AUTHOR NAME DISAMBIGUATION AND CROSS SOURCE DOCUMENT COREFERENCE

A Thesis in
Information Sciences and Technology
by
Ke Dang

© 2011 Ke Dang

Submitted in Partial Fulfillment of the Requirements for the Degree of
Master of Science

December 2011
The thesis of Ke Dang was reviewed and approved* by the following:

Lee Giles  
Professor of Information Sciences and Technology  
Thesis Advisor

John Yen  
Professor of Information Sciences and Technology

Dinghao Wu  
Senior Lecturer and Research Scientist of Information Sciences and Technology

Mary Beth Rosson  
Professor of Information Sciences and Technology  
Chair of College of Information Sciences and Technology

*Signatures are on file in the Graduate School.
ABSTRACT

This thesis deals with two research problems: author name disambiguation in digital library and cross-source document coreference.

The first problem comes from the digital library, which is an important technological tool to maintain the information used by users. However, due to the problem of ambiguous author names, users can not distinguish the exact authors of the articles in the digital library. This ambiguity mainly comes from two problems: polyseme, an author name shared by multiple authors, and synonym, an author with multiple name variant. Successfully addressing this ambiguous author name problem can improve the search quality of the digital library when one intends to search a specific author, which happens quite frequently in the digital library. In addition, when one attempts to compute statistics such as the reputation of an author based on his publications, disambiguating the author name enhances the accuracy. In this thesis, we present a comprehensive and synthetically summarization of the author name disambiguation algorithms. We also survey the evaluation datasets and metrics used in the papers. In addition, based on the survey, we suggest several possible directions and interesting ideas in the future.

For the cross-source document coreference problem, it is a new and important research direction. Cross-source document coreference deals with the problem of disambiguating the
entities in documents of one source to their corresponding identities, if exists, in another source. For example, one source, which is called general source, can be World Wide Web and another source can be the Wikipedia, which is called canonical source. The success of cross-source document coreference can benefit many scenarios. We can automatically construct and enrich the entity information in one source according to its information in another source. Search results of entities in one source can also be grouped by their identities in the other sources. Furthermore, we can compute the reputation of one entity discussed in one source from the information of other sources. In this thesis, we utilize one information extraction tools, OpenCalais, and develop a large number of features, 88 features, to help cross-source document coreference. We also make use of a state-of-the-art machine learning algorithm, random forests, to this new area. In the experiment, we compare the random forests model with three traditional models: Decision Tree, Naïve Bayes and Bayes Network. The experiment results demonstrate that random forests model significantly outperforms Naïve Bayes and Bayes Network by 10.67% and 12.65%. Random forests algorithm also outperforms the Decision Tree by 2.88%.

**Keywords:** author name disambiguation, digital library, cross-source document coreference, random forests
# TABLE OF CONTENTS

List of Figures............................................................................................................................ vii

List of Tables ............................................................................................................................ iix

Acknowledgement...................................................................................................................... x

1 Introduction ............................................................................................................................ 1

1.1 Motivation and Problem Definition................................................................................ 1

1.2 Contribution of the Thesis.............................................................................................. 2

1.3 Outline of the Thesis......................................................................................................... 4

2 Author Name Disambiguation in Digital Library ............................................................... 5

2.1 Introduction to Author Name Disambiguation in Digital Library .................................. 5

2.2 Author Name Disambiguation Algorithms..................................................................... 7

2.2.1 Algorithms for Polyseme Problem.............................................................................. 8

2.2.2 Algorithm for Synonym Problem............................................................................. 13

2.2.3 Algorithms for Both Problems ................................................................................. 15

2.2.4 Summary .................................................................................................................. 45

2.3 Evaluation Datasets and Metrics..................................................................................... 51

2.3.1 Dataset ...................................................................................................................... 51

2.3.2 Evaluation Metrics ................................................................................................. 57

2.3.3 Summary .................................................................................................................. 66

2.4 Summary on Author Name Disambiguation in Digital Library ..................................... 67
3 Cross-source Document Coreference Using Random Forests ........................................ 69
  3.1 Introduction to Cross-source Document Coreference ............................................ 69
  3.2 Related Work ........................................................................................................ 72
  3.3 Methodology ........................................................................................................ 74
  3.4 Experiments ......................................................................................................... 81
  3.5 Summary on Cross-source Document Coreference Using Random Forests ......... 84
4 Conclusion and Future Work ................................................................................... 85
References .................................................................................................................. 88
List of Figures

FIGURE 1: CROSS-SOURCE DOCUMENT COREFERENCE EXAMPLE. GENERAL SOURCE IS ON THE LEFT AND CANONICAL SOURCE IS ON THE
right........................................................................................................................................................................ 2

FIGURE 2: CITATION LABELING ALGORITHM, CITED FROM (LEE ET AL., 2005) ................................................................. 9

FIGURE 3: BLOCK-BASED ALGORITHM, CITED FROM (LEE ET AL., 2005) ........................................................................... 13

FIGURE 4: AGGLOMERATIVE CLUSTERING ALGORITHM, CITED FROM (SONG ET AL., 2007) ................................................. 19

FIGURE 5: HEURISTIC-BASED HIERARCHICAL CLUSTERING ALGORITHM, CITED FROM (COTA ET AL., 2007) ....................... 21

FIGURE 6: STEP 2 OF COAUTHOR-BASED ALGORITHM, CITED FROM (KANG ET AL., 2009) ................................................... 24

FIGURE 7: PROCESS OF WEB-BASED ALGORITHM, CITED FROM (PEREIRA ET AL., 2009) .................................................... 25

FIGURE 8: CLUSTERING FUNCTION OF WEB-BASED AUTHOR DISAMBIGUATION, CITED FROM (PEREIRA ET AL., 2009) .......... 27

FIGURE 9: ONLINE SVM AND DBSCAN CLUSTERING AUTHOR DISAMBIGUATION, CITED FROM (HUANG ET AL., 2006) .......... 33

FIGURE 10: AN AUTHOR NAME WITH THREE DIFFERENT AFFILIATIONS, CITED FROM (CULOTTA ET AL., 2007) ................. 36

FIGURE 11: TRAINING PHASE OF THE ERROR-DRIVEN LEARNING ALGORITHM, CITED FROM (CULOTTA ET AL., 2007) ............ 37

FIGURE 13: HISTOGRAM OF NAME COUNTS IN MEDLINE, CITED FROM (TORVIK AND SMALHEISER, 2009) ......................... 53

FIGURE 14: A DBLP WEBPAGE FOR A SPECIFIC AUTHOR ................................................................................................ 54

FIGURE 15: SEARCH “SUPPORT VECTOR MACHINE” IN CITESEERX .................................................................................... 55

FIGURE 16: PERCENTAGE/RANK RATIO FOR TWO ALGORITHMS, CITED FROM (LEE ET AL., 2005) ................................. 58

FIGURE 17: EXAMPLE OF THE RESULTS BASED ON ACCURACY FOR TOP-K CANDIDATES, CITED FROM (LEE ET AL., 2005) ....... 60

FIGURE 18: EXAMPLE OF RESULTS USING “ACCURACY BASED ON CONFUSION MATRIX” (CITED FROM (HAN ET AL., 2005)) ....... 61

FIGURE 19: EXAMPLE OF THE RESULTS BASED ON F1P AND F1C THE NUMBERS IN X-AXIS INDICATING DIFFERENT AUTHOR NAMES
FROM (SONG ET AL., 2007)) .............................................................................................................................. 63

FIGURE 20: RELATIONSHIP BETWEEN CROSS DOCUMENT COREFERENCE AND CROSS-SOURCE DOCUMENT COREFERENCE ............. 73

FIGURE 21: CONSTRUCT A TREE IN RANDOM FORESTS ALGORITHM (TREEPATITUK AND GILES, 2009). 75
List of Tables

TABLE 1: PARTIALLY CITATION CLUSTERS OF THREE DISAMBIGUATED AUTHORS OF THE SAME NAME LABEL “J. E. SMITH”, CITED FROM (HAN ET AL., 2005) .......................................................................................................................................... 16

TABLE 2: SUMMARY OF THE ALGORITHMS ........................................................................................................................................ 47

TABLE 3: SUMMARY OF DATASETS USED IN THE ALGORITHMS DISCUSSED IN CHAPTER 2 ........................................................................................................... 51

TABLE 4: EXAMPLE OF METADATA IN THREE MEDLINE PAPERS, CITED FROM (TREERATPITUK AND GILES, 2009)........................................................................ 52

TABLE 5: SUMMARY OF MEDLINE, DBLP AND CiteSeerX ................................................................................................ 56

TABLE 6: EXAMPLE OF CONFUSION MATRIX........................................................................................................................................ 60

TABLE 7: MATCHING MATRIX, CITED FROM (KANG ET AL., 2009) .................................................................................................. 61

TABLE 8: SUMMARY OF THE EVALUATION METRICS ........................................................................................................... 64

TABLE 9: ENTITIES EXTRACTED FROM OpenCalais AND USED IN THIS CHAPTER ............................................................................ 76

TABLE 10: CONFUSION MATRIX FOR DIFFERENT MODELS .................................................................................................... 83

TABLE 11: DISAMBIGUATION PRECISION FOR DIFFERENT MODELS ......................................................................................... 83
Acknowledgement

This research thesis would not have been possible without the support of many people. I wish to express my gratitude to my supervisor, Dr. Giles who was abundantly helpful and offered invaluable assistance, support and guidance.

Deepest gratitude is also due to the members of the thesis committee, Dr. Yen and Dr. Wu for their unselfish and unfailing support as my thesis committees.

Special thanks also to all my graduate friends, especially group members: Jian Huang, Treeratpituk Puck, and Wenying Xiong for sharing the literature and invaluable assistance. Not forgetting to my bestfriends who always been there.

I would also like to convey thanks to the Ministry and Faculty for providing the financial means and laboratory facilities.

I wishes to express my love and gratitude to my beloved families, for their understanding and endless love, through the duration of my studies.

Ke Dang
1 Introduction

1.1 Motivation and Problem Definition

In the area of information science and technology, which is the interdisciplinary science dealing with information, technology and people, digital library is an important technological tool to maintain the information used by people. For example, the digital library MEDLINE (Torvik and Smalheiser, 2009) contains 15.3 million records of articles in the 2006 versions. With so much information stored in the digital library, it is very important to ensure that the records are correct. However, due to various problems such as the data-entry errors, ambiguous format and imperfect citation gathering (On et al., 2005), it is challenge to achieve this goal. Ambiguous author name is just one of these challenge problems and it is a very common problem in the digital libraries (Lavender et al., 2008). Author name disambiguation in digital library is the problem of identifying the true authors from the ambiguous author names in the papers recorded in the digital library. It is the first problem discussed in this thesis.

Author name disambiguation in digital library deals with the problem of disambiguating entities within one source. The second problem in this thesis is cross-source document coreference. Cross-source document coreference (CSDC) deals with the problem of
disambiguate the entities in documents of one source to their corresponding identities, if exists, in another source. In this thesis, we will use the individuals with the same name as the example of ambiguous entities. For example, when we search one person name, “Paul Collins” in the search engine, as the left graph in below, the results retrieved by the search engine is one source. We have another canonical source like Wikipedia. There are 12 different “Paul Collins” in the Wikipedia as in the right graph. Cross-source document coreference algorithms disambiguate the person names in the search results to their matching Wikipedia pages.

Figure 1: Cross-source document coreference example. General source is on the left and canonical source is on the right

1.2 Contribution of the Thesis

As discussed above, this thesis mainly deals with two research problems. The first one is author name disambiguation in digital library and the second one is cross-source document coreference.
For author name disambiguation in digital library, there are two challenges in this research area: (1) How to develop an algorithm to disambiguate the author names. (2) How to evaluate the performance of the algorithms. For the first challenge, we will synthetically survey the author name disambiguation algorithms. We will analyze them according to the problems they solve: algorithms for polyseme problem, algorithms for synonym problem and algorithms for both problems. The second research challenge comes from the evaluations of the performance of the author name disambiguation algorithms. First, various evaluation metrics are used to measure the performance of author disambiguation algorithms. It is partly because of the different expressions of this research problem. Although all algorithms attempt to address the author name disambiguation problem as discussed above, the forms of the expressions over this problem are different. Second, various datasets are used in the papers. Thus, now the algorithms are evaluated by different evaluation metrics and over different datasets. In this thesis, we will review these evaluation metrics and datasets.

Cross-source document coreference is a new research direction. In order to conduct cross-source document coreference, there are two steps: constructing the canonical source and disambiguating entities in the documents of other source to the canonical source. The first step is done by the creators of Wikipedia, DBPedia and etc. This paper focuses on the second step: based upon an existing canonical source, designing algorithm to disambiguate the
entities in documents of other sources to their corresponding identities in the canonical source.

The contribution of this thesis comes from two perspectives. First, this thesis leverage one information extraction tools, OpenCalais, and developed a large number of features, 88, to help cross-source document coreference. Second, it utilizes a state-of-the-art machine learning algorithm, random forests, to this new research area. The experiment results demonstrate that random forests outperform all the traditional models in the cross-source document coreference problem.

1.3 Outline of the Thesis

This thesis is structured as following: Chapter 2 synthetically surveys the author name disambiguation algorithms and studies the evaluation datasets and metrics for author name disambiguation algorithms. Chapter 3 provides our algorithm for the cross-source document coreference problem. Finally, we summarize this paper and discuss the future work in the chapter 4.
2 Author Name Disambiguation in Digital Library

2.1 Introduction to Author Name Disambiguation in Digital Library

As we stated in the chapter 1, author name disambiguation in digital library is the problem of identifying the true authors from the ambiguous author names in the papers recorded in the digital library. In the chapter, we synthetically survey the author name disambiguation algorithms and their evaluation datasets and metrics.

We define the range of this research problem and research papers we examined in this chapter in this paragraph. First, we need to know that the author names can be found at two places in the digital libraries. The first place is the records of the articles themselves. Each article can have multiple author names. The second place is the citations of each article. An article can include many citations. In this chapter, we will concentrate on both types of author names. And in the following of the paper, we will consider that the citations is the same as the article since the author name disambiguation algorithms used in the citations can also been applied to articles. Second, there are two types of ambiguity for the author names. The first one is polyseme, which means that multiple individuals who are the authors have the same name.
For example, the author name “D. Johnson” can represent “David B. Johnson” of the Rice University or a different person “David S. Johnson” in the AT&T research lab (Han et al., 2005). The second ambiguity is synonyms, which means that the same individual has different name variants. Take “David S. Johnson” in the AT&T research lab for example, when his name appears in the papers, it can be “D. Johnson” and it can also be “D. S. Johnson”. Particularly, when applied to citations of the papers, the two problems are also defined as mixed citation and split citation by Lee et al. (Lee et al., 2005). Third, we focus on the papers specifically dealing with the author name disambiguation problem in digital library. The disambiguation object is the author name in digital library so the features used in the algorithms of these papers are domain-specific features such as coauthors and title of papers and the evaluation dataset are from the digital libraries. We will not survey the papers regarding to the broader and related research problems such as the record linkage (Fellegi and Sunter, 1969) and person name disambiguation (Mann and Yarowsky, 2003). The algorithms for these papers use different sets of features such as birth date, job titles and neighboring person name in webpages for the web person name disambiguation problem. However, the specific methods of these related areas are good source for references. We can survey them in the future.

Since 1969, this problem has been discussed by Galvez (Galvez and Andmoya, 2007) and recognized as a very important issue. In the MEDLINE digital library, for example, it is
estimated that 23% of all query over the MEDLINE is just about the author names (Herskovic et al., 2007). However, 2/3 of the author names are ambiguous with the type of polyseme and each of the ambiguous names is shared by eight difference individuals on average (Torvik and Smalheiser, 2009). Because of the ambiguity of the author names, the search quality of the digital library can be hindered. In addition, in digital libraries such as CiteSeer (Giles et al., 1998), a digital library storing academic papers, we need to compute the reputations of authors based on their publications or citations. The ambiguity of the author names makes this task difficult. Therefore, successfully addressing this problem can not only improve the search quality but also the computation of the statistics based on the author names in digital libraries.

In this chapter, we will review the author name disambiguation algorithms in section 2.2. In section 2.3, we will study the evaluation datasets and metrics for author name disambiguation algorithms. Finally, we will summarize this chapter in the section 2.4.

### 2.2 Author Name Disambiguation Algorithms

Many researchers have conducted the research regarding the author name disambiguation problem. We know that there are two sub-problems for author name disambiguation problem: polyseme, which means that multiple individuals who are the authors have the same name, and synonym, which means that the same individual has different name variants. While most of algorithms can address both of the sub-problems, there are still algorithms designed
specifically for just one of them. Then we organize the sections in section 2.2 as below: In section 2.2.1, we will introduce the algorithms designed for polyseme problem. In section 2.2.2, we will present the algorithms for synonym problem. In section 2.2.3, we will discuss the algorithms which can handle both of the sub-problems. Finally, we summarize and compare all the algorithms discussed above in section 2.2.4.

2.2.1 Algorithms for Polyseme Problem

Citation Labeling Algorithm (Lee et al., 2005)

Lee et al. (Lee et al., 2005) present an algorithm that will handle the polyseme problem. They consider this problem in the scenario of citations. First, Lee et al. defines the mixed citation problem as:

*Given a collection of citations C, by an author aᵢ, can we quickly and accurately identify false citations by another author aⱼ, when aᵢ and aⱼ have the identical name spelling* (Lee et al., 2005).

In order to solve this problem, they propose a citation labeling function \( f_{cl} : c_i \rightarrow a_j \), where \( c_i \) is a citation with coauthors \( A=\{a_1, ..., a_n\} \), keywords from the title \( T=\{t_1, ..., t_m\} \) and venue name \( V \). After removing the first coauthors \( a_1 \) (\( a_1 \) is assumed to be the ambiguous name) from the coauthor list, the citation labeling function is trying to label an author \( a_j \) to this citation \( c_i \). Then this mixed citation problem can be resolved using the following algorithm(Lee et al., 2005).
For each citation $c_i \in C$

Remove the ambiguous author name $a_1$ from the coauthor list of $c_i$:

Apply $f_{ci}$ to $c_i$ and get author name $a_2$;

If $a_1 \neq a_2$, $c_i$ is a false citation, remove it from $C$; (Lee et al., 2005)

This process can be illustrated from Figure 2. "Daniel Ullman" is the author name removed from the coauthor list. Specifically, the citation labeling function works through computing the similarity between $c_i$ and all the possible authors and returns the author with the maximum similarity. The similarity between a citation $c$ and an author $a$ is computed by the following equation (Lee et al., 2005):

$$\text{sim}(c, a) = \alpha \text{sim}(c^c, a^c) + \beta \text{sim}(c^t, a^t) + \gamma \text{sim}(c^v, a^v)$$
where the $\alpha + \beta + \gamma = 1$, $c_o$, $c_t$ and $c_v$ are token vectors of coauthors, paper title and venue title of citation $c$, $a_o$, $a_t$ and $a_v$ are token vectors of coauthors, paper title and venue title of all the citations related to author $a$. For example, there is a citation “Dongwon Lee, Byung-won On, effective and scalable solutions for mixed and split citation problem in digital libraries, IQIS 2005”. Its token vector $c_c$ is [“Dongwon Lee”, “Byung-won On”]. Its $c_t$ is [“effective”, “scalable”, “solutions”, “mixed”, “split”, “citation”, “problem”, “digital”, “libraries”] (stopwords are removed). Its $c_v$ is [“IQIS”]. For two token vectors, their similarity is computed by the cosine similarity and TFIDF method of standard information retrieval technique (Baeza-Yates and Ribeiro, 1999).

**Search Engine Driven Algorithm (Tan et al., 2006)**

Tan et al. (Tan et al., 2006) have proposed a search engine driven author disambiguation algorithm to handle the polyseme problem. They model the polyseme problem as below:

*For an ambiguous author name, we assume that this name represents $k$ unique individuals. For a list of citations $C$, can we assign each citation $c \in C$ to one of the $k$ individuals.*

The motivation for their algorithm is that there should be hidden information that is difficult to extract from the citation record itself (Tan et al., 2006). For example, two papers discussed the similar topic with different keywords in the titles. This algorithm leverages the search engine to help solve this problem. This algorithm has three steps. In the first step, for each
citation \( c \in C \), this algorithm inputs the title of \( c \) to the search engine and obtain a list of URLs. Then, each citation \( c \) will be represented as a feature vector. The feature values in the feature vectors are the inverse host frequency (IHF) of these URLs, which is the measurement of the relative rarity of an internet host hosting these URLs. According to IHF, the URLs obtained from the aggregator websites will have low value and the URLs obtained from the personal publication websites will be assigned high value. The IHF values are estimated using an external corpus. The authors first build a list of citations from the top 100 author names (by number of citations) in DBLP (Tan et al., 2006). Then they input the titles of citations to the search engine and get a corpus of URLs. The URLs are further trimmed to their hostnames, each of which has frequency \( f(h) \) in the corpus. For each host name \( h \), its IHF is computed using the following equation:

\[
\text{IHF}(h) = \log_2 \frac{\max_h f(h)+1}{f(h)+1}
\]

In the second step, this algorithm will compute the distances of every two citations in the citation list \( C \) using the cosine similarity. In the third step, the hierarchical agglomerative clustering algorithm is used to partition the citation list \( C \) to \( k \) clusters representing \( k \) unique individuals.

**Summary**

Although the two algorithms attempt to solve the same problem, the ways they model the polyseme problem are quite different as discussed above. Both algorithms have their own
advantages and disadvantages.

One of the advantages of the citation labeling algorithm is that it uses the sampling techniques to speed up the author disambiguation process. This sampling technique is introduced in (Gravano et al., 2003) and it ensures that the labeling algorithm only considers the author names similar to the ambiguous author name. For example, in Figure 2, “Liwei Wang” is not considered because it is last name is not “Ullman”. The sampling strategy is similar to the block strategy used by the other algorithms (Huang et al., 2006; Lee et al., 2005). Furthermore, this algorithm can be improved by the introducing of more similarity computation function other than the TFIDF method.

The disadvantage of the search engine driven algorithm is the assumption that there are k unique individuals. For one thing, it is not easily to know the number k for each ambiguous author name. For another thing, for the consistently increasing digital libraries, the number k is not a static value. The advantage of this algorithm is the use of the external resource, i.e., the search engine. In addition, the use of the IHF is a novel way of measuring the similarity of two citations.
2.2.2 Algorithm for Synonym Problem

Block-based Algorithm (Lee et al., 2005)

Lee et al. (Lee et al., 2005) presents a block-based algorithm to handle the synonym problem. They also consider this problem in the scenario of citations. They first define the split citation problem as:

*Given two lists of author names, X and Y, for each author name $x \in X$, find name variants of $x$: $y_1, y_2, \ldots, y_n \in Y$* (Lee et al., 2005)

The solution for this problem can be illustrated from Figure 3. We can see that the algorithm can be summarized to three steps. In the first step, this algorithm creates a block for each of
the author names in X. In the second step, this algorithm assigns the author names in Y to the relevant blocks. For example, “W. Wang” and “Liwei Wang” is assigned to the block of “Wei Wang”. In the third step, within each block, this algorithm computes the distance of each author name from Y to the “block host name”, the author name for which this block is created. If the distance of an author name from Y to the “block host name” is less than a threshold, this author name is viewed as the name variant of the “block host name”. In the paper written by Lee et al. (Lee et al., 2005), they propose four different methods to measure the distance of two author names.

The first method is the Naïve Bayes Model. If they want to estimate the distance between two names $x$, the “block host name”, and $y_i \in Y$. The distance is represented as: $P(y_i) \prod_k P(A_k|y_i)$, where the $A_k$ is the k-th coauthor of $y_i$, being the same coauthor of $x$ (Lee et al., 2005). The probability of $P(y_i)$ and $P(A_k|y_i)$ is estimated from the training set.

The second method is the support vector machines (SVM). For all the author names in a block, this algorithm randomly uses half of the author names as the training set and trains a binary classification model. This classification model can determine whether an author name is the name variant of the “block host name”. This algorithm applies the binary classification model to the testing author names. The distance between the testing author name and “block host name” is represented by the confidence score of the classification model.
The third method is the string-based distance metric. The distance between two author names is represented by the distance between their coauthor lists. For example, when we want to compute the distance between “Jefferey Ullman” and “J. Ullman”, that is dist(“Jefferey Ullman” ,“J. Ullman”). We compute dist(coauthors(“Jefferey Ullman”) ,coauthors(“J. Ullman”)) instead. There are many string-based distance metric. This paper utilizes four methods proposed in (Cohen et al., 2003): Jaccard, TFIDF, Jaro and JaroWinkler. Please check the paper (Cohen et al., 2003) for details.

The fourth method is the vector-based cosine distance. Like the string-based distance metric, the distance between two author names is represented by the distance of their coauthor lists. However, instead of computing the string distance of two coauthor lists directly, this algorithm constructs two feature vector representing two authors and uses the coauthors as the features. Then the distance is the cosine distance of the two vectors: \[ \cos \theta = \frac{\langle \vec{a}, \vec{b} \rangle}{\|\vec{a}\| \|\vec{b}\|}, \]
where \( \vec{a} \) and \( \vec{b} \) are two feature vectors.

### 2.2.3 Algorithms for Both Problems

Most of the author name disambiguation algorithms can address both polyseme and synonym problems. These algorithms can be classified to two classes. The first one uses the unsupervised methods. The second one takes advantage of the supervised methods. While
most of the supervised learning methods also use the clustering methods like the first one, the
difference between the two methods is that the second method makes use of the labeled
training set. We will present the work done by the different research groups for each method.

2.2.3.1 Unsupervised Methods

Given a list of the citations, there should be many authors for this list of citations. If we see a
name, “Smith J. E.” for example, we want to know whether those “Smith J. E.” are referred to
the same person or several different persons. A typical method is that we cluster the citations.
The intuition for this clustering method is that we assume all the articles written by the same
person should share some similarities, which distinct this person from other persons. Then by
the appropriate clustering methods, the citations by the same person should be clustered into
the same cluster. Table 1 is an instance of the result of a clustering algorithm cited from (Han
et al., 2005). The clustering algorithm partitions the 9 citations to 3 clusters, which means that
the algorithm judges that there are three different “J. E. Smith”. All the methods discussed in
this section share two features:

1) Do not use the labeled training set.

2) They will use the clustering method to disambiguate the author names.

Table 1: Partially citation clusters of three disambiguated authors of the same name label “J. E. Smith”,
cited from (Han et al, 2005)
K-way Spectral Clustering (Han et al., 2005)

Han et al. formulates the author name disambiguation problem as partition the collections of citations to different clusters (Han et al., 2005). They assume that each cluster only contains the citations by the same authors. They achieve this by adopting the K-way spectral clustering (Zha et al., 2001) algorithm to the author name disambiguation problem. Table 1 shows the results of this algorithm. The spectral clustering techniques employ the eigen-decomposition methods and try to find the approximation of the global optimal solutions (Zha et al., 2001). This clustering technique has been successfully used in other areas like the data mining. One of the contributions of this paper is that this paper adopted K-spectral clustering algorithm to

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Author citations</th>
</tr>
</thead>
</table>
the author name disambiguation problem. Also, during the experiment, this paper has compared K-means clustering to the K-spectral clustering methods. Another contribution of this paper is the selection of the features for the clustering. This paper use three attributes of the citations as features: author names, paper titles, and publication venue titles. These three attributes will reflect the identification information of a particular author. This paper has also mentioned that their algorithm can also make use of additional features such as the affiliations and addresses of authors.

From this paper, we can see that for the clustering methods there are two important decisions: select the clustering methods and choose the right features. Thus, the advantages of this paper are its clear way of modeling the author disambiguation problem and its two contributions. However, one of the disadvantages for the K-spectral clustering is that although it can find the approximation of the global optimal solutions for a dataset it needs to know the K value of the dataset as a prior value, which is not a static number for the increasing dataset. In other words, the online author name disambiguation is not possible. The second disadvantage for the K-spectral clustering is that its computation complexity is $O(N^2)$, which is not suitable for the large-scale datasets.

**Topic-based Unsupervised Name Disambiguation (Song et al., 2007)**

Song et al. (Song et al., 2007) proposes an efficient topic-based unsupervised author name
disambiguation algorithm. This algorithm adopts both the graphical bayesian models and a hierarchical clustering method. It is based on two-stages. In the first stage it extended two generative models, probabilities latent semantic analysis (PLSA) (Hofmann, 1999) and latent dirichlet allocation (LDA) (Blei et al., 2003), to build two topic-based models. After the first stage, an unsupervised hierarchical agglomerative clustering method (Han et al., 2005) is used to partition the papers to different clusters with different authors as shown in Figure 4.

<table>
<thead>
<tr>
<th>Algorithm 1 Agglomerative Clustering</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: Input:</td>
</tr>
<tr>
<td>a₁,...a_M: names to cluster</td>
</tr>
<tr>
<td>D(aᵢ,aⱼ): point-level distance metric</td>
</tr>
<tr>
<td>C(cᵢ,cⱼ): cluster-level distance metric</td>
</tr>
<tr>
<td>Sim(aᵢ,aⱼ): name-name similarity matrix</td>
</tr>
<tr>
<td>ε,θ: threshold parameter</td>
</tr>
<tr>
<td>2: Initialize</td>
</tr>
<tr>
<td>place each name in a singleton cluster,</td>
</tr>
<tr>
<td>calculate the pairwise distance between</td>
</tr>
<tr>
<td>names according to D,</td>
</tr>
<tr>
<td>set C ← D,</td>
</tr>
<tr>
<td>3: Clustering Procedure</td>
</tr>
<tr>
<td>4: Repeat</td>
</tr>
<tr>
<td>find two names (aᵢ,aⱼ) or name clusters (cᵢ,cⱼ) that</td>
</tr>
<tr>
<td>are closest according to D and C,</td>
</tr>
<tr>
<td>randomly choose a name to represent a cluster,</td>
</tr>
<tr>
<td>if Sim(aᵢ,aⱼ) is greater than θ</td>
</tr>
<tr>
<td>merge the pair to form a new cluster,</td>
</tr>
<tr>
<td>else</td>
</tr>
<tr>
<td>find the next closest pair or quit if no pair satisfy</td>
</tr>
<tr>
<td>the criteria,</td>
</tr>
<tr>
<td>update the distance between clusters according to C,</td>
</tr>
<tr>
<td>5: Until the distance between the closest pair of any two</td>
</tr>
<tr>
<td>clusters is greater than ε,</td>
</tr>
<tr>
<td>6: Output: Clusters c₁,...c_T.</td>
</tr>
</tbody>
</table>

Figure 4: Agglomerative clustering algorithm, cited from (Song et al, 2007)

In order to introduce this algorithm clearly, we will briefly explain how the two stages work. In the first stage, learning the PLSA and LDA models is equal to learning all the probability of aᵢk, where aᵢk means that the probability of author name aᵢ being in the topic k. To compute the
distance of the author name $a_i$ and $a_j$, the Euclidean distance is employed:

$$D(a_i, a_j) = \sqrt{\sum_k (a_{ik} - a_{jk})^2}$$

Then from the Figure 4, we can see how the author name disambiguation algorithm works. In Figure 4, the $Sim(a_i, a_j)$ measures the pairwise similarity between the author names and it is computed by the following equation:

$$Sim(a_i, a_j) = 1 - \frac{Le(a_i, a_j)}{\max (|a_i|, |a_j|)}$$

Where the $Le(a_i, a_j)$ is the Levenshtein distance introduced by Levenshtein (Levenshtein, 1965) and the $|a_i|$ represents the length of name $a_i$. In Figure 4, the cluster level distance metrics is the measurement of the distance between clusters, which is computed by the complete-link metric proposed in (King, 1967). The $\epsilon$ and $\theta$ are used as the stopping criteria and are set in the beginning of the algorithm.

For the CiteSeer dataset with more than 750,000 documents, PLSA yields 418,500 unique authors in 2,570 minutes, while LDA finishes in 4,390 minutes with 418,775 authors (Song et al., 2007). One of the advantages for this paper is that it is more efficient to the spectral clustering method as discussed above. It is also more effective to another supervised method DBSCAN (Huang et al., 2006), which we will discussed later in this chapter. Another advantage for this algorithm is its ability to be extended to more general areas such as the person name disambiguation in web pages.
Heuristic-based Hierarchical Clustering (Cota et al., 2007)

Cota et al. (Cota et al., 2007) propose a heuristic-based hierarchical clustering method to disambiguate the author names. This algorithm is explained by assuming that the first author name in each citation is the ambiguous author name. However, it can also be applied to other author names.

As shown in Figure 5, this algorithm is processed in three steps. In the first step, this algorithm obtains all the citation records in the data set. Then this algorithm divides the citations into two groups. The first group includes all the citations with the ambiguous author name. For example, in Figure 5, all the four papers in Figure 5a are the citations with
ambiguous author name “Gupta, A.”. The citations in this group are the target citations we need to disambiguate. The second group includes the remaining citations which do not include the ambiguous author name. In the second step, this algorithm begins to partition the citations in the first group of the first step. First, the algorithm creates a cluster by adding the first citation record. Second, this algorithm tests the second citation and compares it with the first cluster. If both the first initial of the first name and the last name of the second citation are the same as the first cluster, the second citation is considered to be compatible to the first cluster. If the second citation record is compatible to the cluster and the second citation record has at least one co-author with the cluster, the second citation is added to the cluster; otherwise the second citation will create a new cluster itself. The previous procedure in the second step is conducted for every citation until every citation is in a cluster. The result of second step is illustrated in the Figure 5b: Three clusters are created and for the cluster 2 two citations have the compatible co-author names “Bettati, R.” and “Bettati, Riccardo”. After the second step, the group one has been partition to many fragments. In order to decrease the fragmentation, this algorithm tries to merge the clusters by a pairwise method. This algorithm uses the citations’ title and venue to compute two clusters’ distance. If the distance of two different clusters is less than a threshold, the two clusters are merged to one cluster until no merge can happen. The result of the third step is demonstrated in Figure 5c.

Like the previous algorithms such as K-way spectral clustering algorithm developed by Han
et al. (Han et al., 2005), this paper uses three attributes of the citations: author names, citation titles, and publication venue titles. The three attributes are considered to be the most important features to disambiguate the author names. In the second step, the intuition of utilizing the author names is that it is unlikely for two compatible author names sharing the compatible co-author names represent two different individuals, which has been confirmed in the experimental result of this paper. The intuition for the third step to employ the citation titles and venue is that the same person tends to publish his papers in the similar areas. Actually, the two intuitions are so important that they are used consciously or unconsciously by many author name disambiguation algorithms. For example, the intuition about coauthors will be used in other algorithms like the coauthor-based algorithm (Kang et al., 2009), which is introduced below.

**Coauthor-based Clustering (Kang et al., 2009)**

Kang et al. (Kang et al., 2009) propose an author name disambiguation algorithm based on the co-authorship of the authors. The whole algorithm can be divided to two steps. In the first step, this algorithm intends to retrieve more coauthors. This algorithm inputs two author names, including the ambiguous name, in one citation to the search engines. From the search results, this algorithm can get more coauthors of the ambiguous author name. For example, this algorithm wants to disambiguate the author name “In-su Kang” and a citation is written by “In-su Kang” and “Seung-Hoon Na”. This algorithm will input the two names to the search
engine. If there is one other article written by “In-su Kang”, “Seung-Hoon Na” and “Seungwoo Lee”, “Seungwoo Lee” is also considered to be the coauthor of “In-su Kang”. The second step is illustrated in the Figure 6.

<table>
<thead>
<tr>
<th>Input:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_1, \ldots, a_n$: same-name author occurrences</td>
</tr>
<tr>
<td>$a_i = {v_{1i}, \ldots, v_{mi}}$: each name occurrence $a_i$ has a set of $m \geq 0$ his/her coauthor names $\theta_i$; a cluster-merging threshold</td>
</tr>
<tr>
<td>Initialize:</td>
</tr>
<tr>
<td>$c_i = {a_i}$; consider each name occurrence $a_i$ as an element of cluster $c_i$</td>
</tr>
<tr>
<td>Loop:</td>
</tr>
<tr>
<td>1 DO</td>
</tr>
<tr>
<td>2 For each cluster-pair $(c_i, c_j)$, calculate $\text{CSim}(c_i, c_j)$</td>
</tr>
<tr>
<td>3 $\text{CSim}(c_i, c_j) = \max(\text{ASim}(a_s, a_t))$, $\forall a_s \in c_i$, $\forall a_t \in c_j$</td>
</tr>
<tr>
<td>4 $\text{ASim}(a_s, a_t) =</td>
</tr>
<tr>
<td>5 Find the most similar cluster-pair $(c_{i, j})$</td>
</tr>
<tr>
<td>6 $(c_{i, j}) = \text{argmax}<em>i \text{CSim}(c_i, c</em>{j})$</td>
</tr>
<tr>
<td>7 IF $\text{CSim}(c_{i, j}) &gt; 0$ THEN</td>
</tr>
<tr>
<td>8 $c_{i, j} = c_i \cup c_j$; merge $c_i$ and $c_j$ into a new larger cluster $c_{i, j}$</td>
</tr>
<tr>
<td>9 ENDIF</td>
</tr>
<tr>
<td>10 WHILE ($\text{CSim}(c_i, c_j) &gt; 0$)</td>
</tr>
<tr>
<td>Output:</td>
</tr>
<tr>
<td>Clusters of author occurrences: $[c_k]$</td>
</tr>
</tbody>
</table>

Figure 6: Step 2 of coauthor-based algorithm, cited from (Kang et al., 2009)

In Figure 6, the same-name author occurrence means the author names that need to be disambiguated. Each same-name author occurrence has a set of coauthors. This algorithm adopts the single-link agglomerative clustering method (Jain et al., 1999) and calculates the similarity of two clusters based on the similarity of same-name author occurrences in the two clusters. And the similarity of two same-name author occurrence is defined as the count number of the matched coauthors in the two same-name author occurrences. If the similarity of two clusters is larger than a threshold, the two clusters are merged to one cluster. The previous procedure is conducted repeatedly until no clusters can be merged.
The advantage for this algorithm is that although it only uses the feature of the coauthor its performance is reported to be good over a Korean dataset, because of the characteristic of Korean names: one person’s name is written in Korean in a single form, a surname followed by a given name without delimiters or middle name (Kang et al., 2009). However, the disadvantage for this algorithm is that it may not be useful for author names from other countries since it is not easy for this algorithm to incorporate more features such as the title of the citation and the citation venue. Another advantage of this algorithm is the first step of using the search engine, which can be adopted by other algorithms to get more coauthors. We have already seen two algorithms taking advantage of search engines (Kang et al., 2009; Tan et al., 2006) and will introduce one more such algorithm in the below.

Web-based Clustering (Pereira et al., 2009)

![Diagram of web-based algorithm](image)

Figure 7: Process of web-based algorithm, cited from (Pereira et al., 2009)

Pereira et al. propose an unsupervised web-based author disambiguation algorithm (Pereira et
al., 2009). The motivation of this algorithm is that it wants to find the personal information of the ambiguous names through their curricula and web pages contains publications of the ambiguous authors. This kind of information is valuable resource to disambiguate the author names since these web pages will usually contains the full names and other useful information. This algorithm tries to discover the information by submitting the query to search engines. This algorithm can be summarized to three steps, which are illustrated in Figure 7. This algorithm also assumes that there is a list of citations C and the first author name in each citation is the ambiguous author name.

In the first step, this algorithm obtains the information from the Web. This algorithm first extracts the information from the list of citations C containing the ambiguous name and submits the extracted information to the search engine. Pereira et al. suggests two kinds of queries to submit to the search engine: unquoted author name followed by the word “publication” and the unquoted citation title (e.g.: Pereira, D.A. publication using web information for author name disambiguation); unquoted author name followed by the quoted citation title (e.g.: Pereira, D.A. “using web information for author name disambiguation”). This algorithm collects the result lists of the search engine and create a web document corpus $D$. In the second step, this algorithm tries to identify the documents which are the single author documents, i.e. the documents including the citations of a single person. This algorithm also weights each document according to their importance with the inverse
document frequency (IHF) proposed by Tan et al. (Tan et al., 2006). In the third step, this algorithm uses a hierarchical clustering method to group the citations to clusters representing unique authors as illustrated in Figure 8.

```
Require: Citations \( C = \{c_1, c_2, ..., c_n\} \);
Require: Web document collection \( D = \{d_1, d_2, ..., d_m\} \) with its information extracted in Step 2;
Ensure: \( CL = \{(cc_1, p_1), (cc_2, p_2), ..., (cc_p, p_p)\} \)
where \( CL \) is the set of generated clusters, \( cc_i \) is the set of citations in cluster \( d_i \), and \( p_i \) is the author name of \( d_i \);

1: for each citation \( c_i \) do
2: create new cluster \( cl_i \);
3: \( cc_i \leftarrow c_i \);
4: \( p_i \leftarrow \) author name of \( c_i \);
5: end for
6: \( S \leftarrow \) single author documents in \( D \);
7: sort documents in \( S \) in descending order by \( IHF \);
8: for each \( d_j \in S \) in descending order do
9: \( A \leftarrow \) clusters that contain at least one citation found in \( d_j \);
10: if \( IHF(d_j) > \min IHF \) and author name of \( d_j \) is compatible with the author name of some cluster in \( A \) then
11: fuse all clusters in \( A \);
12: select author name for the fused clusters;
13: else
14: fuse clusters in \( A \) which have compatible author names and two pairs of distinct citations found in a document \( d_k \neq d_j \) whose \( \text{host}(d_k) \neq \text{host}(d_j) \);
15: select author name for each fused set of clusters;
16: end if
17: end for
18: \( R \leftarrow \) not single author documents in \( D \);
19: sort documents in \( R \) in descending order by \( IHF \);
20: for each \( d_j \in R \) in descending order do
21: \( A \leftarrow \) single clusters that contain citation found in \( d_j \);
22: fuse clusters in \( A \) which have compatible author names;
23: select author name for each fused set of clusters;
24: end for
```

Figure 8: Clustering function of web-based author disambiguation, cited from (Pereira et al., 2009)

The advantage of this algorithm is that it takes advantage of the web sites published by the authors themselves, thereby improving the performance of author disambiguation algorithm. This work can be viewed as an improved version of the search engine driven algorithm.
developed by Tan et al. (Tan et al., 2006). Another advantage of this algorithm is that it does not need the specification of the $k$ number in the other unsupervised methods. Also, this algorithm is quite flexible and each of the three steps has plenty space to improve. For example, the first step can use more query options.

**Summary**

We have introduced five different unsupervised algorithms in section 2.2.3.1. Although they have all utilized the clustering methods, the distances of elements for the clustering are computed from different perspectives: eigen-decomposition (Han et al., 2005), topic model (Song et al., 2007), heuristics (Cota et al., 2007), coauthorship (Kang et al., 2009), IHF (Pereira et al., 2009). Besides the advantages and disadvantages discussed right below the description of the algorithms themselves, we will introduce more comparisons of these algorithms in section 2.2.4.

**2.2.3.2 Supervised Methods**

Many research groups have also utilized the supervised learning methods to disambiguate the author name.

**K-means Clustering with Naïve Bayes Model (Han et al., 2003a)**

Han et al. (Han et al., 2003a) tries to disambiguate the author names using the K-means
clustering algorithm (Hartigan and Wong, 1979) with the naïve bayes model. This algorithm has three steps. In the first step, all the citations are randomly and equally assigned to K clusters. In the second step, for each citation in the dataset this algorithm estimates the probability that this citation will be generated by each cluster. Then this algorithm changes this citation to the cluster with the highest generating probability. The second step is conducted iteratively until this algorithm is coverage, when fewer than 1% of the citations change the cluster assignment (Hartigan and Wong, 1979). The algorithm outputs the final clusters as the result of the algorithm in the third step. Specifically, in the second steps, this algorithm uses the naïve bayes model to estimate the probability of one cluster generating one citation. Three features of the citations are utilized in this algorithm: coauthors, publication title and publication venue.

This algorithm is simple and straightforward. Also, more features such as the affiliation of the authors are easily to be incorporated into this algorithm helping improve the performance. The main disadvantage for this algorithm is that we need to preset the value K, which is not easily estimated. This estimation of the K value can be the future work of the K-means disambiguation algorithm.

**Naïve Bayes Model (Han et al., 2004)**

Besides as one of the steps of the disambiguation algorithms, naïve bayes model is used
directly for the author disambiguation problem by Han et al. (Han et al., 2004). They assume that each author’s papers are generated by naïve bayes model. Therefore, they will use the past papers of an author to predict the future papers. They assume that there exists a paper database. This database records each author’s papers indexed by the author’s canonical name, a name that is the minimal invariant and the complete name entity for disambiguation (Han et al., 2004). This database can be collected from the existing digital library such as DBLP.

Assume that there are N canonical author name entries $A_1, A_2, \ldots, A_N$ in the database. For a new citation $C$ with an ambiguous author name $x$, this author disambiguation algorithm determines which name entries this ambiguous name belongs to. This algorithm will remove this ambiguous author name from this paper and find the name entry $A_i$ with the maximum posterior probability. The target function is as below:

$$\arg\max_i P(A_i | C)$$

Based on the Bayes rule, the above function can be changed to:

$$\arg\max_i P(C | A_i) P(A_i) / P(C)$$

Since $P(C)$ is the value independent with the $A_i$, it can be get rid of from the above equation. The target equation can be further changed to:

$$\arg\max_i P(C | A_i) P(A_i)$$

$P(A_i)$ can be estimated by calculating the proportion of the papers of $A_i$ in the database. In order to estimate the $P(C | A_i)$, paper $C$ is represented by three attributes: paper coauthors, paper title, paper venue, symbolized as $T_1, T_2, T_3$, respectively. Based on the assumption that
the three attributes are independent to each other, we can get the function:

$$P(C|A_i) = \prod_{1 \leq j \leq 3} P(T_j|A_i)$$

Finally, the algorithm estimates the probability of $P(T_j|A_i)$ from the database. After all the probabilities are estimated from the database, this algorithm can find the result of the target function.

The advantage for this algorithm is that it not only utilizes the features we can see but also the features such as the prior of an author. However, the disadvantage for this algorithm is that it needs the database with the canonical author name entries. Therefore, this algorithm is not suitable to disambiguate new author names that are not in the database.

**Support Vector Machine (Han et al., 2004)**

Han et al. (Han et al., 2004) also presents an author disambiguation algorithm that utilizes the model of support vector machine (SVM) (Vapnik, 1995). For this SVM algorithm, it considers the author disambiguation problem as a multi-class classification problem. In other words, this algorithm views each author as a class. If there is an ambiguous author name in a paper, this algorithm removes this author name from this paper and tries to classify the paper to a class of an author. First, this algorithm constructs a feature vector for each paper. The features are the coauthors of this paper, the keywords of paper title and journal title with their
frequency in the citation as the feature weight (Han et al., 2004). Second, after the construction of the feature vector, this algorithm considers the multi-class classification problem as multiple binary classification problems. For example, there are N authors A₁, A₂, ..., Aₙ. This algorithm lets the A₁ as the first class and the remaining authors as the second class. If the binary classifier estimates that this paper is relevant to the first class, this paper is classified to the first author; otherwise the binary classifier continues to spit the the remaining authors to two classes until this paper is classified to one author. Third, we just need to know SVM how to solve the binary classification problem. As a supervised learning approach, there is a two-class training dataset including a large number of papers represented by their feature vectors. All these feature vectors are labeled to one of two classes. During the training phase of the SVM model, SVM algorithm finds an optimal hyperplane to separate the two classes of feature vectors in the training dataset as accurate as possible. The hyperplane can be represented as a decision function whose input is the feature vector and output is the class of the feature vector. Specifically, in the paper written by Han et al (Han et al., 2004), they use the SVMlight (Joachims, 2001) to execute their experiments.

If we consider the author disambiguation problem as the multi-class problem, there are many machine learning algorithm to use. The advantage for SVM model is that SVM has good generalization performance and the ability to handle high dimension data (Han et al., 2004). However, the SVM-based author disambiguation algorithm is based on the previous
classification of the author names in the training dataset. It is not suitable to disambiguate the new authors who are not in the training dataset. In addition, it needs to train a SVM model for each ambiguous author name so it is hard to be applied to large scale author names. In order to address the large scale author name disambiguation problem, some researchers use the online SVM as the distance function for a clustering algorithm, which we will introduce below.

**Online SVM and DBSCAN Clustering (Huang et al., 2006)**

![Online SVM and DBSCAN Clustering](image)

_Huang et al. (Huang et al., 2006) find that the scalability problem is a primary concern for the_
large-scale database. Meanwhile, the expandability problem is another issue for the disambiguation algorithm as more paper come into the disambiguation system all the previous results need to be adjusted to adapt to the new papers. In order to address the two problems, they propose a new author disambiguation algorithm as shown in the Figure 9:

This algorithm can be summarized to five steps. In the first step, the algorithm will extract the metadata of the papers using a SVM extractor described in (Han et al., 2003b). After the extraction of the metadata, in the second step, this system will put the author names with the similar variants to the candidate classes. Using the method can decrease the number of the pairwise comparison between the author names. This step is just like the block approach in the (Lee et al., 2005). In the third step, for each two author names $i$ and $j$ within the same candidate class, the similarity $\text{sim}(i,j)$ is computed by a similarity vector: $s(i,j) = [\text{sim}(i_1,j_1), ..., \text{sim}(i_k,j_k), ..., \text{sim}(i_n,j_n)]$, where $i_k$ and $j_k$ are the k-th attributes regarding to $i$ and $j$ ($1 \leq k \leq n$, $n$ is the number of the attributes of the author names). Different similarity metrics are applied to different attributes according to their characteristics: this algorithm uses the edit distance for emails and URLs; it uses the token based Jaccard similarity for addresses and affiliations; it uses the hybrid similarity Soft-TFIDF for name variations (Bilenko et al., 2003). In the fourth step, after getting the similarity vector $s(i,j)$, the support vector machine (SVM) (Vapnik, 1995) is used to classify whether the similarity vector belongs to the class of match, i.e., whether $i$ and $j$ are the name...
variant to each other. The confidence value of the SVM classification result is used as the distance of $i$ and $j$. Particularly, in this step, this algorithm uses the LASVM (Bordes et al., 2005), an online kernel classifier. In the fifth step, based on the pairwise distance calculated previously, this algorithm takes advantage of a cluster method DBSCAN (Ester et al., 1996) to finally partition the authors to clusters and finish the author disambiguation process.

There are multiple advantages for this algorithm. First, this algorithm makes use of the block concepts in the second step to reduce the complexity of the pairwise comparison for the author names. Second, unlike the third step in Block-based Algorithm (Lee et al., 2005) described above in our chapter, which uses the same distance metric for all the attributes, using different similarity metrics for different attributes improved the performance of the algorithm. Third, compared to traditional SVM, which works on the batch setting (Huang et al., 2006), LASVM works on the online setting. When a new paper comes into the system, LASVM model will fast change and integrate this new paper into itself accordingly. This online learning characteristic makes the LASVM, as well as the disambiguation algorithm, suitable for large-scale database, helping solve the scalability and expandability problems of other disambiguation algorithms. Fourth, the DBSCAN clustering can resolve the transitivity problem of the typical pairwise method: while paper $i$ is coherent with paper $j$ and paper $j$ is coherent with paper $k$, paper $i$ is not coherent with $k$. 
Error-driven Machine Learning (Culotta et al., 2007)

Culotta et al. (Culotta et al., 2007) have presented an error-driven machine learning approach for author disambiguation problem. They notice that, typically, most of the supervised learning methods have three steps. In the first step, these methods will train a binary classifier to determine the similarity of two authors. In the second step, each pair of authors will be tested by the binary classifier. In the third step, based on the similarity scores, these methods will partition the authors to clusters. Although these methods can achieve good results, they only utilize the pairwise similarity. For one thing, the pairwise method may result in the transitivity problem as described in (Huang et al., 2006). For another thing, even if some of the methods can ameliorate the transitivity problem by applying special approach, these methods can still omit the effects of the aggregate features. Those features are the evidence helping disambiguate the authors based on more than two authors. For instance, this kind of evidence can be: one author can only be related to a few organizations; one author may not publish more than thirty papers a year. One motivating example shown in (Culotta et al., 2007) is as below:

<table>
<thead>
<tr>
<th>Author</th>
<th>Title</th>
<th>Institution</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y. Li</td>
<td>Understanding Social Networks</td>
<td>Stanford</td>
<td>2003</td>
</tr>
<tr>
<td>Y. Li</td>
<td>Virtual Network Protocols</td>
<td>Peking Univ.</td>
<td>2001</td>
</tr>
</tbody>
</table>

Figure 10: An author name with three different affiliations, cited from (Culotta et al., 2007)

In the above example, three identical author names are observed. Also, the keywords in the
titles of the three papers are similar. They all include the keyword “network”. The pairwise methods may predict they have the same author. However, only the second and third papers are written by the same person. Hence, to employ the aggregate features, Culotta et al. (Culotta et al., 2007) presents the following error-driven algorithm:

Figure 11: Training phase of the error-driven learning algorithm, cited from (Culotta et al., 2007)

Figure 11 is the high-level pseudo-code for the training phase of the error-driven learning algorithm. $D$ is a training dataset with labeled papers $X_1, X_2, \ldots, X_n$. The author disambiguation algorithm attempts to partition the $n$ papers to $m$ sets $S_1,S_2,\ldots,S_m$. The partition of the dataset $D$ is represented as $T(D) = \{S_1,S_2,\ldots,S_m\}$. For each partition $T(D)$, there is a scoring function $G: T(D) \rightarrow \mathbb{R}$, which gives this partition a evaluation score. Then the purpose of author disambiguation algorithm is to find the optimal partition $T^*(D)$ with the highest score:

$$T^* = \arg \max_T G(T(D))$$

In Figure 11, $\Lambda^0$ and $\Lambda^1$ are the parameters for the machine learning algorithm and $A$ is a prediction function:
\[ A : X \times \Lambda \to T^0(X) \times \cdots \times T^r(X) \]

The output of \( A \) is a sequence of partitions, where \( T^r(X) \) is the final partition. In the step 5 of Figure 11, this algorithm will select the examples that will result in the errors during the training. It is in this step that this algorithm takes advantage of the aggregate features to help disambiguate the authors. The advantage of this algorithm is the using of the aggregate features of the authors. Also, the idea of utilizing the error to improve the disambiguation performance is good. In addition, the representation of the author disambiguation problem as searching the optimal partition is not only suitable for the error driven algorithm but also quite flexible and is worth exploring in future.

**Random Forests (Treeratpituk and Giles, 2009)**

Treeratpituk et al. (Treeratpituk and Giles, 2009) presents a random forests algorithm to solve the author disambiguation problem. They formulate the author disambiguation problem as below:

*Given two papers \( paper_A \) and \( paper_B \), both containing an author with the name “Iname, init”, where “Iname” refers to the last name and “init” refers to the first initial. We want to disambiguate whether they refer to the same person* (Treeratpituk and Giles, 2009).

The main purpose of the random forests algorithm is to learn a distance function for a pair of authors. First, in order to be more efficient, like the other algorithms (Huang et al., 2006; Lee et al., 2005), they use a blocking strategy to block the author names with the same last name.
and first initial. Second, for an ambiguous author name with the same last name and first initial in two papers $\text{paper}_A$ and $\text{paper}_B$, they use the following metadata of the two papers to construct a similarity profile (Treeratpituk and Giles, 2009):

$$\text{paper}_A = (\text{lname}_A, \text{fname}_A, \text{init}_A, \text{mid}_A, \text{suf}_A, \text{coauth}_A, \text{aff}_A, \text{title}_A, \text{jour}_A, \text{lang}_A, \text{year}_A, \text{mesh}_A)$$

$$\text{paper}_B = (\text{lname}_B, \text{fname}_B, \text{init}_B, \text{mid}_B, \text{suf}_B, \text{coauth}_B, \text{aff}_B, \text{title}_B, \text{jour}_B, \text{lang}_B, \text{year}_B, \text{mesh}_B)$$

$lname_i$=the author’s last name in paper$i$

$fname_i$=the author’s first name in paper$i$

$init_i$=the author’s first initial in paper$i$

$mid_i$=the author’s middle name in paper$i$, if given

$suf_i$=the author’s suffix in paper$i$

$coauth_i$=set of coauthors’ last name in paper$i$

$aff_i$=affiliation of the paper$i$’s 1$^{st}$ author

$title_i$= paper$i$’s title

$jour_i$= paper$i$’s journal name

$title_i$= paper$i$’s title

$lang_i$= paper$i$’s language

$year_i$= paper$i$’s year of publication

$mesh_i$= paper$i$’s mesh terms (Treeratpituk and Giles, 2009)

The similarity profile for this ambiguous name is a feature vector. This feature vector has 21 features. These 21 features are from the above metadata. Take one of the features for example,
the auth_suf feature: auth_suf is equal to 1 if suf_A and suf_B are given and suf_A is equal to suf_B; otherwise auth_suf is equal to 0. Please check the paper for details of the 21 features. After the determination of the similarity profile for this ambiguous author name, they can use the random forests model to classify the similarity profile. The classification result is the result of the author name disambiguation.

Random forest (Breiman, 2001) combines a set of decision trees. The classification result of the random forest is the vote of these decision trees. When building a tree, there are three differences to the traditional decision tree. First, the tree is built from a bootstrap sample. Second, instead of using all the variables, when splitting, the tree randomly chooses a subset of the variables and conducts the split based on this subset of variables. Third, the tree is not pruned.

There are a number of advantages to use the random forests algorithm over author disambiguation problem. First, since the random forest is the combination of many decision trees and each feature includes just a small number of information, even if some features are weak, it will not affect the overall performance of the random forest. Second, the dependencies and the interactions between the features can be captured by the random forest algorithm. This characteristic can be helpful for some intuitions. For example, there are an intuition: “If two author names with the same last name and first initial have the same
affiliation and coauthors, they are the same person.” Third, random forest training and predicting are fast, so it is suitable for the large scale author disambiguation problem.

**Probabilistic Similarity Metric (Torvik et al., 2005)**

Torik et al. (Torvik et al., 2005) propose a probabilistic model for the author disambiguation problem. This model can estimate the probabilistic of a pair of author names sharing the same last name and first initial to be the same individual. This algorithm has an assumption that two papers written by the same person should share a variety of similarities not only in the coauthors and paper titles but also other attributes of the papers. Thus, this algorithm can be summarized to be two steps. In the first step, this algorithm presents a comparison vector called the “similarity profile” for each two papers with the the same last name and first initial. This similarity profile is constructed based on eight features: paper title, journal name, coauthor names, medical subject headings, language, affiliation, middle initial, and suffix (e.g. Jr. or III). The similarity profile reflects the similarity between the two papers. We need to notice that this algorithm is specifically designed to deal with the author name disambiguation problem in a bio-informatics digital library called the MEDLINE and this Medline digital library records each article’s title, author name(s), affiliation (if available), abstract (if available), journal, date of publication, language, and medical subject headings (Torvik et al., 2005). The eight features can be retrieved directly from the MEDLINE records. In the second step, after the construction of the similarity profile between the two papers, this algorithm
compares this similarity profile with two reference sets: match set and nonmatch set. The match set contains millions pairs of papers which are written by the same authors and the nonmatch set includes millions pairs of papers which are written by different authors. The reason for this algorithm called the probability model is that it represents this likelihood as the probability. If the similarity profile is $x$, the match set is $M$ and the nonmatch set is $N$. The algorithm has transferred the disambiguation problem to the computation of the probability $Pr(M|x)$, which means the probability that $x$ can be included in the match set $M$. Through the bayes theorem:

$$Pr(M|x) = \frac{1}{1 + \frac{1 - Pr(M)}{Pr(M) Pr(x|M) / Pr(x|N)}}$$

Where $Pr(x|M)$ (or $Pr(x|N)$) means the probability of observing similarity profile $x$ when the two papers are written by the same author (or different author, respectively). $Pr(M)$ represents the proportion of pairs of papers that are written by the same author given all the pairs of papers with the same last name and first initial. This algorithm estimates the $Pr(M)$ and $Pr(x|M) / Pr(x|N)$ from the match set and nonmatch set and gets $Pr(M|x)$. One can use this probability to disambiguate the author name: if this probability is larger than a threshold, it is likely that it is written by the same author.

One of the advantages for this algorithm is that it can be combined with other unsupervised clustering methods and used as the pairwise similarity. Another advantage for this algorithm is
its automatic way to generate massive match set. This algorithm uses the pairs of papers with the same last name, first initial, middle initial and suffixes as the match set. These pairs of papers should represent the same author in most cases since the average $\Pr(x|M)/\Pr(x|N)$ is larger than 3.500 (Torvik et al., 2005). The disadvantage of this algorithm is that it takes advantage of the specific feature of the MEDLINE digital library, which records the eight features of each article, so it is not suitable to be applied to other digital libraries without so much information.

**Probabilistic Similarity Metric with Clustering (Torvik and Smalheiser, 2009)**

Based on the previous algorithm (A Probabilistic Similarity Metric for Author Name Disambiguation ) proposed by Torik et al. (Torvik et al., 2005), Torik et al. (Torvik and Smalheiser, 2009) proposes a new algorithm to author name disambiguation. The main difference between the two algorithms is that the new algorithm carries out the agglomerative clustering to disambiguate the author name. This algorithm can be summarized to three steps. In the first step, besides the eight features used by the algorithm in (Torvik et al., 2005), this algorithm takes advantage of additional features: first name and their variants, email addresses, and cross-correlations between specific last names and specific affiliations (Torvik and Smalheiser, 2009). In the second step, this algorithm computes the similarity profiles the same as (Torvik et al., 2005), although the specific way of estimating the $\Pr(M)$ and
Pr(\(x|\mathcal{M}\))/(\(x|\mathcal{N}\)) are a little different. In the third step, after the computation of the pairwise probabilities between every two papers, this algorithm uses an agglomerative clustering method to disambiguate the author names. This agglomerative clustering method first put all the papers in singleton clusters. Then it merges the pairs of clusters with the largest average match probability. Specifically this clustering method iteratively tries to find the two clusters \(c_1\) and \(c_2\) according to the following equation:

\[
\text{argmax}_{c_1, c_2} \prod_{i \in c_1, j \in c_2} \left( \frac{p_{ij}/(1 - p_{ij})}{n(c_1)n(c_2)} \right)
\]

\(p_{ij}\) means the pairwise probability between paper \(i\) in cluster \(c_1\) and paper \(j\) in cluster \(c_2\), which is computed in the first two steps. \(n(c_1)\) and \(n(c_2)\) mean the number of papers in \(c_1\) and \(c_2\). Finally, this algorithm terminates the iteration until the largest pairwise probability is less than 0.5 and outputs the clusters that will represent different individuals.

Like the algorithm in (Torvik et al., 2005), this algorithm has the problem that it is only effective to the digital library with so much information recorded for each paper like the MEDLINE. But for many other digital libraries without so much information, this paper is not suitable for them. Besides the advantage of the algorithm in (Torvik et al., 2005), one of the advantages of this algorithm is that it successfully utilizes a specific clustering method to further improve the author disambiguation performance.
Summary

We have described eight supervised learning algorithms. These supervised methods include naïve bayes model, SVM (Han et al., 2004; Huang et al., 2006), random forests (Treeratpituk and Giles, 2009), probabilistic methods (Torvik and Smalheiser, 2009; Torvik et al., 2005), and an error-driven method (Culotta et al., 2007). Three algorithms take advantage of the clustering method (Culotta et al., 2007; Han et al., 2003a; Torvik and Smalheiser, 2009), but they use different supervised learning methods to estimate the distances of the elements in clusters. We will introduce more summary in section 2.2.4.

2.2.4 Summary

In Table 2, we will summarize all the sixteen algorithms we discussed in chapter 2 in terms of their publication date, the problem they addresses, their input and output, whether they use the clustering method or not, how they compute the clustering distance and whether they need the preset classes.

In Table 2, we find that 13 out of 16 algorithms can address both the polysemic and synonym problem. Also, recent years (after 2006), all the algorithms can address both of the problems. So we can predict that this is the trend to consider the two problems together. Furthermore, we can see the evolution of the algorithms for author name disambiguation problem. In 2003, the researchers use the traditional methods such as the naïve bayes model and K-means clustering.
to disambiguate the papers (Han et al., 2003a). Gradually, the researchers take advantage of the SVM model, a model that is widely used in various areas during recent years, to improve the performance (Han et al., 2004). After noticing the advantage of the online SVM model
<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Year</th>
<th>Problem</th>
<th>input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Citation Labeling</td>
<td>2005</td>
<td>Polyseme</td>
<td>Lists of citations with false citation</td>
<td>Lists of citations without false citation</td>
</tr>
<tr>
<td>Search Engine Driven</td>
<td>2006</td>
<td>Polyseme</td>
<td>List of papers</td>
<td>Clusters of papers</td>
</tr>
<tr>
<td>Block-based</td>
<td>2005</td>
<td>Synonym</td>
<td>Two lists of author names</td>
<td>Clusters of author names</td>
</tr>
<tr>
<td>K-way Spectral</td>
<td>2005</td>
<td>Both</td>
<td>List of papers</td>
<td>Clusters of papers</td>
</tr>
<tr>
<td>Topic-based</td>
<td>2007</td>
<td>Both</td>
<td>List of papers</td>
<td>Clusters of papers</td>
</tr>
<tr>
<td>Heuristic-based</td>
<td>2007</td>
<td>Both</td>
<td>List of papers</td>
<td>Clusters of papers</td>
</tr>
<tr>
<td>Coauthor-based</td>
<td>2009</td>
<td>Both</td>
<td>List of author names</td>
<td>Clusters of author names</td>
</tr>
<tr>
<td>Web-based</td>
<td>2009</td>
<td>Both</td>
<td>List of papers</td>
<td>Clusters of papers</td>
</tr>
<tr>
<td>K-means</td>
<td>2003</td>
<td>Both</td>
<td>List of papers</td>
<td>Cluster of papers</td>
</tr>
<tr>
<td>Naïve Bayes</td>
<td>2004</td>
<td>Both</td>
<td>Canonical name database and a paper</td>
<td>Which canonic name this paper belong to</td>
</tr>
<tr>
<td>SVM</td>
<td>2004</td>
<td>Both</td>
<td>Canonical name database and a paper</td>
<td>Which canonic name this paper belong to</td>
</tr>
<tr>
<td>DBSCAN</td>
<td>2006</td>
<td>Both</td>
<td>List of papers</td>
<td>Clusters of papers</td>
</tr>
<tr>
<td>Error-driven</td>
<td>2007</td>
<td>Both</td>
<td>List of papers</td>
<td>Partition of the papers</td>
</tr>
<tr>
<td>Random Forest</td>
<td>2009</td>
<td>Both</td>
<td>Two papers</td>
<td>Whether they are written by the same author</td>
</tr>
<tr>
<td>Probabilistic Similarity</td>
<td>2005</td>
<td>Both</td>
<td>Two papers</td>
<td>Whether they are written by the same author</td>
</tr>
<tr>
<td>Metric</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Probabilistic Similarity</td>
<td>2009</td>
<td>Both</td>
<td>List of papers</td>
<td>Clusters of author names</td>
</tr>
<tr>
<td>Metric with Clustering</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 2: Summary of the algorithms (continue)

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Clustering</th>
<th>Clustering distance</th>
<th>K</th>
</tr>
</thead>
<tbody>
<tr>
<td>Citation Labeling</td>
<td>No</td>
<td>None</td>
<td>No</td>
</tr>
<tr>
<td>Search engine Driven</td>
<td>Yes</td>
<td>IHF</td>
<td>Yes</td>
</tr>
<tr>
<td>Block-based</td>
<td>No</td>
<td>None</td>
<td>Yes</td>
</tr>
<tr>
<td>K-way Spectral</td>
<td>Yes</td>
<td>Eigen-decomposition</td>
<td>Yes</td>
</tr>
<tr>
<td>Topic-based</td>
<td>Yes</td>
<td>Topic model</td>
<td>Yes</td>
</tr>
<tr>
<td>Heuristic-based</td>
<td>Yes</td>
<td>Heuristic</td>
<td>No</td>
</tr>
<tr>
<td>Coauthor-based</td>
<td>Yes</td>
<td>Coauthors</td>
<td>No</td>
</tr>
<tr>
<td>Web-based</td>
<td>Yes</td>
<td>IHF</td>
<td>No</td>
</tr>
<tr>
<td>K-means</td>
<td>Yes</td>
<td>Naïve bayes</td>
<td>Yes</td>
</tr>
<tr>
<td>Naïve Bayes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>SVM</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DBSCAN</td>
<td>Yes</td>
<td>Online SVM</td>
<td>No</td>
</tr>
<tr>
<td>Error-driven</td>
<td>No</td>
<td>None</td>
<td>No</td>
</tr>
<tr>
<td>Random Forest</td>
<td>No</td>
<td>None</td>
<td>No</td>
</tr>
<tr>
<td>Probabilistic Similarity Metric</td>
<td>No</td>
<td>None</td>
<td>No</td>
</tr>
<tr>
<td>Probabilistic Similarity Metric with Clustering</td>
<td>Yes</td>
<td>Probabilistic metric</td>
<td>No</td>
</tr>
</tbody>
</table>

(Bordes et al., 2005), online SVM is also utilized by the researchers (Huang et al., 2006). Random forest (Breiman, 2001) is a relatively new model compared with naïve bayes, K-means (Hartigan and Wong, 1979) and SVM (Vapnik, 1995). Random forest has recently been successfully applied to classification problems and gains the comparable performance with the top classifier SVM (Breiman, 2001), so it is used to author name disambiguation problems in (Torvik et al., 2005). Based on this trend, we can predict that when a new classification algorithm is proved to be successful in other areas, it is worth trying it on the
author name disambiguation problem. Both polyseme and synonym problem can be viewed as the mapping between multiple entities and one entities. Like the SVM algorithm described in (Huang et al., 2006), it considers author disambiguation problem as classification problem.

We have summed up the input and output of these algorithms in the Table 2. As discussed above, for citation labeling algorithm (Lee et al., 2005) and search engine driven algorithm (Tan et al., 2006), although they deal with the same problem their representations of the polyseme problem are different. The disadvantage for the representation in citation labeling algorithm is that the disambiguation algorithm can only differentiate one author name with the others. In other words, it is just binary classification. The representation of the polyseme problem in the block-based algorithm (Lee et al., 2005) assumes that there are two lists of author names so the input of the block-based algorithm is two lists of author names. This representation is similar to that in the naïve bayes algorithm (Han et al., 2004) and SVM algorithm (Han et al., 2004). The X list in the block-based algorithm is the variant of the canonical name database in naïve bayes algorithm (Han et al., 2004) and SVM algorithm (Han et al., 2004). From the input and output of the random forests algorithm (Treeratpituk and Giles, 2009) and probabilistic similarity metric algorithm (Torvik et al., 2005), we can see that both the two algorithms supply one way of computing the pairwise distance between two author names so that they can be further combined with the clustering algorithms. For example, the probabilistic similarity metric with clustering algorithm (Torvik and Smalheiser,
(2009) is the extensive work of the probabilistic similarity metric algorithm (Torvik et al., 2005). For the rest of the algorithms, their inputs are lists of papers and outputs are clusters of papers. Using the clustering algorithm to address the author name disambiguation problem is the trend.

We also summarize all the algorithms using the clustering methods in Table 2. We find that eight out of the thirteen algorithms which deal with both polyseme and polyseme problems have used the clustering methods. The advantage for the clustering method is obvious: first, author name disambiguation algorithms should find the mapping between multiple entities (canonical authors) and one entity (ambiguous author name). Second, in most cases, we do not know how many candidate canonical authors exist in the ambiguous author names. The clustering methods can generate the clusters without the preset number of classes.

Table 2 has a column named “K”. K is the preset number of the classes. For the classification algorithm such as SVM algorithm (Han et al., 2004) and naïve bayes algorithm (Han et al., 2004), they need to preset the K number so these kinds of algorithm are not suitable for handling the ambiguous author name with new authors. For the k-way spectral algorithm (Han et al., 2005), topic-based algorithm (Song et al., 2007) and k-means algorithm (Han et al., 2003a), although they use the clustering method they still need to preset the K number, which can influence their performance over disambiguating author names with new authors.
Hence, the possible future work for these algorithms is how to estimate the K value. We also summarize the clustering distance method. There are eight clustering distance methods from the simple heuristic method (Cota et al., 2007) to the more advanced SVM method (Han et al., 2004). It is interesting to apply more clustering distance method to the author name disambiguation problem in future.

2.3 Evaluation Datasets and Metrics

In this section, we will summarize the widely used evaluation datasets, from various digital libraries, in the research of the author name disambiguation problem. We will also introduce the evaluation metrics for author name disambiguation algorithms.

2.3.1 Dataset

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Papers</th>
</tr>
</thead>
<tbody>
<tr>
<td>MEDLINE</td>
<td>(Lee et al., 2005; Torvik and Smalheiser, 2009; Torvik et al., 2005; Treeratpituk and Giles, 2009)</td>
</tr>
<tr>
<td>DBLP</td>
<td>(Culotta et al., 2007; Han et al., 2005; Han et al., 2004; Hartigan and Wong, 1979; Lee et al., 2005; Pereira et al., 2009; Treeratpituk and Giles, 2009)</td>
</tr>
<tr>
<td>CiteSeer</td>
<td>(Huang et al., 2006; Song et al., 2007)</td>
</tr>
<tr>
<td>A Korean Dataset</td>
<td>(Kang et al., 2009)</td>
</tr>
<tr>
<td>e-Print</td>
<td>(Lee et al., 2005)</td>
</tr>
<tr>
<td>EconPapers</td>
<td>(Lee et al., 2005)</td>
</tr>
<tr>
<td>Manually Constructed Small Dataset</td>
<td>(Han et al., 2003a; Tan et al., 2006)</td>
</tr>
</tbody>
</table>
In Table 3, we summarize the datasets used for evaluation in the algorithms discussed in chapter 2. We find that MELINE, DBLP and CiteSeer are the most frequently used datasets. The three datasets all contain large-scale publications and are easily obtained online so they are good datasets for us to use in the future. Therefore, we will introduce them as sub-sections below. For the other datasets, we will introduce them briefly in the subsection of summary.

**MELINE**

We will first introduce MEDLINE. This introduction is mainly cited from (Torvik and Smalheiser, 2009). MEDLINE (Torvik and Smalheiser, 2009) is a digital library for the area of biology and medicine. Its articles are from the important journals of these areas. For the whole dataset of 2009 version, it includes more than 17 million papers with their metadata. This dataset is given to the researchers to download with a XML file for free.

Table 4: Example of metadata in three MEDLINE papers, cited from (Treeratpituk and Giles, 2009)

<table>
<thead>
<tr>
<th>ArticleTitle</th>
<th>Affiliation</th>
<th>Authors</th>
<th>JournalTitle</th>
<th>PubDate</th>
<th>MeshHeading</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>The cell biology of blastocyst development.</td>
<td>Dept of Medical Biochemistry, University of Calgary Health Science Center, Alberta</td>
<td>Watson, AJ</td>
<td>Molecular reproduction and development [ENG]</td>
<td>1992 Dec</td>
</tr>
</tbody>
</table>
For each paper in the MEDLINE, MEDLINE will record its paper title, affiliation of the authors, coauthors, journal title, publication date and medical subject heading (MeshHeading) as illustrated in Table 4.

Torik et al. (Torvik and Smalheiser, 2009) has drawn a figure to show the distribution of the author names in MEDLINE:

![Histogram of name counts in MEDLINE](image)

Figure 12: Histogram of name counts in MEDLINE, cited from (Torvik and Smalheiser, 2009)

In Figure 12, they count the unique author name based on the last name and first initial. We can see that while most of the name occurs only a few times there are still names that are associated with a large number of papers thus quite ambiguous. “J. Lee” is the author name for 15,980 papers. From this histogram, we can also know the importance of addressing the author name disambiguation problem.
Digital Bibliography & Library Project (DBLP) Dataset

DBLP (Ley, 2009) is a website aiming at supplying the computer science bibliography (http://dblp.uni-trier.de/). This website is hosted in Universität Trier in German. Beginning from the end of 1993, this website has indexed more than 1.4 million publications until July 2010. These publications are from the most important computer science conferences, journals and books. The dataset including the content of DBLP is available for the researchers to download through providing a XML file.

Figure 13: A DBLP webpage for a specific author

Figure 13 illustrates one typical DBLP webpage for a specific author “Wang-Chien Lee”. In this webpage, it lists all papers written by “Wang-Chien Lee” indexed by publication date.
CiteSeer and CiteSeer\(^x\) Dataset

CiteSeer ([http://citeseer.ist.psu.edu](http://citeseer.ist.psu.edu)) was a digital library for the academic papers primarily from the area of computer and information science. It also provided the search engine service for users to search in the digital library. CiteSeer was hosted in Pennsylvania State University in United States. Now it is replaced by a new website CiteSeer\(^x\) ([http://citeseerx.ist.psu.edu](http://citeseerx.ist.psu.edu)). The new CiteSeer\(^x\) digital library indexes more than 1.6 million documents with 31 million citations until July 2010. Both CiteSeer and CiteSeer\(^x\) provide their metadata, databases and datasets of documents for researchers to download. Figure 14 illustrates the results when search “support vector machine” in CiteSeer\(^x\):

![Figure 14: Search “support vector machine” in CiteSeer\(^x\)](image)
Table 5: Summary of MEDLINE, DBLP and CiteSeerX

<table>
<thead>
<tr>
<th></th>
<th>MEDLINE</th>
<th>DBLP</th>
<th>CiteSeerX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of citations indexed</td>
<td>17 million</td>
<td>1.4 million</td>
<td>31 million</td>
</tr>
<tr>
<td>(July, 2010)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Free</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Easy to download</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Metadata Construction</td>
<td>Manually</td>
<td>Combination of</td>
<td>Automatically</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Automatically</td>
<td>Manually</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5 summarizes MEDLINE, DBLP and CiteSeerX discussed above. MEDLINE, DBLP and CiteSeer are all good datasets to evaluate the author name disambiguation algorithms. First, they include large-scale publications and citations. Second they are free. Third, it is easy to download from the internet. Specifically, for most of the publications, MEDLINE provides metadata such as the affiliation and medical subjective heading as the fields in its XML file (Torvik and Smallheiser, 2009). Researchers can use them directly by parsing from the XML file without extracting from the papers themselves. Also, the medical subjective heading can be viewed as the topic of the papers so it is useful for algorithms concerning the topic information. Furthermore, the metadata of MEDLINE is constructed manually. Therefore, its error rate of the spellings of the author names should be low (Treeratpituk and Giles, 2009). For DBLP, the relative advantage is that it organizes the papers by their authors so it can be used to build the canonical name database as described in (Han et al., 2004). Also, DBLP papers are partly from the publication webpages of the authors’ personal website (Ley, 2009).
which guarantees their correctness. For CiteSeer, the metadata are automatically extracted so it is more challenge for the author disambiguation algorithms. For the Korean dataset used by Kang et al. (Kang et al., 2009), it is a dataset consisting of 8675 papers published in the IT-related conferences in Korean from 1999 to 2006. Besides DBLP and MEDLINE, Lee et al. (Lee et al., 2005) use the e-Print and EconPapers datasets. According to their description, E-Print dataset is a dataset in the Physics domain with 156,627 citations and EconPapers is a dataset in the Economics domain with 20,486 citations. Compared with the three datasets, first, they are not as large as the MEDLINE, DBLP and CiteSeerX; second, it is not easily to download directly online based on our knowledge.

2.3.2 Evaluation Metrics

In this section, we will introduce the evaluation metrics used for evaluating the algorithms described in the chapter 2.
Lee et al. (Lee et al., 2005) use the percentage/rank ratio to measure the performance of their algorithm over the mixed citation problem defined by them. We use the results of their algorithm to illustrate this metric in Figure 15.

In order to solve the mixed citation problem defined by them (check the description in chapter 2 for details), the author disambiguation algorithm should find the false citation among the true citations. This metric measures how much percentage of the false citations are ranked in the bottom 10%, 20%, etc (Lee et al., 2005). For example, there are 100 authors and each author has a list of 99 citations written by himself. For each list of the 99 citations, we insert one false citation to it. This author disambiguation algorithm ranks the 100 citations (including one false citation) in each list and gets the rank of the false citation in each list. If
64 false citations are ranked in the bottom 10% of their lists, then we can get the percentage/rank ratio: 64%/0.1. As illustrated in Figure 15, the sampling-based algorithms are better than the baseline algorithm.

Accuracy for top-k Candidate

Lee et. al (Lee et al., 2005) use the accuracy for top-k candidates to measure the performance of their algorithm over the split citation problem defined by them. In their definition of the split citation problem (check the description in chapter 2 for details), there are two list X and Y. If the top-k candidates of the Y lists includes the name variant of one name \( x \) in list X, this metric defines that the result of this algorithm matches over this name \( x \). If there are \( n \) names in list X, then the accuracy is defined as:

\[
\text{Accuracy}(k) = \frac{\text{number of matches over the } n \text{ names}}{n}
\]

Figure 16 is one example for the results based on this metric. It is the results of seven different algorithms over four datasets when \( k \) is equal to 5.
Accuracy based on Confusion Matrix

Many papers (Han et al., 2003a; Han et al., 2005; Han et al., 2004; Tan et al., 2006; Treeratpituk and Giles, 2009) use the accuracy based on confusion matrix to measure their algorithms. Table 6 is one example of the confusion matrix. The number in A[i,j] means the number of author A_i being predicted as author A_j (Han et al., 2005). The accuracy is calculated as:

\[
\text{Accuracy} = \frac{\sum_i A[i, i]}{\sum_{ij} A[i, j]}
\]

Table 6: Example of confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>A_1</th>
<th>A_2</th>
<th>A_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A_1</td>
<td>50</td>
<td>20</td>
<td>30</td>
</tr>
<tr>
<td>A_2</td>
<td>30</td>
<td>50</td>
<td>20</td>
</tr>
<tr>
<td>A_3</td>
<td>20</td>
<td>30</td>
<td>50</td>
</tr>
</tbody>
</table>

The accuracy for Table 6 is 0.5. This metric is suitable for measuring the accuracy of the
author disambiguation algorithm over one ambiguous author name. For example, Figure 17 is the result based on this metric. We can see that it computes the result for each ambiguous author names. For algorithms considering the author name disambiguation problem as the classification problems, this metric is the same as the classification accuracy.

Figure 17: Example of results using “accuracy based on confusion matrix” (cited from (Han et al., 2005))

**Metrics based on Matching Matrix**

**Table 7: Matching matrix, cited from (Kang et al., 2009)**

<table>
<thead>
<tr>
<th>Machine-generated clusters (M)</th>
<th>Gold standard clusters (G)</th>
<th>Match</th>
<th>Mismatch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Match</td>
<td>a</td>
<td>b</td>
<td></td>
</tr>
<tr>
<td>Mismatch</td>
<td>c</td>
<td>d</td>
<td></td>
</tr>
</tbody>
</table>

Kang et al. (Kang et al., 2009) uses the metrics based on matching matrix as shown in Table 7. There are two sets of clusters M and G. M is generated automatically and G is generated manually called golden standard clusters. The metrics based on the matching matrix are:

\[
\text{Precision} = \frac{a}{a + b} \\
\text{Recall} = \frac{a}{a + c}
\]
\[ F1 = 2 \times \text{Precision} \times \text{Recall}/(\text{Precision} + \text{Recall}) \]

\[ \text{Accuracy} = (a + d)/(a + b + c + d) \]

\[ \text{Over- clustering error} = b/(a + b + c + d) \]

\[ \text{Under- clustering error} = c/(a + b + c + d) \]

In the matching matrix, \(a\) represents the number of pairs of author names partitioned to the same clusters both in M and G; \(b\) represents the number of pairs of author names partitioned to the same cluster in M but not in G; \(c\) is the number of pairs of author names which are placed to the same cluster in G but to different clusters in M; \(d\) is the number of pairs of author names which are assigned to different clusters both in M and G.

**Pair-level Pairwise F1 and Cluster-level Pairwise F1**

Pair-level Pairwise F1 (F1P) and Cluster-level Pairwise F1 (F1C) are used by Song et al. (Song et al., 2007) and Huang et al. (Huang et al., 2006). F1P is based on the pairwise precision \(pp\) and pairwise recall \(pr\). Pairwise precision is computed as the fraction of the co-referent pairs in the same cluster. Pairwise recall is computed as the fraction of the co-referent pairs being put in the same cluster (Huang et al., 2006). F1P is the harmonic mean of \(pp\) and \(pr\). F1C is based on the cluster precision \(cp\) and cluster recall \(cr\). \(cp\) and \(cr\) are computed as:

\[ cp = \frac{\text{number of totally correct clusters}}{\text{number of clusters retrieved by this algorithm}} \]
F1C is the harmonic mean of $cp$ and $cr$.

Average Cluster Purity, Average Author Purity and K metrics

Average cluster purity (ACP), average author purity (AAP) and K metrics are used by many papers including (Cota et al., 2007) and (Pereira et al., 2009). They can be defined as (the same as (Cota et al., 2007) and (Pereira et al., 2009)): 

\[
ACP = \frac{1}{N} \sum_{i=1}^{q} \sum_{j=1}^{R} \frac{n_{ij}^2}{n_i}
\]

\[
AAP = \frac{1}{N} \sum_{j=1}^{R} \sum_{i=1}^{q} \frac{n_{ij}^2}{n_j}
\]

\[
K = \sqrt{ACP \cdot AAP}
\]

Where:

N: the number of citation records in the test dataset

R: number of reference clusters (manually generated)
q: number of clusters automatically generated

n_{ij}: number of elements of cluster i belonging to cluster j

n_i: number of elements of cluster i

2.3.2.1 Summary

Table 8: Summary of the Evaluation Metrics

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Papers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage/Rank ratio</td>
<td>(Lee et al., 2005)</td>
</tr>
<tr>
<td>Accuracy for top-k candidate</td>
<td>(Lee et al., 2005)</td>
</tr>
<tr>
<td>Accuracy based on Confusion Matrix</td>
<td>(Han et al., 2003a; Han et al., 2005; Han et al., 2004; Tan et al., 2006; Treeratpituk and Giles, 2009)</td>
</tr>
<tr>
<td>Metrics based on matching matrix</td>
<td>(Kang et al., 2009)</td>
</tr>
<tr>
<td>F1P, F1C</td>
<td>(Culotta et al., 2007; Huang et al., 2006; Pereira et al., 2009; Song et al., 2007)</td>
</tr>
<tr>
<td>ACP, AAP, K</td>
<td>(Cota et al., 2007; Pereira et al., 2009)</td>
</tr>
</tbody>
</table>

In Table 8, we summarize all the metrics described above. We find that the accuracy based on the confusion matrix is used the most frequently in these papers. It has been utilized in five papers (Han et al., 2003a; Han et al., 2005; Han et al., 2004; Tan et al., 2006; Treeratpituk and Giles, 2009). In (Han et al., 2003a; Han et al., 2005; Han et al., 2004), the output of the algorithms in these papers are clusters of papers. They all make an assumption that they already know the number of the clusters their clustering algorithms need to construct, e.g., the k number in the K-means algorithm (Han et al., 2003a). So the accuracy based on confusion matrix is suitable for evaluating these kinds of algorithms. In (Tan et al., 2006) and (Treeratpituk and Giles, 2009), the algorithms in the two papers view the author name
disambiguation problem as the classification problem. Hence, they use the classification accuracy to measure the performance of their algorithms. We notice that the accuracy based on the confusion matrix is equal to the classification accuracy. So accuracy based on the confusion matrix is also appropriate to measure the algorithms viewing the author name disambiguation problem as the classification problem.

For AAP, ACP and K metrics, we can see that AAP evaluates the purity of the generated clusters with respect to the reference clusters manually generated, i.e., whether the generated clusters include only citations belonging to the reference clusters (Pereira et al., 2009). ACP evaluates the level of splitting of one author into several clusters, i.e., how fragment the generated clusters are (Pereira et al., 2009). K metrics is the geometric means between ACP and AAP. All the three metrics’ ranges are between 0 and 1. The bigger the values are, the better the algorithm is. The three metrics are suitable for the algorithms whose outputs are the clustering. However, unlike the accuracy based on the confusion matrix, the three metrics still work even if the number of golden standard clusters is different from that of the automatically generated clusters.

For F1P, it is same as the F1 score based on the matching matrix in (Kang et al., 2009). Correspondingly, the pairwise precision and pairwise recall consisting of F1P are identical to precision and recall based on the matching matrix in (Kang et al., 2009). F1P and F1C are
suitable for measuring the clustering algorithms. F1P and F1C can also be used along with the K metric. For example, in (Pereira et al., 2009), Pereira et al. use the F1P, F1C and K to measure their algorithms.

For the metrics based on the matching matrix, the precision, recall and F1 are the same as the pairwise precision, pairwise recall and F1P as discussed above. For percentage/Rank ratio, this metric is only suitable for the algorithms whose input includes false citation and whose output is a ranked list. This metric is not appropriate for the algorithms whose outputs are clusters. For accuracy for top-k candidate, this metric is not so strict. For example, if the top 5 candidates in the list Y include a false name variant, this accuracy will still consider it as match. This metric is also not appropriate for the clustering algorithm.

### 2.3.3 Summary

Based on the summarization of evaluation datasets, MEDLINE, DBLP and CiteSeer are good evaluation datasets to use in the future because of their advantages discussed above. We can further choose the specific dataset from them based on their relative characteristics. However, if we intend to evaluate the ability of the disambiguation algorithm to handle the datasets from different areas, we can make use of more datasets discussed above. In addition, we think it should be useful for this research area to have a standard benchmark dataset like the LETOR (Qin et al., 2010) benchmark in the learning to rank area.
For the evaluation metrics, we introduce six different metrics. The percentage/rank ratio and accuracy for top-k candidate are only suitable for their specific algorithms. For the remaining four metrics, they are all good metrics to evaluate the algorithms although they may have some overlaps between each other. Currently, the use of both the “F1P and F1C” and “ACP, AAP and K” metrics in (Pereira et al., 2009) is the most comprehensive evaluation method among all the papers.

2.4 Summary on Author Name Disambiguation in Digital Library

In this chapter, we review the area of the author name disambiguation in digital library. This chapter mainly focuses on the two research challenges in this literature: the algorithms themselves and the evaluation methods. We present a comprehensive and synthetically summarization of the author name disambiguation algorithms. We survey these algorithms according to the problems they can solve. Furthermore, among the thirteen algorithms dealing with both polyseme and synonym, we find that five of them use the unsupervised methods and eight of them take advantage of the supervised methods. In addition, we compare sixteen author names disambiguation algorithms with each other in terms of eight perspectives: their publication date, the problems they addresses, their input, their output, whether they use the clustering method or not, how do they compute the clustering distance, whether they need the
preset classes and how they enhance their efficiency and scalability. We also survey the evaluation datasets and metrics used in the chapters. Based on the survey, we recommend the useful evaluation datasets and metrics for researchers to use in the future.
3 Cross-source Document Coreference Using Random Forests

3.1 Introduction to Cross-source Document Coreference

As one of the largest encyclopedic source, World Wide Web has huge amount of information and knowledge worth to exploit. However, the difficulty of utilizing the valuable information of the World Wide Web comes from its unstructured format and the websites containing errors. To address this problem, the knowledge sharing communities like Wikipedia have already organized the information with structured format using the category, and minimize the noise and error by allowing different people all over the world to edit the same pages. Furthermore, using the advanced automated information extraction and knowledge harvesting algorithms, large scale knowledge base and ontology such as DBPedia and freebase is available now to make the representation of the information and knowledge in World Wide Web more organized, precise and concise. These knowledge bases, including Wikipedia, DBPedia and freebase, can be used as canonical sources differentiated from the unstructured and nonprecise web source. Now we have the problem of how to link the entities in the web source to their descriptions in the corresponding canonical source. Cross-source document coreference is the research area trying to tackle this problem. Specifically, cross-source document coreference (CSDC) deals with the problem of disambiguating the entities in documents of one source to
their corresponding identities, if exists, in another source.

Cross-source document coreference is a very important research problem. The success of cross-source document coreference can benefit the following scenarios (take the person entities for example): (1) When you search a person name in one search engine, the search results can be grouped by their identities in the canonical source. (2) When detecting a group of webpages talking about a person without canonical pages in canonical sources, we can automatically create the canonical page for that person. (3) The person who has a canonical page in the canonical source can filter the webpages discussing him. (4) The success of scenario (3) can further help compute the reputations for one specific person. (5) The success of scenario (3) can also help automatically enrich the information in the canonical webpages. The cross-source document coreference itself can help scenario (1) and (4), from the perspective of information retrieval and filtering. Furthermore, the success of scenario (1) and (4) is the base of the success of scenario (2), (3) and (5). Cross-source document coreference algorithms can also benefit the document summarization, information fusion and question answering.

This is a new research direction. In order to conduct cross-source document coreference, there are two steps: constructing the canonical source and disambiguating entities in the documents of other sources to the canonical source. The first step is done by the creators of the Wikipedia,
This chapter focuses on the second step: based upon an existing canonical source, designing algorithm to disambiguate the entities in documents of other sources to their corresponding identities in the canonical source. In the following of this chapter, we assume there are two sources: canonical source and general source. In this chapter, we will utilize the random forests (Breiman, 2001), an supervised machine learning method, to finish this task. Random forests had been extensively used in the other tasks such as author name disambiguation(Treeratpituk and Giles, 2009). It has been proved to be effective and its performance is similar to other machine learning techniques such as Support Vector Machine. Furthermore, random forests algorithm is able to handle large scale of features efficiently and conduct feature selection since it uses different trees to produce variable importance. The contribution of this chapter comes from two perspectives. First, it utilizes a state-of-the-art machine learning algorithm, random forests, to a new research area: cross-source document coreference. Second, this chapter leverages one information extraction tools, OpenCalais, and develops a large number of features, 88 features, to help cross-source document coreference.

This chapter is structured as following. Related work is reviewed in section 3.2. We will present the cross-source document coreference algorithm in section 3.3. In section 3.4, we will describe the evaluation dataset and experiment results. Finally, we will summarize this chapter in the section 3.5.
3.2 Related Work

Cross-source document coreference is similar to cross document coreference (CDC) (Bagga and Baldwin, 1998). CDC targets at the problem of determining whether entities discussed in different documents referring to the same identity (Huang et al., 2010). As a new direction, cross-source document coreference algorithms can benefit from consulting machine learning algorithms, feature construction methods and useful resources used by the cross documents coreference algorithms. The paper written by Bagga and Baldwin (Bagga and Baldwin, 1998) is the one of the pioneer works which addresses the cross document coreference problem and it uses the vector space model to cluster the individuals. Wan et al. (Wan et al., 2005) proposed a cross document coreference system. The disadvantage for this system is that it only extracted the simple pattern: title, organization, email address and etc. Niu et al. (Niu et al., 2004) utilizes the information extraction tools and leverages the Bayesian framework and maximum entropy modeling to help cross document coreference. Baron and Freedman (Baron and Freedman, 2008) also utilizes the information extraction tools to extract the document entities from the text. After the extraction of the same real world entity, they use an agglomerative clustering to disambiguate the co-referent entities. Huang et al. (Huang et al., 2009) tackle the cross document coreference problem by leveraging the profiles of entities which are constructed by using information extraction tools and reconciled by using a within-document coreference module. Specifically, it uses a kernelized soft relational
clustering algorithm to partition the entities to sets of identities. In addition, Huang et al. (Huang et al., 2010) leverage the document-level categories, sub-document level context and extracted entities and relations for the cross document coreference problem. They also train a categorization model based on thousands of ODP categories and a million web documents and use the categorization model to improve cross document coreference performance.

![Diagram](image.png)

Figure 19: Relationship between cross document coreference and cross-source document coreference

However, the two research directions differ substantially: CDC only deals with one data source while CSDC deals with two sources. Technically, given a set of document D discussing individuals with the same name, CDC algorithms tend to make use of the clustering methodology to generate clusters of documents and represent each individual with one cluster. Cross-source document coreference algorithms will adopt the classification techniques due to the existing individuals with the same name in canonical source. The two research directions can be used as two steps to help construct the canonical source and disambiguate the documents. CDC algorithms can be used as the first step to construct the canonical source.
After the construction of canonical source, i.e. individual categories, there is no need to cluster the previous documents again. Instead, we can just classify the new document to the existing categories or set up a new individual category if new individual comes in.

3.3 Methodology

In this section, we will present our methodology for Cross-source Document Coreference. First, we will discuss the random forests algorithm. Then, we will briefly introduce OpenCalais, an information extraction tool, and the feature construction based on this tool.

3.3.1 Random Forests

Random forests is developed by (Breiman, 2001). It is a machine learning algorithm that combines a set of decision trees. Each tree is constructed independently from the training set. In the training set, there are M variables. For each decision tree, we need to decide its variable number which is randomly selected from the M variables. The variable number can be m, where m is less than M. Then we split the trees in the m variables. The important difference of the decision tree in random forests to the ordinary decision tree is that it does not need to be pruned and it is fully grown. The figure below shows the process of constructing a tree in random forests algorithm. The output of random forests is the mode of the decision trees.
The improvement of random forests over decision trees comes from the low bias and low variance. Low bias comes from the growing tree without pruning. The low variance thanks to the aggregation of the features and the random variable selection (Treeratpituk and Giles, 2009). Random forests used in classification can achieve promising results compared with other classification methods (Breiman, 2001).

In the random forests, we can tune the number of trees, i.e. $N$, we need to construct and the number of features, i.e. $m$, in each tree. The two parameters can affect the correlation of trees and the performance of individual tree. The correlation of trees is in inverse proportion to the performance of random forests; meanwhile, the performance of individual tree is in direct proportion to the performance of random forests. Increasing the number $m$ can enhance both the correlation of trees and the performance of individual tree. Thus, a tradeoff has to be made.
according to the correlation of trees and the performance of individual tree through the selecting of an optimal number \( m \). It is suggested that \( m \) can be set to \( \log_2(M + 1) \), which performs well in the experiment (Breiman, 2001).

Random forests algorithm has many advantages to be applied to cross-source document coreference problem. First, it is highly efficient to handle large scale datasets in this problem because the training process of random forests is very fast. Second, even with weak features, we do not need to delete them manually in random forests.

### 3.3.2 OpenCalais

OpenCalais is a web service that automatically creates rich semantic metadata for the content users submit. Using natural language processing (NLP), machine learning and other methods, Calais analyzes the document and finds the entities within it. Calais goes well beyond classic entity identification and returns the facts and events hidden within the text as well (cited from about.opencalais.com). OpenCalais information extraction module currently can handle English, French and Spanish language. It has five output formats: Simple, RDF, Microformat, JSON and N3. Each format can output different content and extraction results. In this chapter, we use the simple format. In table 9, we illustrate the 40 entities which are extracted from the OpenCalais and used in this chapter. The meanings of these entities are self-explained.

Table 9: Entities extracted from OpenCalais and used in this chapter
<table>
<thead>
<tr>
<th>Anniversary</th>
<th>FaxNumber</th>
<th>OperatingSystem</th>
<th>RadioProgram</th>
</tr>
</thead>
<tbody>
<tr>
<td>City</td>
<td>Holiday</td>
<td>Organization</td>
<td>Region</td>
</tr>
<tr>
<td>Company</td>
<td>IndustryTerm</td>
<td>Person</td>
<td>Region</td>
</tr>
<tr>
<td>Continent</td>
<td>MarketIndex</td>
<td>PhoneNumber</td>
<td>SportsEvent</td>
</tr>
<tr>
<td>Country</td>
<td>MedicalCondition</td>
<td>PoliticalEvent</td>
<td>SportsGame</td>
</tr>
<tr>
<td>Currency</td>
<td>MedicalTreatment</td>
<td>Position</td>
<td>SportsLeague</td>
</tr>
<tr>
<td>EmailAddress</td>
<td>Movie</td>
<td>Product</td>
<td>TVShow</td>
</tr>
<tr>
<td>Event</td>
<td>MusicGroup</td>
<td>ProgrammingLanguage</td>
<td>TVStation</td>
</tr>
<tr>
<td>Facility</td>
<td>NaturalFeature</td>
<td>PublishedMedium</td>
<td>URL</td>
</tr>
</tbody>
</table>

### 3.3.3 Problem definition and Features

We will first present the problem definition and then describe how we construct the classification features for random forests. We have canonical source \( C \) including \( N \) individual entities and we can assume that each individual entity can be represented by one document describing it. In the \( N \) individual entities, there are individual entities sharing the same person name. Given general source \( G \), \( G \) includes documents discussing a number of individual entities. The problem is how to link the individual entities in the \( G \) to their corresponding identities in \( C \). We focus on disambiguating the individual entities with the same person name since we can easily distinct individual entities with different person name using a simple blocking function. Take one person name “Paul Collins” for example, there are 12 different entities with the name “Paul Collins” in \( C \) and 100 documents discussing “Paul Collins” in \( G \). For each of the 100 documents in \( G \), Cross-source document coreference algorithm will determine its corresponding entities in \( C \).
One individual entity in $G$ is $E_g$ and one individual entity in $C$ is $E_c$. We can use two documents, $d_g$ and $d_c$, describing the entities to represent them in both sources. In order to determine whether they are the same entity, we construct the similarity profile between the two documents, based on the text of the document and the entities extracted from the information extraction tools. The metadata of $d_g$ and $d_c$ includes 41 attributes: text of the document, TEXT, and the 40 entities extracted from OpenCalais as shown in table 9.

$$d_g = (TEXT_g, ANNIVERSARY_g, CITY_g, COMPANY_g, CONTINENT_g, \ldots, TVSTATION_g, TECHNOLOGY_g, URL_g)$$

$$d_c = (TEXT_c, ANNIVERSARY_c, CITY_c, COMPANY_c, CONTINENT_c, \ldots, TVSTATION_c, TECHNOLOGY_c, URL_c)$$

We develop 88 features for the profile of the two documents based on the 41 metadata. The 88 features can be categorized to five types: TEXT_SIMILARITY, COPERSON_SIMILARITY, LOCATION_SIMILARITY, CATEGORY_SIMILARITY and SINGLEENTITY_SIMILARITY.

**TEXT_SIMILARITY**

Feature 1 is TEXT_TFIDF. It is the similarity between the TEXT of two documents.
\[
\text{TEXT\_TFIDF} = \sum_{t \in \text{TEXT}_g \cap \text{TEXT}_c} \text{TFIDF}(t, \text{TEXT}_g) \times \text{TFIDF}(t, \text{TEXT}_c)
\]

\[
\text{TFIDF}(t, \text{TEXT}) = \log(\text{TF}(t, \text{TEXT}) + 1) \times \log(\text{IDF}(t))
\]

In the above equation, \( t \) means the term. \( \text{IDF}(t) \) means the inverse document frequency of term \( t \) in the corpus. \( \text{TF}(t, \text{TEXT}) \) means the term frequency of \( t \) in \( \text{TEXT} \).

Feature 2 is \( \text{TEXT\_JACCARD} \). It is the Jaccard similarity between the \( \text{TEXT} \) of two documents.

\[
\text{TEXT\_JACCARD} = \frac{|\text{TEXT}_g \cap \text{TEXT}_c|}{|\text{TEXT}_g| \cap |\text{TEXT}_c|}
\]

**COPERSON\_SIMILARITY**

Feature 3 and feature 4 are the \( \text{COPERSON} \) and \( \text{COPERSON\_JACCARD} \). We define \( \text{COPERSON}(d_c, d_g) \) as the number of the coexist person name except the target ambiguous person name in the two documents \( d_c \) and \( d_g \). For example, \( d_c \) has three person name A, B and C and \( d_g \) has three person name A, B and D where A is the target ambiguous name.

Then \( \text{COPERSON}(d_c, d_g) \) is 1.

\[
\text{COPERSON\_JACCARD} = \begin{cases} \frac{\text{COPERSON}(d_c, d_g)}{|\text{PERSON}_g \cap \text{PERSON}_c|} & \text{if } \text{COPERSON}(d_c, d_g) > 0 \\ 0 & \text{otherwise} \end{cases}
\]

**LOCATION\_SIMILARITY**

Feature 5 and feature 6 are the \( \text{LOCATION\_TFIDF} \) and \( \text{LOCATION\_JACCARD} \). First, we find all the entities in table 9 that may consists of location information: Anniversary, City, Company, Continent, Country, Currency, EntertainmentAwardEvent, Facility, NaturalFeature,
Organization, PoliticalEvent, Product, ProvinceOrState, PublishedMedium, RadioProgram, RadioStation, Region, SportsEvent, SportsGame, SportsLeague, TVShow, TVStation. Second, we include the content of all the entities to one entity: LOCATION.

\[
LOCATION\_TFIDF = \sum_{t \in LOCATION_g \cap LOCATION_c} TFIDF(t, LOCATION_g) \times TFIDF(t, LOCATION_c)
\]

\[
LOCATION\_JACCARD = \frac{|LOCATION_g \cap LOCATION_c|}{|LOCATION_g| \cap |LOCATION_c|}
\]

**CATEGORY\_SIMILARITY**

Features from feature 7 to feature 10 describe the category information. We define four new entities. First, ENTERTAINMENT consists of the content of the entities: EntertainmentAwardEvent, Movie, MusicAlbum and MusicGroup in table 9. Second, SPORT consists of the content of the entities: SportsEvent, SportsGame and SportsLeague in table 9. Third, ECONOMYPOLITIC consists of the content of the entities: MarketIndex, PoliticalEvent and Product in table 9. Fourth, IT consists of the content of the entities: OperatingSystem, ProgrammingLanguage and Technology in table 9. We get four features ENTERTAINMENT\_SUM, SPORT\_SUM, ECONOMYPOLITIC\_SUM and IT\_SUM. Take SPORT\_SUM for example, we first compute SPORT\_NUMBER:

\[
SPORT\_NUMBER = SPORTEVENT\_NUMBER + SPORTGAME\_NUMBER + SPORTLEAGUE\_NUMBER
\]

Where SPORTEVENT\_NUMBER is the number of the SPORTEVENT entities extracted by the OpenCalais. SPORTGAME\_NUMBER and SPORTLEAGUE\_NUMBER are computed
in the same way.

\[
\text{SPORT\_SUM} = \begin{cases} 
\text{SPORT\_NUMBER}_g + \text{SPORT\_NUMBER}_c & \text{if } \text{SPORT\_NUMBER}_g > 2 \text{ and } \text{SPORT\_NUMBER}_c > 2 \\
0 & \text{otherwise}
\end{cases}
\]

We compute the ENTERTAINMENT\_SUM, ECONOMY\_POLITIC\_SUM and IT\_SUM in the same way as SPORT\_SUM. The four features indicate the probabilities of documents \(d_c\) and \(d_g\) discussing the same category.

**SINGLEENTITY\_SIMILARITY**

Feature 11 to feature 88 are the similarities of the single entity in table 9. Except the Person entity, we compute the TFIDF and jaccard similarities for all the 39 entities in Table 9 and get 78 features.

### 3.4 Experiments

The dataset we use in this chapter is from the first Web People Search Evaluation Campaign (WePS). This dataset has been widely used to evaluate the cross document coreference algorithm (Huang et al., 2010; Huang et al., 2009). In the training set, it includes the person name from Wikipedia and the list of participants in the European Conference in Digital Libraries. Also, it composes of the names sampled from the US Census (Mann, 2006). These names are either the highly ambiguous names on the web or the names of famous or historical people. There are totally 40 person names in the training set with each name having up to 100 web pages. In the test set, it contains the person names from the Wikipedia, the ACL 2006
participants and US Census. These names are randomly selected. There are totally 30 person names in the test set with each name having up to 100 web pages.

In our experiments, we construct the canonical source and general source from the gold standard of WePS dataset. For each person name, the gold standard clusters the documents according to whether they contain the same individual entity. In our experiments, for each person name, we only consider the clusters, except the discarded cluster, with at least two documents. Then for each cluster, we select the first document, use it to represent the individual entity of this cluster and pick it into the canonical source. The remaining documents in this cluster are chosen to be the general source. Take figure 22 for example, it is the golden standard document for person name “Abby Watkins”. We only use two entities.
with the id 13 and 14 in our experiments. Specifically, documents ranked 99 and 115 are used as the canonical source and the documents whose ranks are 36, 52, 76 and 14 are used to construct the general source. In the experiments, the person entities with the name “Abby Watkins” in documents whose ranks are 36 and 52 are considered to be relevant to the person entities with the name “Abby Watkins” in document ranked 99.

Table 10: Confusion matrix for different models

<table>
<thead>
<tr>
<th>Models</th>
<th>Relevant</th>
<th>Nonrelevant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision Tree</td>
<td>1331</td>
<td>1074</td>
</tr>
<tr>
<td>Naïve Bayes</td>
<td>922</td>
<td>1483</td>
</tr>
<tr>
<td>Bayes Network</td>
<td>1224</td>
<td>1181</td>
</tr>
<tr>
<td>Random Forests</td>
<td>1486</td>
<td>919</td>
</tr>
</tbody>
</table>

Table 11: Disambiguation precision for different models

<table>
<thead>
<tr>
<th>Models</th>
<th>Disambiguation Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision Tree</td>
<td>83.19%</td>
</tr>
<tr>
<td>Naïve Bayes</td>
<td>75.28%</td>
</tr>
<tr>
<td>Bayes Network</td>
<td>73.32%</td>
</tr>
<tr>
<td>Random Forests</td>
<td>85.97%</td>
</tr>
</tbody>
</table>

We compare the random forests model with three different models: DecisionTree, Naïve Bayes and Bayes Network. All the experiment results are the 5-fold cross-validation disambiguation results. All the experiment results are conducted using the WEKA data mining
tool (Hall et al., 2009). The parameters are the default parameters in WEKA. For the random forests, the number of trees is 10 and for each tree we randomly choose 7 features. For the decision tree, we use the J48 trees. Table 10 presents the classification confusion matrix and Table 11 summarizes the disambiguation precision according to the Table 10. From Table 11, we can see that random forests algorithm significantly outperforms Naïve Bayes and Bayes Network by 10.67% and 12.65%. Random forests algorithm also outperforms the ordinary decision tree by 2.88%.

3.5 Summary on Cross-source Document Coreference Using Random Forests

In this chapter, we introduce the cross-source document coreference problem and make use of a state-of-the-art machine learning algorithm, random forests, to this new area. We utilize one information extraction tools, OpenCalais, and developed a large number of features, 88 features, to help cross-source document coreference. We compare the random forests model with three traditional models and the experiment results demonstrate that random forests algorithm outperforms all the other models in the cross-source document coreference problem.
4 Conclusion and Future Work

In sum, this thesis mainly deals with two research problems: author name disambiguation in digital library and cross-source document coreference. We successfully achieve the following contributions:

1) We present a comprehensive summarization of the author name disambiguation algorithms and classify them according to the problem they can solve. We not only summarize the advantages and disadvantages of the algorithms themselves but also compare and analyze them synthetically with the other algorithms.

2) We synthetically summarize and analyze the evaluation datasets and metrics in author name disambiguation area.

3) We leverage one information extraction tool, OpenCalais, and develop a large number of features, 88 features, to help cross-source document coreference.

4) We utilize a state-of-the-art machine learning algorithm, random forests, to this new research area. The experiments results demonstrate that random forests algorithm outperforms three typical traditional models in the cross-source document coreference problem.

In the thesis, we find several important and challenge issues for the author name
disambiguation algorithms. We can conduct research in these issues in the future. First, for
algorithms considering the author name disambiguation problem as classification problem,
how can we estimate and determine the number of the classes? The first case is that: given a
list of citations with an ambiguous author name how can we determine the number of the
authors we need to classify them to? The second case is that: currently an ambiguous author
name has three possible authors in the digital library. When a new paper comes into the digital
library with that ambiguous author name, how can we determine whether that paper belongs
to the existing authors or a new author? Second, how can we deal with the large scale digital
library? We need to compromise the dilemma of accuracy and efficiency while facing the
large scale information. How can we make the algorithms more efficient without affecting
their accuracy too much? We have already learned three methods to make the algorithm more
efficient: sampling (Lee et al., 2005), blocking (Huang et al., 2006; Lee et al., 2005) and
online learning method (Huang et al., 2006). Could we find more methods? Third, we have
already seen so many clustering and classification methods, can we find more clustering and
classification methods to help improve the accuracy of the algorithms. Fourth, how to select
the appropriate features for a specific clustering or classification algorithm is important. We
are interested in seeing a paper discussing the feature selection in author name disambiguation
problem. Fifth, facing so many evaluation datasets, we think it should be useful for this
research area to have a standard benchmark dataset like the LETOR (Qin et al., 2010)
benchmark in the learning to rank area. For the cross-source document coreference problem,
in the future, it will be worth performing the feature selection in the random forests and trying to analyze the effect of different features. It is also helpful to apply more machine learning algorithm to this problem.
References


Cohen W., Ravikumar P., Fienberg S. (2003) A Comparison of String Distance Metrics for Name-matching tasks. In IIWeb Workshop held in conjunction with IJCAI.


