Incremental Unsupervised Name Disambiguation in Cleaned Digital Libraries

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

[1] Departamento de Ciência da Computação, Universidade Federal de Minas Gerais
[2] Departamento de Computação, Universidade Federal de Ouro Preto
{anapc,ferreira,laender,mgoncalv}@dcc.ufmg.br

Abstract. Name ambiguity in the context of bibliographic citations is one of the hardest problems currently faced by the Digital Library (DL) community. Here we deal with the problem of disambiguating new citations records inserted into a cleaned DL, without the need to process the whole collection, which is usually necessary for unsupervised methods. Although supervised solutions can deal with this situation, there is the costly burden of generating training data besides the fact that these methods cannot handle well the insertion of records of new authors not already existent in the repository. In this article, we propose a new unsupervised method that identifies the correct authors of the new citation records to be inserted in a DL. The method is based on heuristics that are also used to identify whether the new records belong to authors already in the digital library or not, correctly identifying new authors in most cases. Our experimental evaluation, using synthetic and real datasets, shows gains of up to 19% when compared to a state-of-the-art method without the cost of having to disambiguate the whole DL at each new load (as done by unsupervised methods) or the need for any training (as done by supervised methods).

Categories and Subject Descriptors: H.3.3 [**Information Search and Retrieval**]: Information Search and Retrieval; H.3.7 [**Information Storage and Retrieval**]: Digital Libraries

Keywords: Bibliographic Citation, Digital Library, Name Disambiguation

1. INTRODUCTION

Digital Libraries (DLs) are complex information systems that involve rich sets of digital objects and their respective metadata, along with multiple organizational structures and services such as searching, browsing and personalization, being generally built towards a target community with specific interests [Gonçalves et al. 2004].

Bibliographic citations (here regarded as a set of bibliographic attributes such as author and coauthor names, work title, publication venue title and publication year) management within DLs involves a number of tasks. One in particular, *author name disambiguation*, has required a lot of attention from the DL research community due to its inherent difficulty. Specifically, name ambiguity is a problem which occurs when a set of citation records contains ambiguous author names (the same author may appear under distinct names or distinct authors may have similar names). This problem may be caused by a number of reasons, including the lack of standards and common practices, and the decentralized generation of content (e.g., by means of automatic harvesting).

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, publication venue title and publication year. With each attribute in a citation is associated a specific value, which may be composed of several *elements*. In case of the

This research is partially funded by InWeb - The National Institute of Science and Technology for the Web (MCT/CNPq/FAPEMIG grant number 573871/2008-6) and by the authors's individual scholarships and research grants from CAPES, CNPq, and FAPEMIG.

attribute "author names", an element corresponds to the name of a single unique author. Each author name element is a *reference* r_j to an author. In case of the other attributes, an element corresponds to a word/term. The objective of a disambiguation method is to produce a disambiguation function that is used to partition the set of references to authors $\{r_1, r_2, \ldots, r_m\}$ into n sets $\{a_1, a_2, \ldots, a_n\}$, so that each partition a_i contains (all and ideally only all) the references to a same author. We consider that each reference r to an author extracted from a citation record c contains at least the attributes: author name, coauthor names that contain the other references extracted from the record c, work title, publication venue title and publication year of the citation c.

The challenges of dealing with name ambiguity in bibliographic DLs have led to a myriad of disambiguation methods [Bhattacharya and Getoor 2006; 2007; Cota et al. 2007; Cota et al. 2010; Culotta et al. 2007; Fan et al. 2011; Ferreira et al. 2010; Han et al. 2004; Han et al. 2005; Han et al. 2005; Huang et al. 2006; Kanani et al. 2007; Kang et al. 2009; Levin and Heuser 2010; On and Lee 2007; Pereira et al. 2009; Soler 2007; Song et al. 2007; Tang et al. 2008; Torvik and Smalheiser 2009; Treeratpituk and Giles 2009; Yang et al. 2008], that try to assign the records to their authors or to group the records of the same author based on similarity measure. However, despite the fact that most of these methods were demonstrated to be relatively effective (in terms of error rate or similar metrics), most of them were evaluated using a static snapshot collection extracted from a DL.

In real life, there is a constant need of inserting new ambiguous citation records into a disambiguated digital library, being therefore necessary to assign each reference r_i of those new records to their corresponding "real" authors. A possible solution for this problem is to re-run a disambiguation method for all citation records after inserting the new records into the DL. Despite very expensive, as DLs are usually composed of thousands, sometimes millions of records, this is what is usually done, mainly when applying unsupervised solutions (see Section 2). Moreover, any possible manual corrections that have been made, which do occur since the methods are not perfect, would be lost with the re-processing of the whole collection.

Another solution would be to disambiguate only the new records. In this case, if the new records belong to pre-existing authors in the DL (i.e., authors of citations already inserted into a DL), supervised methods that try to assign the citations to their corresponding authors may be used, but if any author of these new records do not exist in the DL (i.e., a new author) these methods will assign these records to incorrect existing authors. Moreover, these methods usually need a learning phase requiring costly training data.

The solution we advocate here is to exploit similarity among records and assign the new records to authors with similar citation records in the DL or to new authors when the similarity evidence is not strong enough. Accordingly, in this article we propose INDi, an Incremental (unsupervised) Name Disambiguation method for assigning references (i.e., author names) of new ambiguous citation records inserted into a disambiguated DL to their correct authors. Our method includes specific heuristics for checking whether references of new citation records belong to pre-existing authors of the DL or if they belong to new ones (i.e., authors without citation records in the DL), avoiding running the disambiguation process in the entire DL. We experimentally evaluate our method and compare it with a state-of-the-art unsupervised method. For this, we used the real BDBComp collection and collections generated by a synthetic data generator with which we can control several parameters of the problem, for instance, the percentage of new authors having records inserted into the DL for each new load, an important challenge for disambiguation methods. Our experimental evaluation, using synthetic and real datasets, shows gains of up to 19% when compared to a state-of-the-art method without the cost of having to disambiguate the whole DL at each new load (as done by supervised methods) or the need for any training (as done by supervised methods). We are not aware of any proposal or evaluation of methods with such properties in the literature.

This article is organized as follows. Section 2 discusses related work. Section 3 describes our proposed method and Section 4 presents the results of our experimental evaluation. Then, Section 5 discusses some failure cases and, finally, Section 6 presents our conclusions and offers possible directions for future work.

2. RELATED WORK

Automatic name disambiguation methods proposed in the literature adopt solutions that are usually based on supervised [Culotta et al. 2007; Ferreira et al. 2010; Han et al. 2004; Huang et al. 2006; Torvik and Smalheiser 2009; Treeratpituk and Giles 2009] or unsupervised techniques [Bhattacharya and Getoor 2006; 2007; Cota et al. 2010; Fan et al. 2011; Han et al. 2005; Han et al. 2005; Kanani et al. 2007; Kang et al. 2009; Levin and Heuser 2010; On and Lee 2007; Pereira et al. 2009; Soler 2007; Song et al. 2007; Tang et al. 2008; Yang et al. 2008]. In this section, we briefly discuss some of these methods. We refer the reader to [Cota et al. 2010] for a more detailed discussion.

The unsupervised methods, i.e., methods based on unsupervised techniques such as clustering, usually exploit similarities between attributes of the references to authors, by means of predefined similarity functions, in order to group those references that are likely to be associated with the same author. These functions are defined over existing attributes in the citations [Bhattacharya and Getoor 2007; Cota et al. 2007; Cota et al. 2010; Fan et al. 2011; Han et al. 2005; Levin and Heuser 2010; On and Lee 2007; Soler 2007], over implicit information such as citation topics [Song et al. 2007; Yang et al. 2008], or over data retrieved from the Web [Kanani et al. 2007; Kang et al. 2009; Pereira et al. 2009; Yang et al. 2008]. There are also unsupervised methods that attempt to optimize the fit between the set of references to authors and some mathematical model used to represent an author [Bhattacharya and Getoor 2006; Han et al. 2005; Tang et al. 2008].

In contrast, supervised methods learn from a training set containing pre-labeled examples a "model" to either predict the author of an citation record [Han et al. 2004; Ferreira et al. 2010] or to determine if two citation records belong to the same author [Culotta et al. 2007; Huang et al. 2006; Torvik and Smalheiser 2009; Treeratpituk and Giles 2009]. The derived model can then be applied to a set of unseen (new) examples (test set). In the former case, the records in the training and test sets represent citation records, while in the latter, the records correspond to comparisons between two citation records (e.g., the difference of the vectors representing the records).

Our proposed method differs from those previously cited by incrementally disambiguating the DL, i.e., our method disambiguates only the ambiguous references (names) in the new citation records inserted into a DL. Moreover, it is able to identify whether these references refer to authors of citation records already present in the DL or refer to new ones (i.e., authors without citation records in the DL). It is also able to decrease the processing cost, since it is not necessary to disambiguate the entire DL at once. Finally, our method is unsupervised and, therefore, does not require any training data which is usually costly to be obtained.

3. PROPOSED METHOD

INDi aims at determining the authors of new citation records added to a DL, avoiding the disambiguation of the whole digital library. It uses some heuristics to disambiguate the author names (i.e., the references to authors) of new citation records. These heuristics are meant to disambiguate the new citation records by prioritizing the assignment of this record to the correct author, i.e., in case of doubts INDi prefers to consider the new record as belonging to a new author (i.e., an author that does not have citation records in the DL yet) instead of assigning the doubtful record to a existing author with a probability of error. In this sense, INDi has a bias towards producing over time groups (clusters) of citations which are very pure, that is, most of the citation records assigned by INDi to a given existing author are most probably correct. The side effect is that authors who are not so prolific (i.e., with few entries in the DL) may have its production split into groups associated with new authors. This is done for a major reason: mixing the citation records of different authors is a much harder problem to fix than putting together split groups in a manual correction, which we do assume that will eventually occur given that current state-of-the-art methods still do produce errors. In fact, it is much easier for a DL administrator to check whether the authors indicated as new by our method are in fact of this type than sorting through all the groups in the DL to find eventual errors. We recall that INDi is an incremental method that disambiguates only new entries, being capable of

Algorithm 1 INDi

Input: Citation record c; Cleaned collection $\mathcal{C}$ of citation records; Similarity thresholds α_{Venue} and α_{Title}; Incremental value δ;

Output: List L of pairs that contain the reference and the author (identification of the author);

```
1: c' ← PreprocessCitationRecord(c);
2: for each reference r ∈ c' do
3:    S ← GetClustersOfSimilarAuthors(r, C);
4:    S' ← PreprocessClusters(S);
5:    if there is cluster s ∈ S' and similarCoauthors(s, r) and (similarTitle(s, r, α_Title) or similarVenue(s,
      r, α_Venue)) then
6:       add(L,r,s.IdAuthor);
7:    else
8:       auxThresVenue ← α_Venue + δ;
9:       auxThresTitle ← α_Title + δ;
10:      if r.coauthorList is empty then
11:         if there is cluster s ∈ S' and (similarTitle(s, r, auxThresTitle) or similarVenue(s, r,
            auxThresVenue)) then
12:            add(L,r,s.IdAuthor);
13:         else
14:            add(L,r,newIdAuthor());
15:         end if
16:      else
17:         if there is s ∈ S' and s.coauthorList is empty and (similarTitle(s, r, auxThresTitle) or
            similarVenue(s, r, auxThresVenue)) then
18:            add(L,r,s.IdAuthor);
19:         else
20:            add(L,r,newIdAuthor());
21:         end if
22:      end if
23:   end if
24: end for
```

benefiting from eventual manual corrections.

In more details, INDi attempts to disambiguate the new citation records by looking for an existing author whose citation records in the DL have a similar author name, at least one coauthor in common and similar work or publication venue titles. For cases in which the new citation record does not have coauthors (i.e., citations records with only one author) or all the existing citation records in a group of an existing similar author do not have coauthors, we avoid the coauthor check, but raises similarity thresholds for publication venue and work title. When all these tests fail, the citation record is considered as belonging to a new author. The alternative to this last step, i.e., to consider only work and publication venue title similarities when there is no coauthor in common would most probably incur in many false positives decreasing the purity of the groups, the error we want to avoid the most. Following, we detail the algorithm used by INDi to disambiguate the citation records.

The main steps of our method are described in Algorithm 1. This algorithm receives as input a citation record c and a collection $\mathcal{C}$ in which c will be inserted, as well as two similarity thresholds α_{Venue} and α_{Title} that are used in the comparison between publication venue titles and work titles, respectively, and δ value that is used to increase the similarity thresholds. The result is a list of references with their corresponding authors.

Firstly, INDi preprocesses the citation record c by removing punctuations and stopwords from the work and publication venue titles. In addition it stems the work title using Porter's algorithm [Porter 1980]. For each reference r to an author extracted from the citation record c, INDi obtains from the collection $\mathcal{C}$ all citation records with an author name similar to the one of the reference r, groups the records that belong to the same author in the same cluster and preprocesses them (line 3 and 4).

(a)	new reference	**Antonio Oliveira**, Yalmar Ponce Atencio, *Claudio Esperanca*, Paulo Roma Cavalcanti. A collision detection and response scheme for simplified physically based animation. XVIII Simposio Brasileiro de Computacao Grafica e Processamento de Imagens.
(b)	existing cluster	**Antonio A. F. de Oliveira**, Paulo Sergio de Souza Coelho, *Claudio Esperanca*. A technique for compensating light source direction in face recognition applications. XI Simposio Brasileiro de Computacao Grafica e Processamento de Imagens.
		Antonio Alberto Fernandes de Oliveira, P. Coelho, *C. Esperanca*. An image processing and belief network approach to face detection. XII Simposio Brasileiro de Computacao Grafica e Processamento de Imagens.

Fig. 1. Illustrative Example. (a) New reference. (b) Cluster including a reference with a similar coauthor name.

From line 5 to 23, it attempts to find the author of the reference r in three steps. In Step 1 (lines 5 and 6), it searches for a cluster s that has an author name similar to the one in reference r. This candidate cluster must have at least one coauthor in common with r and either a similar work title or publication venue title with r. If a cluster with these characteristics is found, the reference r is assigned to the author of this cluster. Fig. 1 illustrates this case. The new reference (Fig. 1(a)) is assigned to an existing cluster (Fig. 1(b)) which, besides having an author name similar to that of the cluster (represented in bold in the figure), also includes a similar coauthor name ("C. Esperanca") and the same publication venue title ("Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens").

We notice that this procedure follows real-world-based heuristics similar to those adopted by HHC [Cota et al. 2010], i.e., very rarely two authors with similar names that share a coauthor in common would be two different people in the real world and authors tend to publish on the same subjects and venues for some portion of their careers.

If this test fails, in Step 2 (lines from 10 to 15) INDi checks the coauthor list of the reference r. In case this list is empty (i.e., the work has a sole author), it searches for a cluster similar to the reference r using the work or publication venue titles. If such a cluster is found, the reference r is assigned to its author. This would happen, for instance, if the new reference in Fig. 1(a) did not include a coauthor name.

Step 3 (lines from 17 to 21) is executed instead of Step 2 if the coauthor list of the reference r is not empty. It tests whether there is a cluster s with an empty coauthor list that is similar to r using work or publication venue titles, and if, the test is positive, the reference r is assigned to the cluster s. This would be the case if the references in the existing cluster in Fig. 1(b) did not include a coauthor name.

Particularly, in steps 2 and 3 the similarity thresholds used to verify the similarity between work or publication venue titles are increased by a value (δ). The intuition behind this is to find work or publication venue titles more similar to the ones in the reference r, since the reference r or the cluster s does not have coauthors, a stronger evidence. If all the tests in Steps-1-3 fail, we include r as belonging to a new author.

In our experimental evaluation (Section 4), for author and coauthor names, we use a function (*similarCoauthors*) based on the *Fragment Comparison* algorithm [Oliveira 2005], a pattern matching algorithm specially designed for persons' names. This algorithm takes as input two strings corresponding to two author names represented in some canonical form. Then, by using an edit distance function, it compares each of the individual fragments that compose those names (including initials), ignoring the order in which they appear in the input strings. For work and publication venue titles, we use the *similarTitle* and *similarVenue* functions which are based on the well-known cosine similarity metric [Salton et al. 1975]. In this case, we use each word in the work or publication venue title as a term and calculate the cosine similarity between a cluster[1] and a new citation record using feature vectors, in which each feature corresponds to the TF-IDF (i.e., term frequency and inverse document

[1]The cluster is represented by a summary vector of all the records presented in it.

frequency) value of its corresponding term[2]. If the similarity is greater than a given threshold, we assign the record to the respective cluster.

To analyze the complexity of the INDi algorithm, we estimate the number of comparisons performed by it. In this analysis we do not consider the cost of the similarity functions because they apply to small strings and are not affected by the size of the collection.

INDi is not executed on the whole DL, but only on the new load. To select the clusters with author names similar to that of a new reference, INDi compares the author name of such a reference with the representative author name of the clusters in the DL with cost $O(n_a)$, where n_a is the number of clusters in the DL. After, INDi matches each reference in the new load with each cluster. Thus, the cost is $O(n_r \cdot (n_a + n_p))$, where n_r is the number of references to authors in the new load and n_p is the number of previously selected clusters. Since these selected clusters are a subset of the total clusters, the final cost is $O(n_r \cdot n_a)$.

4. EXPERIMENTAL EVALUATION

In this section we present experimental results that demonstrate the effectiveness of our method. We first describe the collections, the baseline and the evaluation metric. Then, we discuss the effectiveness of our method in comparison with the baseline. We use real and synthetic collections in our experimental evaluation. Our motivation for using synthetic data is that we can generate controlled data, setting, for instance, the percentage of new authors to be inserted into the DL in each load, an important challenge for disambiguation methods.

4.1 Collections

As our aim is to check the behavior of our method when new citation records are inserted into the collection over time, we used a real dataset extracted from the BDBComp[3] digital library and datasets generated by SyGAR, a synthetic data generator that generates citations records simulating a living digital library that evolves according to various desired patterns. SyGAR and BDBComp datasets are described following.

SyGAR Datasets

SyGAR [Ferreira et al. 2009] is a tool for generating synthetic collections of citation records. It was designed to help with the evaluation of name disambiguation methods in realistic yet controlled scenarios, with evolving digital libraries. It is able to capture aspects of real collections that are key to disambiguation methods, and use them to generate synthetic datasets. These datasets may be larger and span longer periods of time but are representative of the real data with regard to the interest area of the authors.

SyGAR takes as input a real collection of previously disambiguated citation records, referred to as the *input collection*. Each such a record is composed of the three attributes most commonly exploited by disambiguation methods [Cota et al. 2010; Ferreira et al. 2010; Han et al. 2004; Han et al. 2005; Pereira et al. 2009], namely, a list of author names, a list of unique terms present in the work title, and the publication venue. Authors with the same ambiguous name, and their corresponding records, are organized into *ambiguous groups* (e.g., all authors with name “C. Chen"). SyGAR also takes as inputs several other parameters, such as number of loads to be synthesized, total number of records to be generated per load, probability of selecting a *new* coauthor, and so on, which are used in the data generation process. As output, SyGAR produces a representative list of synthetically generated

[2]Term frequencies are considered within the cluster, i.e., a term may be counted multiple times if it appears in several work or publication venue titles of different citation records in the same cluster.
[3]http://www.lbd.dcc.ufmg.br/bdbcomp

Table I. The BDBComp Dataset.

Ambiguous Group	Total Number of Records	Total Number of Distinct Authors
A. Oliveira	52	16
A. Silva	64	32
F. Silva	26	20
J. Oliveira	48	18
J. Silva	36	17
J. Souza	35	11
L. Silva	33	18
M. Silva	21	16
R. Santos	20	16
R. Silva	28	20

citation records, referred to as the corresponding *output collection*. Each generated record consists of the three aforementioned attributes.

For our experiments, SyGAR is used to generate cleaned collections of citation records and ten data loads representing data to be inserted into the collection in each year. As *input collection*, we use a collection extracted from DBLP. This collection sums up 4,272 records associated with 220 distinct authors, which means an average of approximately 20 records per author. Small variations of this collection have been used in several other works [Han et al. 2004; Han et al. 2005; Pereira et al. 2009]. Its original version was created by Han et al. [Han et al. 2004], and they manually labeled the records. For this, they used the authors' publication home page, affiliation name, e-mail, and coauthor names in a complete name format, and also sent emails to some authors to confirm their authorship. However, it should be noticed that there is not temporal information in this dataset which does not allows us to test the dynamic aspects of the evolution of the collection and their impact in the algorithms. The use of Sygar allow us to exploit this temporal dimension in a controlled way.

BDBComp Dataset

As a second dataset, we used a collection of citation records extracted from BDBComp summing up 363 records associated with 184 distinct authors, approximately 2 records per author. Notice that, despite being relatively small, this collection is very difficult to disambiguate, because it has many authors with only one citation record. This collection was created by us and contains the 10 largest ambiguous groups found in BDBComp from 1987 to 2007.

Table I shows more detailed information about the BDBComp dataset and its ambiguous groups. Disambiguation is particularly difficult in ambiguous groups such as the "F. Silva" group, in which the majority of authors has appeared in only one citation record.

As mentioned before, each citation record consists of the author name, a list of coauthor names, the title of the work, the title of the publication venue (conference or journal) and the publication year. Pre-processing involved standardizing coauthor names using only the initial letter of the first name along with the full last name and removing punctuation and stop-words of publication and venue titles.

4.2 Baseline

We choose the Heuristic-based Hierarchical Clustering (HHC) method [Cota et al. 2007; Cota et al. 2010] as our baseline. In [Cota et al. 2010], HHC is compared with other unsupervised and supervised methods proposed in the literature [Han et al. 2004; Han et al. 2005; Huang et al. 2006]. In that comparison HHC achieved the best results in the disambiguation task, significantly outperforming even supervised methods based on state-of-the-art learning techniques (e.g., Support Vector Machines).

HHC attempts to resolve the name ambiguity problem in two main steps after a preprocessing phase, in which it treats citation records by removing stop-words and stemming the words of work and publication venue titles, and then groups references corresponding to ambiguous authors (i.e.,

authors with similar names).

In the first step, HHC receives a collection of citation records $\mathcal{C}$ and returns a list $\mathcal{G}$ of clusters of references to the same author. It processes $\mathcal{C}$ by splitting it into two separate lists: S with records whose author names occur in a short format and L with the remaining ones (i.e., those records whose ambiguous author names are not in a short format). HHC proceeds by first processing L and then S. When processing the lists L and S, HHC groups the references that have similar author name and at least one coauthor name in common. This process generates very pure (i.e., with few wrong references to the same author within a cluster) but very fragmented (i.e., many references related to a same author are spread in different clusters) sets of clusters.

In the second step, HHC tries to reduce fragmentation by using additional information from the citation attributes in the initial clusters (in our implementation we used work and publication venue titles) in order to fuse them. Thus, in the second step, HHC proceeds in a hierarchical manner and fuses the clusters created in step one based on their contents. Clusters are pairwise compared and, to determine whether two clusters can be fused, HHC estimates their similarity based on the elements that compose the work and publication venue title of their respective citations, if the names of the authors of the clusters are similar. If the estimated similarity of the elements of either these attributes is higher than a given threshold, work and venue threshold, the two clusters are fused. This process continues until the termination criterion is met, i.e., until no more clusters can be fused. The final result is a list of clusters with their respective references.

The complexity of HHC, considering the number of comparisons, is $O(n_r^2 + n_c^2 \cdot log\ n_c)$ *in the average case*[4], where n_r is the total number of references in the DL[5] and n_c is the total number of clusters generated in this first iteration (which is usually smaller than n_r). Initially, HHC always processes the whole DL comparing all its references to each other in the first iteration. After this, the process occurs in a hierarchical fashion (with the hierarchical fusion of the clusters). The average case in this scenario occurs when half of the clusters are matched at each new iteration. In the worst case, when only one cluster is matched at each new iteration, the complexity is $O(n_r^2 + n_c^3)$.

4.3 Evaluation Metrics

In order to evaluate the effectiveness of the proposed disambiguation method, we used the K metric [Lapidot 2002]. This metric determines a trade-off between the average cluster purity (ACP) and the average author purity (AAP). Given a collection, ACP evaluates the purity of the empirical clusters (i.e., clusters of references to authors provided by the methods) with respect to the theoretical clusters for this collection. Thus, if the empirical clusters are pure (i.e., they contain only authorship records associated with the same author), the corresponding ACP value will be 1. ACP is defined in Equation 1:

$$\mathrm{ACP} = \frac{1}{N} \sum_{i=1}^{e} \sum_{j=1}^{t} \frac{n_{ij}^2}{n_i} \tag{1}$$

where N is the total number of references to authors, t is the number of theoretical clusters, e is the number of empirical clusters, n_i is the total number of references in the empirical cluster i, and n_{ij} is the total number of references in the empirical cluster i which are also in the theoretical cluster j.

For a given collection, AAP evaluates the fragmentation of the empirical clusters with respect to the theoretical clusters. If the empirical clusters are not fragmented, the corresponding AAP value will be 1. AAP is defined in Equation 2:

$$\mathrm{AAP} = \frac{1}{N} \sum_{j=1}^{t} \sum_{i=1}^{e} \frac{n_{ij}^2}{n_j} \tag{2}$$

[4]We did not consider any type of blockage in this analysis.
[5]This is usually much bigger than the number of references in the new load

Table II. Distribution of Average Number of Publications per Year per Author (DBLP: 1984 - 2008).

	Average Number of Publications per Year			
	One	Two	Three	> Four
New Authors	55%	30%	10%	5%
Existing Authors	14%	42%	28%	16%

Table III. The Parameter Values used by INDi and HHC in Each Dataset.

	INDi			HHC	
Dataset	α_{Title}	α_{Venue}	δ	title threshold	venue threshold
BDBComp	0.0	0.2	0.2	0.4	0.4
Synthetic 5	0.1	1.0	0.2	0.3	1.0
Synthetic 10	0.1	0.9	0.2	0.2	0.6

where n_j is the total number of references in the theoretical cluster j.

The K metric consists of the geometric mean between ACP and AAP values. It evaluates the purity and fragmentation of the empirical clusters extracted by disambiguation methods. The K metric is given in Equation 3:

$$K = \sqrt{\text{ACP} \times \text{AAP}} \tag{3}$$

4.4 Results

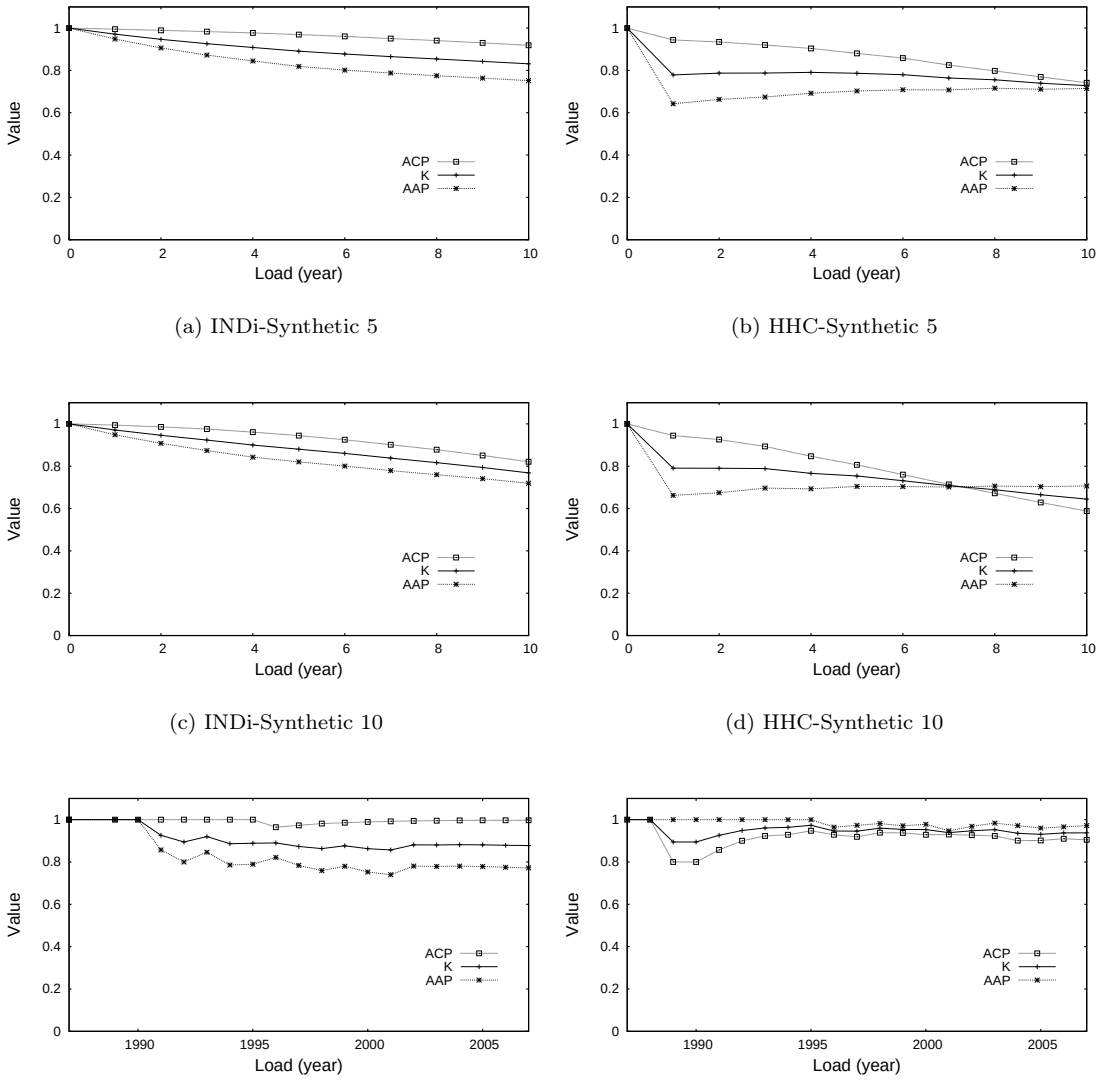
In order to evaluate INDi, we compare its performance with that of HHC in the BDBComp and the synthetic datasets generated by SyGAR, simulating DLs that evolve over a period of 10 years. Each synthetic dataset was generated containing, in its initial state (year), the same number of records of DBLP static collection (4272 citation records) and, at the end of each year, a new load with new citation records are inserted into the DLs. In each year, the number of citations generated for each author follows the average yearly publication rates of authors shown in Table II. These values were extracted from DBLP by counting the number of publications of each author of three selected ambiguous groups during the period of 1984-2008. We selected groups with very different author population sizes. These groups compose the SyGAR input collection extracted from DBLP.

New authors are added to the synthetic datasets at a given rate in each successive load. We experiment with rates of 5% and 10% of new authors, in comparison with the current population of the dataset. Using both rates, for each load (year), SyGAR generated five datasets, using different random seeds for data generation. In our evaluation, the results in the synthetic datasets in each point in time (load) correspond to the average results using the five datasets, obtained by HHC (our baseline) and INDi (our method). Hereafter, we will refer to the group of synthetic datasets created with rates of 5% and 10% of new authors as Synthetic 5 and Synthetic 10, respectively.

In case of BDBComp, a real dataset, we used the date of publication of the citation records to simulate the yearly loads. It is obvious that, in this case, each yearly load (point in time) is unique, i.e., there are no multiples runs. In BDBComp, in average, 56% of the records are from new authors, a completely different behavior when compared to the synthetic datasets, which is justified by the characteristics of the Brazilian computing community.

To compare INDi and HHC, we use the best parameter values found for each method in each dataset. These parameter values are showed in Table III. Particularly, notice that the use of $\alpha_{Title} = 0.0$ by INDi in the BDBComp dataset means that in Step 1 of our method the candidate citation record has to have *some* similarity with the cluster of the correct author in order to be assigned to it, since the comparison forces the similarty to be greater, nor greater or equal, than the threshold.

Fig. 2 shows the results obtained by INDi and HHC in the datasets during each year (load). The initial state corresponds to a disambiguated DL. The performance at load 1 corresponds to the first



(a) INDi-Synthetic 5

(b) HHC-Synthetic 5

(c) INDi-Synthetic 10

(d) HHC-Synthetic 10

(e) INDi-BDBComp

(f) HHC-BDBComp

Fig. 2. Performance obtained by INDi and HHC on whole DL (i.e., the disambiguation of the DL obtained by INDi and HHC).

execution of each method. At each year, a new load of records is inserted into the DL and the results correspond to the execution of each method in terms of the ACP, AAP and K metrics. Notice that HHC reprocesses the whole DL each time a new load is added to the DL.

In Synthetic 5, we can notice that HHC has a sharp decline of the K value (see Fig. 2(b)) at the first year (load). This decline occurs due to the increase of fragmentation (AAP=0.642) while the purity of the clusters has a small decline (ACP=0.943). In the next years, HHC has a very slight decrease of the fragmentation while the purity remains falling more steadily. The final result of HHC after ten loads (K=0.728) is shown in Table IV. On the other hand, INDi does not present the same sharp decline at the first year (see Fig. 2(a)) and its performance decreases softly over time obtaining, after the last year, K=0.831. In the end of the considered period of successive loads INDi performance is 14% superior to the HHC's performance considering the K metric. We notice that HHC presents a

Table IV. Results Obtained by INDi and HHC at the Last Year with 95% of Confidence Interval.

	INDi			HHC		
Dataset	K	ACP	AAP	K	ACP	AAP
Synthetic 5	0.831±0.007	0.919±0.010	0.752±0.007	0.728±0.009	0.742±0.011	0.715±0.013
Synthetic 10	0.768±0.009	0.821±0.013	0.719±0.009	0.644±0.018	0.588±0.025	0.707±0.023
BDBComp	0.877	0.997	0.772	0.937	0.905	0.972

Table V. Running time (seconds) of INDi and HHC disambiguating Synthetic 10 dataset with 95% of confidence interval.

	Load (with average number of references)									
Method	1(4920)	2(5612)	3(6348)	4(7090)	5(7915)	6(8804)	7(9682)	8(10598)	9(11604)	10(12663)
INDi	0.81±0.16	1.17±0.10	1.51±0.15	1.92±0.21	2.91±0.24	3.71±0.39	4.56±0.40	5.64±0.47	7.10±0.80	8.40±0.68
HHC	11.50±1.06	14.89±0.81	17.49±1.79	21.51±1.11	25.79±1.96	31.48±1.98	38.83±1.91	45.93±2.30	59.05±5.01	73.07±5.30

sharp decline of AAP after the first load because, as it disambiguates the whole DL again, it may reintroduce some ambiguity that might have been previously corrected (remind that the initial state corresponds to a disambiguated DL).

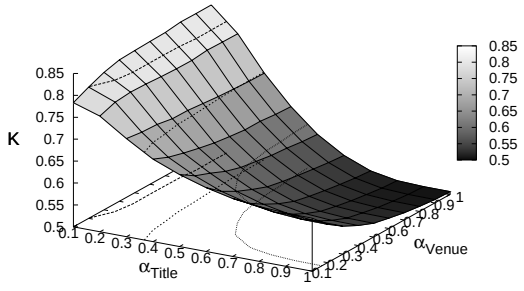
The behavior of INDi and HHC in Synthetic 10 (see Fig. 2(c) e (d)) is very similar to that obtained in Synthetic 5, but the result after 10 years is worst for both methods, as we should expect given the introduction of more new authors. Comparing the performance of both methods on Synthetic 5 against Synthetic 10, we notice that the performance of HHC decreases by about 11% while the performance INDi decreases by 7%. This shows that the introduction of records with new authors in the DLs hurts more the performance of HHC in its successive runs. Overall, after 10 successive loads the performance of INDi is around 19% superior to that of HHC.

We also run INDi and HHC in the BDBComp dataset over a period of eleven years, from 1987 to 2007 (see Fig. 2(e) e (f)). Using HHC, the fragmentation remains almost the same until 1995 while the purity decreases in the first two years improving a bit from 1991 until 1995. After this period, the purity and fragmentation have small variations with regards the K, ACP and AAP metrics. With INDi, the K metric remains equals to 1 until 1990, the purity remains very high throughout the whole period while the fragmentation sharply increases from 1990 to 1992 and remains in similar levels over the remainder years. By the end of the eleven-year period the values of K, ACP and AAP are 0.877, 0.997 and 0.772, respectively for INDi. HHC outperforms INDi by around 6% when considering the K metric in 2007. However, it is very important to notice that the purity of the groups produced by INDi remains very high over the entire period, an important situation in disambiguation since, as said before, mixing citations records of different authors is a much harder problem to fix than putting together split groups in an eventual manual correction. We recall that only INDi can benefit from these manual corrections when compared to other unsupervised methods such as HHC. We will further discuss issues related to the results in the BDBComp dataset in the analysis of cases of failure in Section 5.

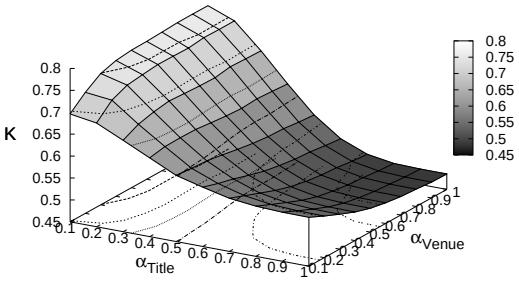
To investigate efficiency issues of INDi, we compare its running time for disambiguating the Synthetic 10 dataset with the corresponding running time spent by HHC. The experiments were executed on an Intel Xeon E5620 machine with a clock of 2.40GHz, 8 GBytes of RAM memory and running Linux. We notice that both methods were implemented in Java. Table V shows the time spent by each method in seconds.

As we can see, INDi is much faster than HHC. Moreover, the time spent by HHC increases more sharply as new loads are added to the collection. These results are in consonance with the complexity analysis presented before.

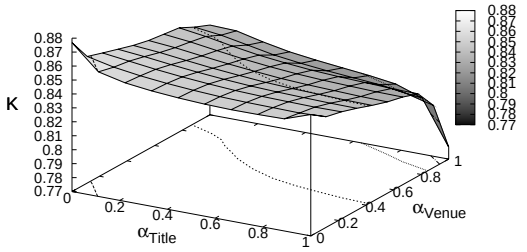
Finally, we analyze the sensitivity of INDi to the several parameters. Fig. 3 shows the average K values obtained by INDi after 10 loads, varying α_{Venue} and α_{Title} similarity thresholds, in the Synthetic 5, Synthetic 10 and BDBComp datasets, using $\delta = 0.2$. We notice that the best K values are obtained with low values of α_{Title}, for instance, with $\alpha_{Title} = 0.1$ and $\alpha_{Venue} = 0.9$ the K values are equal to 0.831 and 0.768 in Synthetic 5 and Synthetic 10 datasets, respectively. The worst K values are obtained with high values of α_{Title}, for instance, with $\alpha_{Title} = 1.0$ and $\alpha_{Venue} = 1.0$ the K



(a) Synthetic 5



(b) Synthetic 10



(c) BDBComp

Fig. 3. Sensitivity analysis of the similarity thresholds α_{Title} and α_{Venue} (δ=0.2).

Table VI. Percentage of New and Existing Authors Correctly and Incorrectly Identified by INDi.

	New Authors		Existing Authors	
Dataset	Correct	Incorrect	Correct	Incorrect
Sinthetic 5	41.803	58.197	87.289	12.711
Sinthetic 10	31.235	68.765	88.723	11.277
BDBComp	99.507	0.493	62.821	37.179

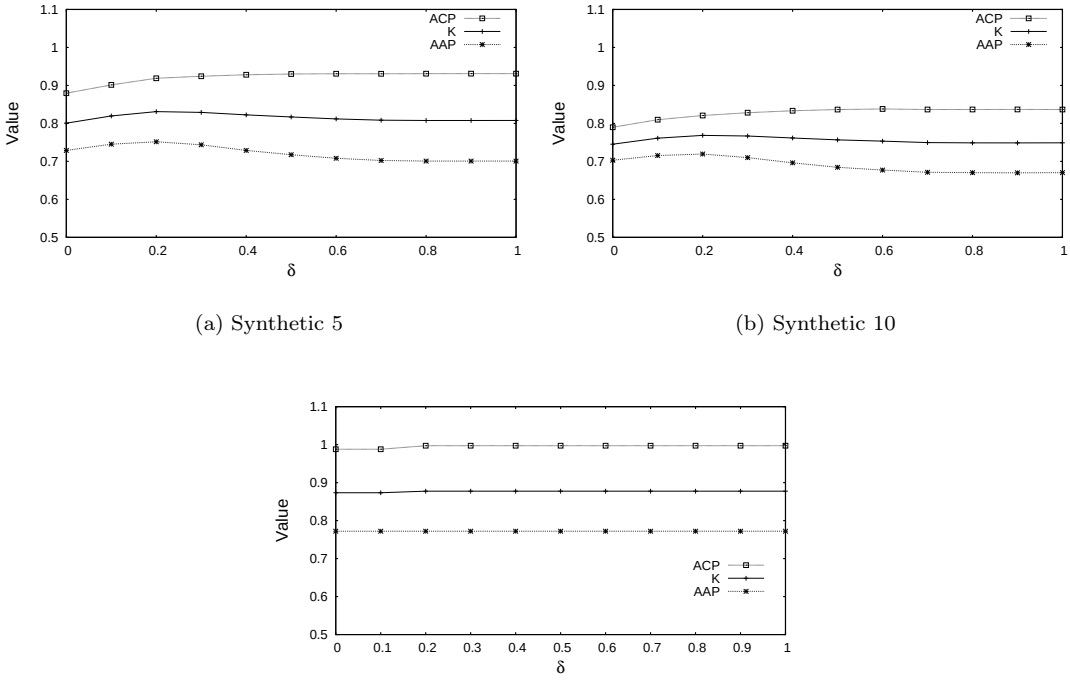
value are equal to 0.507 and 0.485 in the Synthetic 5 and Synthetic 10 datasets, respectively. Using the BDBComp dataset, the performance of INDi decreases when the α_{Venue} is set to higher values.

Fig. 4 shows the average K values obtained by INDi after 10 loads, varying the δ value. For this analysis, we use the best values of α_{Title} and α_{Venue} in each dataset. We notice that $\delta = 0.2$ produces the best results in all datasets and that, by using other values, there is little, if any, impact in the INDi effectiveness.

5. ANALYSIS OF CASES OF FAILURE

In this section, we discuss some of our results, mainly analyzing the cases of failure of our proposed method. We start by showing in Table VI the percentage of citation records of new and existing authors correctly and incorrectly identified by INDi.

In general, the percentage of correct identifications of existing authors is very high in the synthetic datasets although there is room for improving the percentages of identification of new authors. In any case, the overall results are very good (e.g., in terms of the K metric) given that, in these datasets, which were generated based on DBLP, the number of existing authors is much higher than the new



(a) Synthetic 5

(b) Synthetic 10

(c) BDBComp

Fig. 4. Sensitivity Analysis of the δ_{Venue} and $\delta Title$.

ambiguous ones (5% and 10% yearly). On the other hand, INDi is extremely effective in identifying new authors in BDBComp which has, comparatively, many more new authors, as discussed before, corresponding to authors publishing for the first time in a Brazilian national conference covered by this DL. As before, the overall performance is very high. It is interesting to notice how the changes in the parameter settings allow INDi to adapt itself and perform well in these completely different situations, showing its flexibility.

We turn our attention now to the incorrect assignments. For these, we were able to identify six of such cases:

- Failure 1. The new record has a similar coauthor name when compared with records of the correct author but does not have a similar work or publication venue title.
- Failure 2. The new record, which has coauthors, does not have coauthor names similar to the ones present in records of the correct author.
- Failure 3. The new record does not have coauthors and its work or publication venue titles are similar to those of records of an incorrect author.
- Failure 4. A cluster of an incorrect author, with records without coauthors, has records similar to the new record considering the work or publication venue titles.
- Failure 5. The new record has similar coauthor names and work or publication venue titles to those of an incorrect author.
- Failure 6. The new record does not have coauthors and its work or publication venue titles are not similar to any citation record of any author, but this record belongs to a pre-existing author in the DL. This record is considered as belonging to a new author.

Table VII shows six examples in which INDi fails in determining the correct author of a citation record for each case of failure. In the first example (Failure 1) INDi fails because the work and publication venue titles of the new record (R_1) are not similar to any of the records of the correct author (i.e., author of the records in cluster C_1), although they have coauthors with similar names.

Table VII. Examples of failure cases. The labels R_i's correspond to the identification of the new citation records. C_1, C_2 and C_6 correspond to the correct clusters of the authors and C_3, C_4 and C_5 correspond to the incorrect clusters assigned to the new citation record.

Failure	Label	Author	Coauthors	Work Title	Publication Venue Title
Failure 1	R_1	Ashish K. Gupta	D. Suciu, A. Halevy	Search Experi Novel Queri	WEBDB
	C_1	Ashish Gupta	M. Onizuka, A. Halevy, D. Raven, I. Campillo, V. Narasayya	Point Rotorua Xpath us Compress	SIGMOD
Failure 2	R_2	A. Mauro B. Oliveira	J. Peyrin	MINHONIX- A Distributed System for Teaching	SBRC
	C_2	Antonio Mauro Barbosa de Oliveira	J.Coelho Filho, J. Silva Jr., M. Vieira	Minhoca Plus, uma Rede Local para Fins Didaticos	SBRC
Failure 3	R_3	A. Gupta	-	Librari Fruit D	Interacting with Computers
	C_3	Avaneendra Gupta	J. Hayes J. Hayes J. Hayes	Optim Program Clip Compens Tool Spatial Object Program Invoc Nest Tool Compens Chain Spatial CMO Distribut Program Reader Cell Nest	DAC Mobile HCI Mobile HCI
Failure 4	R_4	J. Robinson	A. Burns	Delai Stabl Develop VLSI Dialogu Knowledg Test	INFOCOM
	C_4	J. Robinson	- - - -	Repli Web Independ CLP Revenu Time Critic Site Outlin Evalu Real Diagram Discov	CACM CACM WWW SIGMETRICS
Failure 5	R_5	A. Gupta	Z. Yang	Sat Applic Comput Based Imag Detect	FMCAD
	C_5	A. Gupta	Z. Yang, T. Blalock, A. Biere P. Ashar, O. Shrichman, Z. Yang	Condit Fring Comput Imag Size Capacit Claus Applic Manipul Demov Reachab Detect BDD Comput Imag Based Applic Sat	FMCAD FMCAD
Failure 6	R_6	Lena Veiga e Silva	-	Um Servico de Auto-arquivamento de Publicacoes Cientificas Compativel com o Padrao OAI	Informatica Publica
	C_6	Lena Veiga e Silva	D. Pozo, A. Laender, M. Goncalves	Modelagem de Bibliotecas Digitais usando a Abordagem 5S: Um Estudo de Caso	SBBD

In this case, the new record is assigned to a new author, i.e., an author that does not belong to the DL yet. This could have been avoided by considering only the coauthorship in Step 1, but we have decided to adopt a stronger heuristics that also considers the relatedness of the works as given by the similarity of their titles as well as of the titles of their publication venues. Remind that one of our goals is to preserve as much as possible the purity of the clusters while also being able to correctly identify new authors.

The second example corresponds to Failure 2. The new citation record (R_2) also was erroneously assigned to a new author in the DL, because R_2 corresponds to the first work of the author "A. Mauro B. Oliveira" with the coauthor "J. Peyrin". In fact, the majority of INDi's failures in the BDBComp dataset falls in this category. This happens because in this DL there is a much higher percentage of records that represent the first publication of an ambiguos author with a new coauthor with whom she has never published before. In this case INDi considers this author as new, although she may already have a few publications in the dataset. An alternative for DLs that present similar patterns is to relax our heuristic by considering, in the cases in which there is no coauthor in common, only the similarity between work and publication venue titles, but using higher thresholds in order to avoid decreasing purity too much. We made this small modification to INDi and run it on the BDBComp

dataset. By using $\delta = 0.4$, which is basically twice the value used before, we obtained a new K value of 0.906 that is only around 3% worst than the one obtained by HHC while still keeping high levels of purity (ACP=0.936). We leave for future work the study of strategies to decide when is worth trying this small relaxation or not. We also expect that eventual manual corrections in the dataset will help to ameliorate this situation.

The third example is related to Failure 3. In this case, INDi fails in the second step. The new citation record (R_3) does not have coauthors and its publication venue title is similar to the publication venue titles of records of an incorrect ambiguous author (C_3). In this case INDi incorrectly assigns the citation record R_3 to the author of the cluster C_3.

The fourth example is related to Failure 4, which occurs in the third step. The citation records in cluster C_4 do no have neither coauthor names nor venue titles similar to the the new citation record R_4, but both have similar work titles. So record R_4 is incorrectly assigned to the author of the records in C_4.

In fifth example, corresponding to Failure 5, INDi incorrectly assigns the new citation record R_5 to the author of the record in the cluster C_5 because both have coauthors in common and similar terms in work and publication venue titles. This failure is very hard to fix.

In the last example, which corresponds to Failure 6, the new citation record R_6 does not include any coauthor name and, because there are no similar terms between the work and the publication venue titles, it is incorrectly assigned to a new author.

6. CONCLUSIONS

In this article, we present INDi, a method that disambiguates the new citation records inserted into a cleaned digital library. INDi is able to detect whether the new records belong to pre-existing authors in the DL (i.e., authors already having records in the DL) or to a new author. Being unsupervised, INDi does not need any training.

Our experimental evaluation, using datasets generated by a synthetic data generator, shows gains of up of to 19% when compared to a state-of-the-art method without the cost of having to disambiguate the whole DL at each new load or the need for any training. Using data extracted from BDBComp, INDi presents small loses when compared to the same baseline. These loses are mainly due to the fact that the new records being inserted correspond to the first works with new coauthors of an existing author. However, in this dataset, INDi produces fewer cases of mixed citations, which is a problem that is much harder to manually fix afterwards. Again, we recall that INDi is the only disambiguation method that can benefit from these manual corrections which do occur in real digital libraries as the automatic methods are not perfect. INDi is also able to identify basically all the new authors in BDBComp.

As future work, we intend to investigate and propose alternatives to properly address the cases of failure generated by our method. We also want to design strategies to automatically discover the best thresholds for a given dataset.

REFERENCES

Bhattacharya, I. and Getoor, L. A latent dirichlet model for unsupervised entity resolution. In *Proceedings of the Sixth SIAM International Conference on Data Mining*. Bethesda, MD, USA, pp. 47–58, 2006.

Bhattacharya, I. and Getoor, L. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data* 1 (1): 1–35, 2007.

Cota, R. G., Ferreira, A. A., Gonçalves, M. A., Laender, A. H. F., and Nascimento, C. An unsupervised heuristic-based hierarchical method for name disambiguation in bibliographic citations. *Journal of the American Society for Information Science and Technology* 61 (9): 1853–1870, 2010.

Cota, R. G., Gonçalves, M. A., and Laender, A. H. F. A heuristic-based hierarchical clustering method for author name disambiguation in digital libraries. In *Proceedings of the XXII Brazilian Symposium on Databases*. João Pessoa, Paraiba, Brazil, pp. 20–34, 2007.

Culotta, A., Kanani, P., Hall, R., Wick, M., and McCallum, A. Author disambiguation using error-driven machine learning with a ranking loss function. In *Proceedings of the International Workshop on Information Integration on the Web*. Vancouver, Canada, 2007.

Fan, X., Wang, J., Pu, X., Zhou, L., and Lv, B. On graph-based name disambiguation. *ACM Journal of Data and Information Quality* 2 (2): 10:1–10:23, 2011.

Ferreira, A. A., Gonçalves, M. A., Almeida, J. M., Laender, A. H. F., and Veloso, A. SyGAR - A Synthetic Data Generator for Evaluating Name Disambiguation Methods. In *Proceedings of the 13th European Conference on Research and Advanced Technology for Digital Libraries*. Corfu, Greece, pp. 437–441, 2009.

Ferreira, A. A., Veloso, A., Gonçalves, M. A., and Laender, A. H. F. Effective self-training author name disambiguation in scholarly digital libraries. In *Proceedings of the 2010 ACM/IEEE Joint Conference on Digital Libraries*. Gold Coast, Queensland, Australia, pp. 39–48, 2010.

Gonçalves, M. A., Fox, E. A., Watson, L. T., and Kipp, N. A. Streams, structures, spaces, scenarios, societies (5s): A formal model for digital libraries. *ACM Transaction Information System* 22 (2): 270–312, 2004.

Han, H., Giles, C. L., Zha, H., Li, C., and Tsioutsiouliklis, K. Two supervised learning approaches for name disambiguation in author citations. In *Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries*. Tucson, AZ, USA, pp. 296–305, 2004.

Han, H., Xu, W., Zha, H., and Giles, C. L. A hierarchical naive Bayes mixture model for name disambiguation in author citations. In *Proceedings of the 2005 ACM Symposium on Applied Computing*. Santa Fe, New Mexico, USA, pp. 1065–1069, 2005.

Han, H., Zha, H., and Giles, C. L. Name disambiguation in author citations using a k-way spectral clustering method. In *Proceedings of the 5th ACM/IEEE Joint Conference on Digital Libraries*. Denver, CO, USA, pp. 334–343, 2005.

Huang, J., Ertekin, S., and Giles, C. L. Efficient name disambiguation for large-scale databases. In *Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases*. Berlin, Germany, pp. 536–544, 2006.

Kanani, P., McCallum, A., and Pal, C. Improving author coreference by resource-bounded information gathering from the web. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*. Hyderabad, India, pp. 429–434, 2007.

Kang, I.-S., Na, S.-H., Lee, S., Jung, H., Kim, P., Sung, W.-K., and Lee, J.-H. On co-authorship for author disambiguation. *Information Processing & Management* 45 (1): 84–97, 2009.

Lapidot, I. Self-Organizing-Maps with BIC for Speaker Clustering. Tech. rep., IDIAP Research Institute, Martigny, Switzerland, 2002.

Levin, F. H. and Heuser, C. A. Evaluating the use of social networks in author name disambiguation in digital libraries. *Journal of Information and Data Management* 1 (2): 183–197, 2010.

Oliveira, J. W. A. *A Strategy for Removing Ambiguity in the Identification of the Authorship of Digital Objects*. M.S. thesis, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil, 2005. (in Portuguese).

On, B.-W. and Lee, D. Scalable name disambiguation using multi-level graph partition. In *Proceedings of the 7th SIAM International Conference on Data Mining*. Minneapolis, Minnesota, USA, pp. 575–580, 2007.

Pereira, D. A., Ribeiro-Neto, B. A., Ziviani, N., Laender, A. H. F., Gonçalves, M. A., and Ferreira, A. A. Using web information for author name disambiguation. In *Proceedings of the 2009 ACM/IEEE Joint Conference on Digital Libraries*. Austin, TX, USA, pp. 49–58, 2009.

Porter, M. F. An algorithm for suffix stripping. *Program* 14 (3): 130–137, 1980.

Salton, G. M., Wong, A., and Yang, C. S. A vector space model for automatic indexing. *Communications of the ACM* 18 (11): 613–620, 1975.

Soler, J. M. Separating the articles of authors with the same name. *Scientometrics* 72 (2): 281–290, 2007.

Song, Y., Huang, J., Councill, I. G., Li, J., and Giles, C. L. Efficient topic-based unsupervised name disambiguation. In *Proceedings of the 7th ACM/IEEE Joint Conference on Digital Libraries*. Vancouver, BC, Canada, pp. 342–351, 2007.

Tang, J., Zhang, J., Zhang, D., and Li, J. A unified framework for name disambiguation. In *Proceeding of the 17th international conference on World Wide Web*. Beijing, China, pp. 1205–1206, 2008.

Torvik, V. I. and Smalheiser, N. R. Author name disambiguation in medline. *ACM Transactions on Knowledge Discovery from Data* 3 (3): 1–29, 2009.

Treeratpituk, P. and Giles, C. L. Disambiguating authors in academic publications using random forests. In *Proceedings of the 2009 ACM/IEEE Joint Conference on Digital Libraries*. Austin, TX, USA, pp. 39–48, 2009.

Yang, K.-H., Peng, H.-T., Jiang, J.-Y., Lee, H.-M., and Ho, J.-M. Author name disambiguation for citations using topic and web correlation. In *Proceedings of the 12th European Conference on Research and Advanced Technology for Digital Libraries*. Aarhus, Denmark, pp. 185–196, 2008.