A Brief Survey of Automatic Methods for Author Name Disambiguation

Anderson A. Ferreira[1,2] Marcos André Gonçalves[2] Alberto H. F. Laender[2]

[1]Departamento de Computação
Universidade Federal de Ouro Preto
35400-000 Ouro Preto, Brazil

[2]Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
31270-901 Belo Horizonte, Brazil

{ferreira, mgoncalv, laender}@dcc.ufmg.br

ABSTRACT

Name ambiguity in the context of bibliographic citation records is a hard problem that affects the quality of services and content in digital libraries and similar systems. The challenges of dealing with author name ambiguity have led to a myriad of disambiguation methods. Generally speaking, the proposed methods usually attempt to group citation records of a same author by finding some similarity among them or try to directly assign them to their respective authors. Both approaches may either exploit supervised or unsupervised techniques. In this article, we propose a taxonomy for characterizing the current author name disambiguation methods described in the literature, present a brief survey of the most representative ones and discuss several open challenges.

1. INTRODUCTION

Several scholarly digital libraries (DLs), such as DBLP[1], CiteSeer[2], PubMed[3] and BDBComp[4], provide features and services that facilitate literature research and discovery as well as other types of functionality. Such systems may list millions of bibliographic citation records (here understood as a set of bibliographic attributes such as author and coauthor names, work and publication venue titles of a particular publication[5]) and have become an important source of information for academic communities since they allow the search and discovery of relevant publications in a centralized manner. Studies based on the DL content can also lead to interesting results such as topic coverage, research tendencies, quality and impact of publications of a specific subcommunity or individuals, collaboration patterns in social networks, etc. These types of analysis and information, which are used, for instance, by funding agencies on decisions for grants and for individual's promotions, presuppose *high quality content* [29, 31].

According to Lee et al. [31], the challenges to have high quality content comes from data-entry errors, disparate citation formats, lack of (enforcement of) standards, imperfect citation-gathering software, ambiguous author names, and abbreviations of publication venue titles. Among these challenges, *author name ambiguity* has required a lot of attention from the DL research community due to its inherent difficulty. Specifically, name ambiguity is a problem that occurs when a set of citation records contains ambiguous author names, i.e., the same author may appear under distinct names (synonyms), or distinct authors may have similar names (polysems). This problem may be caused by a number of reasons, including the lack of standards and common practices, and the decentralized generation of content (i.e., by means of automatic harvesting [30]).

To illustrate the problem, Table 1 shows a set of three citations $\{c_1, c_2, c_3\}$ so that each citation has its author names identified by $r_j, 1 \leq j \leq 16$. For each citation c_i, each name r_j is a reference to an author and has a list of attributes associated with it, such as, coauthor names (i.e., the list of references to other authors of the same citation), work title, publication venue title, publication year and so on. Examining Table 1, we see examples of synonyms and polysems, which, as mentioned before, are subproblems of the name ambiguity problem. Author names r_3 and r_{15} are examples of polysems where r_3 refers to "Ajay Gupta" from IBM Research India and r_{15} refers to "Aarti Gupta" from NEC Laboratories America, USA. Author names r_3 and r_7 are examples of synonyms. Both refer to "Ajay Gupta" from IBM Research India.

More formally, the name disambiguation task may be formulated as follows: Let $C = \{c_1, c_2, ..., c_k\}$ be a set of citation records. Each citation record c_i has a list of attributes which includes at least author names, work title and publication venue title. With each attribute in a citation is associated a specific value, which may have

[1]http://dblp.uni-trier.de
[2]http://citeseer.ist.psu.edu
[3]www.ncbi.nlm.nih.gov/pubmed
[4]http://www.lbd.dcc.ufmg.br/bdbcomp
[5]We use the terms "citation" and "citation record" interchangeably.

Table 1: Illustrative Example (Ambiguous Group of A. Gupta)

Citation Id	Citation
c_1	(r_1) S. Godbole, (r_2) I. Bhattacharya, (r_3) A. Gupta, (r_4) A. Verma. Building re-usable dictionary repositories for real-world text mining. CIKM, 2010.
c_2	(r_5) Indrajit Bhattacharya, (r_6) Shantanu Godbole, (r_7) Ajay Gupta, (r_8) Ashish Verma, (r_9) Jeff Achtermann, (r_{10}) Kevin English. Enabling analysts in managed services for CRM analytics. KDD, 2009.
c_3	(r_{11}) T. Nghiem, (r_{12}) S. Sankaranarayanan, (r_{13}) G. E. Fainekos, (r_{14}) F. Ivancic, (r_{15}) A. Gupta, (r_{16}) G. J. Pappas. Monte-carlo techniques for falsification of temporal properties of non-linear hybrid systems. HSCC, 2010.

several components. In case of the *author names* attribute, a component corresponds to the name of a single unique author and is a reference r_j to a real author. In case of the other attributes, a component corresponds to a word/term. The objective of a disambiguation method is to produce a function that is used to partition the set of references to authors $\{r_1, \ldots, r_m\}$ into n sets $\{a_1, \ldots, a_n\}$, so that each partition a_i contains (all and ideally only all) the references to a same author.

To disambiguate the bibliographic citations of a DL, first we may split the set of references to authors into groups of references whose values of the author name attribute are ambiguous. These are called ambiguous groups (i.e., groups of references having the value of the author name attribute with similar names). The ambiguous groups may be obtained by using blocking methods [37] which address scalability issues avoiding the need for comparisons among all references.

The challenges of dealing with name ambiguity in citation records have led to a myriad of disambiguation methods [3, 4, 7, 9, 15, 16, 20, 21, 22, 24, 26, 27, 33, 35, 38, 40, 41, 42, 43, 44, 46, 49]. One such a challenge is that, usually, only a minimum set of attributes is available to work with (in most case only author names and publication and venue titles). In any case, existing disambiguation methods usually attempt to either group citation records of the same author using some type of similarity between them or try to directly assign the citation records to their respective authors.

An early survey with some preliminary disambiguation methods is found in [28]. In that work, Klass classifies the methods into supervised or unsupervised ones and describes some methods published until 2006. However, the area has been very prolific in the last years, with many methods recently proposed. In this article, we propose a new taxonomy for characterizing the current methods for disambiguating author names and present a brief survey of some of the most representative ones.

This article is organized as follows. Section 2 proposes our taxonomy for characterizing the author name disambiguation methods. Section 3 presents an overview of representative author name disambiguation methods. A summary of characteristics of the methods is presented in Section 4. Section 5 discusses some open challenges in the author name disambiguation task. Finally, Section 6 presents our conclusions.

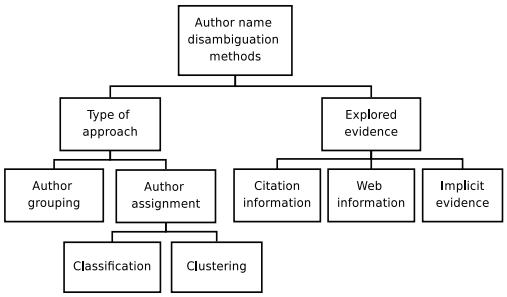

Figure 1: Proposed taxonomy

2. A TAXONOMY FOR AUTHOR NAME DISAMBIGUATION METHODS

This section presents a hierarchical taxonomy for grouping the most representative automatic author name disambiguation methods found in the literature. The proposed taxonomy is shown in Figure 2. The methods may be classified according to the main type of exploited approach: *author grouping* [4, 7, 9, 15, 16, 22, 24, 26, 27, 36, 38, 41, 42, 44, 45, 46, 35, 49], which tries to group the references to the same author using some type of similarity among reference attributes, or *author assignment* [3, 16, 20, 21, 43], which aims at directly assigning the references to their respective authors. Alternatively, the methods may be grouped according to the evidence explored in the disambiguation task: the citation attributes (only), Web information, or implicit data that can be extracted from the available information.

We should notice that in this survey we cover only automatic methods. Other types of method, such as manual assignment by librarians [39] or collaborative efforts[6], heavily rely on human efforts, which prevent them from being used in massive name disambiguation tasks. For this reason, they are not addressed in this article. There are also efforts to establish a unique identification to each author, such as the use of an Open Researcher Contributor Identification[7] (ORCID), but these are also not covered here.

Since the name disambiguation problem is not restricted to a single context, it is also worth noticing that several other name disambiguation methods, which exploit distinct pieces of evidence or are targeted at other applications (i.e., name disambiguation in Web search

[6]http://meta.wikimedia.org/wiki/WikiAuthors
[7]http://www.orcid.org

results), have been described in the literature [2, 12, 18, 48, 50]. However, a discussion of these methods is outside the scope of this article.

Finally, we should stress that the categories in our taxonomy are not completely disjoint. For instance, there are methods that use two or more types of evidence or mix approaches. In the next subsections, we detail our proposed taxonomy.

2.1 Type of Approach

As said before, one way to organize the several existing author name disambiguation methods is according to the type of approach they exploit. We elaborate this distinction further in the discussion below.

2.1.1 Author Grouping Methods

Author grouping methods apply a similarity function to the attributes of the references to authors (or group of references) to decide whether to group the corresponding references using a clustering technique. The similarity function may be predefined (based on existing ones and depending on the type of the attribute) [4, 7, 22, 36, 41], learned using a supervised machine learning technique [9, 24, 44, 45, 46], or extracted from the relationships among authors and coauthors, usually represented as a graph [15, 33, 35]. This similarity function is then used along with some clustering technique to group references of a same author, trying to maximize intra and minimize inter-cluster similarities, respectively.

Defining a Similarity Function

Here, a similarity function is responsible for determining how similar two references (or groups of references) to authors are. The goal is to obtain a function that returns high similarity values for references to the same author and returns low similarity values for references to different authors. Moreover, it is desirable that the similarity function be transitive. More specifically, let c_1, c_2 and c_3 be three citations, if c_1 and c_2 are very similar (according to the function) and c_2 and c_3 are also very similar, then c_1 and c_3 should have high similarity according to our function. Next, we discuss the ways to determine this similarity function.

Using Predefined Functions. This class of methods has a specific predefined similarity function $\mathcal{S}$ embedded in their algorithms to check whether two references or groups of references refer to the same author. Examples of such function $\mathcal{S}$ include [6]: the Levenshtein distance, Jaccard coefficient, cosine similarity, soft-TFIDF and others [6], applied to elements of the reference attributes. Ad-hoc combinations of such functions have also been used (e.g., in [4, 41])

These methods do not need any type of supervision in terms of training data but their similarity functions are usually tuned to disambiguate a specific collection of citation records. For different collections, a new tuning procedure may be required. Finally, not all the functions used in these methods are transitive by nature.

Learning a Similarity Function. Learning a specific similarity function usually produces better results, since these learned functions are directly optimized for the disambiguation problem at hand. To learn the similarity function, the disambiguation methods receive a set $\{s_{ij}\}$ of pairs of references (the training data) along a special variable that informs whether these two corresponding references refer to the same author. The pair of references, r_i and $r_j \in R$ (the set of references) are usually represented by a similarity vector $\vec{s_{ij}}$. Each similarity vector $\vec{s_{ij}}$ is composed of a set $\mathcal{F}$ of q features $\{f_1, f_2, \ldots, f_q\}$. Each feature f_p of these vectors represents a comparison between attributes $r_i.A_l$ and $r_j.A_l$ of two references, r_i and r_j.

The value of each feature is usually defined using other functions, such as Levenshtein distance, Jaccard coefficient, Jaro-Winkler, cosine similarity, soft-TFIDF, euclidean distance, etc., or some specific heuristic, such as the number of terms or coauthor names in common, or special values such as the initial of the first name along with the last names, etc.

The training data is then used to produce a similarity function $\mathcal{S}$ from R x R to $\{0, 1\}$, where 1 means that the two references do refer to the same author and 0 means that they do not. As mentioned before, methods relying in learning techniques to define the similarity function are quite effective in different collections of citations, but they usually need many examples and sufficient features to work well, which can be very costly to obtain.

Exploiting Graph-based Similarity Functions. The methods that exploit graph-based similarity functions for author name disambiguation usually create a coauthorship graph $G = (V, E)$ for each ambiguous group. Each element of the author name and coauthor name attributes is represented by a vertex v. The same coauthor names are usually represented by only a unique vertex. For each coauthorship (i.e., a pair of authors who publishes an article) an edge $\langle v_i, v_j \rangle$ is created. The weight of each edge $\langle v_i, v_j \rangle$ is related to the amount of articles coauthored by the corresponding author names represented by vertices v_i and v_j.

A graph-based metric (e.g., shortest path as in [33]) may be combined with other similarity functions on the attributes of the references to authors or used as a new feature in the similarity vectors.

Clustering Techniques

The author grouping methods usually exploit a clustering technique in their disambiguation task. The most used techniques include: 1) *partitioning* [23], which creates a pre-defined number k of partitions of the set of references to authors in an iterative process; 2) *hier-*

archical agglomerative clustering [23], which groups the references to authors in a hierarchical manner; 3) *density-based clustering* [23], in which a cluster corresponds to a dense region of references to authors surrounded by a region of low density (according to some density criteria) – references in regions with low density are considered as noise; and 4) *spectral clustering* [51], which corresponds to graph-based techniques that compute the eigenvalues and eigenvectors, the spectral information, of a Laplacian Matrix that, in the the author name disambiguation task, represents a similarity matrix of a weighted graph. In general, these clustering techniques rely on a "good similarity function" to group the references.

2.1.2 Author Assignment Methods

Author assignment methods directly assign each reference to a given author by constructing a model that represents the author (e.g., the probabilities of an author publishing an article with other (co-)authors, in a given venue and using a list of specific terms in the work title) using either a supervised classification technique [16, 20] or a model-based clustering technique [3, 21].

Classification. Methods in this class assign the references to their authors using a supervised machine learning technique. More specifically, they receive as input a set of references to authors with their attributes, the *training data* $\mathcal{D}$, consisting of examples or, more specifically, references for which the correct authorship is known. Each example is composed of a set $\mathcal{F}$ of m features $\{f_1, f_2, \ldots, f_m\}$ along with a special variable called the *author*. This *author* variable draws its value from a discrete set of labels $\{a_1, a_2, \ldots, a_n\}$, where each label uniquely identifies an author. The training examples are used to produce a disambiguation function (i.e., the disambiguator) that relates the features in the training examples to the correct author. The *test set* (denoted as $\mathcal{T}$) for the disambiguation task consists of a set of references for which the features are known while the correct author is unknown. The disambiguator, which is a function from $\{f_1, f_2, \ldots, f_m\}$ to $\{a_1, a_2, \ldots, a_n\}$, is used to predict the correct author for the references in the test set. In this context, the disambiguator essentially divides the records in $\mathcal{T}$ into n sets $\{a_1, a_2, \ldots, a_n\}$, where a_i contains (ideally all and only all) the references in which the ith author is included.

These methods are usually very effective when faced with a large number of examples of citations for each author. Another advantage is that, if the collection has been disambiguated (manually or automatically), the methods may be applied only to references of the new citations inserted into the collection by simply running the learned model on them. Although successful cases of the application of these methods have been reported, the acquisition of training examples usually requires skilled human annotators to manually label references. DLs are very dynamic systems, thus manual labeling of large volumes of examples is unfeasible. Further, the disambiguation task presents nuances that impose the need for methods with specific abilities. For instance, since it is not reasonable to assume that examples for all possible authors are included in the training data and the authors change their interesting area over time, new examples need be insert into training data continuously and the methods need to be retrained periodically in order to maintain their effectiveness.

Clustering. Clustering techniques that attempt to directly assign references to authors work by optimizing the fit between a set of references to an author and some mathematical model used to represent that author. They use probabilistic techniques to determine the author in a iterative way to fit the model (or estimate the parameters in probabilistic techniques) of the authors. For instance, in the first run of such a method each reference may be randomly distributed to an author a_i and a function, from a set of features $\{f_1, f_2, \ldots, f_m\}$ to $\{a_1, a_2, \ldots, a_n\}$, is derived using this distribution. In the second iteration, this function is used to predict the author of each reference and a new function is derived to be used in the next iteration. This process continues until a stop condition is reached, for instance, after a number of iterations. Two algorithms commonly used to fit the models in disambiguation tasks are Expectation-Maximization (EM) [11] and Gibbs Sampling [19].

These methods do not need training examples, but they usually require privileged information about the correct number of authors or the number of author groups (i.e., group of authors that publish together) and may take some time to estimate their parameters (e.g., due to the several iterations). Additionally, these methods may be able to directly assign authors to their references in a new citations using the final derived function.

2.2 Explored Evidence

In this section, we describe the kinds of evidence most commonly explored by the disambiguation methods.

Citation Information. Citation information are the attributes directly extracted from the citations, such as author/coauthor names, work title, publication venue title, year, and so on. These attributes are the ones commonly found in all citations, but usually are not sufficient to perfectly disambiguate all references to authors. Some methods also assume the availability of additional information such as emails, addresses, paper headers, which is not always available or easy to obtain, although if existent, they usually help the process (a lot!).

Web Information. Web information represents data retrieved from the Web that is used as additional information about an author publication profile. This informa-

tion is usually obtained by submitting queries to search engines, based on the values of citation attributes and the returned Web pages are used as new evidence (attributes) to calculate the similarity among references. The new evidence usually improves the disambiguation task. One problem is the additional cost of extracting all the needed information from the Web documents.

Implicit Evidence. Implicit evidence is inferred from visible elements of attributes. Several techniques have being implemented to find implicit evidence, such as the latent topics of a citation. One example is the Latent Direchlet Location (LDA) [5] that estimates the topic distribution of a citation (i.e, LDA estimates the probability of each topic given a citation). This estimated distribution is used as new evidence (attribute) to calculate the similarity among references to authors.

2.3 Evaluation Metrics

Although not part of our taxonomy, one important point to understand the discussion that follows is the evaluation metrics that are used by each proposed method in their experimental evaluations. The most used metrics are: *accuracy*, which is basically the proportion of correct results among all predictions; the traditional metrics of *precision*, *recall*, and *F1*, commonly used for information retrieval and classification problems[8]; *pairwise F1*, a variation of F1 that considers pairs of citations correctly assigned to the same author (or not); *Cluster F1*, that calculates precision and recall of the correct clusters (i.e., the clusters that contain all and only all the references to an author); the *K* metric [7], a combination of purity (a pure cluster contains citations of only one author) and fragmentation of clusters (fragmentation occurs when the production of one author is split into one or more clusters); B-Cubed [1], that calculates precision and recall for each reference to an author; and MUC [1]. In this last metric, recall is calculated by summing up the number of elements in the ground-truth clusters minus the number of empirical clusters (obtained with the method) that contain these elements and dividing this by the total of elements minus the number of theoretical clusters. Precision is calculated similarly.

3. OVERVIEW OF REPRESENTATIVE METHODS

In this section, we present a brief overview of representative author name disambiguation methods which fall under one or more of the categories of the proposed taxonomy. Our main focus here is on those methods that have been specifically designed to address the name ambiguity problem in the context of bibliographic citations, since they are more related to the scope of this work. In the next subsections, we describe each method under the category we consider that best fits it. We notice that most of the described methods explore citation information in the disambiguation task. Thus, we leave to Subsection 3.3 the discussion of those methods that use additional evidence.

[8]In this last case, the authors are considered as classes and the correct assignments need to be known a priori.

3.1 Author Grouping Methods

Using Predefined Functions. Han et al. [22] represent each reference as a feature vector where each feature corresponds to an element of a given instance of one of its attributes. The authors consider two options for defining the feature weights: TFIDF and NTF (Normalized Term Frequency), being NTF given by $ntf(i,d) = freq(i,d)/maxfreq(i,d)$ where $freq(i,d)$ refers to the feature frequency i within the record d, and $maxfreq(i, d)$ refers to the maximal term frequency of feature i in the record d. The authors propose the use of K-way spectral clustering with QR decomposition [51] to construct clusters of references to the same author. To use this clustering technique, the correct number of clusters to be generated needs to be informed. The K-way spectral clustering method represents each reference as a vertex of an undirected graph and the weight of an edge represents the similarity between the attributes associated with the connected references. K-way spectral clustering splits the graph so that records that are more similar to each other will belong to the same cluster. This method was evaluated using data obtained from the Web and DBLP. Experimental results achieved 63% of accuracy in DBLP and up to 84.3% in the Web collection.

An algorithm for collective entity resolution (i.e., an algorithm that uses only disambiguated coauthor names when disambiguating an author name of a citation) that exploits attribute elements (i.e., attribute values present in the citation records) and relational information (i.e., authorship information between entities referred in the citations records) is proposed in [4] by Bhattacharya and Getoor. The authors propose a combined similarity function defined on attributes and relational information. As the initial step, they create clusters of disambiguated references verifying if two references have at least k coauthor names in common (they used only the author names in their experiments but they mention that other attributes may be used). The experiments were performed using soft-TFIDF, Jaro-Winkler, Jaro and Scaled Levenshtein metrics for name attributes, and Common Neighbours, Jaccard coefficient, Adamic-Adar similarity and Higher-order neighbourhoods metrics for relational attributes. The authors exploit a greedy agglomerative strategy that merges the most similar clusters in each step. The collections used in the experiments were: a subset of CiteSeer containing machine learning documents, a collection of high energy physics

publications from arXiv[9] and BioBase[10] that contains biological publications from Elsevier. The method obtained around 0.99 of F1 in the CiteSeer and arXiv collections and around 0.81 in the BioBase collection.

In [41], Soler proposes a new distance metric between two citations, c_i and c_j, (or clusters of citations) based on the probability of these publications having terms and author names in common. The proposed algorithm creates clusters of articles using the proposed metric and summarizes the clusters by means of a representative citation that includes the distance from it to the others. It groups the citations whose distances among them is minimum using as evidence author names, email, address, title, keywords, research field, journal and publication year. The final decision whether two candidate clusters belong to the same author or not is given by a specialist. Soler presents some illustrative cases of clusters obtained by using the proposed metric with records extracted from ISI-Thomson Web of Science database[11].

In [7], Cota et al. propose a heuristic-based hierarchical clustering method for author name disambiguation that involves two steps. In the first step, the method creates clusters of references with similar author names that share at least a similar coauthor name. Author name similarity is given by a specialized name comparison function called *Fragments*. This step produces very pure but fragmented clusters. Then, in the second step, the method successively fuses clusters of references with similar author names according to the similarity between the citation attributes (i..e., work title and publication venue) calculated using the cosine measure. In each round of fusion, the information of fused clusters is aggregated (i.e., all words in the titles are grouped together) providing more information for the next round. This process is successively repeated until no more fusions are possible according to a similarity threshold. The authors used pairwise F1 and K metrics on collections extracted from DBLP and BDBComp to evaluate the method and obtained around 0.77 and 0.93 for K in DBLP and BDBComp, respectively. An extension of this method that allows the name disambiguation task to be incrementally performed is presented in [10].

Learning a Similarity Function. In [44], Torvik et al. propose to learn a probabilistic metric for determining the similarity among MEDLINE records. The learning model is created using similarity vectors between two references containing features resulting of the comparison between the common citation attributes along with medical subject headings, language, and affiliation of two references. They also propose heuristics for generating training sets (positive and negative) automatically. In a subsequent work [45], Torvik and Smalheiser extend the method by including additional features, new ways of automatically generating training sets, an improved algorithm for dealing with the transitivity problem and a new agglomerative clustering algorithm for grouping records. They estimate recall around 98.8%, that only 0.5% of the clusters have mixed references of different authors (purity), and that only in 2% of the cases the references of a same authors are split into two or more clusters (fragmentation).

In [24], Huang et al. present a framework in which a blocking method is first applied to create blocks of references to authors with similar names. Next DBSCAN, a density-based clustering method [14], is used for clustering references by author. For each block, the distance metric between pairs of citations used by DBSCAN is calculated by a trained online active support vector machine algorithm (LASVM), which yields, according to the authors, a simpler and faster model than the standard support vector machines. The authors use different functions for each different attribute, such as the edit distance for emails and URLs, Jaccard similarity for addresses and affiliations and soft-TFIDF for names. The authors have applied their framework to a manually annotated dataset with 3,335 citation records and 490 distinct authors. Experiments were performed with pairs of references in which the disambiguator informs whether two references are of the same author or not. They obtained 90.6% in terms of pairwise F1. It should be noticed that these results were obtained by exploiting additional sources of evidence such as the headers of papers obtained from CiteSeer.

In [9], Culotta et al. aim to learn a score function to be applied to the disambiguation result, such that higher scores correspond to the more correct disambiguations. Instead of calculating the score using pairs of references, the authors propose a score function that considers all references in a cluster together, with the goal of maximizing the result of the score function in the resulting disambiguation. To learn this function, they propose a training algorithm that is error-driven, i.e., training examples are generated from incorrect predictions in the training data and ranked, i.e., the classifier uses a ranking of candidate predictions to tune its parameters. The authors evaluated two loss functions to tune the parameters, Ranking Perceptron [17] and Ranking MIRA [8]. The experimental evaluation used two collections extracted from DBLP (one which is called Penn, because disambiguation was performed manually by students from Penn State University) and other from the Rexa[12] Digital Library. As evaluation metrics, they used pairwise F1, MUC and B-Cubed [1]. As evidence, they ex-

[9] http://arxiv.org

[10] http://www.elsevier.com/wps/find/bibliographicdatabasedescription.cws_home/600715/description#description

[11] http://isiknowledge.com

[12] http://rexa.info

ploited features such as first and middle names of the authors, number of coauthors in common, rarity of the last name, similarity between work titles, e-mails, affiliations and publication venue titles as well as the minimum, maximum and average values for real-valued features, among several others. They also used a greedy agglomerative clustering technique to group the references. Ranking Perceptron generated the best results in DBLP and Penn, with 0.52 and 0.86 of pairwise F1, respectively. Ranking MIRA generates the best result on the other DBLP collection with 0.931 of pairwise F1.

Treeratpituk and Giles [46] propose a learned similarity function for author name disambiguation in the MEDLINE digital library. The authors exploit a large feature set obtained from MEDLINE metadata, similar to that proposed in [44]. The authors also use similarity vectors to learn the similarity function using a Random Forest classifier. They compare the use of Random Forests with decision trees, support vector machines, naïve Bayes and logistic regression to learn the function to be used along with some clustering technique (left unspecified). They also investigate the performance of subsets of the features capable of reaching good effectiveness. The authors obtain almost 96% of accuracy in their experiments by exploiting this large set of features.

Exploiting Graph-based Similarity Functions. In [35], On et al. address synonym in the group entity resolution problem (i.e., a reference to a person associated with a group of items, e.g., an author with a list of publications) by proposing an approach that uses the quasi-clique graph-mining technique for exploiting, besides simple textual similarities, "contextual information" extracted from the group items' attributes (e.g., the citation attributes) as additional evidence. This contextual information is obtained constructing a graph for each group to represent relationships between the author names (i.e., references) and the attribute values (e.g., co-authors). This graph is then superimposed on the pre-built graph constructed using the entire set of author names. Using this contextual information, the authors also propose a graph-based distance function based on common quasi-clique between the graphs of two entities (i.e., references). They compared their graph-based function (distQC) with Jaccard, TF-IDF and IntelliClean functions [32] by measuring the precision and recall at the top k most similar references using three collections extracted from ACM[13], BioMed (a dataset of medical publications) and IMDb. On average, the experiments show an improvement of 63%, 83% and 46% over Jaccard, TFIDF and IntelliClean functions in terms of precision at top-k records returned by their algorithm in ACM. Similar results were obtained for the other collections.

In [33], Levin and Heuser propose social network metrics that, along with string metrics, generate match functions to verify whether two references represent the same author. These functions were used in (very small) collections extracted from Cora[14], BDBComp and DBLP. The authors construct a graph with two kinds of vertices: one represents a reference to an author occurring in a citation and the other represents the citation itself; and two kinds of edges: one links the reference to the citation and the other links the vertices that share the same author name value. The authors obtained in their experiments around 95%, 82% and 95% of F1 in versions of Cora, BDBComp and DBLP, respectively.

In [15], Fan et al. propose the GHOST (GrapHical framewOrk for name diSambiguaTion) framework. GHOST solves the polysem problem using only the coauthor name attribute in five steps. In the first one, GHOST represents a collection as a graph G=(V, E), where each vertex $v \in V$ represents a reference to be disambiguated and each undirected edge (v_i, v_j) represents a coauthorship whose label S_{ij} is a set of citations coauthored by v_i and v_j. In the second step, GHOST identifies the valid paths eliminating the invalid ones between two nodes, i.e., a path that contains a subpath $v_i S_{ik} v_k S_{kj} v_j$ where S_{ik} is equal to S_{kj} and both have only one citation. In the third step, GHOST creates a matrix representing similarities between the vertices. For this, the authors propose a new similarity function based on the formula that calculates the resistance of a parallel circuit. In the fourth step, the Affinity Propagation clustering algorithm [13] is used to group the references to the same author. Finally, in the last step, GHOST makes use of user feedback to improve the results. Experimental evaluation was performed in collections extracted from DBLP and MEDLINE. GHOST obtained on average 0.86 and 0.98 of pairwise F1 in DBLP and MEDLINE, respectively.

3.2 Author Assignment Methods

Classification. In [20], Han et al. propose two methods based on supervised learning techniques that use coauthor names, work titles and publication venues as evidence for assigning a reference to its author. The first method uses naïve Bayes (NB), a generative statistical model frequently used in word sense disambiguation, to capture all writing patterns in the authors' citations. The second method is based on Support Vector Machines (SVMs), which are discriminative models basically used as a classifier [34]. An important difference between the two techniques is that a NB model requires only positive examples to learn about the writing patterns whereas SVMs require both positive and negative examples to learn how to identify the author. Both methods have been evaluated with data taken from the Web and DBLP. Experimental results show that, on average,

[13] http://portal.acm.org

[14] http://www.cs.umass.edu/ mccallum/code-data.html

using all attributes, the SVM-based method was more accurate (accuracy=95.6%) than the NB method (accuracy=91.3%) for the Web collected dataset while for the DBLP dataset the NB method performed better (SVM accuracy was 65.4% while NB's was 69.1%).

In [47], Veloso et al. propose SLAND, a disambiguation method that infers the author of a reference by using a supervised rule-based associative classifier. The method uses author names, work title and publication venue title attributes as features and infers the most probable author of a given reference r_i using the confidence of the association rules $\mathcal{X} \rightarrow a_i$ where $\mathcal{X}$ only contains features of r_i. The method also works on demand, i.e., association rules to infer the correct author of a reference are generated in the moment of disambiguation. The method is also capable of inserting new examples into the training data during the disambiguation process, using reliable predictions, and detecting authors not present in the training data. Experiments were conducted in two collections extracted from DBLP and BDBComp and the proposed method outperformed representative supervised (SVM and NB) considering the Micro and Macro F1 metrics. In the DBLP and BDBComp collections, the (Micro) F1 values were 0.911 and 0.457, respectively. To reduce the cost of obtaining training data, this method was extended [16] to become self-trained, i.e., it is now capable of producing its own training examples using (test) references to be disambiguated. Initially, the method extracts pure clusters of references by exploiting highly discriminative features, such as co-author names. The most dissimilar clusters are then selected to represent training examples for their authors. Next, the references in the rest of clusters are classified according to these training examples. In the experiments with the same collections, the self-trained method outperformed by far KWAY and SVM-DBSCAN and the associative method was the best choice for classifying the remaining test references not incorporated into the training data when compared to SVM and NB.

Clustering. In [21], Han et al. present an unsupervised hierarchical version of the naïve Bayes-based method for modeling each author. The authors assume that each citation is generated by a mixture of K authors. They then calculate the probability of a citation record c_m given an author a_i P($c_m|a_i$) using the probability of each attribute of this record given such author, in a hierarchical way. To estimate the parameters, the authors use the Expectation Maximization algorithm [11] aiming to maximize the likelihood of the citation records. The method obtained on average 54% and 58% of accuracy on data extracted from DBLP and the Web, respectively.

In [3], Battacharya and Getoor extend the generative model Latent Dirichlet Allocation (LDA) and propose a probabilistic model for collective entity resolution that uses the cooccurence of the references to authors in each work to determine the entities jointly, i.e., they use the disambiguated references to disambiguate other references in the same citation. In their model, the authors associate an attribute v_a, that contains the author name in the citation, with each author a. They assume that each citation is constructed choosing their authors from an author group (i.e., a group of authors that publish some article together) distribution. That is, initially a distribution that determines the probability of each author group having a specific author chosen to write the article is selected. Next using this distribution, the authors and a variation of their names are chosen for this citation. The proposed method receives as input only an approximation of the number of author groups in the collection. Experiments were performed using citations extracted from CiteSeer and arXiv reaching up to 0.99 and 0.98 respectively of pairwise F1.

In [43], Tang et al. propose a probabilistic framework based on Hidden Markov Random Models (HMRF) for the polysem subproblem. In this work, the authors use author names, work title, publication venue title, publication year, abstract and bibliographic references as content-based evidence and relationships between citations as structure-based evidence for disambiguating author names. Each relationship represents the fact that two citations were published in the same publication venue, have a coauthor name in common, cite each other, have distinct coauthor names that were coauthors in another citation, or have some specific user-provided constraint in common. Content and structure-based evidence are modeled as feature functions (used to represent the similarity between two citations by their content or relationships) which are then incorporated into a HMRF used to estimate the weights of the feature functions and to assign the citations to their authors. The authors also use Bayesian Information Criterion to estimate the number of authors of the collection. Experimental evaluation was performed on citations extracted from ArnetMiner[15]. Pairwise F1 values were 0.888 and 0.805 when the method uses the correct number of authors and when it estimates this number, respectively.

3.3 Using Additional Evidence

Web Information. In [26], Kanani et al. present two approaches for author name disambiguation that gather additional evidence from the Web. They construct a graph in which each vertex corresponds to a reference to an author and the edges are weighted with values that represent the probability of the two vertices (i.e., references) being the same author. This weight is initially calculated using the citation attributes. In the first approach, they use the result of searches submitted to a

[15] http://arnetminer.org

Web search engine for the work titles of citation records of the corresponding references to authors to change the weight of the edge between two references. In the second one, they use one of the returned pages of the search as a new type of vertex in the graph (web vertex), adding new edges from this new vertex to each previously existing reference vertex, indicating the probability of the reference and the web page belonging to the same author. In both cases, a maximum entropy or logistic regression model is learned for a pair of references a_i and a_j and the weight of the edge $\langle a_i,a_j \rangle$ is given by the probability of the corresponding references refer to the same author minus the probability of these references refer to the different authors. DBLP, Penn and the Rexa collections were used in their experiments. Using the results of searches to Google to change the weight of the edges their method obtain around 0.905, 0.877 and 0.918 of accuracy and around 0.886, 0.814 and 0.747 of pairwise F1 in the DBLP, Rexa and Penn collections, respectively. The method that uses the returned Web pages as vertices in the graph was run only with DBLP, producing 0.882 of accuracy and 0.903 of pairwise F1.

In [49], Yang et al. address the author name ambiguity problem using topics and correlations found on the Web. They determine the topics of the citation from venue information using an extraction algorithm based on association rules in order to create a topic association network. They also use the Web for retrieving publication pages of authors or coauthors to be disambiguated. Then, they create a similarity function making use of an SVM classifier on the top of all these features. The authors represent the references to authors as vertices in a graph and the similarity function is used to create the edges between vertices. Their clustering technique removes a bridge edge when each resulting connected component has at least a given number of vertices. They tested their approach on the collection constructed by Han et al. [20] and obtained an increase of accuracy around 66% (0.75 of accuracy) when compared to the use of citations without topics and Web correlations.

In [27], Kang et al. exploit coauthorship information using a Web-based technique that obtains other (implicit) coauthors of the reference to be disambiguated. They submit a pair of author names of a same citation as a query to Web search engines to retrieve documents containing both author names and then extract new names found in these documents as new implicit coauthors of this pair. The authors measure the similarity between two references by counting the number of coauthors in common and use the single-link agglomerative clustering technique [25] to group the references to the same author. They used a collection of citations published in Korean during 1999-2006 that has only the polysem problem obtaining around 0.85 of pairwise F1.

In [38], Pereira et al. also exploit Web information to disambiguate author names. Their method attempts to find Web documents corresponding to curricula vitae or Web pages containing publications of a single author. It works in three steps. The first step receives a list of citations whose references must be disambiguated and, for each citation, submits a query containing data from its attributes to a Web search engine. It then inserts the top-m documents in the answer set into a set $\mathcal{D}$ of documents. The second step selects the documents in $\mathcal{D}$ that contain publications from a given author. The third step groups the reference to authors whose citations occur in a same document in a hierarchical manner, i.e., if citations of two ambiguous references occur in the same Web document, these citations are considered as belonging to the same author and are fused in a same cluster. The experimental evaluation, performed using DBLP data, obtained on average 0.80, 0.76 and 0.14 of K, pairwise F1 and cluster F1 metrics, respectively.

Implicit Evidence. In [42], Song et al. propose a two-step unsupervised method. The first step uses Probabilistic Latent Semantic Analysis (PLSA) and Latent Dirichlet Allocation (LDA) to assign a vector of probabilities of topics to each citation. The PLSA and LDA proposed by them introduce a variable for persons (authors) in the generative model, that does not exist in general generative models. The second step considers the distributions of the probability of topics with respect to citations as a new attribute for name disambiguation. The authors use the Levenshtein distance to measure the similarity between two names. When two names are considered similar, they use the probability vectors of two corresponding citations and the Euclidean distance to merge the citations of the same authors. The authors compared their method with a greedy agglomerative clustering, K-way spectral clustering and LASVM + DBSCAN on citations extracted from CiteSeer and personal names on the Web. Their experiments demonstrate that their method, when faced with a lot of citation information is more effective than the baselines obtaining on average around 0.911 and 0.936 of pairwise F1 on the Web and CiteSeer collections, respectively.

In [40], Shu, Long and Meng extend the Latent Dirichlet Allocation model (LDA) for obtaining the topic distribution of each citation by adding the assumption that every topic is a Dirichlet distribution over all author names, that each document is a mixture of topics, and that each topic is a Dirichlet distribution over all the words. They train a classifier (C4.5 and SVMs) based on the similarity on topics, coauthor names, title and venue, as well as on the minimum distance between coauthor names, to predict whether two references are of the same author or not. The authors attempt to solve the problem of name ambiguity by trying to solve first

the polysem problem and then the synonymy. They use K-way spectral clustering to split the references into k sets, one for each author, in order to deal with the polysem problem. Next, they compare two sets of references of authors whose names have a distance below a given threshold and count the number of citations from these two sets which are assigned to the same author by the classifier. This value is divided by the total number of pairs of those two sets and if the result is greater than a given threshold they are merged. The authors show the effectiveness of their method by applying it to data extracted from DBLP. For the polysem problem the precision an recall were over 0.9 for the most ambiguous groups while for the synonym problem the precision was around 0.99 and recall equals to 0.917.

4. SUMMARY OF CHARACTERISTICS

In this section, we present an overview of the characteristics found in the described author name disambiguation methods, summarized in Tables 2 and 3.

Among the collections used to evaluate the methods we have: (1) versions of CiteSeer, DBLP, BDBComp, ArnetMiner, and Rexa containing computer science publications; (2) arXiv that contains citations from high energy physics publications; (3) BioBase, containing citations from biological publications; (4) MEDLINE and BioMed with data from biomedical publications; (5) ISI-Thomson with publications from several knowledge areas; (6) Cora, which is constituted of duplicated citations in Computer Science and person names extracted from the Web; and (7) IMDb with data about actors.

The majority of the described methods [4, 7, 9, 15, 22, 24, 26, 27, 33, 35, 38, 40, 41, 42, 45, 46, 49] try to disambiguate references to authors by using a similarity function to indicate whether two references refer to the same author instead of directly assigning the corresponding author to each reference, as in [3, 16, 20, 21, 43, 47].

Some of these methods receive the correct number of authors in the collection as input ([15, 21, 22]) ors this number corresponds to the number of authors in the training data [20]. Other methods, such as those proposed in [3], [43] and [16], try to estimate this number.

Almost half of the methods [3, 4, 7, 15, 16, 20, 21, 22, 33, 35, 40] uses at most the three main citation attributes: author names, work title and publication venue title, as evidence. These attributes are the most commonly found in citation records, constituting in most cases the hardest situation for disambiguation. Few methods [26, 27, 38, 49] exploit additional evidence such as emails, addresses, paper headers etc., which are not always available or easy to obtain.

Tables 2 and 3 also summarize the evaluation metrics used by each method as well as the type of subproblem (i.e., synonym, polysem, or both) addressed.

5. OPEN CHALLENGES

There are several open challenges that need to be addressed in order to produce more reliable solutions that can be employed in a production mode in real digital libraries. Below we discuss some of them.

Very Little Data in the Citations. In most cases we have only the basic information about the citations available: author (coauthor) names, work and publication venue titles, and publication year. Furthermore, in some cases author names contain only the initial and the last surname and the publication venue title is abbreviated. New strategies that try to derive implicit information (e.g., topics) or gather additional information from the Web are promising in this scenario.

Very Ambiguous Cases. Several methods exploit coauthor-based heuristics, by explicitly assuming the hypotheses that: (i) very rarely ambiguous references will have coauthors in common who have also ambiguous names; or (ii) it is rare that two authors with very similar names work in the same research area. These hypotheses work in most cases, but when they fail, the errors they generate are very hard to fix. For example, in the case of authors with Asian names, the first hypothesis fails more frequently than for authors with English or Latin names.

Citations with Errors. Errors occur in citation data which are sometimes impossible to detect. The methods need to be tolerant to such errors.

Efficiency. With the high amount of articles being published nowadays in the different knowledge areas, the solutions need to deal with the problem efficiently. Few proposed methods have this explicit concern.

Different Knowledge Areas. As we have seen, most of the collections used to evaluate the methods are related to Computer Science. However, other knowledge areas (e.g., Humanities, Medicine) may have different publication patterns (e.g., publications with a sole author or with a lot of coauthors) causing additional difficulties for the current generation of methods.

Incremental Disambiguation. Ideally disambiguation should be performed incrementally as new citations are incorporated into the DL, since it is not reasonable to assume that the whole DL should be disambiguated at each new load. However, most, if not all, methods ignore this fact. A promising solution is presented in [10].

Author Profile Changes. It is common that the research interests of an author change over time. This can happen due to new collaborations, change in research group or institution, natural evolution of a research field, etc. These changes cause modifications in the model representing the author profile causing difficulties for the methods. A possible solution may involve retraining, but determining when to retrain is a challenge. However, this issue has been largely ignored by all methods.

New Authors. The methods should be capable of iden-

Table 2: Summary of characteristics - Author grouping methods

Method	Similarity function	Clustering technique	Evidence	Collections	Evaluation metric	Subproblem	# of authors
Bhattacharya and Getoor [4]	Common neighbours, Jaccard, Adamic/Adar and Higher-order neighbourhoods	Agglomerative	Author name	CiteSeer, arXiv and BioBase	F1	Both	Unknown
Cota et al. [7]	Fragment comparison and cosine	Agglomerative	Citation attributes	DBLP and BDBComp	Pairwise F1 F1 and K	Both	Unknown
Culotta et al. [9]	Error-drive and hank-based learning	Agglomerative	All of each collection	DBDL and Rexa	Pairwise F1, MUC and B-Cubed	Both	Unknown
Fan et al. [15]	graph-based	Affinity Propagation	Author names	DBLP and MEDLINE	Pairwise F1	Polysem	Unknown
Han et al. [22]	Cosine	Spectral clustering	Citation attributes	DBLP and Web	Accuracy	Both	Known
Huang et al. [24]	Learned using LASVM	DBScan	First page of the articles	CiteSeer	Pairwise F1	Both	Unknown
Kanani, McCallum and Pal [26]	Learned using maximum entropy or logistic regression	Partitioning	Citation attributes and Web pages	DBLP, Penn and Rexa	Accuracy and pairwise F1	Both	Unknown
Kang et al. [27]	Heuristic	Agglomerative	Author names and Web pages	Korean citations	F1 and under/ over-clustering error	Polysem	Unknown
Levin and Heuser [33]	Social network metrics	-	Citation attributes	DBLP, Cora and BDBComp	F1	Both	Unknown
On et al. [35]	Quasi-clique	-	Citation/Movie attributes	ACM, BioMed and IMDb	Ranked recall and precision	Synonym	Unknown
Pereira et al. [38]	Heuristic	Agglomerative	Citation attributes	DBLP	Pairwise and cluster F1 and K	Both	Unknown
Shu, Long and Meng [40]	Learned using C4.5/SVMs and edit distance	Spectral and agglomerative clustering	Citation attributes	DBLP	Pairwise F1	Both	Known
Soler [41]	Probabilistic metric	Agglomerative	Citation attributes, email, address, keywords and research field	ISI-Thomson	-	Both	Unknown
Song et al. [42]	Levenshtein and Euclidean distance	Agglomerative	Citation attributes and latent topics (LDA/PLSA)	CiteSeer and Web	Pairwise and cluster F1	Both	Unknown
Torvik and Smalheiser [45]	Learn a probabilist metric	Agglomerative	MEDLINE metadata	MEDLINE	Recall	Both	Unknown
Treeratpituk and Giles [46]	Learned using random forest classifier	-	MEDLINE metadata	MEDLINE	Accuracy	Both	Unknown
Yang et al. [49]	Learned using SVM	Partitioning	Citation attributes, topics and Web pages	DBLP	Accuracy, precision and recall	Both	Unknown

Table 3: Summary of characteristics - Author assignment methods

	Method	Technique	Attributes	Collections	Evaluation metric	Subproblem	# of authors
Classification	Ferreira et al. [16]	Associative classifier	Citation attributes	DBLP and BDBComp	Pairwise F1 and K	Both	Estimated
	Han et al. [20]	SVM and naïve Bayes classifiers	Citation attributes	DBLP and Web	Accuracy	Both	Known
	Veloso et al. [47]	Associative classifier	Citation attributes	DBLP and BDBComp	F1	Both	Estimated
Clustering	Battacharya and Getoor [3]	LDA with Gibbs sampling	Author names	CiteSeer and arXiv	F1	Both	Estimated
	Han et al. [21]	Hierarchical naïve Bayes with EM	Citation attributes	DBLP and Web	Accuracy	Both	Known
	Tang et al. [43]	Hidden Markov Random Fields	Citation attributes	ArnetMiner	Pairwise F1	Polysem	Estimated

tifying references to new ambiguous authors who do not have citations in the DL yet. Only one of the reported methods [47] has explicitly addressed this issue.

6. CONCLUSIONS

This article presented a brief survey on author name disambiguation methods. We proposed a taxonomy to classify the methods and provided an overview of the most representative ones. Some patterns became clear. The majority of the surveyed methods perform disambiguation by comparing citation records using some type of similarity function. This function, which can be predefined or learned specifically for the disambiguation task, is directly applied to the citation attributes, which can be enhanced with additional information retrieved from the Web or inferred from the own citation attributes (e.g., topics). A few other methods disambiguate by directly assigning the citation records to their authors using supervised and unsupervised machine learning techniques. Some open problems were also discussed.

One major gap in the field is the lack of direct comparisons among the methods under the same circumstances: e.g., same collections (e.g., many methods used different versions of collections such as DBLP), same computational environment, same experimental design. This is probably due to the lack of standard collections like those provided by the TREC competitions. Moreover, the few comparisons that exist involve at most three or four methods and were performed in static scenarios. In fact, there is no study about how these methods would perform in a real-word scenario of a dynamic and living digital library. These issues along with the open problems previously discussed are in our opinion what should guide the research efforts for developing new author name disambiguation methods in the near future.

Acknowledgments

This research is partially funded by InWeb (MCT/CNPq/FAPEMIG grant 573871/2008-6), CAPES, CNPq, and FAPEMIG.

7. REFERENCES

[1] A. Bagga and B. Baldwin. Algorithms for scoring coreference chains. In *LREC*, pages 563–566, 1998.
[2] R. Bekkerman and A. McCallum. Disambiguating web appearances of people in a social network. In *WWW*, pages 463–470, 2005.
[3] I. Bhattacharya and L. Getoor. A latent dirichlet model for unsupervised entity resolution. In *SDM*, 2006.
[4] I. Bhattacharya and L. Getoor. Collective entity resolution in relational data. *ACM TKDD*, 1(1), 2007.
[5] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.
[6] W. W. Cohen, P. D. Ravikumar, and S. E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *IIWeb*, pages 73–78, 2003.
[7] R. G. Cota, A. A. Ferreira, M. A. Gonçalves, A. H. F. Laender, and C. Nascimento. An unsupervised heuristic-based hierarchical method for name disambiguation in bibliographic citations. *JASIST*, 61(9):1853–1870, 2010.
[8] K. Crammer and Y. Singer. Ultraconservative online algorithms for multiclass problems. *JMLR*, 3:951–991, 2003.
[9] A. Culotta, P. Kanani, R. Hall, M. Wick, and A. McCallum. Author disambiguation using error-driven machine learning with a ranking loss function. In *IIWeb*, 2007.
[10] A. P. de Carvalho, A. A. Ferreira, A. H. F. Laender, and M. A. Gonçalves. Incremental unsupervised name disambiguation in cleaned digital libraries. *JIDM*, 2(3):289–304, 2011.
[11] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B*, 39(1):1–38, 1977.
[12] C. P. Diehl, L. Getoor, and G. Namata. Name reference resolution in organizational email archives. In *SDM*, pages 70–91, 2006.
[13] D. Dueck and B. J. Frey. Non-metric affinity propagation for unsupervised image categorization. In *ICCV*, 2007.
[14] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, pages 226–231, 1996.
[15] X. Fan, J. Wang, X. Pu, L. Zhou, and B. Lv. On graph-based name disambiguation. *JDIQ*, 2:10:1–10:23, 2011.
[16] A. A. Ferreira, A. Veloso, M. A. Gonçalves, and A. H. F. Laender. Effective self-training author name disambiguation in scholarly digital libraries. In *JCDL*, pages 39–48, 2010.
[17] Y. Freund and R. Schapire. Large margin classification using the perceptron algorithm. *Machine learning*, 37(3):277–296, 1999.
[18] C. Galvez and F. de Moya Anegón. Approximate personal name-matching through finite-state graphs. *JASIST*, 58(13):1960–1976, 2007.
[19] T. Griffiths and M. Steyvers. Finding scientific topics. *The National Academy of Sciences*, 101(1):5228–5235, 2004.
[20] H. Han, C. L. Giles, H. Zha, C. Li, and K. Tsioutsiouliklis. Two supervised learning approaches for name disambiguation in author citations. In *JCDL*, pages 296–305, 2004.
[21] H. Han, W. Xu, H. Zha, and C. L. Giles. A hierarchical naive Bayes mixture model for name disambiguation in author citations. In *SAC*, pages 1065–1069, 2005.
[22] H. Han, H. Zha, and C. L. Giles. Name disambiguation in author citations using a k-way spectral clustering method. In *JCDL*, pages 334–343, 2005.
[23] J. Han and M. Kamber. *Data mining: concepts and techniques*. Morgan Kaufmann, 2005.
[24] J. Huang, S. Ertekin, and C. L. Giles. Efficient name disambiguation for large-scale databases. In *ECML-PKDD*, pages 536–544, 2006.
[25] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.
[26] P. Kanani, A. McCallum, and C. Pal. Improving author coreference by resource-bounded information gathering from the web. In *IJCAI*, pages 429–434, 2007.
[27] I.-S. Kang, S.-H. Na, S. Lee, H. Jung, P. Kim, W.-K. Sung, and J.-H. Lee. On co-authorship for author disambiguation. *Inf. Process. Manage.*, 45(1):84–97, 2009.
[28] V. C. Klaas. Who's who in the world wide web: Approaches to name disambiguation. Master's thesis, Diplomarbeit, LMU München, Informatik, 2007.
[29] A. H. F. Laender, M. A. Gonçalves, R. G. Cota, A. A. Ferreira, R. L. T. Santos, and A. J. C. Silva. Keeping a digital library clean: new solutions to old problems. In *DocEng*, pages 257–262, 2008.
[30] C. Lagoze and H. V. de Sompel. The open archives initiative: building a low-barrier interoperability framework. In *JCDL*, pages 54–62, 2001.
[31] D. Lee, J. Kang, P. Mitra, C. L. Giles, and B.-W. On. Are your citations clean? *Comm. ACM*, 50(12):33–38, 2007.
[32] M.-L. Lee, T. W. Ling, and W. L. Low. IntelliClean: a knowledge-based intelligent data cleaner. In *KDD*, pages 290–294, 2000.
[33] F. H. Levin and C. A. Heuser. Evaluating the use of social networks in author name disambiguation in digital libraries. *JIDM*, 1(2):183–197, 2010.
[34] T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.
[35] B.-W. On, E. Elmacioglu, D. Lee, J. Kang, and J. Pei. Improving grouped-entity resolution using quasi-cliques. In *ICDM*, pages 1008–015, 2006.
[36] B.-W. On and D. Lee. Scalable name disambiguation using multi-level graph partition. In *SDM*, pages 575–580, 2007.
[37] B.-W. On, D. Lee, J. Kang, and P. Mitra. Comparative study of name disambiguation problem using a scalable blocking-based framework. In *JCDL*, pages 344–353, 2005.
[38] D. A. Pereira, B. A. Ribeiro-Neto, N. Ziviani, A. H. F. Laender, M. A. Gonçalves, and A. A. Ferreira. Using web information for author name disambiguation. In *JCDL*, pages 49–58, 2009.
[39] C. L. Scoville, E. D. Johnson, and A. L. McConnell. When A. Rose is not A. Rose: the vagaries of author searching. *Medical Reference Services Quaterly*, 22(4):1–11, 2003.
[40] L. Shu, B. Long, and W. Meng. A latent topic model for complete entity resolution. In *ICDE*, pages 880–891, 2009.
[41] J. M. Soler. Separating the articles of authors with the same name. *Scientometrics*, 72(2):281–290, 2007.
[42] Y. Song, J. Huang, I. G. Councill, J. Li, and C. L. Giles. Efficient topic-based unsupervised name disambiguation. In *JCDL*, pages 342–351, 2007.
[43] J. Tang and *et al*. A unified probabilistic framework for name disambiguation in digital library. *TKDE*, 24(6):975–987, 2012.
[44] V. I. Torvik , M. Weeber, D. R. Swanson, and N. R. Smalheiser. A probabilistic similarity metric for Medline records: A model for author name disambiguation. *JASIST*, 56(2):140–158, 2005.
[45] V. I. Torvik and N. R. Smalheiser. Author name disambiguation in MEDLINE. *ACM TKDD*, 3(3):1–29, 2009.
[46] P. Treeratpituk and C. L. Giles. Disambiguating authors in academic publications using random forests. In *JCDL*, pages 39–48, 2009.
[47] A. Veloso, A. A. Ferreira, M. A. Gonçalves, A. H. F. Laender, and W. Meira Jr. Cost-effective on-demand associative author name disambiguation. *Inf. Process. Manage.*, 48(4):680– 697, 2012.
[48] Q. M. Vu, T. Masada, A. Takasu, and J. Adachi. Using a knowledge base to disambiguate personal name in web search results. In *SAC*, pages 839–843, 2007.
[49] K.-H. Yang, H.-T. Peng, J.-Y. Jiang, H.-M. Lee, and J.-M. Ho. Author name disambiguation for citations using topic and web correlation. In *ECDL*, pages 185–196, 2008.
[50] M. Yoshida and *et al*. Person name disambiguation by bootstrapping. In *SIGIR*, pages 10–17, 2010.
[51] H. Zha and *et al*. Spectral relaxation for K-means clustering. In *NIPS*, pages 1057–1064, 2001.