \sim 70$	1
$21 \sim 30$	6		

old

We experimented with SKB2 using different threshold values for document frequency ratio threshold: $r = 1, 2, 5, 10$. The directory structures used in these experiments were Dmoz10, Google10, and Yahoo10. Table 5.6 shows the experimental results. As can be seen from this table, SKB2 achieves good performances, especially when $r = 5$ and 10. This result agrees with the characteristic that the stronger the relation to a topic that a term has, the larger the document frequency ratio it has.

Performance of SKB systems when varying the window size

Tables 5.7 and 5.8 show the performance variations with different window size parameters. In these experiments, we used the Google20 directory structure with the number of representative directories set to 10. As can be seen from the results in these two tables, the SKB1 and SKB2 methods achieve better performance when the window size increases. We also experimented with the VSM method with different window size parameters. As shown in Table 5.9, we noted that the performance values of the VSM method decreased slightly when we increased the window size.

From the performance value decrease of the VSM method, we learn that the further the text is from the personal names, the more noise it contains. On the other hand, from the increased performance values of the SKB methods, we found that the SKB methods can effectively filter out noisy text and select relevant text far from the personal names.

Performance of SKBs when varying the number of representative directories

Tables 5.10 and 5.11 show the different performances with different number of representative directories $k = 10, 20, 30$. In this experiment, we used the Google20 directory structure with the window size fixed at 50. We recognize from the results shown in these two Tables that SKB1 and SKB2 methods achieved improved performance when

Table 5.4: Comparison between VSM, NER and SKB1

Method	P_{aver}	Method	P_{aver}
VSM	58.5%	SKB1_Yahoo10	64.1%
NER	54.1%	SKB1_Yahoo20	62.2%
SKB1_Google10	**64.2**%	SKB1_Dmoz10	62.4%
SKB1_Google20	63.8%	SKB1_Dmoz20	60.8%

the numbers of representative directories changed from 30 to 20 and 10.

Performance for each method on real namesake document sets

Table 5.12 shows the performance comparisons between VSM, NER, and SKB2 in the experiments on real namesake document sets. We use the averaged precision values at 11 recall points for all possible re-ranking sequences in the comparisons. The results show that our SKB2 performs best in 18 sets, follows by the NER performs best in 5 sets, and the VSM performs best in 1 sets. On average, our SKB2 also outperforms the baseline methods VSM and NER.

5.3.3 Discussions

Disambiguation of people in web documents is challenging because web documents are published by resources of different kinds, and useful information is mixed with noise. To improve effectiveness when processing web documents, we propose a new method that uses web directories to aid the extraction of the documents' features and the measurement of documents' similarities. We use information from directories to improve the calculation of the vector space model. The key to our approach is that web directories provide information about the relationship between directories' documents themselves and the relationship between directories' documents and other documents; these relationships cannot be found in the conventional vector space model method. We have proposed two approaches to exploit information from web directories. First, by investigating the relationship between documents referring ambiguous personal names and the documents on the web directories, we can improve the measurement of term frequencies in a document. Compared with the vector space model method and the named entity recognition method, we have improved the averaged precisions

Table 5.5: Comparison between VSM, NER and SKB2

Method	P_{aver}	Method	P_{aver}
VSM	58.5%	SKB2_Yahoo10	64.5%
NER	54.1%	SKB2_Yahoo20	63.2%
SKB2_Google10	**66.1%**	SKB2_Dmoz10	63.4%
SKB2_Google20	65.5%	SKB2_Dmoz20	62.5%

from 3.9% to 9.7%, and from 12.4% to 18.7%, respectively. Furthermore, we can exploit the relationship between the documents in the same web directories. Using this relationship, we can differentiate topic terms from common terms, even if they have the same characteristic in that they appear frequently in some documents. This exploitation can be regarded as an attempt to measure topic frequencies of terms. Although, we cannot count topic frequencies precisely, we can use web directories to modify term frequencies in documents to approach topic frequencies. This second exploitation results in a further improvement of averaged research precision from 6.8% to 12.9%, and from 15.5% to 22.2% over the VSM method and over the NER method, respectively.

We investigated the robustness of our approaches over changes of directory structure and variation of system parameters. The experimental results with different directory structures and different system parameters support the conclusion that our SKB methods achieve stable performance.

From the practical point of view, our approach has advantages as well as limitations. The greatest advantage is that it requires little preparation because the existing web directories can be used directly with virtually no preprocessing. In addition, the broad coverage of web directory topics enables the use of our approach with a broad range of people. On the other hand, the most significant limitation of our approach is its increasing cost of computation. The increase is proportional to the number of directories used.

Table 5.6: Average precisions of SKB2 with different threshold values of document frequency ratio

Directory	SKB1	SKB2, $r = 1$	SKB2, $r = 2$	SKB2, $r = 5$	SKB2, $r = 10$
Dmoz10	62.4%	62.5%	62.4%	63.4%	**63.6%**
Google10	64.2%	65.3%	65.4%	**66.1%**	66.0%
Yahoo10	64.1%	64.7%	**64.8%**	64.5%	64.4%

Table 5.7: Performance of SKB1 method with different window sizes

Window size	P_{aver}	Window size	P_{aver}
10	62.6%	60	65.4%
20	64.8%	70	65.3%
30	65.1%	80	**65.5%**
40	64.9%	90	65.4%
50	65.1%	100	65.4%

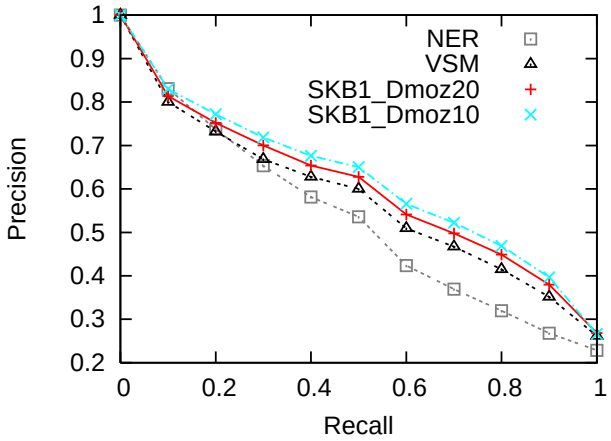


Figure 5.2: Performance of SKB1 with Dmoz directories

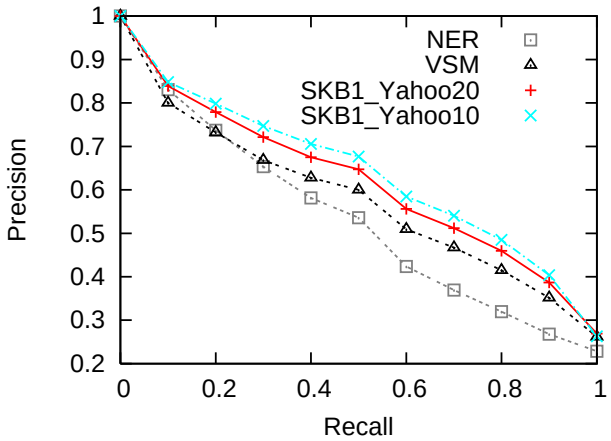


Figure 5.3: Performance of SKB1 with Yahoo directories

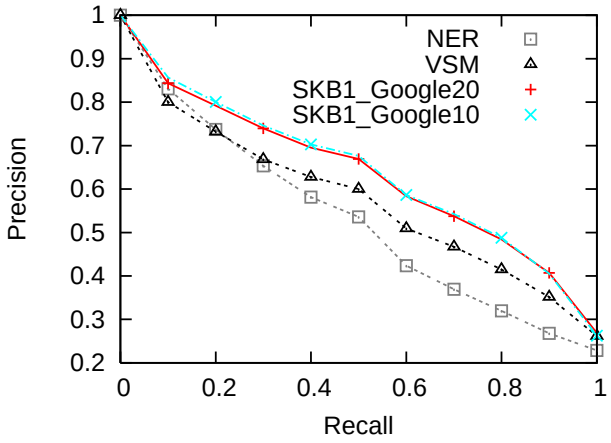


Figure 5.4: Performance of SKB1 with Google directories

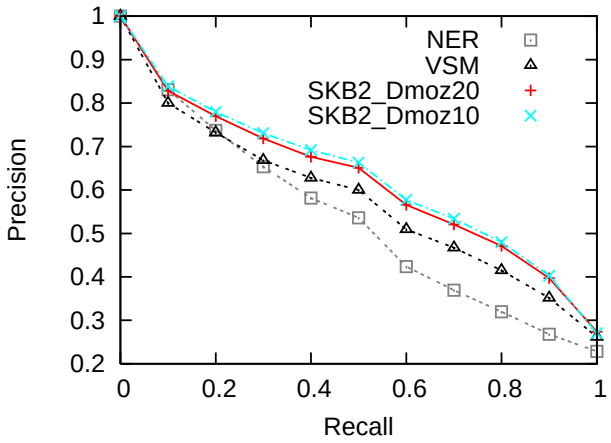


Figure 5.5: Performance of SKB2 with Dmoz directories

Table 5.8: Performance of SKB2 method with different window sizes

Window size	P_{aver}	Window size	P_{aver}
10	63.4%	60	66.65%
20	66.1%	70	66.69%
30	66.3%	80	**66.75%**
40	66.4%	90	66.73%
50	66.4%	100	66.68%

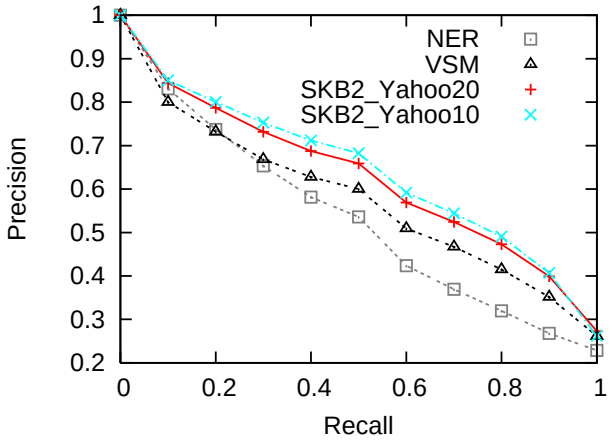


Figure 5.6: Performance of SKB2 with Yahoo directories

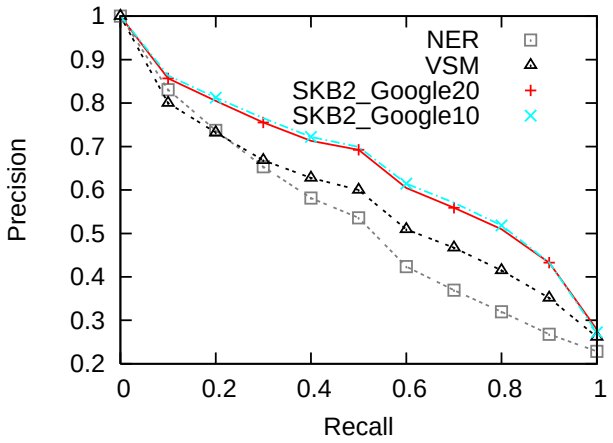


Figure 5.7: Performance of SKB2 with Google directories

Table 5.9: Performance of VSM method with different window sizes

Window size	P_{aver}	Window size	P_{aver}
10	**59.1%**	60	58.4%
20	58.6%	70	58.3%
30	58.6%	80	58.4%
40	58.6%	90	58.4%
50	58.5%	100	58.3%

Table 5.10: Performance of SKB1 with different number of representative directories

Number of representative directories	P_{aver}
10	**65.1**%
20	63.8%
30	62.0%

Table 5.11: Performance of SKB2 with different number of representative directories

Number of representative directories	P_{aver}
10	**66.4%**
20	65.5%
30	63.5%

Table 5.12: Performance for each method on real namesake document sets

Name query	NER	VSM	SKB2
Adachi Jun	59.0%	59.5%	**64.2%**
Sakai Shuichi	73.8%	77.1%	**79.7%**
Tom M. Mitchell	71.9%	79.9%	**81.5%**
Tanaka Katsumi	**79.5%**	68.6%	72.4%
John D. Lafferty	76.9%	81.4%	**89.7%**
Andrew McCallum	83.5%	84.8%	**88.5%**
Paul G. Hewitt	64.2%	69.5%	**72.2%**
Edwin F. Taylor	74.6%	74.1%	**85.6%**
Paul W. Zitzewits	**85.7%**	84.2%	83.8%
Frank Bridge	**55.8%**	50.9%	55.1%
Kenneth W. Ford	52.6%	51.9%	**72.0%**
Michael A. Dubson	**73.1%**	70.1%	72.5%
Scott Hammer	76.2%	68.6%	**82.0%**
Thomas F. Patterson	57.1%	65.0%	**81.5%**
Michele L. Pearson	53.2%	57.0%	**64.4%**
Henry F. Chambers	54.2%	56.6%	**64.2%**
David C. Hooper	44.3%	52.5%	**60.0%**
Lindsay E. Nicolle	83.1%	85.7%	**88.8%**
John M. Roberts	76.9%	74.0%	**81.1%**
David Reynolds	69.9%	69.4%	**72.8%**
Thomas E. Woods	**91.4%**	90.7%	84.5%
Thomas A. Brady	63.4%	57.1%	**63.8%**
William L. Cleveland	60.4%	59.0%	**80.8%**
Peter Haugen	50.2%	**59.7%**	59.4%
Average	67.6%	68.6%	**75.1%**

5.4 Experiments on Extraction of Topic from Web Directories

We applied the LDA method and the Gibbs sampling algorithm for web directories to extract topics. We removed stop words and ignored words that appeared in less than 10 documents to get a vocabulary of roughly 10000 words. The parameters were selected as follows: $\alpha = \frac{50}{T}, \beta = \frac{200}{W}$, and $k = 1, 2, 5, 10, 20, 50, 100, 200$.

5.4.1 Results of Document Similarities Measurement Using Extracted Topics

We carried experiments using the 216 test sets described in Section 5.2.2. The six directory structures as described in Section 5.2.1 were used independently to modify documents. The parameters used in experiments were the window size parameter $n = 10, 20, ..., 100$, the bias factor $k = 1, 2, 5, 10, 20, 50, 100, 200$, and the number of representative topics $m = 10, 20, 30$.

5.4.2 The overall performance for each method

Figs. 5.9, 5.10, 5.8 show the precision-recall results for our method using different directory structures and the comparisons with the baseline methods. Table 5.13 shows the comparison among methods in terms of the averaged precision values. In these experiments, we set $n = 50$, $k = 10$, and $m = 10$. As we can see from these results, our approach outperforms the baseline methods VSM and NER.

5.4.3 Performances of our approach when varying parameters

Variation of the bias factor

Table 5.13: Comparison between VSM, NER and SKB1

Method	P_{aver}	Method	P_{aver}
VSM	58.5%	SKB_Yahoo10	61.2%
NER	54.1%	SKB_Yahoo20	61.3%
SKB_Google10	63.4%	SKB_Dmoz10	64.2%
SKB_Google20	64.4%	SKB_Dmoz20	**64.6%**

We varied the bias factor k used in the step of extraction latent topics described in Section 4.3. The values of bias factor used for six directory structures were $k = 1, 2, 5, 10, 20, 50, 100, 200$. When $k = 1$, the topics are symmetric to all directories. When $k > 1$, the topics are asymmetric and each directory is biased to its own preference topic. The performances with different bias factors in terms of averaged precision values are summarized in Fig. 5.11 and Tables 5.14, 5.15.

Variation of the window size

We varied the window size $n = 10, 20, ..., 100$ for our approach and the VSM approach. In these experiments, we used the directory Dmoz20 with the bias factor $k = 10$, and the number of representative directory $m = 10$. The results in terms of averaged precision values are shown in Tables 5.16 and 5.17.

Variation of the number of representative topics

We varied the number of representative topics $m = 10, 20, 30$ for our approach. The results in terms of averaged precision values are shown in Table 5.18.

5.4.4 Discussions

There are some difficulties when working with web documents comparing with other kinds of documents like scientific publications and news articles. First, since the WWW is a heterogeneous environment, where documents come from different kinds of publishers, documents on the web are informal and noisy. Second, unlike scientific publications and news articles, where the entire of document is relevant to concerned person, in web documents, only some parts of a document surrounding the personal name tend to

Table 5.14: Performance of SKB with different bias factors

Bias	Google10	Yahoo10	Dmoz10
1	63.6%	61.2%	64.9%
2	64.3%	60.6%	64.4%
5	63.7%	61.3%	64.4%
10	63.4%	61.3%	64.2%
20	63.8%	61.8%	64.9%
50	64.2%	61.4%	65.5%
100	64.5%	61.9%	65.1%
200	63.9%	61.9%	64.5%

contain information relevant to the person in concerned. We have proposed a new method that uses web directories as additional information resources to recognize important terms in name ambiguous web documents and to evaluate these terms with more appropriate weights. The key point in our approach is that web directories are used as a collection of contexts. Web directories have abundant of texts so important terms will appear there more frequently than in name ambiguous documents. Therefore, we use web directories to find overlapping contexts between them and other web documents and then, to modify web documents so that important terms will have more weights. Contexts in web directories are mixed together, so we use the LDA method to preprocess web directories and get a set of topics and distributions of terms for topics. These topics are then used to amplify contexts in web documents so that the extraction of contexts and the calculations of documents' common contexts is easier. Experiment results show that our approach achieves better performances than the conventional approaches. It improves the performance from 3.6% to 12.8%, and from 12.0% to 22.0%, comparing to the VSM approach and the NER approach, respectively. We have also tried different parameter values for our approach and we have been able to verify the stableness and the robustness of our approach.

The use of web directories has advantages as well as disadvantages. In terms of preparation costs, it is advantageous since there are well-known and profusion web directories already existed on the WWW. In addition, web directories cover a broad range of topics, so our approach is able to disambiguate people in different scenarios. On the other hand, our approach also has disadvantages because it requires more com-

Table 5.15: Performance of SKB with different bias factors

Bias	Google20	Yahoo20	Dmoz20
1	61.2%	61.3%	64.1%
2	64.2%	61.1%	64.4%
5	62.8%	61.1%	64.5%
10	64.4%	61.2%	64.6%
20	64.2%	61.3%	65.7%
50	64.4%	61.6%	66.3%
100	65.1%	61.6%	66.0%
200	65.5%	60.5%	66.2%

putation cost when working with directories. The increasing of cost is proportional to the number of directories used.

Table 5.16: Performance of SKB with different window sizes

Window size	Averaged precision	Window size	Averaged precision
10	65.2%	60	64.5%
20	**65.5%**	70	64.2%
30	65.2%	80	64.0%
40	65.0%	90	63.8%
50	64.6%	100	63.6%

Table 5.17: Performance of VSM with different window sizes

Window size	Averaged precision	Window size	Averaged precision
10	**59.1%**	60	58.4%
20	58.6%	70	58.3%
30	58.6%	80	58.4%
40	58.6%	90	58.4%
50	58.5%	100	58.3%

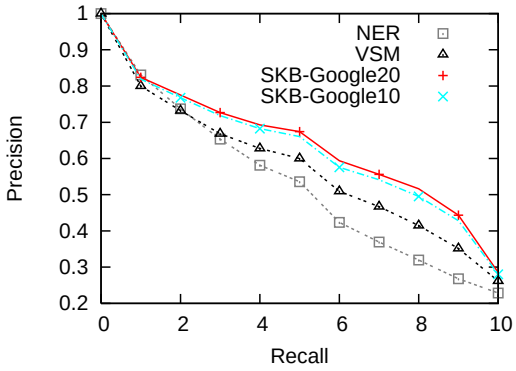


Figure 5.8: Performance of SKB with Google directories

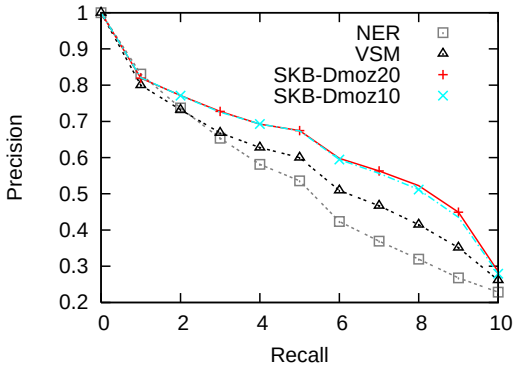


Figure 5.9: Performance of SKB with Dmoz directories

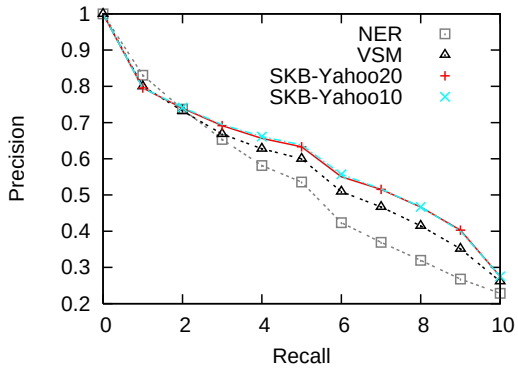


Figure 5.10: Performance of SKB with Yahoo directories

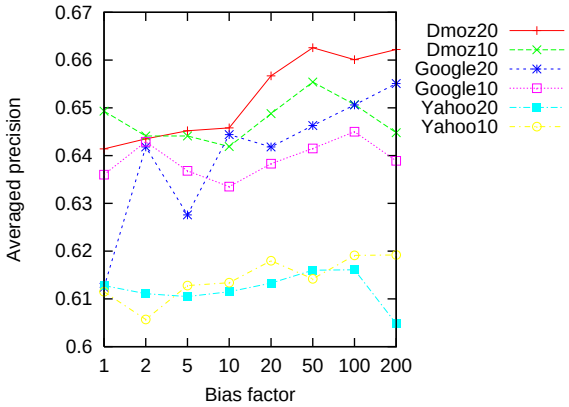


Figure 5.11: Performance of SKB with different bias factors

Table 5.18: Performance of SKB with different number of representative topics

Number of representative topics	Averaged precision
10	**64.6%**
20	64.1%
30	63.9%

Chapter 6

Name Disambiguation Demo System

In this Chapter, we present our name disambiguation demo system. The purpose of this demo system is two-fold. First, it shows how our approach disambiguates ambiguous names. Second, it helps us to understand the internal executions of our approach so that we can understand the details about how our approach utilize information from knowledge base. By understanding the details of our approach, we can hope to improve it further.

6.1 Overview

The name disambiguation approach that we proposed is assumed to cooperate with a search engine system. The name disambiguation function is developed as an embedded module to the search engine. The demo system that we introduce in this Chapter is to demonstrate how such a name disambiguation module can cooperate with the search engine. In our demo system, we assume that the input for the module is search results for a personal name query. The system interacts with end users to get a feedback document selected by users. This notifies the system the person that users are interested in. The system then calculates similarities between documents and re-ranks documents in the order of similarities to the feedback document.

Fig. 6.1 shows the overview of our demo system. The system' operation in steps is

as follows.

1. **Step 1**
   Queries from users are sent to the Google search engine to get some documents from the top of the result. In our demo system, we get 100 documents from the top.

2. **Step 2**
   The result set is first shown to end users where documents are in the order as produced by the search engine.

3. **Step 3**
   Next, users select from the result set a document that mentions to the person of their interests.

4. **Step 4**
   The selected document is sent back to the system. The system then calculates document similarities and outputs the final results which are re-ranked so that documents with more similarities to the selected document go to the top.

In order to understand the difference between our approach and other approaches, in the document similarity calculation step, users can choose which approach is used to calculate similarities. The system can show the differences between approaches in terms of performances. They can also show the details of similarity calculations for each approach so that we can understand and compare the specific features of each approach.

The system has two running modes: a online running mode and a offline running mode. In the online running mode, interactions with search engines and document similarity calculations are in real time. The retrieved documents and similarities results are recorded in hard disks for the offline mode. In the offline mode, recorded results in online mode are replayed to users. The online mode is to demonstrate the execution time to users while the offline mode is to save time for users when they are care about the features of the system.

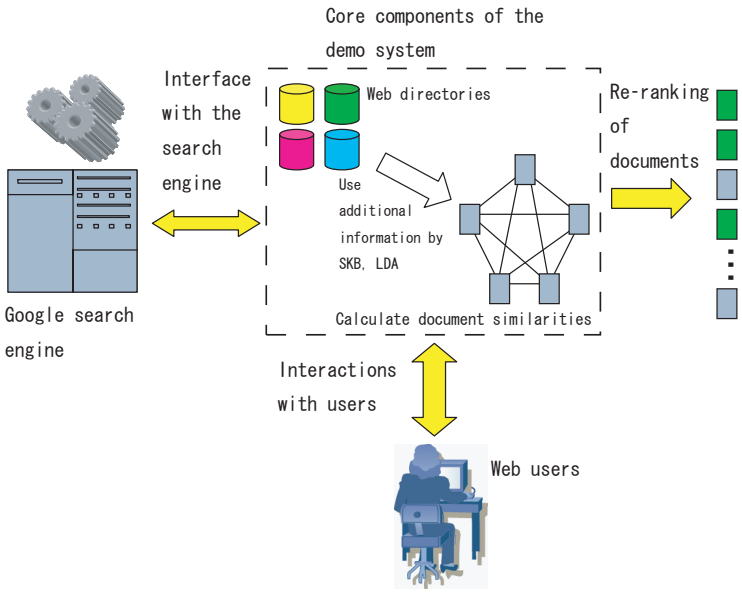


Figure 6.1: Overview of our name disambiguation demo system

6.2 Components of the System

Our demo system works as a server for end-users. The daemon server is built using the Ruby on Rails framework [3, 13, 102]. The Ruby on Rails framework is a new web application development framework that uses the Ruby programming language[103]. Its main advantage is that modules to interact with users are generated automatically. For example, the code to get value data from users' forms, the code to produce output pages to users are generated by the Ruby on Rails framework. By using the Ruby on Rails framework, we can focus on core components of web applications which manipulate data.

The main components of the demo system and the details for all components are explained in sequence in this Section.

1. **Interface to search engines**
   In this module, we get name queries from users and send queries to the Google

search engine to get result pages. We do query transactions with the Google using the normal html get method. Results from the search engine are parsed to extract the url addresses of result pages. After that, we use addresses of web pages to retrieve the results. For each name query, we collect top 100 pages in the result set.

2. **Offline data preprocessing**
   In this module, we collect web directories from three web directories: the Dmoz directory[1], the Google directory[2], and the Yahoo directory[3]. We crawl web directories starting from the root node until we reach all child nodes at level two. Then, directories with the number of documents less than ten are removed because their topics may be weak. Selected directories are then preprocessed by our algorithms. We stem words to root forms, remove stop words, calculate statistical information, and apply the latent Dirichlet allocation method to extract topics in web directories.

3. **Online data preprocessing**
   In this module, we process result pages of online name queries. We stem words to root forms and remove stop words. We also extract names of entities in documents using the LingPipe software[4] for the baseline method that disambiguate personal names by names of entities being relevant to people.

4. **Document similarity calculation**
   In this module, we calculate similarities of all document pairs using our approaches and baseline approaches. We also record information about common terms, phrases in document pairs, so that we can understand the details of calculation in each approach.

5. **Document re-ranking**
   In this module, we interact with users to get their feedbacks and re-rank result documents according to the feedbacks. We have some data sets being used in our

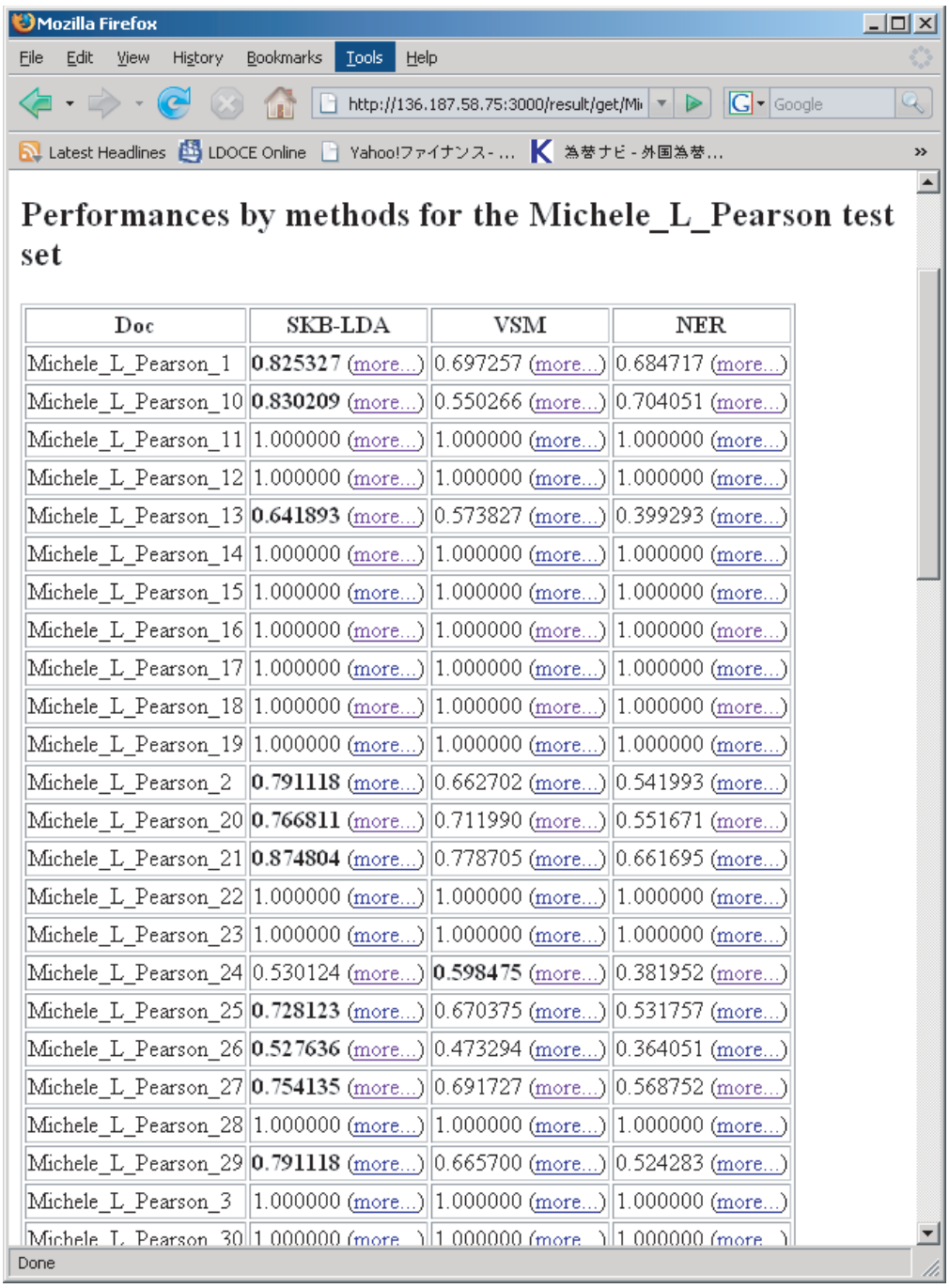
[1] http://www.dmoz.org
[2] http://directory.google.com
[3] http://dir.yahoo.com
[4] http://www.alias-i.com/lingpipe/

research experiments with their truth results prepared. For these data sets, we calculate the disambiguation performance for each method and show the comparisons of results to users.

Fig. 6.2, Fig. 6.3, and Fig. 6.4 are the screenshots of our demo system. They show the overall performances of data sets by methods, performances of each data set by methods, and an example of result re-ranking.

Dir name	SKB-LDA	VSM	NER
Adachi_Jun	**0.814**	0.785	0.778
Andrew_McCallum	**0.936**	0.902	0.892
David_C_Hooper	**0.902**	0.883	0.862
David_Reynolds_History	**0.862**	0.807	0.817
Edwin_F_Taylor	**0.931**	0.842	0.848
Frank_Bridges	0.867	0.886	**0.889**
Henry_F_Chambers	**0.904**	0.892	0.883
J_M_Roberts	**0.915**	0.848	0.869
John_D_Lafferty	**0.941**	0.913	0.891
Kenneth_W_Ford	**0.918**	0.853	0.856
Lindsay_E_Nicolle	**0.951**	0.924	0.891
Michael_A_Dubson	**0.874**	0.784	0.807
Michele_L_Pearson	**0.915**	0.893	0.869
Paul_Hewitt	**0.88**	0.839	0.806
Paul_W_Zitzewitz	**0.911**	0.884	0.891
Peter_Haugen	0.821	**0.839**	0.792
Sakai_Shuichi	**0.887**	0.858	0.835
Scott_Hammer	**0.914**	0.856	0.886
Tanaka_Katsumi	**0.81**	0.783	0.802
Thomas_A_Brady	0.81	0.802	**0.842**
Thomas_E_Woods	0.912	0.929	**0.931**
Thomas_F_Patterson	**0.925**	0.911	0.895
Tom_M_Mitchell	**0.944**	0.917	0.878
William_L_Cleveland	**0.922**	0.904	0.908

Figure 6.2: A screen shot showing performances by data sets and methods



Performances by methods for the Michele_L_Pearson test set

Doc	SKB-LDA	VSM	NER
Michele_L_Pearson_1	**0.825327** (more...)	0.697257 (more...)	0.684717 (more...)
Michele_L_Pearson_10	**0.830209** (more...)	0.550266 (more...)	0.704051 (more...)
Michele_L_Pearson_11	1.000000 (more...)	1.000000 (more...)	1.000000 (more...)
Michele_L_Pearson_12	1.000000 (more...)	1.000000 (more...)	1.000000 (more...)
Michele_L_Pearson_13	**0.641893** (more...)	0.573827 (more...)	0.399293 (more...)
Michele_L_Pearson_14	1.000000 (more...)	1.000000 (more...)	1.000000 (more...)
Michele_L_Pearson_15	1.000000 (more...)	1.000000 (more...)	1.000000 (more...)
Michele_L_Pearson_16	1.000000 (more...)	1.000000 (more...)	1.000000 (more...)
Michele_L_Pearson_17	1.000000 (more...)	1.000000 (more...)	1.000000 (more...)
Michele_L_Pearson_18	1.000000 (more...)	1.000000 (more...)	1.000000 (more...)
Michele_L_Pearson_19	1.000000 (more...)	1.000000 (more...)	1.000000 (more...)
Michele_L_Pearson_2	**0.791118** (more...)	0.662702 (more...)	0.541993 (more...)
Michele_L_Pearson_20	**0.766811** (more...)	0.711990 (more...)	0.551671 (more...)
Michele_L_Pearson_21	**0.874804** (more...)	0.778705 (more...)	0.661695 (more...)
Michele_L_Pearson_22	1.000000 (more...)	1.000000 (more...)	1.000000 (more...)
Michele_L_Pearson_23	1.000000 (more...)	1.000000 (more...)	1.000000 (more...)
Michele_L_Pearson_24	0.530124 (more...)	**0.598475** (more...)	0.381952 (more...)
Michele_L_Pearson_25	**0.728123** (more...)	0.670375 (more...)	0.531757 (more...)
Michele_L_Pearson_26	**0.527636** (more...)	0.473294 (more...)	0.364051 (more...)
Michele_L_Pearson_27	**0.754135** (more...)	0.691727 (more...)	0.568752 (more...)
Michele_L_Pearson_28	1.000000 (more...)	1.000000 (more...)	1.000000 (more...)
Michele_L_Pearson_29	**0.791118** (more...)	0.665700 (more...)	0.524283 (more...)
Michele_L_Pearson_3	1.000000 (more...)	1.000000 (more...)	1.000000 (more...)
Michele_L_Pearson_30	1.000000 (more...)	1.000000 (more...)	1.000000 (more...)

Figure 6.3: A screen shot showing performances of one data set by methods

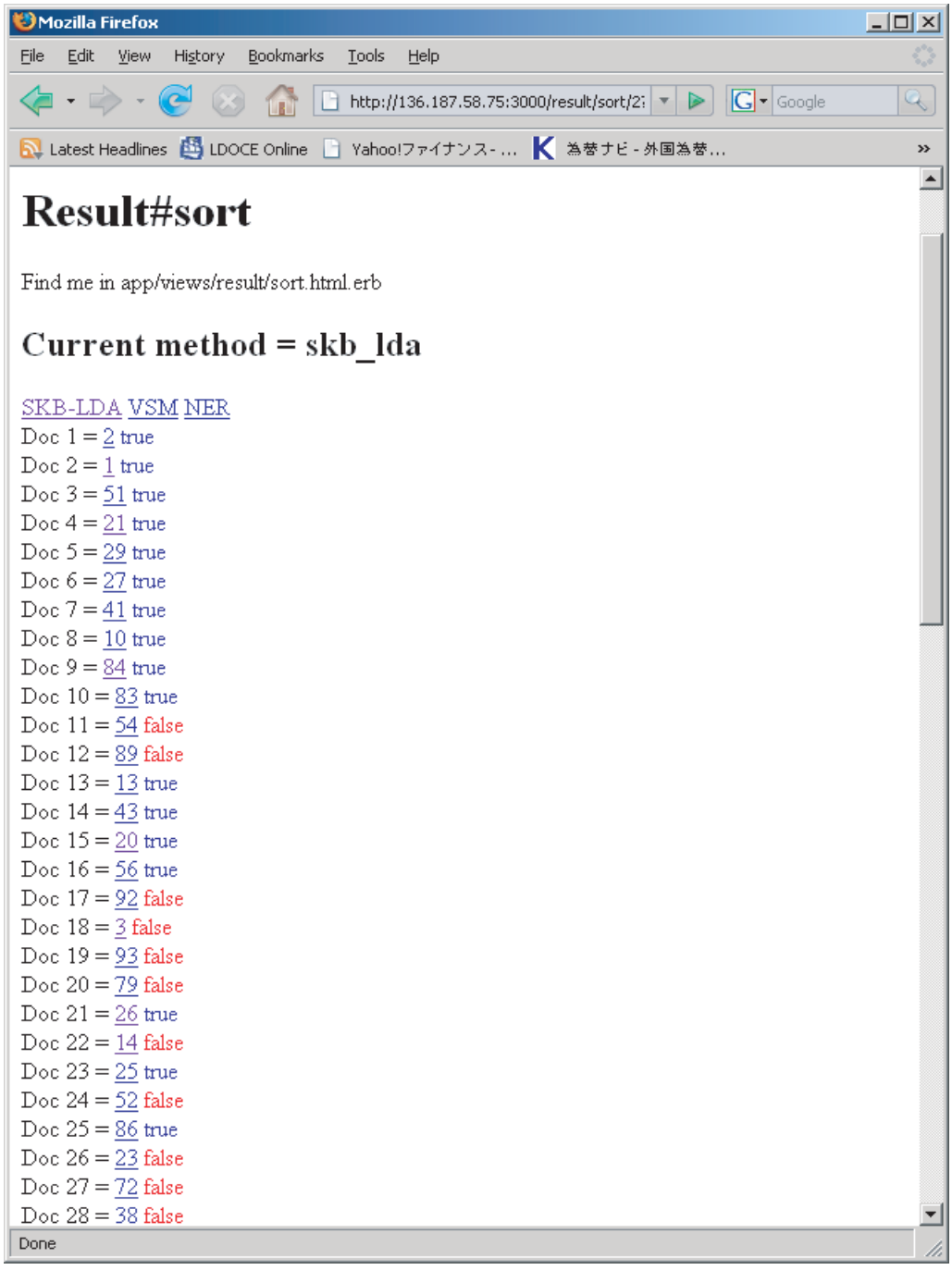


Figure 6.4: A screen shot showing a re-ranking performance

Chapter 7

Conclusions

In this Chapter, we give the general conclusions of our research. We begin with the review of name disambiguation problems. We review the problem definition, how the name disambiguation is important in the WWW. Next, we summarize our approach to the name disambiguation problem. We discuss how our approach conquers the name disambiguation, the main key points that make the originalities in our approach. After that, we discuss other kind of disambiguation problems that may have some relationships with the name disambiguation problem. We consider the ability to extend our approach to these general sense disambiguation problems.

Besides disambiguation applications, many other applications that mine data from the WWW are useful for users. The idea of using external knowledge in our approach can be extended to this range of web mining applications. First, we discuss how our idea is different from other approaches that also use external information to improve the performance. Then, we mention the advantages and disadvantages when using external information. Finally, we present the basic framework to extend our idea to other problems in web mining.

7.1 Summary of Our Research

In our research, we focus on the need for information about people in the web. A basic characteristic of data in the web is that data is created by many individuals and they are published in different places. Therefore, it is required to collect data

together so that users can access to the useful data easily. However, data collected automatically by computer programs may contain useless information. For example, the namesake problem that a name is shared by several people may cause noise to information. Because of the namesake problem, when we use personal names to search for information of people, the result sets often contain documents relevant to several people. Our research target is to differentiate the result sets so that we can separate the person that users are looking for from other people.

The main idea in our name disambiguation method is to use web directories as a kind of additional information to the documents that we want to disambiguate names. Basically, by using additional information we can hope to improve the disambiguation performance. In previous approaches, several methods have also proposed to use additional information. For example, training data and extraction rules in some methods can be regarded as additional information from outside. Additional information in our approach has some features that make our approach a big difference from previous approaches. In previous approaches, additional information in the training phase and documents in the test phase are often from the same resource. They have similar characteristics and the system learns these characteristics of data in the training phase. However, in our research, since we can not prepare training data with close characteristics to real data, we can only use a kind of neutral information for training data.

We have proposed two methods to utilize web directories. In the first method, we use web directories directly to modify documents. In web directories, since the amount of text is abundant, the frequencies of terms related to the directory topic is large. This makes the feature vectors of web directories to have more weights for terms related to the topic than other terms. We use directory feature vectors to modify document feature vectors so that the system can recognize the topics contained in documents more easily. In the second method, web directories are preprocessed before being used in document modifications. We apply the latent Dirichlet allocation method[16, 45] to extract latent topics contained in web directories. Using this method, we are able to extract probabilities of terms in topics. Then, the extracted probabilities are used to modify documents to amplify weights for terms that are strongly related to the document topic. After the step of modification of documents using one of the two

approaches above, for each original document, we get a set of modified documents associated with different directories and topics. These sets of modified documents are combined together to calculate the similarities of document pairs.

In our research, we use the re-ranking method to help users disambiguate ambiguous names. This way of using re-ranking method is different from some previous researches that use the clustering method to disambiguate names. We choose to use the re-ranking method because the personal name disambiguation in web search has a different target from the general word sense disambiguation problems. In the word sense disambiguation, the target is to understand all senses of an ambiguous word. Therefore, the clustering method is suitable to this goal because we try to group senses with the same meaning together and each sense is associated with a cluster. However, the target is different in the problem of personal name disambiguation. In this problem, the users do not need to understand all the people in the results but only the person that they are looking for. The clustering method may solve a problem harder than the required problem. Since clustering algorithms require parameters like the number of output clusters or the similarity threshold, it is difficult to tune these parameters so that the algorithm will run well for an arbitrary name. Further more, if we cluster names into clusters and the clustering results have some mistakes, documents relevant to one person may go to several clusters. As a result, users have to check several clusters in order to retrieve all the useful documents. Because of these reasons, the method of re-ranking documents is more suitable to goal in name disambiguation tasks. The re-ranking algorithm is also simpler than the clustering algorithm and it does not require the tuning of parameters. In the re-ranking procedure, we first need a feedback from users. First, users select a document that matches the person of interest and notify the system. The system then re-ranks documents in the order of similarities to the selected document.

We conducted experiments to verify the effectiveness of our approach. We used real data from the web in our experiments. For web directories, we used three well-known web directories: the Dmoz directory[1], the Google directories[2], and the Yahoo directories[3]. For documents of ambiguous names, we got from the Google search engine[4] some

[1] http://www.dmoz.org
[2] http://directory.google.com
[3] http://dir.yahoo.com
[4] http://www.google.com

results for some personal name queries. We compared our approach with some previous approaches: the approach based on the vector space model method and the approach based on the named entity recognition method. The experiment results showed that our approach had advantage over baseline approaches. We also developed a name disambiguation demo system. The demo system demonstrated the way to cooperate our approach with a search engine to disambiguate personal names. It also showed the details of internal operations of our approach. This helped us to analyze the different features in our approach comparing to previous approaches and to understand the advantages and disadvantages of our approach.

7.2 Using Knowledge Base for Other Applications

The human always combine several information resources when solving problems. For example, when we study computer science, we use our knowledge in mathematics, statistics, physics, etc to understand computer science lectures. Our name disambiguation approach is an imitation of the technique by human. We use additional information beside the information provided in the applications. Basically, our method of using additional data can be applied to applications that could exploit additional data. For example, additional set of documents can be used in the feedback process in information retrieval, in the clustering algorithms, and in many other data mining applications.

When using additional information, it is worth to take into consideration that the approach should not require the additional data with high preparation cost. This is because the documents in the web themselves form a huge amount of data, so if the additional information has a larger order of amount than the data in the problem itself, the effort to prepare additional information will be much more difficult than the problem being solved. The quality of additional information is also important. From the experience of our research, it is more the quality of additional information than its quantity that causes the improvement in performances. Because of this reason, it is worth to pay human cost to refine a reasonable amount of additional information rather than to pay exhaustive computation cost to collect a huge amount of additional information.

7.3 Conclusions

In the era of information explosion like the present internet and WWW era, it is the people who are able to manage, to control and to utilize the information will acquire advantages in daily competitions such as political activities, business competitions, academic researches and studies, etc. In order to manage information well, computers are useful tools which can process information automatically and quickly. Web mining applications appear as important tools to utilize useful information. These applications crawl web documents automatically and find out useful information quickly. They play as gateways for people who want to use the information in the WWW. Our research target is to provide web citizens such a useful tool. We help users to acquire information about people on the web more easily. When users get a document collection that refers to several people, our approach can differentiate the person of interest in an semi-automatic process. Extensions of our problem are disambiguation for names of organizations, names of places, and names of products. These problems are in the class of homonym problems where the same name is used to refer different entities. Another class of problems that is close to these homonym problems is the class of synonym problems. In synonym problems, an entity is referred by different names. Both the homonym problems and the synonym problems need to be solved when we want to identify entities in the web.

Appendix A

Expectation Maximization Algorithm

In this appendix, we introduce the general expectation maximization (EM) algorithm and its application to the unsupervised approach of the word sense disambiguation problem. The more details about the EM algorithm can be found in [12, 32, 75, 79, 89, 90, 101]

The general expectation maximization

Suppose we have some samples of data, each sample produces two output values x_i, y_i. However, we can only observe the values x_i and we have to guess the hidden value y_i. Of course, without any more information, we are unable to calculate $P(y_i|x_i)$ and guess the most possible value for y_i. In fact, there are applications that we are able to model the probability $P(X, Y|\theta)$, where θ is a set of parameters for the model. Our task is to find the optimal θ so that the generation of X is the most possible. That is we have to solve the following problem.

$$\theta = \arg\max_{\theta} P(X|\theta) \tag{A.1}$$

$$P(X|\theta) = \prod_{x_i} P(x_i|\theta)$$

$$= \prod_{x_i} \sum_{y_j} P(x_i, y_j|\theta) \tag{A.2}$$

$$\log P(X|\theta) = \log \prod_{x_i} \sum_{y_j} P(x_i, y_j|\theta)$$

$$= \sum_{x_i} \log \sum_{y_j} P(x_i, y_j|\theta)$$

$$= \sum_{x_i} \log \sum_{y_j} \frac{P(x_i, y_j|\theta)}{q(y_j)} q(y_j) \tag{A.3}$$

where $q(y_j)$ is a distribution of y_j, that is $\sum_j q(y_j) = 1$. Since $\log x$ is a convex function, we have the following inequality.

$$\log\left(\alpha x + (1-\alpha)\, y\right)) \geq \alpha \log x + (1-\alpha) \log y \tag{A.4}$$

$$\log\left(\sum_i \alpha_i x_i\right) \geq \sum_i \alpha_i \log x_i \text{ , where } \sum_i \alpha_i = 1 \tag{A.5}$$

Apply this inequality into (A.3), we get.

$$\log P(X|\theta) = \sum_{x_i}\sum_{y_j} q(y_j)\log\frac{P(x_i,y_j|\theta)}{q(y_j)} + \Delta,\ \Delta \geq 0 \tag{A.6}$$

$$\mathcal{L}(q,\theta) = \sum_{x_i}\sum_{y_j} q(y_j)\log\frac{P(x_i,y_j|\theta)}{q(y_j)} \tag{A.7}$$

$$\begin{aligned}\Delta &= \sum_{x_i}\sum_{y_j} q(y_j)\left(\log\frac{P(x_i,y_j|\theta)}{q(y_j)} - \log\sum_{y_j} P(x_i,y_j|\theta)\right)\\ &= \sum_{x_i}\sum_{y_j} q(y_j)\left(\log\frac{P(x_i,y_j|\theta)}{q(y_j)} - \log P(x_i|\theta)\right)\\ &= \sum_{x_i}\sum_{y_j} q(y_j)\left(\log\frac{P(y_j|x_i,\theta)}{q(y_j)}\right)\end{aligned} \tag{A.8}$$

Equa. (A.6) is an important part for the EM algorithm. It is used to increase the value of $\log(X|\theta)$. The increasing effect is by an iterative algorithm that consists of two steps in each iteration as follows.

1. Step 1: **Expectation step**
   In this step, we try to keep the value of $logP(X|\theta)$ and to maximize the value of $\mathcal{L}(q,\theta)$ in Equa. (A.6). As this Equa. (A.6) guides us, the maximization achieves when $\Delta = 0$, that is when the following condition is satisfied.

$$\log\frac{P(y_j|x_i,\theta)}{q(y_j)} = 0 \tag{A.9}$$

$$\text{which mean that } q(y_j) = P(y_j|x_i,\theta) \tag{A.10}$$

   .

2. Step 2: **Maximization step**
   In this step, we try to fix values of q and vary θ to maximize $\mathcal{L}(q,\theta)$. This step requires to find the solution for θ using analytical mathematics. The Lagrange method is often used to solve this optimization problem [51, 54, 105].

Unsupervised approach for word sense disambiguation

In 2.1.3, we introduced the a previous approach that uses expectation maximization algorithm for the unsupervised approach in word sense disambiguation. Here, we derive the details of this algorithm using the general EM algorithm we presented in the above. The calculation of $P(v_j|s_k)$ by 2.10 in the expectation step is straightforward from the general EM algorithm. The calculation of Equas. (2.11), (2.11), and (2.11) in the maximization step can be derived using the Lagrange method as follows.

In the maximization step, our objective function is as follows.

$$L(C|\mu) = \sum_{i=1}^{I} \log \sum_{k=1}^{K} P(c_i|s_k)P(s_k) \tag{A.11}$$

$$P(c_i|s_k) = \prod_{v_j} P(v_j|s_k)^{\text{count}(v_j \text{ in } c_i)} \tag{A.12}$$

$$\begin{aligned} L(C|\mu) &= \sum_{i=1}^{I} \log \sum_{k=1}^{K} \prod_{v_j} P(v_j|s_k)^{\text{count}(v_j \text{ in } c_i)} P(s_k) \\ &= \sum_{i=1}^{I} \log \sum_{k=1}^{K} \frac{P(s_k|c_i)}{P(s_k|c_i)} \prod_{v_j} P(v_j|s_k)^{\text{count}(v_j \text{ in } c_i)} P(s_k) \\ &\geq \sum_{i=1}^{I} \sum_{k=1}^{K} P(s_k|c_i) \log \left(\frac{P(s_k)}{P(s_k|c_i)} \prod_{v_j} P(v_j|s_k)^{\text{count}(v_j \text{ in } c_i)} \right) \\ &= \mathcal{L}(\theta) \end{aligned} \tag{A.13}$$

The set of parameters are $P(s_k)(k = 1, 2, ..., K)$ and $P(v_j|s_k)(j = 1, 2, ..., J; k = 1, 2, ..., K)$. They satisfy the following probabilistic properties.

$$\sum_{k} P(s_k) = 1 \tag{A.14}$$

$$\sum_{j} P(v_j|s_k) = 1 \text{ , where } k = 1, 2, ..., K \tag{A.15}$$

In order to find parameters that satisfy Equas. A.14, A.15 and maximize Equa. (A.13), we can use the Lagrange method as follows.

$$\frac{\partial \mathcal{L}}{\partial P(v_j|s_k)} + \lambda_k = 0 \text{ , where } j = 1, 2, ..., J; k = 1, 2, ..., K \tag{A.16}$$

$$\frac{\partial \mathcal{L}}{\partial P(s_k)} + \mu = 0 \text{ , where } k = 1, 2, ..., K \tag{A.17}$$

Equa. (A.16) becomes

$$\sum_i \frac{P(s_k|c_i) \cdot \text{count}(v_j \text{ in } c_i)}{P(v_j|s_k)} + \lambda_k = 0 \tag{A.18}$$

$$\sum_i P(s_k|c_i) \cdot \text{count}(v_j \text{ in } c_i) + \lambda_k \cdot P(v_j|s_k) = 0$$

By summarizing over j, we have

$$\sum_{i,j} P(s_k|c_i) \cdot \text{count}(v_j \text{ in } c_i) + \lambda_k \cdot \sum_j P(v_j|s_k) = 0$$

$$\sum_{i,j} P(s_k|c_i) \cdot \text{count}(v_j \text{ in } c_i) + \lambda_k = 0$$

$$\sum_{i,j} P(s_k|c_i) \cdot \text{count}(v_j \text{ in } c_i) = -\lambda_k \tag{A.19}$$

Substituting λ_k into A.18 we get:

$$\begin{aligned} P(v_j|s_k) &= -\frac{\sum_i P(s_k|c_i) \cdot \text{count}(v_j \text{ in } c_i)}{\lambda_k} \\ &= \frac{\sum_i P(s_k|c_i) \cdot \text{count}(v_j \text{ in } c_i)}{\sum_{i,j} P(s_k|c_i) \cdot \text{count}(v_j \text{ in } c_i)} \end{aligned} \tag{A.20}$$

which is the Equa. (2.12).

In the similar manner, Equa. (A.17) becomes

$$\sum_i \frac{P(s_k|c_i)}{P(s_k)} + \mu = 0 \tag{A.21}$$

$$\sum_i P(s_k|c_i) + \mu \cdot P(s_k) = 0$$

By summarizing over k, we have

$$\begin{aligned}
\sum_i \frac{P(s_k|c_i)}{P(s_k)} + \mu &= 0 \\
\sum_{i,k} P(s_k|c_i) + \mu \cdot \sum_k P(s_k) &= 0 \\
\sum_{i,k} P(s_k|c_i) + \mu &= 0 \\
\sum_{i,k} P(s_k|c_i) &= -\mu
\end{aligned} \tag{A.22}$$

Substituting μ into Equa. (A.21), we have:

$$\begin{aligned}
P(s_k) &= -\frac{\sum_i P(s_k|c_i)}{\mu} \\
&= \frac{\sum_i P(s_k|c_i)}{\sum_{i,k} P(s_k|c_i)}
\end{aligned} \tag{A.23}$$

which is the Equa. (2.13).

Appendix B

Gibbs Sampling Method

In this Chapter, we study the Gibbs sampling [25, 26, 41, 43, 52, 65, 66, 82] method which is used to infer parameters in the latent Dirichlet allocation (LDA) [14, 15, 42, 44, 116] method to model documents. We first study the overview of the latent Dirichlet allocation method. Then, we study the derivation process for the algorithm used to infer parameters for the latent Dirichlet allocation.

The latent Dirichlet allocation method is a method to model the generation of documents. Suppose we have a set of documents, the LDA assumes that the topics of documents are from a predefined collection of topics. A document is not about only one topic but it is assumed to be about several topics, a mixture of topics. To model the general mixture of topics, the LDA uses a vector of topic to model a topic mixture of a document. Topics in turn are mixture of words. A topic is modeled by a vector whose elements is the probability of generating words under the concerned topic.

Denote the collection of topic as $T = \{t_1, t_2, ..., t_T\}$ as the set of topic, $D = \{d_1, d_2, ..., d_D\}$ as the set of documents, $V = \{v_1, v_2, ..., v_W\}$ as the vocabulary of words in the set of documents. The generation of a document d in the LDA model is as follows.

1. **Selection of document topic mixtures**
   Generate a topic vector $\Theta_d = (\vartheta_{d,1}, \vartheta_{d,2}, ..., \vartheta_{d,T})$ for the document. Since the Θ_d

models the mixture of topics, it should satisfy the following probability condition.

$$\sum_{t} \vartheta_{d,t} = 1 \tag{B.1}$$

The generation of Θ_d follows a Dirichlet distribution as follows.

$$\begin{aligned} P(\vartheta_{d,1}, \vartheta_{d,2}, ..., \vartheta_{d,T}|\vec{\alpha}) &= \frac{\Gamma(\sum_{t=1}^{T} \alpha_t)}{\prod_{t=1}^{T} \Gamma(\alpha_t)} \prod_{t=1}^{T} \vartheta_t^{\alpha_t - 1} \\ &= \frac{1}{\Delta(\vec{\alpha})} \prod_{t=1}^{T} \vartheta_t^{\alpha_t - 1} \end{aligned} \tag{B.2}$$

where $\vec{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_T)$ are the parameters of the Dirichlet distribution.

2. **Selection of word topics**
   The LDA assumes that a document may have some topics. Therefore, each word in document is generated from different topics. According to this assumption, the LDA assigns a topic t for each word according to the distribution of topic Θ_d.

3. **Generation of words**
   Use the selected topic t to generate a word. The distribution of words in a topic is assumed to follow a Dirichlet distribution.

$$\sum_{v} \varphi_{t,v} = 1 \tag{B.3}$$

$$\begin{aligned} P(\varphi_{t,1}, \varphi_{t,2}, ..., \varphi_{t,V}|\vec{\beta}) &= \frac{\Gamma(\sum_{i=1}^{V} \beta_i)}{\prod_{i=1}^{V} \Gamma(\beta_i)} \varphi_1^{\beta_1 - 1} ... \varphi_V^{\beta_V - 1} \\ &= \frac{1}{\Delta(\vec{\beta})} \varphi_1^{\beta_1 - 1} ... \varphi_V^{\beta_V - 1} \end{aligned} \tag{B.4}$$

where $\vec{\beta} = (\beta_1, \beta_2, ..., \beta_V)$ are the parameters of the Dirichlet distribution.

Suppose we put all documents in the set in sequence create an entire document. Denote the word vector made from all words in the entire document as $\vec{w} = (w_1, w_2, ..., w_W)$, where W is the total number of words in all documents, the topic vector made from topics of all words in the entire document as $\vec{z}$. Given a set of documents, the word

vector $\vec{w}$ is the only information that we can observe. We have the following parameters that we have to infer.

1. **Topic distribution vectors of documents**
   $\Theta_d = (\vartheta_{d,1}, \vartheta_{d,2}, ..., \vartheta_{d,T})$, for all $d \in \{d_1, d_2, ..., d_D\}$.

2. **Word distribution vectors of topics**
   $\Phi_t = (\varphi_{t,1}, \varphi_{t,2}, ..., \varphi_{t,V})$, for all $t \in \{t_1, t_2, ..., t_T\}$.

3. **Topic IDs of words in all documents**
   $\vec{z} = (z_1, z_2, ..., z_W)$, where W is the total number of words in all documents. These values z_i are not required in the LDA model, but they are used in the Gibbs sampling process to infer the above two sets of parameters.

The parameters $\vec{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_T)$ and $\vec{\beta} = (\beta_1, \beta_2, ..., \beta_V)$ are required to select the Dirichlet distributions for the LDA. They are called hyper-parameters in the LDA model.

Given the vector $\vec{z}$, the probability of generating $\vec{w}$ is derived as follows.

$$p(\vec{w}|\vec{z}) = \int p(\vec{w}|\vec{z}, \Phi) p(\Phi|\vec{\beta}) d\Phi \tag{B.5}$$

$$p(\Phi|\vec{\beta}) = \prod_{t=1}^{T} \frac{1}{\Delta(\vec{\beta})} \prod_{v} \varphi_{t,v}^{\beta_v - 1} \tag{B.6}$$

$$p(\vec{w}|\vec{z}, \Phi) = \prod_{t=1}^{T} \prod_{v} \varphi_{v}^{n_t^{(v)}} \tag{B.7}$$

$$\begin{aligned} p(\vec{w}|\vec{z}) &= \int \prod_{t=1}^{T} \frac{1}{\Delta(\vec{\beta})} \prod_{v} \varphi_{t,v}^{n_t^{(v)} + \beta_v - 1} d\Phi \\ &= \prod_{t=1}^{T} \left(\frac{1}{\Delta(\vec{\beta})} \int \prod_{v} \varphi_{t,v}^{n_t^{(v)} + \beta_v - 1} d\Phi \right) \\ &= \prod_{t=1}^{T} \frac{\Delta(\vec{\beta} + \vec{n}_t)}{\Delta(\vec{\beta})} \end{aligned} \tag{B.8}$$

where $n_t^{(v)}$ is the number of times v is assigned to topic t, and $\vec{n}_t = (n_t^{(1)}, n_t^{(2)}, ..., n_t^{(V)})$.

Given the hyper-parameter vector $\vec{\alpha}$, the probability of generating $\vec{z}$ is derived as follows.

$$p(\vec{z}|\vec{\alpha}) = \int p(\vec{z}|\Theta)p(\Theta|\vec{\alpha})d\Theta \tag{B.9}$$

$$\begin{aligned} p(\vec{z}|\Theta) &= \prod_{w=1}^{W} \vartheta_{d_i,z_i} \\ &= \prod_{d=1}^{D}\prod_{t=1}^{T} \vartheta_{d,t}^{n_d^{(t)}} \end{aligned} \tag{B.10}$$

where $n_d^{(t)}$ is the number of times topic t appears in document d.

$$p(\Theta|\vec{\alpha}) = \prod_{d=1}^{D}\left(\frac{1}{\Delta(\vec{\alpha})}\prod_{t=1}^{T}\vartheta_t^{(\alpha_t-1)}\right) \tag{B.11}$$

Substituting into Equa. (B.9), we get

$$\begin{aligned} p(\vec{z}|\vec{\alpha}) &= \int\left(\prod_{d=1}^{D}\prod_{t=1}^{T}\vartheta_{d,t}^{n_d^{(t)}}\right)\prod_{d=1}^{D}\left(\frac{1}{\Delta(\vec{\alpha})}\prod_{t=1}^{T}\vartheta_t^{(\alpha_t-1)}\right)d\Theta \\ &= \int\prod_{d=1}^{D}\left(\frac{1}{\Delta(\vec{\alpha})}\prod_{t=1}^{T}\vartheta^{n_d^{(t)}+\alpha_t-1}\right)d\Theta \\ &= \prod_{d=1}^{D}\frac{\Delta(\vec{\alpha}+\vec{n}_d)}{\Delta(\vec{\alpha})} \end{aligned} \tag{B.12}$$

where $\vec{n}_d = (n_d^{(1)}, n_d^{(2)}, ..., n_d^{(T)})$.

Combining Equas. (B.8) and (B.12), we get

$$p(\vec{z},\vec{w}|\vec{\alpha},\vec{\beta}) = \prod_{t=1}^{T}\frac{\Delta(\vec{\beta}+\vec{n}_t)}{\Delta(\vec{\beta})}\prod_{d=1}^{D}\frac{\Delta(\vec{\alpha}+\vec{n}_d)}{\Delta(\vec{\alpha})} \tag{B.13}$$

In the Gibbs sampling procedure, in each step we remove the old topic ID of a word w_i and try to generate a new topic ID for the word. In order to generate z_i^{new}, we need to calculate the probability $P(z_i = t)$ given the information of the rest parameters. That is, we have to calculate $P(z_i|\vec{z}_{-i})$, where $\vec{z}_{-i} = (z_1, z_2, ..., z_{i-1}, z_{i+1}, ..., z_W)$.

$$\begin{aligned}
p(z_i|\vec{z}_{-i}) &= \frac{p(\vec{z},\vec{w})}{p(\vec{z}_{-i},\vec{w})} \\
&= \frac{\prod_{t=1}^{T}\frac{\Delta(\vec{\beta}+\vec{n}_t)}{\Delta(\vec{\beta})}\prod_{d=1}^{D}\frac{\Delta(\vec{\alpha}+\vec{n}_d)}{\Delta(\vec{\alpha})}}{\prod_{t=1}^{T}\frac{\Delta(\vec{\beta}+\vec{n}_{-i,t})}{\Delta(\vec{\beta})}\prod_{d=1}^{D}\frac{\Delta(\vec{\alpha}+\vec{n}_{-i,d})}{\Delta(\vec{\alpha})}} \\
&= \frac{\Delta(\vec{\beta}+\vec{n}_{t=z_i})\Delta(\vec{\alpha}+\vec{n}_{d=d_i})}{\Delta(\vec{\beta}+\vec{n}_{-i,t=z_i})\Delta(\vec{\alpha}+\vec{n}_{-i,d=d_i})} \\
&= \frac{\frac{\Gamma(n_{t=z_i}^{(v=w_i)}+\beta_{v=w_i})}{\Gamma(n_{t=z_i}^{(\Sigma)}+\beta_\Sigma)}\cdot\frac{\Gamma(n_{d=d_i}^{(t=z_i)}+\alpha_{t=z_i})}{\Gamma(n_{d=d_i}^{(\Sigma)}+\alpha_\Sigma)}}{\frac{\Gamma(n_{-i,t=z_i}^{(v=w_i)}+\beta_{v=w_i})}{\Gamma(n_{-i,t=z_i}^{(\Sigma)}+\beta_\Sigma)}\cdot\frac{\Gamma(n_{-i,d=d_i}^{(t=z_i)}+\alpha_{t=z_i})}{\Gamma(n_{-i,d=d_i}^{(\Sigma)}+\alpha_\Sigma)}} \\
&= \frac{n_{-i,t=z_i}^{(v=w_i)}+\beta_{v=w_i}}{n_{-i,t=z_i}^{(\Sigma)}+\beta_\Sigma}\cdot\frac{n_{-i,d=d_i}^{(t=z_i)}+\alpha_{t=z_i}}{n_{-i,d=d_i}^{(\Sigma)}+\alpha_\Sigma}
\end{aligned} \tag{B.14}$$

For a document d, the probability that it select a topic t' for a new word w' is as follows.

$$\begin{aligned}
\vartheta_{d,t'} &= P(z' = t'|\vec{z}) \\
&= \frac{P(\vec{z}')}{P(\vec{z})} \\
&= \frac{n_d^{(t')}+\alpha_{t'}}{n_d^{(\Sigma)}+\alpha_\Sigma}
\end{aligned} \tag{B.15}$$

For a topic t, the probability that topic t generates a new word w' is as follows.

$$\begin{aligned} \varphi_{t,w'} &= P(w'|\vec{z'}) \\ &= \frac{P(\vec{w'}|\vec{z'})}{P(\vec{w}|\vec{z})} \\ &= \frac{n_t^{(v=w')} + \beta_{v=w'}}{n_t^{(\Sigma)} + \beta_\Sigma} \end{aligned} \tag{B.16}$$

The derivations of Equas. (B.15) and (B.16) are similar to that of Equa. (B.14).

Publications in doctor researches

[1] Q. Minh Vu, A. Takasu, and J. Adachi. Using web directories for similarity measurement in personal name disambiguation. In *Journal of Information Processing and Management,* Elsevier, to appear (refereed).

[2] Q. Minh Vu, T. Masada, A. Takasu, and J. Adachi. Using a knowledge base to disambiguate personal name in web search results. In *SAC '07: Proceedings of the 2007 ACM Symposium on Applied Computing*, pages 839–843. ACM Press (refereed, acceptance rate of 32.3%).

[3] Q. Minh Vu, T. Masada, A. Takasu, and J. Adachi. Using web directories for similarity measurement in personal name disambiguation. In *AINA Workshop/Symposia, The 2007 IEEE International Symposium on Data Mining and Information Retrieval* (refereed, acceptance rate of 34.3%).

[4] Q. Minh Vu, T. Masada, A. Takasu, and J. Adachi. Personal Name Disambiguation in Web Search Using Knowledge Base. In *DBSJ Letters*, Vol. 5, No. 4, pages 53-56, March, 2007. The Database Society of Japan (refereed).

[5] Q. Minh Vu, T. Masada, A. Takasu, and J. Adachi. Disambiguation of People in Web Search Using a Knowledge Base. In *Proceedings of the RIVF2007 International Conference*, 2007 (refereed, best student paper award, acceptance rate of 29%).

[6] Q. Minh Vu, T. Masada, A. Takasu, and J. Adachi. Name Disambiguation in Web Search Using Knowledge Base. In *Summer Database Workshop DBWS2006*, July, 2006.

Award

Best student paper award in the RIVF2007 International Conference, Hanoi, Vietnam, March, 2007.

Publications in other researches

[1] Q. Minh Vu, T. Hashiguchi, Y. Sun, X. Wang, H. Morikawa, T. Aoyama. Bandwidth Guarantee for Optical Burst Switched Networks with Periodical Wavelength Sharing. In *IEICE General Conference*, March 2005.

[2] Y. Sun, T. Hashiguchi, Q. Minh Vu, X. Wang, H. Morikawa, T. Aoyama. Design and Implementation of an Optical Burst-Switched Network Testbed. In *IEEE Communications Magazine*, vol. 43, no. 11, pp. s48-s55, November 2005 (refereed).

[3] Y. Sun, T. Hashiguchi, Q. Minh Vu, X. Wang, H. Morikawa, T. Aoyama. A Burst Switched Photonic Network Testbed: It's Architecture, Protocols and Experiments. In *IEICE Transactions on Communications*, October, 2005 (refereed).

[4] Y. Sun, T. Hashiguchi, Q. Minh Vu, X. Wang, H. Morikawa, T. Aoyama. Design and Implementation of Burst Switching Nodes for WDM Optical Networks. In *Proceedings of Asia-Pacific Optical Communications Conference (APOC2004)*, November, 2004 (refereed).

[5] Y. Sun, T. Hashiguchi, Q. Minh Vu, X. Wang, H. Morikawa, T. Aoyama. A Burst Switched Network Testbed for Future Photonic Internet. In *IEICE Technical Report*, PN2004-63, December 2004.

[6] Y. Sun, T. Hashiguchi, Q. Minh Vu, X. Wang, H. Morikawa, T. Aoyama. On the Design of Control Plane for WDM Burst Switched Networks. In *Proceedings of the 4th International Conference on Optical Internet (COIN2005)*, May, 2005 (refereed).

[7] Y. Sun, T. Hashiguchi, Q. Minh Vu, X. Wang, H. Morikawa, T. Aoyama. Optical

Burst Switching Testbed. In *Proceedings of the International Conference on IP + Optical Network (iPOP2005)*, demo, Tokyo, Japan, February, 2005.

[8] H. Sugita, Q. Minh Vu, T. Masuzaki, H. Tsutsui, T. Izumi, T. Onoye, Y. Nakamura. JPEG2000 HIGH-SPEED PROGRESSIVE DECODING SCHEME. In *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS2004)*, May, 2004 (refereed).

Bibliographies

[1] A. Aizawa. The feature quantity: an information theoretic perspective of tfidf-like measures. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 104–111, New York, NY, USA, 2000. ACM Press.

[2] A. Aizawa. An information-theoretic perspective of tf-idf measures. *Inf. Process. Manage.*, 39(1):45–65, 2003.

[3] M. Bachle and P. Kirchberg. Ruby on rails. *IEEE Software*, 24(6):105–108, 2007.

[4] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley Longman Publishing, 1999.

[5] A. Bagga and B. Baldwin. Entity-based cross-document coreferencing using the vector space model. In *ACL1998*, 1998.

[6] B. Baldwin, M. Collins, J. Eisner, A. Ratnaparkhi, J. Rosenzweig, and A. Sarkar. University of pennsylvania: description of the university of pennsylvania system used for muc-6. In *MUC6 '95: Proceedings of the 6th conference on Message understanding*, pages 177–191, Morristown, NJ, USA, 1995. Association for Computational Linguistics.

[7] L. Baresi, F. Garzotto, and P. Paolini. From web sites to web applications: New issues for conceptual modeling. In *ER '00: Proceedings of the Workshops on Conceptual Modeling Approaches for E-Business and The World Wide Web and Conceptual Modeling*, pages 89–100, London, UK, 2000. Springer-Verlag.

[8] R. Bekkerman and A. McCallum. Disambiguating web apprearances of people in a social network. In *The Fourteenth International World Wide Web Conference, WWW2005*, 2005.

[9] T. Berners-Lee. WWW: past, present, and future. *Computer*, 29(10):69–77, 1996.

[10] T. Berners-Lee and M. Fischetti. *Weaving the Web; The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor (2 Cassettes)*. Harper Audio, 1999.

[11] Bird, Gourley, Devanbu, Gertz, and Swaminathan. Mining email social networks.

[12] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, August 2006.

[13] D. Black. *Ruby for Rails: Ruby Techniques for Rails Developers*. Manning Publications Co., Greenwich, CT, USA, 2006.

[14] D. M. Blei and M. I. Jordan. Modeling annotated data. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 127–134, New York, NY, USA, 2003. ACM Press.

[15] D. M. Blei and J. D. Lafferty. Correlated topic models. In *Advances in Neural Information Processing Systems 2006*, 2005.

[16] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[17] D. Bollegala, Y. Matsuo, and M. Ishizuka. Extracting key phrases to disambiguate personal name queries in web search. In *Proceedings of the workshop "How can Computational Linguistics improve Information Retreival?", COLING-ACL 2006*, 2006.

[18] E. Brill. A simple rule-based part of speech tagger. In *Proceedings of the third conference on Applied natural language processing*, pages 152–155, Morristown, NJ, USA, 1992. Association for Computational Linguistics.

[19] A. Broder. A taxonomy of web search. *SIGIR Forum*, 36(2):3–10, 2002.

[20] P. F. Brown, S. A. D. Pietra, V. J. D. Pietra, and R. L. Mercer. Word-sense disambiguation using statistical methods. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics*, pages 264–270, Morristown, NJ, USA, 1991. Association for Computational Linguistics.

[21] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[22] D. Cai, Z. Shao, X. He, X. Yan, and J. Han. Mining hidden community in heterogeneous social networks. In *LinkKDD '05: Proceedings of the 3rd international workshop on Link discovery*, pages 58–65, New York, NY, USA, 2005. ACM.

[23] R. Cailliau and J. Gillies. *How the Web Was Born: The Story of the World Wide Web*. Oxford University Press, 2000.

[24] J. Carroll and D. McCarthy. Word sense disambiguation using automatically acquired verbal preferences. *Computers and the Humanities*, 34(1–2), 2000.

[25] C. K. Carter and R. Kohn. On gibbs sampling for state space models. *Biometrika*, 81(3):541–553, September 1994.

[26] G. Casella and E. I. George. Explaining the gibbs sampler. *The American Statistician*, 46(3):167–174, 1992.

[27] S. Clark, J. R. Curran, and M. Osborne. Bootstrapping pos taggers using unlabelled data. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, pages 49–55, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[28] M. Collins. Ranking algorithms for named-entity extraction: boosting and the voted perceptron. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 489–496, Morristown, NJ, USA, 2001. Association for Computational Linguistics.

[29] D. Cutting, J. Kupiec, J. Pedersen, and P. Sibun. A practical part-of-speech tagger. In *Proceedings of the third conference on Applied natural language processing*, pages 133–140, Morristown, NJ, USA, 1992. Association for Computational Linguistics.

[30] D. R. Cutting, J. O. Pedersen, D. Karger, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 318–329, 1992.

[31] I. Dagan and A. Itai. Word sense disambiguation using a second language monolingual corpus. *Comput. Linguist.*, 20(4):563–596, 1994.

[32] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[33] W. D. E. Knowledge resource tools for accessing large text files. In *Machine Translation: Theoretical and methodological issues*, pages 247–261. Cambridge University Press, 1987.

[34] M. Eckert and M. Strube. Resolving discourse deictic anaphora in dialogues. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, pages 37–44, Morristown, NJ, USA, 1999. Association for Computational Linguistics.

[35] M. Evans, N. Hastings, and B. Peacock. *Statistical Distributions 3rd ed.* Wiley, New York, 2000.

[36] K. T. Frantzi, S. Ananiadou, and J. Tsujii. The c-value/nc-value method of automatic recognition for multi-word terms. In *ECDL '98: Proceedings of the Second European Conference on Research and Advanced Technology for Digital Libraries*, pages 585–604, London, UK, 1998. Springer-Verlag.

[37] A. Fujii, T. Tokunaga, K. Inui, and H. Tanaka. Selective sampling for example-based word sense disambiguation. *Comput. Linguist.*, 24(4):573–597, 1998.

[38] G. W. Furnas, S. Deerwester, S. T. Dumais, T. K. Landauer, R. A. Harshman, L. A. Streeter, and K. E. Lochbaum. Information retrieval using a singular value decomposition model of latent semantic structure. In *SIGIR '88: Proceedings of the 11th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 465–480, New York, NY, USA, 1988. ACM.

[39] W. Gale, K. Church, and D. Yarowsky. A method for disambiguating word senses in a large corpus. In *Computers and the Humanities*, pages 415–439, 1992.

[40] S. Ghosal, J. K. Ghosh, and R. V. Ramamoorthi. Posterior consistency of dirichlet mixtures in density estimation. *The Annals of Statistics*, 27(1):143–158, 1999.

[41] W. R. Gilks. *Markov Chain Monte Carlo in Practice*. Chapman & Hall/CRC, December 1995.

[42] M. Girolami and A. Kabán. On an equivalence between PLSI and LDA. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 433–434, New York, NY, USA, 2003. ACM.

[43] P. Glasserman. *Monte Carlo Methods in Financial Engineering*. Series: Stochastic Modelling and Applied Probability , Vol. 53, 2003.

[44] T. Griffiths. Gibbs sampling in the generative model of latent dirichlet allocation, 2002.

[45] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proc Natl Acad Sci U S A*, 101 Suppl 1:5228–5235, April 2004.

[46] R. Guha and A. Garg. Disambiguating people in search. In *The Thirteenth International World Wide Web Conference, WWW2004*, 2004.

[47] R. W. Hamming. *Coding and information theory (2nd ed.)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1986.

[48] J. Han. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[49] J. Hockenmaier. Parsing with generative models of predicate-argument structure. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 359–366, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[50] N. Ide and J. Véronis. Introduction to the special issue on word sense disambiguation: the state of the art. *Comput. Linguist.*, 24(1):2–40, 1998.

[51] S. Jensen. An Introduction to Lagrange Multipliers, 2004.

[52] M. Johannes. Mcmc methods for financial econometrics, 2005.

[53] C. Jordan and C. Watters. Extending the rocchio relevance feedback algorithm to provide contextual retrieval. *Advances in Web Intelligence*, pages 135–144, 2004.

[54] Karabulut and Hasan. The physical meaning of lagrange multipliers. *European Journal of Physics*, 27(4):709–718, July 2006.

[55] M. Kitsuregawa, M. Toyoda, and I. Pramudiono. Web community mining and web log mining: commodity cluster based execution. In *ADC '02: Proceedings of the 13th Australasian database conference*, pages 3–10, Darlinghurst, Australia, Australia, 2002. Australian Computer Society, Inc.

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

[57] T. Kohonen, S. Kaski, K. Lagus, J. Salojärvi, V. Paatero, and A. Saarela. Organization of a massive document collection. *IEEE Transactions on Neural Networks, Special Issue on Neural Networks for Data Mining and Knowledge Discovery*, 11(3):574–585, May 2000.

[58] J. D. Lafferty, A. Mccallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[59] D. L. Lee, H. Chuang, and K. Seamons. Document ranking and the vector-space model. *Software, IEEE*, 14(2):67–75, 1997.

[60] S.-Z. Lee, J. ichi Tsujii, and H.-C. Rim. Part-of-speech tagging based on hidden markov model assuming joint independence. In *ACL '00: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 263–269, Morristown, NJ, USA, 2000. Association for Computational Linguistics.

[61] Y. Lee, H. Ng, and T. Chia. Supervised Word Sense Disambiguation with Support Vector Machines and Multiple Knowledge Sources. In *Proceedings of SENSEVAL-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 137–140, 2004.

[62] G. Levow. Corpus-based techniques for word sense disambiguation.

[63] Li. Random texts exhibit Zipf's law-like word frequency distribution. *IEEETIT: IEEE Transactions on Information Theory*, 38, 1992.

[64] K. Lindén. Evaluation of linguistic features for word sense disambiguation with self-organized document maps. *Computers and the Humanities*, 38(4):417–435, November 2004.

[65] J. S. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer, October 2002.

[66] D. J. C. Mackay. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, June 2002.

[67] M. Makrehchi and M. S. Kamel. Learning social networks from web documents using support vector classifiers. In *WI '06: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 88–94, Washington, DC, USA, 2006. IEEE Computer Society.

[68] B. Malin. Unsupervised name disambiguation via social network similarity. In *Workshop on Link Analysis, Counterterrorism, and Security, SIAM2005*, 2005.

[69] G. S. Mann and D. Yarowsky. Unsupervised personal name disambiguation. In *Proceedings of Computational Natural Language Learning 2003*, 2003.

[70] W. C. Mann. Discourse structures for text generation. In *Proceedings of the 22nd annual meeting on Association for Computational Linguistics*, pages 367–375, Morristown, NJ, USA, 1984. Association for Computational Linguistics.

[71] C. D. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*. The MIT Press, 2003.

[72] T. Masada, A. Takasu, and J. Adachi. Citation data clustering for author name disambiguation. In *In Proceedings of INFOSCALE 2007*, 2007.

[73] A. McCallum and W. Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, pages 188–191, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[74] D. McCarthy and J. Carroll. Disambiguating nouns, verbs, and adjectives using automatically acquired selectional preferences. *Comput. Linguist.*, 29(4):639–654, 2003.

[75] G. J. Mclachlan and T. Krishnan. *The EM Algorithm and Extensions (Wiley Series in Probability and Statistics)*. Wiley-Interscience, October 2007.

[76] M. Meilă. Comparing clusterings—an information based distance. *J. Multivar. Anal.*, 98(5):873–895, 2007.

[77] D. Merkl, M. A. Tjoa, and G. Kappel. A self-Organizing Map that learns the semantic similarity of reusable software components. In A. C. Tsoi and T. Downs, editors, *Proc. ACNN'94, 5th Australian Conf. on Neural Networks*, pages 13–16, St. Lucia, Australia, 1994. Univ. Queensland.

[78] L. Michael. Automatic sense disambiguation: How to tell a pine cone from an ice cream cone. In *Proceedings of the 1986 SIGDOC conference*, pages 24–26. Association for Computating Machinery, 1986.

[79] T. M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[80] G. Muresan. Review of "trec - experiment and evaluation in information retrieval" by ellen m. voorhees and donna k. harman (eds.), the mit press, cambridge, ma, 2005. *Inf. Process. Manage.*, 43(1):285–287, 2007.

[81] I. Nancy and V. Jean. Word sense disambiguation: The state of the art, 1998.

[82] R. M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, University of Toronto, 1993.

[83] J. P. Neeraj. A framework for semantic web services discovery.

[84] J. Ontrup and H. Ritter. Large-scale data exploration with the hierarchically growing hyperbolic som. *Neural Netw.*, 19(6):751–761, 2006.

[85] T. O'reilly. What is web 2.0? design patterns and business models for the next generation of software., September 2005.

[86] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[87] M. Palmer, D. Gildea, and P. Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–105, 2005.

[88] T. Pedersen, A. Kulkarni, R. Angheluta, Z. Kozareva, and T. Solorio. Name discrimination by clustering similar contexts. In *Proceedings of the Sixth International Conference on Intelligent Text Processing and Computational Linguistics*, 2005.

[89] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Readings in speech recognition*, pages 267–296, 1990.

[90] E. Ramasso, M. Rombaut, and D. Pellerin. Forward-backward-viterbi procedures in the transferable belief model for state sequence analysis using belief functions. In K. Mellouli, editor, *ECSQARU*, volume 4724 of *Lecture Notes in Computer Science*, pages 405–417. Springer, 2007.

[91] N. Sahoo, J. Callan, R. Krishnan, G. Duncan, and R. Padman. Incremental hierarchical clustering of text documents. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 357–366, New York, NY, USA, 2006. ACM.

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

[93] M. Sanderson. Word sense disambiguation and information retrieval. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 142–151, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[94] A. Sandholm and M. I. Schwartzbach. A type system for dynamic web documents. In *POPL '00: Proceedings of the 27th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 290–301, New York, NY, USA, 2000. ACM.

[95] H. Schutze. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123, 1998.

[96] F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 134–141, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[97] D. Shen, J. Zhang, G. Zhou, J. Su, and C.-L. Tan. Effective adaptation of a hidden markov model-based named entity recognizer for biomedical domain. In *Proceedings of the ACL 2003 workshop on Natural language processing in biomedicine*, pages 49–56, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[98] N. Slonim and N. Tishby. Agglomerative information bottleneck, 1999.

[99] N. Slonim and N. Tishby. Document clustering using word clusters via the information bottleneck method. In *SIGIR '00: Proceedings of the 23rd annual in-*

ternational ACM SIGIR conference on Research and development in information retrieval, pages 208–215, New York, NY, USA, 2000. ACM Press.

[100] Y. Sure, P. Hitzler, A. Eberhart, and R. Studer. The semantic web in one day. *IEEE Intelligent Systems*, 20(3):85–87, 2005.

[101] C. Sutton and A. Mccallum. An introduction to conditional random fields for relational learning. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2006.

[102] D. Thomas, D. Hansson, L. Breedt, M. Clark, J. D. Davidson, J. Gehtland, and A. Schwarz. *Agile Web Development with Rails*. Pragmatic Bookshelf, December 2006.

[103] D. Thomas and A. Hunt. *Programming Ruby: the pragmatic programmer's guide*. The Pragmatic Programmers, LLC., Raleigh, NC, USA, 2 edition, August 2005.

[104] P. Tonella, F. Ricca, E. Pianta, and C. Girardi. Using keyword extraction for web site clustering. In *WSE*, pages 41–48, 2003.

[105] J. S. Treiman. Lagrange multipliers for nonconvex generalized gradients with equality, inequality, and set constraints. *SIAM J. Control Optim.*, 37(5):1313–1329, 1999.

[106] P. Turney. Word sense disambiguation by web mining for word co-occurrence probabilities.

[107] V. Vinay, K. Wood, N. Milic-Frayling, and I. J. Cox. Comparing relevance feedback algorithms for web search. In *WWW '05: Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1052–1053, New York, NY, USA, 2005. ACM.

[108] E. M. Voorhees. *TREC: Experiment and Evaluation in Information Retrieval*. Digital Libraries and Electronic Publishing. MIT Press, September 2005.

[109] M. N. Vrahatis, B. Boutsinas, P. Alevizos, and G. Pavlides. The new k-windows algorithm for improving the k-means clustering algorithm. *J. Complex.*, 18(1):375–391, 2002.

[110] Q. M. Vu, T. Masada, A. Takasu, and J. Adachi. Name disambiguation in web search using knowledge base. In *Summer Database Workshop DBWS2006*, Japan, 2006. The Database Society of Japan.

[111] Q. M. Vu, T. Masada, A. Takasu, and J. Adachi. Personal name disambiguation in web search using knowledge base. *DBSJ Letters*, 5(4), March 2007.

[112] Q. M. Vu, T. Masada, A. Takasu, and J. Adachi. Using a knowledge base to disambiguate personal name in web search results. In *SAC '07: Proceedings of the 2007 ACM Symposium on Applied Computing*, pages 839–843, New York, NY, USA, 2007. ACM Press.

[113] Q. M. Vu, T. Masada, A. Takasu, and J. Adachi. Using a knowledge base to disambiguate personal name in web search results. In *Proceedings of the RIVF2007 International Conference*, pages 839–843, 2007.

[114] Q. M. Vu, T. Masada, A. Takasu, and J. Adachi. Using web directories for similarity measurement in personal name disambiguation. In *AINA Workshop/Symposia, The 2007 IEEE International Symposium on Data Mining and Information Retrieval*, volume 1, pages 379–384, Los Alamitos, CA, USA, 2007. IEEE Computer Society.

[115] Q. M. Vu, A. Takasu, and J. Adachi. Using web directories for similarity measurement in personal name disambiguation. In *Information Processing and Management, Elsevier*, to appear.

[116] X. Wei and B. W. Croft. Lda-based document models for ad-hoc retrieval. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 178–185, New York, NY, USA, 2006. ACM Press.

[117] R. White, I. Ruthven, and J. M. Jose. The use of implicit evidence for relevance feedback in web retrieval. In *Proceedings of the 24th BCS-IRSG European Colloquium on IR Research*, pages 93–109, London, UK, 2002. Springer-Verlag.

[118] S. K. M. Wong, W. Ziarko, and P. C. N. Wong. Generalized vector spaces model in information retrieval. In *SIGIR '85: Proceedings of the 8th annual international*

ACM SIGIR conference on Research and development in information retrieval, pages 18–25, New York, NY, USA, 1985. ACM.

[119] B. Yang, W. Cheung, and J. Liu. Community mining from signed social networks. *IEEE Transactions on Knowledge and Data Engineering*, 19(10):1333–1348, 2007.

[120] D. Yarowsky. Word-sense disambiguation using statistical models of roget's categories trained on large corpora. In *Proceedings of the 14th conference on Computational linguistics*, pages 454–460, Morristown, NJ, USA, 1992. Association for Computational Linguistics.

[121] J. Zhang, D. Shen, G. Zhou, J. Su, and C. L. Tan. Enhancing hmm-based biomedical named entity recognition by studying special phenomena. *J Biomed Inform*, 37(6):411–422, December 2004.

[122] 長尾 真. 岩波講座ソフトウェア科学 15 自然言語処理. 岩波書店, 1996.