

On the combination of domain-specific heuristics for author name disambiguation: the nearest cluster method

Alan Filipe Santana¹ · Marcos André Gonçalves¹ · Alberto H. F. Laender¹ · Anderson A. Ferreira²

Received: 2 December 2014 / Revised: 9 June 2015 / Accepted: 22 June 2015 / Published online: 7 July 2015
© Springer-Verlag Berlin Heidelberg 2015

Abstract Author name disambiguation has been one of the hardest problems faced by digital libraries since their early days. Historically, supervised solutions have empirically outperformed those based on heuristics, but with the burden of having to rely on manually labeled training sets for the learning process. Moreover, most supervised solutions just apply some type of generic machine learning solution and do not exploit specific knowledge about the problem. In this article, we follow a similar reasoning, but in the opposite direction. Instead of extending an existing supervised solution, we propose a set of carefully designed heuristics and similarity functions, and apply supervision only to optimize such parameters for each particular dataset. As our experiments show, the result is a very effective, efficient and practical author name disambiguation method that can be used in many different scenarios. In fact, we show that our method can beat state-of-the-art supervised methods in terms of effectiveness in many situations while being orders of magnitude faster. It can also run without any training information, using only default parameters, and still be very competitive when compared to these supervised methods (beating several of them)

and better than most existing unsupervised author name disambiguation solutions.

Keywords Name disambiguation · Supervised methods · Heuristics

1 Introduction

It is a consensus that author name disambiguation (AND) has been one of the hardest problems faced by digital libraries since their early days. This can be demonstrated by the large volume of literature published on the topic in the last decade (e.g., a recent survey cites literally dozens of works [7]) and the continuous interest by the research community in the problem. Although some efforts do exist to provide a global unique identifier to all authors, this will not work in all cases, for instance, while processing textual citations in papers for bibliographic analysis. Thus, automatic solutions, which are highly effective, efficient and practical in most situations, are still in need.

Most automated solutions in the literature exploit either some problem-specific heuristics to define similarity functions to be used by clustering or author assignment solutions or exploit supervised methods that learn such functions [7]. Historically, supervised solutions have empirically outperformed the ones based on heuristics, with the burden of having to rely on manually labeled training sets for the learning process. Such training sets are usually very expensive and cumbersome to obtain. Furthermore, such supervised solutions may not be practical at all in real-world situations in which new ambiguous authors (not present in a “static” training set) do appear all the time and changes in the publication patterns of known authors are common.

Moreover, most supervised solutions just apply some type of generic machine learning solution and do not exploit

✉ Marcos André Gonçalves
mgoncalv@dcc.ufmg.br

Alan Filipe Santana
alanfs@dcc.ufmg.br

Alberto H. F. Laender
laender@dcc.ufmg.br

Anderson A. Ferreira
ferreira@iceb.ufop.br

¹ Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, 31270-901 Belo Horizonte, Brazil

² Departamento de Computação, Universidade Federal de Ouro Preto, 35400-000 Ouro Preto, Brazil

specific knowledge about the problem that is usually embedded in the heuristic-based solutions. The few exceptions we know [25] extend some generic supervised solutions to consider a few aspects inherent to the problem, thus presenting the best effectiveness results reported in the literature.

In this article, we follow a similar reasoning, but in the opposite direction. Instead of extending an existing supervised solution, we propose a set of carefully designed heuristics and similarity functions based on our experience of almost a decade working on the problem. We use the proposed heuristics and similarity functions to find the nearest author, represented by a cluster of citations (thus the name of our method: nearest cluster), and assign the ambiguous citation to that author. If no cluster is “near enough”, the method assumes that a new author is being inserted into the digital library. There are also some heuristics to incorporate reliable predictions into the disambiguation process and to merge clusters of similar authors. As our solution requires some parameters to be defined, supervision is used mostly to optimize such parameters for each particular dataset. Notice, however, that our solution can be run without such supervised step as it does not learn any particular model from the training set; the model is already encoded in the heuristics and similarity functions we propose.

Our experiments with several collections, using only the minimum amount of information present in bibliographic citations, namely, author names and publication and venue titles, demonstrate that our proposed method has a number of desired properties. First of all, it is highly effective—in the experiments our method has produced significant effectiveness gains against the best state-of-the-art (supervised) method. Second, it is highly efficient, having a low computational complexity. In fact, when compared to this aforementioned method, our solution is orders of magnitude faster (about 6,500 times). If compared with another fast “natural” baseline, our method is as fast as it but up to 58 % more effective. And third, it is highly practical—besides using only the minimum amount of information available in a citation (and thus not relying on other types of information usually hard to obtain or simply not available at all, such as author affiliation or emails), we show that our method is very insensitive to several parameters and very easy to configure using some general rules.

In addition, the proposed solution can run without any supervision, still producing very reasonable results with default parameters, or by exploiting a previously proposed technique for automatically building training sets for AND tasks [6,9]. Moreover, our method can run in an incremental way, disambiguating only new citations inserted to the digital library (differently from clustering solutions that always disambiguate the digital library as whole) and is able to automatically incorporate new information to the process (e.g.,

the existence of new authors not previously seen in the training data).

We also perform a thorough analysis of cases of error of our method, i.e., cases in which citations were assigned to the wrong authors. This analysis shows that more than half of the errors are due to the lack of information in the training data or due to the very high ambiguity of the references, showing that improvements are very difficult to obtain and that we are almost on the limits of the effectiveness that can be achieved with such type of solution.

Finally, we provide a qualitative analysis of our solution considering all baselines used for comparison in our experiments and show that our proposed method possesses most of the qualities of a good AND solution should have, mainly when compared to the alternative approaches.

This article is organized as follows. Section 2 covers related work. Section 3 describes our proposed method, including details about how its parameters are estimated. Section 4 details our experimental evaluation. Finally, Sect. 5 concludes the paper, including perspectives for future work.

2 Related work

According to Ferreira et al. [7], AND methods may be broadly classified in two categories according to the type of approach they perform: author grouping methods and author assignment methods.

2.1 Author grouping methods

Author grouping methods exploit the similarity among the citations in order to group them by using some clustering technique. In this category, we may cite several methods [2, 4, 5, 12, 14, 19, 23, 24, 26].

Bhattacharya and Getoor [2] propose a combined similarity function based on terms present in a citation and relational information between disambiguated coauthor names of the citations. A greed agglomerative algorithm uses such a function to group the most similar citations in clusters.

Cota et al. [4] propose a heuristic-based hierarchical clustering method (HHC) that is based on two assumptions: (1) very rarely two citations with similar author names sharing similar coauthor names are two different authors and (2) authors usually publish on the same subjects or venues during their careers. HHC has two steps. The first step creates clusters of citations using the coauthor names and the second step successively fuses similar clusters based on the publication and venue titles. Each cluster contains aggregated information of all citations for each attribute in the cluster, providing more information for the next round of fusion.

In [5], Fan et al. propose GHOST (GrapHical framewOrk for name diSambiguaTion), a framework that represents a collection of citations as a graph in which each vertex represents a citation and each undirected edge represents a coauthorship between two citations. Fan et al. also propose a new similarity function based on the formula that calculates the resistance of a parallel circuit and use the Affinity Propagation clustering algorithm to group the citations of a same author.

Han et al. [12] propose the use of K-way spectral clustering with QR decomposition to obtain a given number of citation clusters where each cluster is associated to an author. To calculate the similarity among the citations, Han et al. apply the cosine similarity function on the references.

Huang et al. [14] exploit DBSCAN, a density-based clustering algorithm, for clustering references by author. The similarity function used by DBSCAN is learned by an active support vector machine algorithm (LaSVM), representing the comparison between two citations by a similarity vector, where each feature represents the comparison of an attribute of the two citations.

In [26], Wu et al. propose a hierarchical agglomerative clustering algorithm based on Dempster-Shafer theory in combination with Shannons entropy to disambiguate the author names. Dempster-Shafer theory fuses evidences to obtain more reliable candidate clusters for fusion and Shannons entropy uncovers the importance of each feature.

Some works propose disambiguating author names in PubMed¹ citations [19,23,24]. Torvik et al. [23] propose to learn a probabilistic metric for determining the similarity among PubMed citations while, in [24], Treeratpituk and Giles use a Random Forest classifier to learn the similarity function between citations. Torvik et al. [23] also propose a heuristic for automatically generating training examples and a new agglomerative clustering algorithm for grouping citations of a same author. In [19], Liu et al. present a system for disambiguating author names in PubMed that automatically obtains training examples based on low-frequency author names and pairs of citations in different ambiguous groups, besides using a Huber classifier to learn weight functions jointly with an agglomerative clustering technique to group the citations of a same author.

2.2 Author assignment methods

Author assignment methods directly assign the citations to their corresponding authors using either a supervised classification [6,8,10,25] or a model-based clustering technique [11,22]. These methods use a training set or perform in an interactive way to obtain models to predict the author of a citation. For example, in [10], Han et al. propose two

supervised methods based on naïve Bayes and Support Vector Machines learning techniques. Both methods learn a disambiguation function using a set of training examples to predict the author of each citation. Aiming to eliminate the set of training examples, in [11], Han et al. present an unsupervised hierarchical version of the naïve Bayes-based method for modeling each author by estimating the parameters using the Expectation Maximization Algorithm.

In [22], Tang et al. present a probabilistic framework based on Hidden Markov Random Models for the polysemy subproblem. In such a work, the authors use evidence based on content (i.e., terms of the citation attributes) and relationships between citations (e.g., coauthor names in common) for disambiguating author names. The authors also use Bayesian Information Criterion to estimate the number of authors of a collection.

In [25], Veloso et al. propose SLAND, a method that infers the author of a citation by using a supervised rule-based associative classifier. The method is also capable of improving the coverage of the training set by means of reliable predictions, and detecting authors without any citation in the training set. Aiming to reduce the number of training examples to compose the training set, Ferreira et al. [8] present SAND, a new active sampling strategy based on associative rules for the disambiguation task. Then, in [6,9], they extend their previous work to eliminate the need of manually providing the training examples.

2.3 Final remarks

Besides classifying the methods according to the type of approach they use, alternatively, we can group them according to the evidence exploited in the AND task: citation attributes (only) [4–6,10,24,25], Web information [15,16,20], or implicit data that can be extracted from the available attributes [21]. Some methods [23] also assume the availability of additional information such as emails, affiliations, addresses, paper headers, which is not always available or easy to obtain, although, if existent, may help a lot the process.

Our method, which can be considered an author assignment one, follows a totally different path when compared to what have been historically done in AND tasks. In here, instead of simply applying a generic machine learning solution or adapting it to the problem, we use our accumulated experience on the problem to propose a set of domain-specific heuristics to solve the AND task, using supervision (i.e., training data) only to adapt the method to specific idiosyncrasies of the target dataset. Notice that our proposed method can work without any supervision by using a set of default parameters. We also take advantage of a strategy previously proposed in the literature [6,9] for automatically building training sets for AND tasks with no manual supervision. Our

¹ <http://www.ncbi.nlm.nih.gov/pubmed/>.

Table 1 Notation table

Notation	Description
$c_i = c_i^c \cup c_i^t \cup c_i^v$	Citation represented as sets of terms for coauthors (c_i^c) and publication and venue titles (c_i^t and c_i^v , respectively)
$c_i^x = \{t_l, t_m, \dots\}$	Attribute of a citation represented as a set of terms
$a_i = \{c_l, c_m, \dots\}$	Author i represented as a group of citations
$A = \{a_1, a_2, \dots, a_n\}$	Set of authors defined in the training data
$D = \bigcup_{i=1}^n a_i$	Set of citations in the training data
$n_{t_i} = \{a_j : a_j \in A \wedge \exists c_l \in a_j, t_i \in c_l\} $	Number of authors who used term t_i in some citation
$\text{freq}_{t_i} = \{c_j : c_j \in D \wedge t_i \in c_j\} $	Number of citations having term t_i
$\text{freq}_{i,j} = \{c_k : c_k \in a_j \wedge t_i \in c_k\} $	Number of citations in group a_j that have term t_i
w_c, w_t, w_v	Weights assigned to the coauthor, publication and venue title attributes, respectively

experimental results show that our method combines aspects of effectiveness, efficiency and practicability rarely found in any other method proposed in the literature.

3 Proposed method

In this section, we present our proposed AND method and the procedures used to estimate its parameters. In our method, citations are represented as sets of terms occurring in the list of coauthors and publication and venue titles². From the list of coauthors, we obtain terms formed by the initial letter of their first name appended to the coauthors' last name. Terms in the publication and venue titles are obtained after the removal of stop words and stemming of the remaining words, by using Porter's algorithm [1]. For the equations defined next, we consider the notations presented in Table 1.

3.1 Nearest cluster model

An author publication profile may be characterized by the distribution of the terms found in her bibliographic citations. The list of coauthors captures her collaboration network while the set of terms in the venue and publication titles capture her research interests. Thus, each term present in a citation captures some evidence of the association between a citation and a reference in this citation. The strength of such an evidence varies according to the attribute to which it belongs and its discriminative capability. For instance, the presence of coauthors in common in two citations is a stronger evidence that these two citations belong to the same author than the occurrence of a common term in the titles. Such differences in the

importance of the attributes have been exploited by several disambiguation methods (e.g., [18]).

Based on such observations, a simple disambiguation method consists in defining a similarity function between authors (represented as a group/cluster of citations) and an ambiguous citation. Disambiguation of a citation c_k is performed by identifying the author whose group has the highest similarity with c_k . Equation 1 presents the similarity function between a citation c_k and an author a_j using the proposed method. Such function consists of a weighted sum of the similarities produced considering each attribute in isolation.

$$\text{sim}(c_k, a_j) = w_c f(c_k^c, a_j) + w_t f(c_k^t, a_j) + w_v f(c_k^v, a_j) \quad (1)$$

It should be noticed that a related idea is presented in [18] to identify false citations not belonging to an author due to homonyms. In that work, the similarities between vectors of terms in attributes and the set of citations of an author are computed using the traditional cosine similarity. We, on the other hand, propose our own similarity functions specific for the problem (see next). In any case, we use the above mentioned work as one of our baselines.

In this work, our proposed method is applied in a generic way for solving the disambiguation task, using our own similarity function defined as

$$f(c_k^x, a_j) = \frac{1}{|c_k^x|} \sum_{t_i \in c_k^x} w(t_i, a_j) \quad (2)$$

where,

$$w(t_i, a_j) = \begin{cases} \left(1 + \frac{1 - n_i}{n}\right) \left(\frac{\text{freq}_{i,j}^2 + 1}{|a_j| \text{freq}_{i,j} + 2}\right)^\alpha & \text{if } n_i > 0, \\ 0 & \text{otherwise.} \end{cases}$$

² We here work with the minimum amount of information found in a citation in order to illustrate the practical capabilities of our method in real-world scenarios, but in other contexts citations may include other attributes such as authors' affiliations or emails.

Function $w(t_i, a_j)$ weights the terms of each citation, returning a value between 0 and 1, defined according to the following problem-specific heuristics:

- The higher the number of authors who used a term t_i , the lower its discriminative power. The sum $(1 + (1 - n_i)/n)$ returns 1 if the terms are used only in one group or $1/n$ if the terms has been used in all groups.
- Given the occurrence of a term t_i , the value of the conditional probability $P(a_j|t_i)$ gives us evidence of the strength of the association between the citation that possesses term t_i and author a_j . Such a probability may be estimated by the fraction $f_{req_{i,j}}/f_{req_i}$, however such an estimation may be biased due to the high imbalance commonly found in training data for disambiguation tasks. It is well know that publication patterns have a very skewed distribution: few authors publish a lot while most authors have smaller figures. To consider and reduce this possible bias we multiply this estimate by the distribution of terms in the group, estimated by the fraction $f_{req_{i,j}}/|a_j|$.
- Considering that in disambiguation tasks the training data are usually not very large (due to the inherent costs in generating it), in order to produce accurate estimates of probabilities and terms' distributions we added constants (1 and 2, see Eq. 2) and, mainly, the α exponent to smooth the calculations of these factors. The value of the α parameter, defined in the interval $[0, 1]$, controls the influence on the discriminative factor in the final weight of the term. The value for this factor can be determined by cross-validation in the training set.

3.2 Beyond similarity: domain-specific heuristics

In the course of our 10-year history working on this problem, we have found that some domain-specific issues are very important and should be incorporated by any AND solution. For instance, treatment of “new” ambiguous authors, i.e., authors not previously known to the methods, and adaptability to changes in publication patterns, are two phenomena that do occur in real-world collections and must be taken into consideration. Thus, the similarity values between a citation and groups of citations associated with authors in the training set can be used to cope with both issues.

The main steps of our proposed method are shown in Algorithm 1. During disambiguation, we keep two sets of citations: the training set D , defined by a set of authors A and a set of already assigned citations E but not (yet) included in the training data D . Such division is performed to allow that doubtful citations be re-evaluated at the moment that new citations are included in the training set. In the following, we detail the steps of the algorithm referring to the adopted strategies to consider the automatic inclusion of new citations and new authors in the training set.

Algorithm 1 Name Disambiguation

```

Require: Training Set  $A$ , test citation  $c_k$ , set  $E$  of already assigned citations not (yet)
included in  $D$ 
Ensure: Associate an author  $a_l$  with a citation  $c_k$ 
1:  $G \leftarrow \emptyset$ 
2: for all  $a_j \in A$  do ▷ Calculating similarities
3:   Calculate  $\text{sim}(c_k, a_j)$  according to Equation 1
4:   if  $\text{sim}(c_k, a_j) > 0$  then
5:      $G \leftarrow G \cup \{a_j\}$ 
6:   end if
7: end for
8: Let  $a_l$  be the author with highest  $\text{sim}()$ 
9: Let  $a_m$  be the author with the second highest  $\text{sim}()$ 
10: Calculate  $\Delta(c_k)$  according to Equation 3

11: if  $\text{sim}(c_k, a_l) < \gamma$  then ▷ Checking for new author
12:    $a_z \leftarrow \{c_k\}$ 
13:    $A \leftarrow A \cup \{a_z\}$ 
14: else if  $\Delta(c_k) > \Delta_{min}$  then ▷ Reliable prediction?
15:    $a_l \leftarrow a_l \cup \{c_k\}$ 
16:   for all  $a_j \in G$  do ▷ Checking clusters similarities
17:     Let  $d_{l,j}$  be the cosine similarity between  $a_l$  and  $a_j$ 
18:     if  $d_{l,j} > \phi$  then
19:        $a_l \leftarrow a_l \cup a_j$ 
20:        $A \leftarrow A - \{a_j\}$ 
21:     end if
22:   end for
23:    $E_l \leftarrow \{c_j | c_j \in E \wedge c_j \cap c_k \neq \emptyset\}$ 
24:   for all  $c_j \in E_l$  do ▷ Checking doubtful predictions
25:     Update  $c_j$  similarities
26:     if  $\Delta(c_j) > \Delta_{min}$  then
27:       Include  $c_j$  in the training set
28:       Perform a cluster union, if necessary
29:      $E \leftarrow E - \{c_j\}$ 
30:   end if
31: end for ▷ Just classify and include citation in set E
32: else
33:   Associate an author  $a_l$  with a citation  $c_k$ 
34:    $E \leftarrow E \cup \{c_k\}$ 
35: end if

```

3.2.1 Including new citations into the training set

Adaptability issues try to cope with two aspects inherent to AND: lack of information in the current training set and the capability of capturing slowly changing publication patterns of existing authors (i.e., present in the training data). We cope with both issues by proposing heuristics to enhance and incorporate new information into the training data.

In more details, if a large portion of the terms in a citation c_k is mainly used by only one specific author a_j , specially regarding coauthor names, there is a large confidence that c_k belongs to a_j . By using Equ. 1, a simple confidence metric on such assignment can be obtained based on the similarity between c_k and the first and second most similar authors, a_l and a_m , respectively, as given by the following equation:

$$\Delta(c_k) = \text{sim}(c_k, a_l) \left(\frac{2 \cdot \text{sim}(c_k, a_l)}{\text{sim}(c_k, a_l) + \text{sim}(c_k, a_m)} - 1 \right) \quad (3)$$

This metric combines information about how close c_k is to a_l and how distant it is from the other authors. In the SLAND method [25], a close baseline, the confidence estimate exploits only the distance between the highest conditional probability $\hat{p}(a_l|c_k)$ and the others, independently

on how confident this highest probability is. This allows the inclusion of examples in the training data with low confidence. Given a limit Δ_{\min} , the assignments with $\Delta(c_k) > \Delta_{\min}$ can be considered confident enough and can be included in the training data. At this moment, the assigned citations not included in the training data (set E) that have similar terms are analyzed again (lines 23–31 in Algorithm 1). This approach allows the original method to automatically correct eventual misclassifications, given that the amount of noise introduced in the training is low. In Sect. 3.3, we present the proposed strategy to define Δ_{\min} from the training data.

3.2.2 Detecting new authors

A low similarity between a citation and its most similar group in the training set may indicate the presence of a new author (not in the current training data) or simply a shift of research interests of an existing author. Thus, we introduced the following two additional thresholds to help defining the probability of finding a new author and to control the amount of fragmentation inserted into the training, i.e., distinct groups whose citations belong to the same author: (i) minimum similarity value between a citation and a group of citations, γ , needed to assign a citation to an author; (ii) minimum similarity value between two clusters of citations, ϕ , to consider them as belonging to the same author.

The γ parameter also helps us to control the purity of the groups as it introduces a minimal requirement for performing an actual author assignment. Notice, however, that even with small values for the γ threshold, the citation groups for authors who work in different research lines may still result in some fragmentation. To alleviate this problem, every time a new citation is inserted into the training set, the similarities between the group assigned to c_k and all the other groups that have at least one term shared with c_k are re-calculated (lines 17 and 28 in Algorithm 1). If such similarity is greater than ϕ , these clusters are merged. Notice that in this procedure, we do not take into account manually labeled authors/groups, since we assume the authorship of these citations is correct for most cases.

For comparing two groups, any distance function may be used. Here we used the cosine similarity applied to the term vectors of all citations in the groups weighted according to Eq. 4:

$$tfidf(t_i, a_j) = (1 + \log_2 \text{freq}_{i,j}) \cdot \left(1 + \log_2 \frac{|A|}{n_i} \right) \cdot w_x \tag{4}$$

This function exploits, besides the TFIDF (term frequency—inverse document frequency) weighting scheme, the w_x weights defined for each attribute x containing term

i , the same weight used in Eq. 1. The values of parameters γ and ϕ may be obtained by cross-validation in the training set. Next, we detail the parameter estimation procedures.

3.3 Estimation of parameters

Our proposed method has seven parameters: $w_c, w_t, w_v, \alpha, \Delta_{\min}, \gamma$ and ϕ . The best values for these parameters may be obtained using standard procedures of cross-validation in the training set. However in order to increase efficiency of this search, we proposed a few strategies. The process is performed in five steps in the following order: (1) definition of parameter α ; (2) definition of the values for weights w_c, w_t and w_v ; (3) definition of value for Δ_{\min} ; (4) definition of value for γ ; and (5) definition of value for ϕ .

Parameter α is defined by testing values 0.1, 0.2, 0.3, 0.4 and 0.5 using cross-validation in the training set. These values, determined empirically, define the weight of each term based mainly on its discriminative capability. In this procedure, the weights w_c, w_t and w_v were all configured with a standard value $1/3$. In this and some other steps of the parameter search, the method does not identify new authors nor includes new citations in the training. For the other parameters, the procedures are described next.

3.3.1 Attribute weights

For a citation c_k to be correctly associated with an author a_l , the weights must be defined so that $\text{sim}(c_k, a_l) > \text{sim}(c_k, a_m), \forall a_m \in A - \{a_l\}$. Based on the training data, it is possible to define a set of inequalities considering the difference of similarities between each attribute in the form $w_c \cdot \text{diff}_c + w_t \cdot \text{diff}_t + w_v \cdot \text{diff}_v > 0$, where diff_x represents the difference $f(c_k^x, a_l) - f(c_k^x, a_m)$.

By using, again, a cross-validation procedure in the training set, we obtain approximately $\bar{n}|D|$ inequalities that compose a system, possibly unsolvable. However, the values of the calculated differences, i.e., $\text{diff}_c, \text{diff}_t$ and diff_v , reflect the degree of importance of each attribute. For instance, if many authors publish in the same venues, some values of diff_v will be very low indicating that the weight w_v should not be higher than w_c or w_t .

Considering such observations, the adopted strategy for defining the attribute weights consists of using the summations of the differences of the similarities between each attribute weighted by the total similarity as shown in Eq. 5:

$$w_x \approx \sum_{c_k \in D} \sum_{a_m \in A - \{a_l\}} \text{diff}_x \cdot \text{sim}(c_k, a_m) \tag{5}$$

where x represents a given attribute and a_l is the correct author of the citation. The weights are defined after a normalization that consists of calculating the proportion of the

total value with respect to the other values. If the minimum value is negative, all values are updated by summing up the absolute minimum plus 1.

3.3.2 Minimum confidence

The Δ_{\min} value must be defined so that, for a given citation c_k with $\Delta(c_k) > \Delta_{\min}$, the probability of c_k being correctly assigned is close to 1. Based on the definition of the α , w_c , w_t and w_v parameters, a new cross-validation procedure is performed to fill in a matrix with the calculated values of Δ and a variable indicating a correct or wrong assignment. These values are ordered and an accuracy rate is calculated for each line in the matrix. The lowest value of Δ associated with an accuracy of at least 90% is attributed to Δ_{\min} . Such accuracy has been empirically defined, but it can be adjusted to control the level of noise *allowed* in the training set during the disambiguation process.

If the amount of training is very low, it may occur that no suitable value for Δ_{\min} can be found. In this case, we used the formula $\Delta_{\min} = (w_x + w_y)/2$ to define the value of this parameter, where w_x represents the lowest weight and w_y the second lowest weight. This value expresses the expected similarity value when a citation has few discriminative terms related to a group.

3.3.3 Minimum evidence

The value for the γ parameter was calculated by maximizing the tradeoff between the rate of correct assignments and the rate of identification of new authors. The higher the value of γ , the higher the chance of identifying new authors, with the downside of the possibility of increasing fragmentation in the final result. This tradeoff may be represented as the following sum of probabilities:

$$\hat{p}(\text{sim}(c_k, a_x) < \gamma | a_z \notin A) \hat{p}(a_z \notin A) \\ + \hat{p}(\text{sim}(c_k, a_x) \geq \gamma | a_z \in A) \hat{p}(a_z \in A)$$

where c_k represents a citation, a_x the group with highest similarity with c_k and a_z the correct group of c_k . The probability that a citation belongs to an author absent in the training set $\hat{p}(a_z \notin A)$ was approximated by the fraction $(|A| + 1)/(|D| + 2)$, although this value may be manually set in order to control the purity of the generated clusters. The conditional probabilities were estimated based on the following steps:

1. For each citation $c_k \in D$ and author $a_j \in A$, we calculate similarities $\text{sim}(c_k, a_j)$ not considering the presence of c_k in the training set; this is similar to a *leave-one-out* validation procedure.

2. For each citation $c_k \in D$ we store in a matrix, the similarity value of the citation group $\text{sim}(c_k, a_z)$ and the highest obtained similarity not considering the similarity with the correct group $\text{sim}(c_k, a_x)$.
3. The matrix is stored in decreasing order of $\text{sim}(c_k, a_x)$ values so that the positions of each value correspond to the number of citations in the training set that would not be identified if the value for γ was smaller than $\text{sim}(c_k, a_x)$. These positions are used to estimate the probabilities $\hat{p}(\text{sim}(c_k, a_x) < \gamma | a_z \notin A)$.
4. The matrix is ordered in increasing value of $\text{sim}(c_k, a_z)$ so that the positions of each value correspond to the number of citations with authors in the training set that would not be correctly assigned if the parameter γ was higher than $\text{sim}(c_k, a_z)$. These positions are used to estimate the probabilities $\hat{p}(\text{sim}(c_k, a_x) \geq \gamma | a_z \in A)$.

With the estimated probabilities, the final value for γ is defined as the one that maximizes the sum presented in the beginning of this section. If the amount of training is very low, the value for γ may also be low, thus we empirically define a minimum value for this parameter corresponding to $\frac{1}{4}$ of the attribute value with the lowest weight. We analyze the sensitivity of the method to this parameter in our experimental evaluation.

3.3.4 Minimum cluster similarity

The ϕ parameter was designed to reduce fragmentation caused by the lack of evidence in assigning a citation. To calculate the value of this parameter, the training set was split into two halves. With this we expect to have approximately two clusters per author. The similarities between each pair of clusters belonging to the two different partitions of the training are stored in a matrix along with a variable indicating whether these two groups belong to the same author or not. Based on this procedure, we define the value of ϕ as the one corresponding to the lowest similarity value associated with an error of at most one incorrect cluster match. In the case of small amounts of training, this value can be very low, therefore we define for it a minimal empirical value of 0.075. We also analyze the sensitivity of the method to this parameter in our experimental evaluation.

3.3.5 Self-training

When no training data are available, our method can be manually configured (for this, some guidelines are presented in Sect. 4.5.5) or we may use a strategy to automatically create the training data. Ferreira et al. [6] have proposed a procedure to create and select pure clusters in order to train the SLAND method. Here, this same strategy is used to train NC. This procedure is based on two steps: (1) extract pure clusters, i.e.,

clusters with high probability of containing citations belong to only one author, and (2) discard fragmented clusters. Pure clusters are extracted by exploiting recurring patterns in the coauthorship graph, that is, two citations are placed together in the same cluster if both have coauthors in common.

In order to produce the best possible training set, which ideally contains only one cluster per author in the collection, we need to discard fragmented clusters. This step starts by sorting clusters in descending order of size resulting in a sorted list \mathcal{C} . We then, iteratively select the largest cluster in \mathcal{C} that is most dissimilar to the clusters already selected to compose the training data. Ferreira et al. [6] have shown that using a specialized author name similarity function it is possible to obtain a low fragmentation rate in this last step while keeping the purity of the clusters very high.

3.4 Computational complexity

Our disambiguation method can be efficiently implemented using hash tables. The analysis presented next considers that insertion and search operations in this data structure have a time complexity of $O(1)$.

The number of instructions executed to perform the calculation of the similarity function defined by Equ. 2 is proportional to the number of terms of the citation, thus the steps defined in lines 2–7 of Algorithm 1 have time complexity of $O(|c_k||A|)$. If the value of $\text{sim}(c_k, a_l)$ is less than γ , a new cluster is created and included in the training set in time $O(|c_k|)$. If the value of $\Delta(c_k)$ is less or equal than Δ_{\min} , the citation is included in the set E also in time $O(|c_k|)$. Finally, if the citation is considered reliable, the complexity of each step of the algorithm is given by:

- $O(|c_k|)$ for the inclusion of c_k in the training set (line 15);
- $O(|G|m)$ for calculating the similarities between a_l and each cluster in G and to perform the merges if necessary, where m represents the average number of terms per cluster (lines 16–22);
- $O(|E||c_k|)$ for getting E_l (line 23);
- $O(|E_l|(|A|t + gm))$ for reclassifying each citation in E_l , where t represents the average number of terms per citation and g the average number of clusters related to each citation (lines 24–31).

Since we have that $g \leq |A|$ and $m \geq t$, the time complexity of the entire disambiguation procedure can be represent by $O(|A|(|c_k| + |E|m))$.

4 Experimental evaluation

In this section, we present experimental results that demonstrate the effectiveness, efficiency and practicability of our

method, hereafter called NC (from nearest cluster model). We first describe the baselines, collections and evaluation metrics.

4.1 Baselines

We used as baselines five author assignment methods (four supervised, one self-trained)—SLAND [25], SAND [9], SVM [10], NB [10] and Cosine—and two unsupervised author grouping methods—LAVSM-DBSCAN [14] and HHC [4]. We explain them next³.

SLAND infers the author of a citation by using a *lazy* supervised rule-based associative classifier. The method infers the most probable author of a given record 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 works *on demand*, i.e., association rules to infer the correct author of a citation record are generated at the moment a citation is presented to the method for 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. SAND extends this method by using the procedure described in Sect. 3.3.5 to automatically produce training data, based on pure clusters, for the associative classifier, i.e., SAND does not need any manually labeling process, predicts the author by cluster instead of by citation and performs without any user parameter set up. For these two baselines, we have used the original implementations provided by the authors.

SVM⁴ associates each author with an author class and train the classifier for that class. Each citation is represented by a feature vector with the elements of their attributes (author and coauthor names, and terms from publication and venue titles) and their TFIDF as the feature weight. The method uses an “one class versus all others” approach to multi-class classification.

NB assumes that each citation is generated by a naïve Bayes model. Thus, to estimate the model parameters some citations are used as training data. Let a_i be an author class corresponding to a unique single person, where $i \in [1, n]$ and n is the number of authors, and let r be a citation. The probability of each class a_i generates a citation r is calculated by using the naïve Bayes rule. The citation r is attributed to the class that has the maximal posterior probability of produce it. The parameters of the model for a given author a_i , which include, for example, the probability of a_i writing an article with previously seen and unseen coauthors and

³ The implementations of all methods used in our experimental evaluation are available at <http://www.lbd.dcc.ufmg.br/lbd/collections/disambiguation/author-name-disambiguation-methods>.

⁴ For this baseline, we have used the libSVM package available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>.

the probability of a_i writing a title or publishing in a venue containing a specific word, are all defined in [10]. We have implemented this baseline by ourselves.

Cosine is a version of the method proposed in [18] that uses the cosine distance with TFIDF to calculate the similarity among citations and groups of citations. Similar to our method, it also proposes to use different weights for the citation attributes. Cosine needs that examples of citations for each author be given a priori in order to calculate the similarities and perform the assignments. In this sense, it can be considered as a supervised method. We also have used our own implementation of this baseline.

LASVM-DBSCAN uses DBSCAN⁵, a density-based clustering method [14], for clustering citations by author. First, the distance metric between pairs of citations used by DBSCAN is calculated by a trained active SVM 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 to make the similarity vectors, such as the edit distance for emails and URLs, Jaccard similarity for addresses and affiliations, and soft-TFIDF for author and coauthor names. In our work, we use cosine for work and publication venue title and soft-TFIDF for author and coauthor names.

HHC is a two-step heuristic-based method [4]. In the first step, it creates clusters of citations with similar author names that share at least a similar coauthor name. Author name similarity is given by an ad-hoc name comparison algorithm. This step produces very pure but fragmented clusters. In the second step, it successively merges clusters with similar author names according to the similarity between the other citation attributes (i.e., publication and venue titles) calculated using the cosine measure. In each round of merging, the information of merged clusters is aggregated providing more information for the next round. This process is successively repeated until no more merges are possible according to a similarity threshold. For this baseline, we have used the original implementation provided by the authors.

4.2 Collections

In order to evaluate our disambiguation method, we used collections of citations extracted from DBLP⁶ and BDBComp⁷. The collections are described below⁸.

⁵ We used the LASVM package [3] available at <http://leon.bottou.org/projects/lasvm> and the DBSCAN version available from Weka at <http://www.cs.waikato.ac.nz/ml/weka/>.

⁶ <http://dblp.uni-trier.de>.

⁷ <http://www.lbd.dcc.ufmg.br/bdbcomp>.

⁸ All collections used in our experimental evaluation are available at <http://www.lbd.dcc.ufmg.br/lbd/collections/disambiguation/collections-the-nearest-cluster-method>.

The first collection derived from DBLP sums up 8,418 citations associated with 477 distinct authors, which means an average of approximately 17.6 citations per author. This collection includes 5,585 citations whose author names are in short format, i.e., initial of the first name appended to the last name. The version we use is the same adopted by several other works [4, 6, 8, 9, 20, 25]. It was based on [11], in which the authors manually labeled all citations. For this, they used the author's 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. We used the 14 original ambiguous groups considered in [11] with few corrections performed in [4] due to the identification of some labeling mistakes. Working in a cleaner collection may help to better separate the limitations of the technique from errors caused by bad labeling.

In more details, the original DBLP collection used in [11] contained a few errors. This probably has several causes. First, the citation records listed in an author's homepage were associated with publications written by that same author. However, there are many cases where an author's homepage contains citation records that are related to different authors and, as a result, there might be errors that merge different citation records into the same ambiguous group. Second, there were several cases of duplicated citations. One example, is the citation "The design of a rotating associative memory for relational database applications" by Chyuan Shiun Lin, Diane C. P. Smith, and John Miles Smith" in the "J. Smith" group whose duplicated entry was removed from the collection. In [4], the authors manually examined and closely analyzed the original DBLP collection and removed errors using more recent information found on the Web and in other digital libraries and similar systems (e.g., ArnetMiner⁹ and Microsoft Academic Search¹⁰). In any case, the differences between the collection we use and its original version [11] correspond to only 64 records¹¹ or less than 0.8 % of the collection.

The second collection derived from DBLP, hereafter referred to as KISTI, was built by the Korea Institute of Science and Technology Information [17] for English homonym author name disambiguation. The top 1,000 most frequent author names from late-2007 DBLP version were obtained jointly with their citations. To disambiguate this collection, the authors submitted a query composed of the surname of the author and the publication title of each citation to Google, aiming at finding personal publication pages. The first 20 web pages retrieved for each query were manually checked

⁹ <http://arnetminer.org>.

¹⁰ <http://academic.research.microsoft.com>.

¹¹ This does not count the expansion, performed in [4], from short to full author names in some records to better resemble a more realistic situation, in which there is a more balanced mix of both cases.

to identify the correct personal publication page for each citation. This identified page was then used to disambiguate the citation record. This collection has 41,659 name instances of 867 name groups and 6,908 authors¹².

The collection of citations extracted from BDBComp sums up 361 citations associated with 184 distinct authors, approximately two citations per author, in which only eight author names are in short format. Notice that, although much smaller than the DBLP and KISTI collections, this collection is very difficult to disambiguate, because it has many authors with just a few (one or two) citations. This collection contains the 10 largest ambiguous groups found in BDBComp at the time of the dataset creation [4].

We have used KISTI and BDBComp collections as originally created, while we have promoted a few corrections in the DBLP collection based on mistakes found during some error analysis procedure, as explained above.

4.3 Evaluation metrics

In order to evaluate our proposed disambiguation method, we use two evaluation metrics: the K and pairwise F1 metrics. These are standard metrics that have been largely used by the community (e.g., [14, 20]). In the discussion that follows, we describe these metrics. The key idea is to compare the clusters extracted by disambiguation methods against ideal, perfect clusters, which were manually extracted. Hereafter, a cluster extracted by a disambiguation method will be referred to as an *empirical cluster*, while a perfect cluster will be referred to as a *theoretical cluster*.

The K metric determines the trade-off between the average cluster purity (ACP) and the average author purity (AAP). Given an ambiguous group, ACP evaluates the purity of the empirical clusters with respect to the theoretical clusters for this ambiguous group. Thus, if the empirical clusters are pure (i.e., they contain only citations associated with the same author), the corresponding ACP value will be 1. ACP is defined as:

$$\text{ACP} = \frac{1}{N} \sum_{i=1}^e \sum_{j=1}^t \frac{n_{ij}^2}{n_i} \quad (6)$$

where N is the total number of citations in the ambiguous group, t is the number of theoretical clusters in the ambiguous group, e is the number of empirical clusters for this ambiguous group, n_i is the total number of citations in the empirical cluster i , and n_{ij} is the total number of citations in the empirical cluster i which are also in the theoretical cluster j .

For a given ambiguous group, AAP evaluates the fragmentation of the empirical clusters with respect to the theo-

retical clusters. If the empirical clusters are not fragmented, the corresponding AAP value will be 1. AAP is defined as:

$$\text{AAP} = \frac{1}{N} \sum_{j=1}^t \sum_{i=1}^e \frac{n_{ij}^2}{n_j} \quad (7)$$

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

Thus, the K metric consists of the geometric mean between ACP and AAP values. It evaluates the purity and cohesion of the empirical clusters extracted by each method, being defined as:

$$K = \sqrt{\text{ACP} \cdot \text{AAP}} \quad (8)$$

Finally, pairwise F1 ($pF1$) is the F1 metric calculated using pairwise precision and pairwise recall. Pairwise precision (pP) is calculated as $pP = \frac{a}{a+c}$, where a is the number of pairs of citations in an empirical cluster that are (correctly) associated with the same author, and c is the number of pairs of citations in an empirical cluster not corresponding to the same author. Pairwise recall (pR) is calculated as $pR = \frac{a}{a+b}$, where b is the number of pairs of citations associated with the same author that are not in the same empirical cluster. Thus, the pairwise F1 metric is defined as:

$$pF1 = 2 \cdot \frac{pP \times pR}{pP + pR} \quad (9)$$

4.4 Experimental setup

Experiments were conducted within each ambiguous group splitter into training (50%) and test (50%) sets. First we compare NC with supervised author assignment methods using the training and test sets, and, then, we compare the results obtained using only the test sets with unsupervised or author grouping methods. All results shown correspond to the performance in the test sets and are the average of 10 runs. They were all found to be statistically significant at the 95% confidence level when tested with the two-tailed paired t-test with Holm–Bonferroni correction [13].

Unless otherwise stated, RBF kernels were used for SVM and we employed the LibSVM GRID¹³ tool for finding their optimum parameters for each training data on each ambiguous group. We estimate the parameter of the NB method as in [10]. Finally, for SLAND, the state-of-the-art supervised author assignment method, the best parameters were discovered using cross-validation in the training set.

¹² <http://www.kisti.re.kr>.

¹³ <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>.

Table 2 Results obtained in DBLP collection

Method	K	pF1
NC	0.919 ± 0.020	0.919 ± 0.023
Cosine	0.884 ± 0.028	0.898 ± 0.029
SLAND	0.878 ± 0.027	0.869 ± 0.034
SVM	0.777 ± 0.038	0.702 ± 0.070
NB	0.711 ± 0.045	0.616 ± 0.080

Best results are highlighted in bold

Table 3 Results obtained in KISTI collection

Method	K	pF1
NC	0.940 ± 0.002	0.816 ± 0.009
SLAND	0.927 ± 0.002	0.806 ± 0.008
Cosine	0.883 ± 0.003	0.757 ± 0.008
SVM	0.797 ± 0.004	0.623 ± 0.009
NB	0.768 ± 0.005	0.596 ± 0.009

Best results are highlighted in bold

Table 4 Results obtained in BDBComp collection

Method	K	pF1
NC	0.917 ± 0.020	0.633 ± 0.165
SLAND	0.882 ± 0.031	0.512 ± 0.205
Cosine	0.746 ± 0.041	0.400 ± 0.199
SVM	0.579 ± 0.042	0.201 ± 0.124
NB	0.537 ± 0.067	0.246 ± 0.157

Best results are highlighted in bold

4.5 Results

Table 2 shows the average results of NC compared with Cosine, SLAND, SVM, and NB in DBLP. NC outperforms all baselines under both metrics. Gains, under K, are around 18.3 and 29.3 % compared with SVM and NB, respectively, and, under pF1, gains surpass 30 % compared with the same baselines. Compared with Cosine and SLAND, gains are smaller (around 4 % under K against both baselines) but still significant. The worst performances of SVM and NB are due to the fact that both have static training sets and exploit generic classification techniques not directly adapted to the problem.

In KISTI (Table 3), NC also outperforms all baselines under both metrics. Gains, under K, vary from 1.3 % against SLAND and 22.3 % against NB, and 1.2 and 36.9 % under pF1, over the same baselines.

Our best gains are obtained in BDBComp (Table 4), the hardest collection, as can be verified by the much lower pF1 results obtained by all methods. Against Cosine, gains are around 21.9 and 58.3 % under K and pF1, respectively. Against SLAND, the best baseline in this collection, gains

range from 4 % under K to 23 % under pF1. Finally, against SVM and NB, gains surpass 70.0 % under K and more than 157.4 % under pF1. We notice that this collection has a lot of authors with just a few citations. Thus, several authors do not have any example in the training data. NC and SLAND are able to identify citations belonging to new authors (i.e., authors without any citation in the training data) and add such citations to the training data, justifying their superior performance.

Tables 5 and 6 show, for every single ambiguous group in DBLP and BDBComp, the results obtained by our proposed method and the two baselines with the best average performances: SLAND and Cosine. Similar results are shown in Table 7 for the 40 largest ambiguous groups of KISTI (space reasons prevent us to show results for all 881 groups).

We can notice in Table 5 that NC produces the best results in all ambiguous groups in DBLP, with four statistical ties with SLAND and five with Cosine under K. NC reaches gains of up to 14.8 % and 11 % (“J. Martin” ambiguous group) against SLAND and Cosine, respectively, under K. In terms of pF1, NC outperforms or is statistically tied with SLAND and Cosine in all ambiguous groups with gains reaching 22.0 % against SLAND and 10.7 % against Cosine (both in “J. Martin”).

Likewise, in BDBComp (Table 6) NC is not outperformed by any method in any single group, with six ties with SLAND under K and seven ties under pF1. Comparing with Cosine, NC outperforms it in all ambiguous groups under both metrics, but one ambiguous groups under K and five under pF1¹⁴. Gains under K reach up to 11.1 % against SLAND (“L. Silva”) and 45.9 % against Cosine (“R. Santos”). Under pF1, gains reach up to 53.3 % against SLAND and 729.4 % against Cosine (“F. Silva”).

Table 7 shows the results of SLAND, Cosine and NC in the KISTI collection for the 40 largest ambiguous groups. Under K, NC obtains gains up to 15.8 % (“S. Lee”) against SLAND and 18.6 % (“J. Halpern”) against Cosine. In terms of pF1, NC obtains up to 34.4 % and 18.8 % (“J. Kim”) of gains against SLAND and Cosine, respectively. Notice that, among these 40 largest ambiguous groups, NC does not obtain the best performance under both metrics only in the “M. Vardi” group. The best result obtained by SLAND in this group is explained by its number of authors and citation records. The “M. Vardi” group has only two authors, with one of them having 174 citation records while the other one has only two. Thus, the number of rules generated by SLAND for the most prolific author is much higher than those for the other author. In this case, SLAND always assign the records

¹⁴ As the pF1 metric is calculated based on the number of pairs of citations in the empirical clusters, when the cluster has only one citation, no pair is formed, and the obtained value is equal to 0, as in the case of the “M. Silva” group.

Table 5 Results obtained by SLAND, Cosine and NC in each ambiguous group in the DBLP collection

Ambiguous Group	SLAND		Cosine		NC	
	K	pF1	K	pF1	K	pF1
A. Gupta	0.886 ± 0.024	0.897 ± 0.029	0.913 ± 0.014	0.937 ± 0.011	0.941 ± 0.008	0.957 ± 0.010
A. Kumar	0.927 ± 0.022	0.919 ± 0.027	0.922 ± 0.019	0.928 ± 0.023	0.934 ± 0.015	0.925 ± 0.026
C. Chen	0.845 ± 0.014	0.845 ± 0.025	0.815 ± 0.014	0.829 ± 0.024	0.848 ± 0.018	0.814 ± 0.036
D. Johnson	0.861 ± 0.035	0.849 ± 0.054	0.847 ± 0.017	0.848 ± 0.023	0.898 ± 0.024	0.909 ± 0.026
J. Lee	0.843 ± 0.012	0.835 ± 0.013	0.847 ± 0.013	0.873 ± 0.013	0.904 ± 0.007	0.905 ± 0.007
J. Martin	0.824 ± 0.038	0.745 ± 0.055	0.852 ± 0.038	0.821 ± 0.048	0.946 ± 0.018	0.909 ± 0.033
J. Robinson	0.926 ± 0.029	0.917 ± 0.034	0.928 ± 0.024	0.936 ± 0.026	0.937 ± 0.018	0.939 ± 0.016
J. Smith	0.896 ± 0.013	0.913 ± 0.011	0.891 ± 0.014	0.930 ± 0.013	0.914 ± 0.014	0.930 ± 0.019
K. Tanaka	0.943 ± 0.017	0.941 ± 0.019	0.963 ± 0.014	0.971 ± 0.014	0.974 ± 0.013	0.980 ± 0.016
M. Brown	0.920 ± 0.036	0.911 ± 0.049	0.891 ± 0.057	0.896 ± 0.060	0.922 ± 0.026	0.909 ± 0.037
M. Jones	0.782 ± 0.033	0.765 ± 0.037	0.839 ± 0.026	0.860 ± 0.028	0.882 ± 0.020	0.891 ± 0.023
M. Miller	0.945 ± 0.013	0.949 ± 0.014	0.978 ± 0.006	0.987 ± 0.007	0.980 ± 0.011	0.986 ± 0.008
S. Lee	0.820 ± 0.009	0.818 ± 0.014	0.805 ± 0.009	0.829 ± 0.013	0.869 ± 0.007	0.880 ± 0.011
Y. Chen	0.876 ± 0.010	0.869 ± 0.013	0.885 ± 0.009	0.932 ± 0.010	0.926 ± 0.011	0.939 ± 0.014
Average	0.878 ± 0.027	0.869 ± 0.034	0.884 ± 0.028	0.898 ± 0.029	0.919 ± 0.020	0.919 ± 0.023

Best results are highlighted in bold

Table 6 Results obtained by SLAND, Cosine and NC in each ambiguous group in the BDBComp collection

Ambiguous Group	SLAND		Cosine		NC	
	K	pF1	K	pF1	K	pF1
A. Oliveira	0.897 ± 0.043	0.878 ± 0.077	0.798 ± 0.077	0.765 ± 0.114	0.912 ± 0.040	0.877 ± 0.076
A. Silva	0.869 ± 0.023	0.744 ± 0.059	0.783 ± 0.032	0.599 ± 0.075	0.932 ± 0.028	0.776 ± 0.099
F. Silva	0.931 ± 0.029	0.422 ± 0.284	0.714 ± 0.048	0.078 ± 0.073	0.955 ± 0.041	0.647 ± 0.329
J. Oliveira	0.822 ± 0.054	0.637 ± 0.103	0.683 ± 0.039	0.485 ± 0.084	0.890 ± 0.032	0.764 ± 0.085
J. Silva	0.842 ± 0.058	0.620 ± 0.139	0.801 ± 0.033	0.586 ± 0.081	0.901 ± 0.070	0.778 ± 0.140
J. Souza	0.894 ± 0.049	0.807 ± 0.085	0.797 ± 0.062	0.693 ± 0.092	0.856 ± 0.025	0.747 ± 0.050
L. Silva	0.840 ± 0.043	0.559 ± 0.153	0.770 ± 0.051	0.501 ± 0.116	0.934 ± 0.047	0.795 ± 0.134
M. Silva	0.859 ± 0.029	0.000 ± 0.000	0.769 ± 0.071	0.152 ± 0.120	0.938 ± 0.041	0.418 ± 0.239
R. Santos	0.963 ± 0.030	0.267 ± 0.314	0.635 ± 0.045	0.050 ± 0.058	0.927 ± 0.045	0.261 ± 0.262
R. Silva	0.903 ± 0.035	0.189 ± 0.156	0.713 ± 0.048	0.091 ± 0.099	0.927 ± 0.074	0.263 ± 0.263
Average	0.882 ± 0.031	0.512 ± 0.205	0.746 ± 0.041	0.400 ± 0.199	0.917 ± 0.020	0.633 ± 0.165

Best results are highlighted in bold

to the author with the largest number of citations. However, it is worth to notice that NC performs only 1.8 % lower than SLAND under K in this group.

4.5.1 Error analysis

In here, we present an analysis of some errors detected during our experiments, based on the values of the similarity function (Eq. 2) and the confidence metric (Eq. 3) defined for our method. By using these values, we can identify very ambiguous references as well as how the lack of information in the training set affects the disambiguation process. Given a test citation c_k , the cluster a_l with the highest similarity

value that represents its author¹⁵ and the cluster a_z with the highest similarity value, we have defined the following error cases:

1. A new cluster is incorrectly created when $\text{sim}(c_k, a_l) < 0.05$. This case represents errors that may be considered inevitable due to the lack of evidence in the training data.
2. A new cluster is incorrectly created when $\text{sim}(c_k, a_l) \geq 0.05$. Some of these errors could be avoided with a change in the γ value (the “new author” parameter).

¹⁵ We consider that a cluster represents an author if most of its citations belong to this author.

Table 7 Results obtained by SLAND, Cosine and NC in the 40 largest ambiguous groups in the KISTI collection

Ambiguous Group	SLAND		Cosine		NC	
	K	pF1	K	pF1	K	pF1
A. Choudhary	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
A. Gupta	0.883 ± 0.034	0.878 ± 0.043	0.902 ± 0.025	0.901 ± 0.031	0.918 ± 0.020	0.905 ± 0.033
D. Eppstein	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
D. Lee	0.927 ± 0.026	0.931 ± 0.028	0.890 ± 0.024	0.878 ± 0.038	0.959 ± 0.007	0.957 ± 0.012
H. Chen	0.830 ± 0.035	0.776 ± 0.053	0.860 ± 0.023	0.859 ± 0.043	0.870 ± 0.038	0.801 ± 0.099
H. Wang	0.896 ± 0.031	0.867 ± 0.051	0.907 ± 0.018	0.897 ± 0.030	0.943 ± 0.015	0.938 ± 0.019
J. Chen	0.843 ± 0.036	0.799 ± 0.050	0.889 ± 0.026	0.952 ± 0.025	0.952 ± 0.016	0.981 ± 0.014
J. Halpern	0.918 ± 0.011	0.915 ± 0.013	0.768 ± 0.033	0.779 ± 0.043	0.912 ± 0.020	0.925 ± 0.022
J. Kim	0.789 ± 0.020	0.563 ± 0.047	0.808 ± 0.039	0.637 ± 0.071	0.903 ± 0.010	0.757 ± 0.036
J. Lee	0.814 ± 0.021	0.674 ± 0.045	0.830 ± 0.027	0.754 ± 0.054	0.864 ± 0.024	0.710 ± 0.053
J. Li	0.852 ± 0.032	0.749 ± 0.062	0.874 ± 0.021	0.884 ± 0.030	0.937 ± 0.009	0.929 ± 0.032
J. Liu	0.841 ± 0.028	0.808 ± 0.049	0.890 ± 0.026	0.918 ± 0.026	0.933 ± 0.019	0.952 ± 0.015
J. Mitchell	0.986 ± 0.010	0.987 ± 0.010	0.965 ± 0.020	0.966 ± 0.019	0.998 ± 0.003	0.998 ± 0.003
J. Smith	0.830 ± 0.024	0.828 ± 0.033	0.871 ± 0.026	0.918 ± 0.022	0.929 ± 0.021	0.946 ± 0.028
J. Wang	0.848 ± 0.025	0.797 ± 0.044	0.885 ± 0.019	0.884 ± 0.023	0.932 ± 0.023	0.902 ± 0.040
J. Wu	0.936 ± 0.015	0.937 ± 0.018	0.909 ± 0.019	0.918 ± 0.015	0.952 ± 0.022	0.955 ± 0.025
J. Zhang	0.852 ± 0.027	0.810 ± 0.050	0.861 ± 0.029	0.818 ± 0.036	0.934 ± 0.009	0.901 ± 0.024
L. Zhang	0.865 ± 0.030	0.829 ± 0.055	0.898 ± 0.016	0.897 ± 0.025	0.926 ± 0.023	0.897 ± 0.041
M. Chen	0.886 ± 0.026	0.873 ± 0.031	0.935 ± 0.013	0.958 ± 0.012	0.972 ± 0.010	0.978 ± 0.014
M. Pedram	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
M. Vardi	0.990 ± 0.004	0.991 ± 0.003	0.893 ± 0.041	0.889 ± 0.045	0.972 ± 0.015	0.972 ± 0.016
N. Jennings	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
N. Jha	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
N. Lynch	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
P. Yu	0.983 ± 0.009	0.985 ± 0.007	0.963 ± 0.021	0.964 ± 0.021	0.993 ± 0.009	0.993 ± 0.010
Q. Yang	0.923 ± 0.026	0.942 ± 0.026	0.911 ± 0.012	0.959 ± 0.008	0.943 ± 0.027	0.968 ± 0.025
S. Jajodia	0.945 ± 0.013	0.948 ± 0.016	0.894 ± 0.022	0.916 ± 0.022	0.949 ± 0.016	0.962 ± 0.017
S. Kim	0.843 ± 0.044	0.715 ± 0.108	0.869 ± 0.021	0.837 ± 0.044	0.928 ± 0.026	0.893 ± 0.064
S. Lee	0.812 ± 0.031	0.700 ± 0.084	0.844 ± 0.029	0.827 ± 0.053	0.941 ± 0.036	0.904 ± 0.111
T. Henzinger	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
W. Wang	0.807 ± 0.039	0.677 ± 0.061	0.801 ± 0.022	0.761 ± 0.039	0.885 ± 0.018	0.836 ± 0.044
X. Li	0.804 ± 0.023	0.740 ± 0.047	0.830 ± 0.026	0.785 ± 0.044	0.874 ± 0.032	0.804 ± 0.062
X. Zhou	0.924 ± 0.033	0.910 ± 0.044	0.915 ± 0.044	0.889 ± 0.055	0.964 ± 0.008	0.959 ± 0.011
Y. Chen	0.828 ± 0.025	0.713 ± 0.062	0.849 ± 0.021	0.790 ± 0.024	0.919 ± 0.011	0.873 ± 0.037
Y. Li	0.860 ± 0.023	0.756 ± 0.049	0.837 ± 0.026	0.761 ± 0.046	0.911 ± 0.016	0.838 ± 0.039
Y. Liu	0.894 ± 0.019	0.886 ± 0.025	0.920 ± 0.011	0.938 ± 0.011	0.947 ± 0.011	0.939 ± 0.021
Y. Wang	0.851 ± 0.018	0.804 ± 0.030	0.875 ± 0.012	0.864 ± 0.016	0.899 ± 0.012	0.858 ± 0.020
Y. Yang	0.903 ± 0.020	0.892 ± 0.030	0.894 ± 0.031	0.889 ± 0.036	0.958 ± 0.012	0.945 ± 0.032
Y. Zhang	0.836 ± 0.017	0.750 ± 0.031	0.870 ± 0.029	0.839 ± 0.061	0.937 ± 0.022	0.904 ± 0.048
Z. Zhang	0.871 ± 0.024	0.844 ± 0.037	0.856 ± 0.015	0.842 ± 0.015	0.921 ± 0.018	0.908 ± 0.026
Average	0.897 ± 0.022	0.857 ± 0.037	0.899 ± 0.020	0.889 ± 0.027	0.944 ± 0.012	0.925 ± 0.023

Best results are highlighted in bold

3. Wrong assignment of c_k to a cluster due to noise in the training data, that is, if all citations in a_z that belong to the same author of c_k were removed from a_z , the citation would be correctly assigned to a_l .
4. Wrong assignment of c_k to a cluster, when for each attribute x , $f(c_k^x, a_z) \geq f(c_k^x, a_l)$. In this situation, no matter the values of the attribute weights, the citation cannot be correctly assigned to a_l .

Table 8 Error analysis

#	Error case	% of errors			Total (%)
		BDBComp	DBLP	KISTI	
1	Incorrectly created clusters when $\text{sim}(c_k.a_l) < 0.05$	3.00 %	28.78 %	5.68 %	9.35 %
2	Incorrectly created clusters when $\text{sim}(c_k.a_l) \geq 0.05$	36.64 %	16.22 %	30.49 %	28.29 %
3	Incorrectly classified citations due to noise in the training set	0.00 %	2.38 %	0.68 %	0.95 %
4	Incorrectly classified citations when $\text{diff}_x \leq 0 \quad \forall x$	54.05 %	28.54 %	50.05 %	46.65 %
5	Incorrectly classified citations when $\Delta(c_k) \geq 0.5$	0.00 %	0.06 %	0.16 %	0.14 %
6	Incorrectly classified citations when $\Delta(c_k) < 0.5$ and $\Delta(c_k) \geq 0.25$	0.00 %	1.74 %	1.10 %	1.19 %
7	Incorrectly classified citations when $\Delta(c_k) < 0.25$ and $\Delta(c_k) \geq 0.1$	0.60 %	3.56 %	2.80 %	2.89 %
8	Incorrectly classified citations when $\Delta(c_k) < 0.1$	4.80 %	18.67 %	8.84 %	10.36 %
9	Incorrectly classified citations due to wrong merges	0.90 %	0.03 %	0.18 %	0.17 %
10	Incorrect merges of clusters due to noise in the training set	0.00 %	0.03 %	0.02 %	0.02 %

5. Wrong assignment with high $\Delta(c_k)$ values. Once the confidence metric combines the information about how close c_k is to a_l and how distant it is from a_z , an error with this metric greater than 0.5 indicates a very ambiguous case (or a noise in the training data). Some of the errors associated with low values of this metric can be avoided with changes in the attribute weights.
6. Wrong assignment due to a wrong merge of clusters.
7. Wrong merge of clusters due to noise in the training data, that is, if the clusters were pure the cosine similarity would not be higher than ϕ .

Table 8 shows, for each collection, the percentage of errors considering each case described above. As we can see, more than 56 % of all errors can be considered inevitable due to the lack of information in the training data or due to the high ambiguity of the citations (cases 1, 4 and 5). About 40 % of all errors could be avoided by changing the parameter values, but we do not know how the change would affect the correct classifications (cases 2 and 8). In BDBComp, the high number of errors in case 2 reflects the lack of information in the training data due the low number of citations per author. In DBLP, the high percentage of errors in cases 1 and 8 may be explained by the lack of discriminate terms in the training data (most of the terms occur in more than one cluster). In KISTI, more than 50 % of the errors fall in case 4, which indicates that it would be difficult to get further improvements

using only a similarity function based on bag-of-words. It is also interesting to notice that only a small number of errors is due to noise in the training set (0.95 %).

4.5.2 Runtime analysis

To show the high efficiency of our method, we empirically measured the runtime of NC and the baselines. All experiments were performed on a Linux-based PC (Linux Mint 16 Cinnamon 64 bits) with an Intel Core i5-3570, CPU 3.4 GHz x 4 and 8GBytes RAM. Most methods were implemented using Java, namely: NC, Cosine, SVM, NB, and HHC. SVM also uses tools for training and classifying available in C. SLAND and LASVM are also implemented in C.

Table 9 shows the average runtime in seconds of all methods in the whole disambiguation task (i.e., the time for training and classification). As we can see, NB is the fastest method while ours obtained the second best performance in all collections. However, NC is at least 25 % more effective than NB in these same collections. The runtime of NC is statistically tied with Cosine, but effectiveness gains of NC over this method range from 4 % (DBLP) to 58 % (BDBComp). Compared with the other baselines (SVM and SLAND), our method is orders of magnitude faster. The SVM runtime is around 5900, 4500 and 400 times higher than NC runtime in the BDBComp, DBLP and KISTI collections, respectively, though most of this time is for training (i.e., parametrization).

Table 9 Runtime analysis

Method	Runtime (seconds)		
	BDBCOMP	DBLP	KISTI
NB	0.03 ± 0.01	0.50 ± 0.10	0.53 ± 0.03
NC	0.09 ± 0.02	1.97 ± 0.32	2.69 ± 0.02
Cosine	0.10 ± 0.03	6.05 ± 1.01	5.12 ± 0.05
SVM	527.8 ± 0.1	8862.6 ± 1646.6	1071.5 ± 7.4
SLAND	239.6 ± 15.9	21256.7 ± 3738.9	16337.1 ± 144.7

Regarding SLAND, the best baseline, its runtime is around 2700, 10800 and 6000 times higher than the NC runtime¹⁶.

Remind that SLAND is a lazy method with no a priori training, so most of this is in fact real-time parameter fitting and classification time. The low performance of SLAND can be explained by its computational complexity. The number of instructions executed by SLAND is proportional to the number of rules generated during disambiguation. Given a test citation c_k , the maximal number of association rules that can be induced per training instance is $2^{|c_k|}$. The procedure used to abstain unreliable predictions can, in the worst case, double the number of times that the each citation is classified. Thus, the disambiguation time of SLAND can be represented by $O(|D|2^{|c_k|})$, which is proportional to the number of training examples and exponential on the citation size, for each (test) citation to be disambiguated. This explains the high cost of this method.

4.5.3 Analyzing the components of our solution

To analyze the impact of each component of our solution in the performance of the proposed method, we have performed experiments removing some capabilities of the method. Thus, we evaluate four scenarios: (1) NC-No-Training-Update—the method does not add new examples to the training set, but maintains the capability of identifying new authors inserting them into the training set; (2) NC-Cosine—the algorithm remains the same, but our own similarity function is replaced by the cosine similarity function and the terms are weighted using TFIDF; (3) NC-Similarity-Only—the method does not improve the coverage of the training set and is not capable of identifying new authors, i.e., the method uses only the similarity function in the AND task; and (4) NC-No-New-Authors—the method does not identify new authors.

In BDBComp (Table 10) and KISTI (Table 11), the effectiveness is not hurt only when we do not add new examples to the training data. In all other four situations there is some drop in effectiveness, as already mentioned. BDBComp

¹⁶ To put in perspective, with the reported times, it would take more than two weeks to disambiguate a digital library with 1 million citations with SLAND. With NC, this would take, on average, about three minutes.

Table 10 Results obtained in BDBComp collection

Method	K	pF1
NC	0.917 ± 0.020	0.633 ± 0.165
NC-No-Training-Update	0.918 ± 0.022	0.631 ± 0.162
NC-Cosine	0.861 ± 0.032	0.542 ± 0.212
NC-Similarity-Only	0.747 ± 0.041	0.404 ± 0.192
NC-No-New-Authors	0.738 ± 0.037	0.393 ± 0.181

Best results are highlighted in bold

Table 11 Results obtained in KISTI collection

Method	K	pF1
NC	0.940 ± 0.002	0.816 ± 0.009
NC-No-Training-Update	0.937 ± 0.002	0.810 ± 0.009
NC-Cosine	0.927 ± 0.003	0.803 ± 0.009
NC-No-New-Authors	0.888 ± 0.003	0.762 ± 0.008
NC-Similarity-Only	0.888 ± 0.003	0.761 ± 0.008

Best results are highlighted in bold

and KISTI have several authors with only few citations. Thus, many authors do not have any citation example in the training set. This characteristic leads to effectiveness losses when there is no identification of new authors (NC-No-New-Authors) and, consequently, when it uses only our similarity function (NC-Similarity-Only). In such situations, the performance decreases around 18 and 36 %, under the K and pF1 metrics, respectively, in BDBComp, and around 5 % and 6.7 % in KISTI for the same metrics. These results show why identifying new authors is important in scenarios in which new researchers are frequently inserted into a digital library.

In DBLP (Table 12), the most impacting factor when removed from our solution is the joint removal of the capability of improving coverage of the training set and the capability of identifying new authors as well as the change in the similarity functions. DBLP has many examples in the training set for each author in each ambiguous group, so the removal of the first two capabilities isolated has not much impact, but when removed together, they cause a drop in effectiveness.

To sum up, we can say that, in the tested datasets, all components of our solution are, in one way or another, important for improving effectiveness, with the least important component being the addition of new training examples.

4.5.4 Effectiveness without any training

To analyze the effectiveness of our method without any training example, we performed NC with default parameters (i.e., $w_c = 0.5$, $w_t = 0.3$, $w_v = 0.2$, $\alpha = 0.2$, $\delta = 0$, $\gamma = 0.2$ and $\phi = 0.1$) and with the self-training strategy described in Sect. 3.3.5 (NC-Self-Training). We defined the standard

Table 12 Results obtained in DBLP collection

Method	K	pF1
NC	0.919 ± 0.020	0.919 ± 0.023
NC-No-New-Authors	0.911 ± 0.023	0.915 ± 0.022
NC-No-Training-Update	0.910 ± 0.022	0.909 ± 0.026
NC-Similarity-Only	0.897 ± 0.025	0.903 ± 0.026
NC-Cosine	0.891 ± 0.028	0.887 ± 0.032

Best results are highlighted in bold

configuration as follows: (1) for the attribute weights, we used the values suggested in [18]; (2) for the other attributes, we used the set of common parameters in all collections that maximized effectiveness using cross-validation in the training set.

Note that besides having no information about the best parameters, in the experiments with no training data, NC has also no information about the authors who used a given ambiguous name. The groups of citations representing these authors have to be automatically discovered by using NC's incremental disambiguation capabilities (i.e., training data expansion and fusion of groups). In this context, the comparison with the methods NB, SVM and Cosine is not possible since they need examples of citations for each author to perform the assignments. Thus, in this section we compare our results with those obtained by the methods that can perform with no manually labeled training example, including the author grouping methods HHC and LASVM-DBSCAN. It is important to notice that SAND uses SLAND in its disambiguation task, thus, we do not include SLAND in this discussion.

The results obtained using the test set are shown in Tables 13, 14, and 15 for DBLP, KISTI and BDBComp, respectively. Considering the results obtained by NC with default parameters, in the DBLP and KISTI collections, the effectiveness decreases at most 13.3 and 23 % under K and pF1, respectively. Notice that, in these collections, NC is still better than SVM and NB which uses 50 % of the training data and know the structure of the disambiguation space a priori. In the BDBComp collection, the decrease is even smaller. The K and pF1 results decrease around 0.9 and 6.0 %, respectively. In fact, with no training data, NC still outperforms all author assignment baselines in BDBComp.

Comparing the results with the methods that need no manually labeled example, we notice that NC and NC-Self-Training are statistically tied with SAND, the best baseline, in the BDBComp collection. In the DBLP and KISTI collections, NC-Self-Training outperforms SAND under K metric with gains around 11.2 and 3.9 %, respectively. The results also show the effectiveness of the self-training procedure in BDBComp where NC-Self-Training achieves of about 7.1 % of gain under pF1 metric compared with NC manually con-

Table 13 Comparison of NC applied without training set with the author grouping methods in DBLP

Method	K	pF1
NC	0.797 ± 0.043	0.710 ± 0.094
NC-Self-Training	0.749 ± 0.070	0.661 ± 0.129
HHC	0.692 ± 0.084	0.598 ± 0.143
SAND	0.674 ± 0.091	0.556 ± 0.151
LASVMDSCAN	0.479 ± 0.097	0.319 ± 0.092

Best results are highlighted in bold

Table 14 Comparison of NC applied without training set with the author grouping methods in KISTI

Method	K	pF1
NC-Self-Training	0.927 ± 0.002	0.803 ± 0.008
SAND	0.892 ± 0.003	0.734 ± 0.008
NC	0.888 ± 0.004	0.731 ± 0.009
HHC	0.862 ± 0.003	0.671 ± 0.008
LASVMDSCAN	0.858 ± 0.004	0.638 ± 0.011

Best results are highlighted in bold

Table 15 Comparison of NC applied without training set with the author grouping methods in BDBComp

Method	K	pF1
SAND	0.942 ± 0.022	0.691 ± 0.155
HHC	0.937 ± 0.021	0.722 ± 0.156
NC-Self-Training	0.927 ± 0.020	0.678 ± 0.139
NC	0.905 ± 0.027	0.592 ± 0.142
LASVMDSCAN	0.883 ± 0.042	0.333 ± 0.231

Best results are highlighted in bold

figured and almost tying in KISTI under both metrics. Only in the DBLP there are more substantial losses due to the high ambiguity of the “C. Chen”, “J. Lee”, “S. Lee” and “Y. Chen” groups.

4.5.5 Sensitivity to parameters γ and ϕ

When we apply our method in an unsupervised manner, it is important to perform a careful adjustment of the parameters γ and ϕ in order to balance the degree of purity and fragmentation in the generated clusters. Figure 1 shows how the K metric varies in function of these values, considering the interval from 0 to 0.5, keeping constant the other parameters values ($w_c = 0.5$, $w_t = 0.3$, $w_v = 0.2$, $\alpha = 0.2$ and $\delta = 0$). We notice that values lower than 0.1, for both parameters in all collections, generate clusters with a low purity rate and, therefore, a lower K value. In DBLP and KISTI, the best results are obtained in the range from 0.1 to 0.2 for both parameters. In BDBComp, considering the γ parameter, the

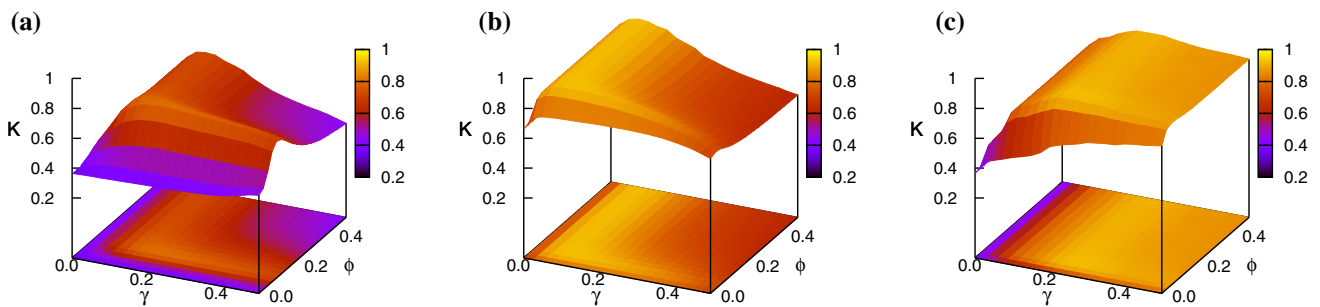


Fig. 1 Sensitivity to parameters γ and ϕ , **a** DBLP, **b** KISTI, **c** BDBComp

Table 16 Qualitative overview of the methods

Method	Efficiency	Effectiveness	Detect new authors	Improve the coverage of training data	Need for training data	Merge fragmented clusters	Sensitivity to parameters values
NC	High	High	Yes	Yes	None	Yes	Medium
SAND	Low	High	Yes	Yes	None	No	None
SLAND	Low	High	Yes	Yes	Medium	No	Medium
HHC	High	Medium	No	–	None	No	Medium
Cosine	High	High	No	No	High	No	Medium
LASVM-DBSCAN	Medium	Medium	No	–	Low	No	High
SVM	Low	Medium	No	No	High	No	High
NB	High	Low	No	No	High	No	None

best results are observed for the values from 0.2 to 0.3, while for the parameter ϕ there is no significant changes for values higher than 0.05, due to the small number of citations per author in this collection.

To set the other parameters, we suggest some guidelines: (1) the coauthor attribute weight must be higher than the other weights (configurations like $w_c = 0.5, w_t = 0.3, w_v = 0.2$ or $w_c = 0.4, w_t = 0.3, w_v = 0.3$ are usually good choices), (2) the α value should be between 0.2 and 0.5 in order to avoid great variations in the term weights due to the lack of information in the training set, and (3) the value of Δ_{min} should be close or equal to 0 so that the method can use the maximum available information in the test set.

4.5.6 Qualitative overview of the methods

Table 16 summarizes some characteristics of each evaluated method based on the results previously discussed. NC is the only method able to detect fragmented groups, i.e., distinct groups that contains citation of a unique author. Furthermore, it requires no training data, have low time complexity and high efficiency. SAND and SLAND have high effectiveness but low efficiency which hampers their use in large datasets. NC, SAND and SLAND are capable of detecting new authors, i.e., they are able to infer whether a citation belongs to an author in the disambiguated repository

or not. HHC, LASVM-DBSCAN and SVM effectiveness is medium, but, SVM has a major drawback as it requires training data associated with each author.

Considering the sensitivity to parameter values, we have observed that all methods, but SAND and NB that do not have parameters, can vary considerable in performance if not properly configured. However, as shown in Sect. 4.5.4, NC can be suitably configured automatically by using a self-training strategy similar to that of SAND.

5 Conclusions and future work

We have proposed a highly effective, efficient and practical method that smoothly combines several domain-specific heuristics to solve the AND task. Our experiments demonstrate the superiority of NC in a rich set of scenarios. In fact, NC produces some of the best results ever reported in the literature using a minimal set of features, with very low computational costs, while presenting most of the properties that an “ideal” AND method should possess.

As future work, we intend to exploit NC in incremental tasks, to improve its effectiveness by considering co-occurrence of terms within attributes, and to exploit relevance feedback in the method. We also intend to investigate techniques to detect noise in the cluster to discard them in

the similarity measure and, as training data change during the disambiguation process, how we can adapt the parameter values to these changes.

Acknowledgments This research is funded by INWeb (CNPq grant 57.3871/2008-6) and by the authors' individual grants from CNPq, CAPES, and FAPEMIG.

References

- Baeza-Yates, R.A., Ribeiro-Neto, B.: Modern information retrieval. Addison-Wesley Longman Publishing Co., Inc., Boston (1999)
- Bhattacharya, I., Getoor, L.: Collective entity resolution in relational data. *ACM Trans Knowl Discov Data* **1**(1) (2007)
- Bordes, A., Ertekin, S., Weston, J., Bottou, L.: Fast kernel classifiers with online and active learning. *J Mach Learning Res* **6**, 1579–1619 (2005)
- Cota, R.G., Ferreira, A.A., Nascimento, C., Gonçalves, M.A., Laender, A.H.F.: An unsupervised heuristic-based hierarchical method for name disambiguation in bibliographic citations. *J Am Soc Inform Sci Technol* **61**(9), 1853–1870 (2010)
- Fan, X., Wang, J., Pu, X., Zhou, L., Lv, B.: On graph-based name disambiguation. *J Data Inform Qual* **2**, 10:1–10:23 (2011)
- Ferreira, A.A., Veloso, A., Gonçalves, M.A., Laender, A.H.F.: Effective self-training author name disambiguation in scholarly digital libraries. In: Proceedings of the 10th Annual Joint Conference on Digital Libraries, pp. 39–48 (2010)
- Ferreira, A.A., Gonçalves, M.A., Laender, A.H.F.: A brief survey of automatic methods for author name disambiguation. *SIGMOD Record* **41**(2), 15–26 (2012)
- Ferreira, A.A., Silva, R., Gonçalves, M.A., Veloso, A., Laender, A.H.F.: Active associative sampling for author name disambiguation. In: Proceedings of the 12th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 175–184 (2012)
- Ferreira, A.A., Veloso, A., Gonçalves, M.A., Laender, A.H.F.: Self-training author name disambiguation for information scarce scenarios. *J Am Soc Inform Sci Technol* **65**(6), 1257–1278 (2014)
- Han, H., Giles, L., Zha, H., Li, C., Tsioutsouliklis, K.: Two supervised learning approaches for name disambiguation in author citations. In: Proceedings of the 4th ACM/IEEE Joint Conference on Digital Libraries, pp. 296–305 (2004)
- Han, H., Xu, W., Zha, H., Giles, C.L.: A hierarchical naive bayes mixture model for name disambiguation in author citations. In: Proceedings of the ACM Symposium on Applied Computing, pp. 1065–1069 (2005)
- Han, H., Zha, H., Giles, C.L.: Name disambiguation in author citations using a K-way spectral clustering method. In: Proceedings of JCDL, pp. 334–343 (2005)
- Holm, S.: A Simple Sequentially Rejective Multiple Test Procedure. *Scand J Stat* **6**(2), 65–70 (1979)
- Huang, J., Ertekin, S., Giles, C.L.: Efficient name disambiguation for large-scale databases. In: Proceedings of European Conference on Principles and Practice of Knowl. Discovery in Databases, pp. 536–544 (2006)
- Kanani, P., McCallum, A., Pal, C.: Improving author coreference by resource-bounded information gathering from the web. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 429–434 (2007)
- Kang, I.S., Na, S.H., Lee, S., Jung, H., Kim, P., Sung, W.K., Lee, J.H.: On co-authorship for author disambiguation. *Inform Process Manag* **45**(1), 84–97 (2009)
- Kang, I.S., Kim, P., Lee, S., Jung, H., You, B.J.: Construction of a large-scale test set for author disambiguation. *Inform Process Manag* **47**(3), 452–465 (2011)
- Lee, D., On, B.W., Kang, J., Park, S.: Effective and scalable solutions for mixed and split citation problems in digital libraries. In: Proceedings of the 2nd International Workshop on Inf. Quality in Inf. Systems, pp. 69–76 (2005)
- Liu, W., Islamaj Doan, R., Kim, S., Comeau, D.C., Kim, W., Yeganova, L., Lu, Z., Wilbur, W.J.: Author name disambiguation for pubmed. *J Assoc Inform Sci Technol* **65**(4), 765–781 (2014)
- Pereira, D.A., Ribeiro-Neto, B.A., Ziviani, N., Laender, A.H.F., Gonçalves, M.A., Ferreira, A.A.: Using web information for author name disambiguation. In: Proceedings of the 9th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 49–58 (2009)
- Shu, L., Long, B., Meng, W.: A latent topic model for complete entity resolution. In: Proceedings of the IEEE International Conference on Data Engineering, pp. 880–891 (2009)
- Tang, J., Fong, A.C.M., Wang, B., Zhang, J.: A unified probabilistic framework for name disambiguation in digital library. *IEEE Trans Knowl Data Eng* **24**(6), 975–987 (2012)
- Torvik, V.I., Smalheiser, N.R.: Author name disambiguation in medline. *ACM Trans Know Discov Data* **3**(3), 1–29 (2009)
- Treeratpituk, P., Giles, C.L.: Disambiguating authors in academic publications using random forests. In: Proceedings of the 9th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 39–48 (2009)
- Veloso, A., Ferreira, A.A., Gonçalves, M.A., Laender, A.H.F., Meira Jr, W.: Cost-effective on-demand associative author name disambiguation. *Inform Process Manag* **48**(4), 680–697 (2012)
- Wu, H., Li, B., Pei, Y., He, J.: Unsupervised author disambiguation using DempsterShafer theory. *Scientometrics* **101**(3), 1955–1972 (2014)