A Collective Approach to Scholar Name Disambiguation

Anonymous Author(s)

ABSTRACT

Scholar name disambiguation remains a hard and unsolved problem, which brings various troubles for bibliography data analytics. Most existing methods handle name disambiguation separately that tackles one name at a time, and neglect the fact that disambiguation of one name affects the others. Further, it is typically common that only limited information is available for bibliography data, *e.g.,* only basic paper and citation information is available in DBLP. In this paper, we propose a collective approach to name disambiguation, which takes the connection of different ambiguous names into consideration. We reformulate bibliography data as a heterogeneous multipartite network, which initially treats each author reference as a unique author entity, and disambiguation results of one name propagate to others of the network. To further deal with the sparsity problem caused by limited available information, we also introduce word-word and venue-venue similarities, and we finally measure author similarities by assembling similarities from four perspectives. Using real-life data, we experimentally demonstrate that our approach is both effective and efficient.

1 INTRODUCTION

Scholar name ambiguity is a common data quality problem for digital libraries such as DBLP [1], Google Scholar [2] and Microsoft Academic Search [3], and has raised various troubles in scholar search, document retrieval and so on [20, 27, 38]. For example, we read an interesting paper written by "Wei Wang" in DBLP, and we want to find more his/her recent publications. However, there are over 200 authors sharing the same name "Wei Wang" in DBLP [18], and the total number of their publications is over 2,000. Hence, it is really time-consuming to find those publications written by the "Wei Wang" in whom we are interested. It is also common that only limited information is available in bibliography data. For example, DBLP only provides basic paper and citation information, *e.g.,* author names, publication title, venue and publication year. This makes name disambiguation even more challenging to attack.

Most existing methods tackle name disambiguation individually [5, 9, 10, 12, 14–18, 24, 27, 30–35]. For each single name to be disambiguated, these methods deal with the papers having the specific name as an author only. However, by tackling each name separately and independently, these methods neglect the connection between these sub-problems. Figure 1 is an example to demonstrate this problem, which shows two papers written by "Ying Zhang" and "Wei Xu" in DBLP. When disambiguating the name "Wei Xu", single name disambiguation methods consider two "Ying Zhang"

WWW, May, 2019, San Francisco.

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Ying Zhang, Wei Xu, Liang Wang, Bruno Rossetto:
Phytoplankton Zooplankton system with bounded random parameter. FSKD 2011: 1606-1610

Gang Wu, Wei Xu, Ying Zhang, Yimin Wei:
A preconditioned conjugate gradient algorithm for GeneRank with application to microarray data mining. Data Min. Knowl. Discov. 26(1): 27-56 (2013)

Figure 1: Example taken from DBLP

(author references) are the same person (author entity). As a result, these two "Wei Xu" have the same coauthor. Coauthors are used as a strong evidence in many methods [10, 27, 35]. Hence, it is very likely that they refer to the same person. However, there are two different "Wei Xu" and two different "Ying Zhang" in this example. More troubles may appear when multi-hop coauthorships are used as features [10, 27]. For example, "Jianxin Li" in UWA is a coauthor and 2-hop coauthor of "Wei Wang" in UNSW, and "Jianxin Li" in BUAA is a 2-hop coauthor of "Wei Wang" in UCLA.

Contributions & Roadmap. To this end, we propose a collective approach to dealing with scholar name disambiguation.

(1) We propose an iterative method via collective clustering, referred to as NDCC, to deal with scholar name disambiguation (Sections 3 and 5). Our collective clustering method uses a heterogeneous multipartite network model, and the disambiguation results of one name affect the others. By representing each author reference as a unique author in the beginning, NDCC alleviates the problem caused by ambiguous coauthor names. In each iteration, a single name is disambiguated, and the network is updated (author nodes merging) according to the disambiguation results. The process repeats until the network converges.

(2) We develop a novel metric for determining the author similarity by assembling the similarities of four features (*i.e.,* coauthors, venues, titles and coauthor names) available in bibliography data (Section 4). Here we differentiate coauthors from coauthor names, as the latter is an ambiguous feature (as illustrated by the example in Figure 1). To overcome the sparsity of certain venues and title words, a word embedding method is utilized to capture the semantic similarity of titles, and the similarity of venues is measured by the overlapping degree of authors publishing papers on venues.

(3) Using three real-life datasets (AMiner, ACM, DBLP), we finally conduct an extensive experimental study (Section 6). We find that our approach NDCC is both effective and efficient, compared with the state-of-the-art methods CE [7], GHOST [10], CSLR [18] and MIX [17]. (a) NDCC on average improves the Macro-F1 over (CE, GHOST, CSLR, MIX) by (17.87%, 23.25%, 16.65%, 45.39%) on AMiner, (25.36%, 24.26%, 14.16%, 37.46%) on ACM, and (13.11%, 23.31%, 8.47%, 50.37%) on DBLP, respectively. (b) NDCC is on average (18, 195, 19) times faster than (CE, CSLR and MIX) on AMiner, (15, 8) times faster than (CE, MIX) on ACM, and 10 times faster than MIX on DBLP, respectively. (c) While GHOST on (AMiner, ACM, DBLP), CSLR on (ACM, DBLP) and CE on DBLP could not finish within

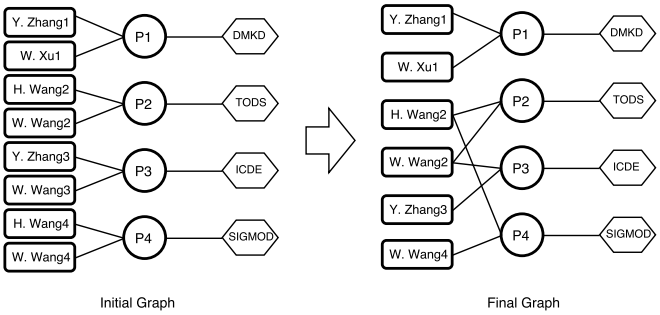


Figure 2: Example heterogeneous multipartite network, such that the left represents the initial scholarly data, and the right represents the final disambiguation results.

6 hours, our NDCC finished on (AMiner, ACM, DBLP) in (98, 543, 2106) seconds, respectively.

2 PROBLEM FORMULATION

In this section, we first introduce basic notations, and then present a formal definition of scholar name disambiguation.

Basic notations. For bibliography data D, each its citation record contains title, author names, venue, and publication year, and we use the Heterogeneous Information Networks (HINs), which are used widely in complex network analysis [19, 37], to model D. Considering that there are no direct relations among nodes with the same type, we refer to this type of HINs as *heterogeneous multipartite networks*, which is formally defined as follows.

A heterogeneous multipartite network is an HIN [25] whose node set can be divided into several disjoint sets $V_0, V_1, \ldots, V_n$ such that every edge connects a node in V_i to another in V_j with $i \neq j$. Node sets V_0 , $V_1, \ldots, V_n$ are called the parts of the network, and the types of nodes in the same part are identical.

We consider each author reference as a unique author entity initially, then the bibliography data is represented as a 4-part heterogeneous multipartite network, containing the sets of author nodes (A), paper nodes (P), venue nodes (V) and title word nodes (T). Figure 3 shows the network schema [25] of the heterogeneous multipartite network for scholar name disambiguation, where there are three types of edges in this network, and edges connecting author nodes to paper nodes, paper nodes to venue nodes and paper nodes to word nodes. We use three matrices to represent the heterogeneous multipartite network $\mathcal{G}$: $\mathbf{W}^{AP}$, $\mathbf{W}^{PT}$ and $\mathbf{W}^{PV}$, storing author-paper edges, paper-(title) word edges and paper-venue edges in heterogeneous multipartite network $\mathcal{G}$, respectively.

We now formalize scholar name disambiguation with the definition of heterogeneous multipartite networks.

Problem statement. Given a heterogeneous multipartite network $\mathcal{G}$, the task of scholar name disambiguation is to adjust author nodes and edges between author and paper nodes, such that for each author a in A, the set of paper nodes P_a connected to a ideally contains all and only those papers written by author a.

Besides the relationships directly available from the heterogeneous multipartite network $\mathcal{G}$, we also use the following indirect relationships for the author similarity computation.

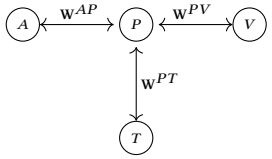


Figure 3: Heterogeneous multipartite network for scholar name disambiguation. There are four parts: author (A), paper (P), venue (V) and title word (T), and their relationships are labelled with the corresponding adjacency matrices.

(1) Coauthorship $\mathbf{W}^{AA}$ is a matrix for valid coauthorship in $\mathcal{G}$, where $\mathbf{W}^{AA}_{i,j}$ is the times that authors i and j collaborates. A coauthor relation is valid if two authors have different names. Note that, it is possible that a paper is written by more than one author with the same name. However, we cannot distinguish them without additional information, such as email addresses. In this case, we just keep an arbitrary author reference and neglect the others. We also dismiss self-coauthorships by setting all $\mathbf{W}^{AA}_{i,i} = 0$, such that $\mathbf{W}^{AA} = \mathbf{W}^{AP} \times (\mathbf{W}^{AP})^{\mathbf{T}} - \mathbf{diag}(\mathbf{W}^{AP} \times (\mathbf{W}^{AP})^{\mathbf{T}})$.

(2) 2-hop coauthorship $\mathbf{W}^{AA^2}$ is a matrix for 2-hop coauthorship in $\mathcal{G}$, where $\mathbf{W}^{AA^2}_{i,j}$ is the number of valid 2-hop coauthorship paths connecting authors i and j. To avoid the redundant information, we only consider *valid 2-hop coauthorship paths* connecting two authors [10]. Specifically, a valid 2-hop coauthorship path in the heterogeneous multipartite network $\mathcal{G}$ is an $APAPA$ path a_i-p_i-a_j-p_j-a_k, where $a_i \neq a_j$, $a_i \neq a_k$, $a_j \neq a_k$ and $p_i \neq p_j$.

(3) Matrices $\mathbf{W}^{AN}$ and $\mathbf{W}^{AN^2}$ are for author-(coauthor name) relations and author-(2-hop coauthor name) relations, respectively.

(4) Matrices $\mathbf{W}^{AV}$ and $\mathbf{W}^{AT}$ are for author-venue relations and author-word relations, respectively. $\mathbf{W}^{AV}_{a,v}$ is the number of papers that author a publishes in venue v, and $\mathbf{W}^{AT}_{a,t}$ is the times that author a uses word t. That is, $\mathbf{W}^{AV} = \mathbf{W}^{AP} \times \mathbf{W}^{PV}$ and $\mathbf{W}^{AT} = \mathbf{W}^{AP} \times \mathbf{W}^{PT}$.

(5) Considering title words or venues may be limited for an author's publication, we expand these words and venues by considering their similar words and venues. We use matrices $\mathbf{W}^{TT}$ and $\mathbf{W}^{VV}$ for word-word similarity and venue-venue similarity, respectively. We present author-(similar word) relations and author-(similar venue) relations with matrices $\mathbf{W}^{AST}$ and $\mathbf{W}^{ASV}$, respectively, such that $\mathbf{W}^{AST} = \mathbf{W}^{AT} \times \mathbf{W}^{TT}$ and $\mathbf{W}^{ASV} = \mathbf{W}^{AV} \times \mathbf{W}^{VV}$.

Table 1 lists the main symbols and their definitions.

3 SOLUTION FRAMEWORK

In this section, we introduce our solution framework NDCC, as illustrated in Figure 4.

(1) Data representation. We represent the bibliography data as a heterogeneous multipartite network, which brings a couple of benefits. First, scholar name disambiguation is formulated with a single network. Specifically, the author nodes in the network are either single author references or atomic clusters (each has several closely related author references) in the beginning. This is a good way to alleviate the error propagation problem caused by ambiguous coauthor names. We disambiguate the author names by updating the network, and the final network represents the name

Table 1: Main symbols

Symbols	Definitions
$\mathcal{G}$	heterogeneous multipartite network
A, P, V, T	set of author/paper/venue/word nodes in $\mathcal{G}$
$A^{(0)}$	set of author nodes in the initial network
N	set of author names in the bibliography data
$\mathbf{W}^{AP}, \mathbf{W}^{PT}, \mathbf{W}^{PV}$	adjacency matrices for (A-P), (P-T) and (P-V), respectively
$\mathbf{W}^{TT}, \mathbf{W}^{VV}$	matrices for (word-word) and (venue-venue) similarity
$\mathbf{W}^{AA}, \mathbf{W}^{AA^2}$	matrices for coauthorship and 2-hop coauthorship
$\mathbf{W}^{AN}, \mathbf{W}^{AN^2}$	matrices for author-(2-hop coauthor name) relations
$\mathbf{W}^{AV}, \mathbf{W}^{ASV}$	matrices for author-venue & author-(similar venue) relations
$\mathbf{W}^{AT}, \mathbf{W}^{AST}$	matrices for author-word & author-(similar word) relations
$\mathbf{d}^A, \mathbf{d}^V, \mathbf{d}^T$	vector for degree of each author/venue/word node
$\mathbf{d}^N$	vector for number of papers of each name
$\mathbf{k}$	vector for estimated author number of each name

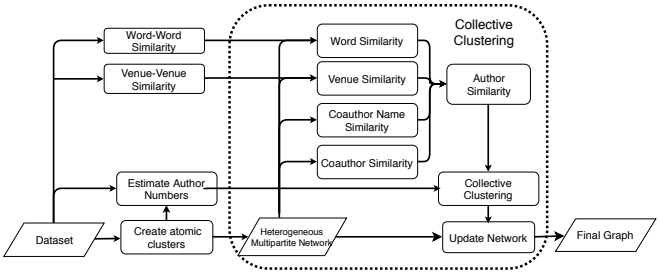


Figure 4: Framework of NDCC

disambiguation results. Second, it is flexible to incorporate extra types of entities such as affiliations and homepages when available.

(2) Similarity measurement. Because of the ambiguity of coauthor names, such as "Ying Zhang" and "Wei Xu" illustrated in Figure 1, we differentiate coauthors from coauthor names. Then we determine the author similarity by assembling the similarities from four perspectives (coauthor, venue, title and coauthor name). It is common that some authors only publish a small number of papers. In this case, venues and title words of their papers are not enough to capture their preferences and research interests, especially in the initial heterogeneous multipartite network, where each author node may only connect to a small number of paper nodes. To alleviate this sparsity problem, we extend the words for authors by considering the words similar to their title words, so do venues. We compute the venue-venue and word-word similarities before name disambiguation begins, as a preprocessing step.

(3) Collective clustering. Obviously the name disambiguation for one name may influence the others. For example, in Figure 2, merging of "H. Wang2" and "H. Wang4" leads to new common coauthor to "W. Wang2" and "W. Wang4", which affects the disambiguation of name "W. Wang". On the other hand, the disambiguation result of "H. Wang" is also affected by its coauthors. Based on this observation, we propose a bottom-up collective clustering method to deal with scholar name ambiguity. In collective clustering, disambiguation of one name affects others by changing the structure of the heterogeneous multipartite network. We iteratively select an author name and calculate the pairwise similarities of its author nodes. We then merge the pairs of author nodes with high similarity scores, and update the network accordingly. Each name needs to be disambiguated several times until it is fully disambiguated. For determining the stop condition, we estimate the number of authors for each name. A name is considered to be fully disambiguated if the number of its author nodes reaches the estimated number.

4 AUTHOR SIMILARITY MEASUREMENT

In this section, we present the author similarity measurement. First, we introduce the preprocessing step to deal with the sparsity problem, which is incorporated into author similarities. Then we propose a novel metric to assemble the similarities from four perspectives: coauthors, venues, titles and coauthor names.

4.1 Dealing with Sparsity

As pointed out in Section 3, some authors only publish a small number of papers, especially in the initial heterogeneous multipartite network, and it is hard to make a good judgment for these authors. To deal with this sparsity problem, we introduce word-word and venue-venue similarities to expand the limited information.

(1) Word-word similarity. The title is an important feature to calculate pairwise similarities of authors. The traditional unigram model treats each word separately and neglects their correlations. It is likely that two titles, which do not share common words, are correlated. For example, one title contains word "hardware" and the other contains word "circuit". Both are related to computer hardware. In this case, traditional unigram model returns a low similarity score. In [17, 18], string level or character level tolerance are used when comparing two titles. However, these methods cannot capture the semantic relation between two words either.

We propose to use Word2vec [21], which is an effective word embedding method, to capture the semantic correlations between words. It takes a text corpus as input and maps each word in the text corpus to a vector in a low dimensional space. First, we normalize all titles using NLTK [8] by turning them into lowercase, removing punctuation, tokenizing and removing stop words. All normalized titles are used as the training text corpus for Word2vec. Then the cosine similarity of word vectors is used as the word-word similarity, which is stored in a matrix denoted by $\mathbf{W}^{TT}$. We keep the pairs whose similarity scores are larger than a threshold σ_t, and disregards the others by setting the similarity scores to zeroes.

(2) Venue-venue similarity. We expand venues for each author, based on an observation that two venues are similar if a large portion of authors both publish papers in these two venues. For example, "SIGMOD" and "VLDB" are both top database conferences, and many authors both publish papers in "SIGMOD" and "VLDB". Hence, "SIGMOD" and "VLDB" are two similar venues.

Based on this observation, we propose to use the Jaccard index of the authors to measure venue-venue similarity. Formally, given two venues i and j, N_i and N_j represent the sets of author names who publish at least one paper in i and j, respectively. The similarity between venue i and j is defined as

$$\mathbf{W}^{VV}_{i,j} = \frac{|N_i \cap N_j|}{|N_i \cup N_j|}.$$

Here we only keep pairs whose similarity scores are above a threshold σ_v, and neglect the others by setting their scores to 0 in $\mathbf{W}^{VV}$.

4.2 Author Similarity Assembling

The *author similarity* is assembled by four similarities (coauthor, venue, title and coauthor name). Given two authors i and j with the same name n, inspired by [18], their similarity is defined as:

$$sim = \sqrt{\sum_{x \neq y} sim_x \times sim_y}, \tag{1}$$

where $x, y \in \{n, t, v, a\}$ and $sim_a, sim_n, sim_t, sim_v$ are coauthor, coauthor name, title and venue similarities, respectively. We omit (i, j) in Equation (1) and the equations in the sequel for simplicity.

We argue that two authors are likely to be the same person if they are similar in at least two aspects. For example, in Figure 2, "H. Wang2" and "H. Wang4" are similar in terms of coauthor names and venues, so it is likely that they are the same person. On the other hand, in Figure 1, although these two "Wei Xu" are similar in the perspective of coauthor name, they are not similar in other perspectives. Thus, they are not similar.

(1) Coauthor similarity. Inspired by the Histogram Intersection Kernel [26], which is often used as a similarity measurement of two histograms, we use the normalized Histogram Intersection Kernel to calculate the coauthor similarity sim_a, defined as

$$sim_a = \sum_k \frac{1}{\mathbf{d}_k^A} \min(\mathbf{W}_{i,k}^{AA}, \mathbf{W}_{j,k}^{AA}) + t(n)\{\sum_k \frac{1}{\mathbf{d}_k^A} \min(\mathbf{W}_{i,k}^{AA}, \mathbf{W}_{j,k}^{AA^2}) + \sum_k \frac{1}{\mathbf{d}_k^A} \min(\mathbf{W}_{i,k}^{AA^2}, \mathbf{W}_{j,k}^{AA})\}, \tag{2}$$

where $t(n) = \begin{cases} 1 & \text{if } \mathbf{k}_n \leq \theta \\ 0 & \text{otherwise} \end{cases}$. Here $\mathbf{d}_k^A$ is the number of papers written by author k, which serves the normalization factor, and θ is a threshold determining whether to use multi-hop coauthorships. The first part of the right side of Equation (2) measures the similarity between coauthors of i and j. The second considers multi-hop coauthors. Comparing with (1-hop) coauthor, multi-hop coauthors are less evidential. We notice that for names with high ambiguities, such as "Wei Wang", using weak evidential features like multi-hop coauthors may introduce errors, and decrease the accuracy performance. So for those names, we neglect the multi-hop coauthorship, and only use the first part.

Coauthor name, title and venue similarities are defined similarly.

(2) Coauthor name similarity.

$$sim_n = \sum_m \frac{1}{\mathbf{d}_m^N} \min(\mathbf{W}_{i,m}^{AN}, \mathbf{W}_{j,m}^{AN}) + t(n)\{\sum_m \frac{1}{\mathbf{d}_m^N} \min(\mathbf{W}_{i,m}^{AN}, \mathbf{W}_{j,m}^{AN^2}) + \sum_m \frac{1}{\mathbf{d}_m^N} \min(\mathbf{W}_{i,m}^{AN^2}, \mathbf{W}_{j,m}^{AN})\}, \tag{3}$$

where $t(n)$ is the same as the one in Equation (2), and $\mathbf{d}_m^N$ is the number of papers written by authors with name m.

(3) Title similarity.

$$sim_t = \sum_t \frac{1}{\mathbf{d}_t^T} \min(\mathbf{W}_{i,t}^{AT}, \mathbf{W}_{j,t}^{AT}) + \{\sum_t \frac{1}{\mathbf{d}_t^T} \min(\mathbf{W}_{i,t}^{AT}, \mathbf{W}_{j,t}^{AST}) + \sum_t \frac{1}{\mathbf{d}_t^T} \min(\mathbf{W}_{i,t}^{AST}, \mathbf{W}_{j,t}^{AT})\}, \tag{4}$$

where $\mathbf{d}_t^T$ is the number of papers containing the word t, and we use the bag-of-words model to represent titles. The first part of the right side of Equation (4) measures the similarity between words both author i and j used in their paper titles. The second part takes the similar words into consideration.

(3) Venue similarity.

$$sim_v = \sum_v \frac{1}{\mathbf{d}_v^V} \min(\mathbf{W}_{i,v}^{AV}, \mathbf{W}_{j,v}^{AV}) + \{\sum_v \frac{1}{\mathbf{d}_v^V} \min(\mathbf{W}_{i,v}^{AV}, \mathbf{W}_{j,v}^{ASV}) + \sum_v \frac{1}{\mathbf{d}_v^V} \min(\mathbf{W}_{i,v}^{ASV}, \mathbf{W}_{j,v}^{AV})\}, \tag{5}$$

where $\mathbf{d}_v^V$ is the number of papers published in venue v. The first part of the right side of Equation (5) measures the similarity between venues where both author i and j publish papers in. The second part takes similar venues into consideration.

Intuitively, the more two authors share the same related entities (coauthors, title words, venues and coauthor names), the more similar they are. Histogram Intersection Kernel is a common way to measure this similarity. At the same time, the weights of different entities are normalized by their frequencies. For example, if a coauthor publishes a lot of papers, then it should be considered as a weak evidence comparing to those who only publish one or two papers. Because productive authors are believed to be the experts connecting different communities. It is likely that they may collaborate with two or more authors with the same name. For instance, "Haixun Wang" in WeWork has over 100 papers, and collaborates both with "Wei Wang" in UCLA and "Wei Wang" in UNSW. From the title perspective, common words, such as "approach" and "system" are less representative comparing to uncommon words like "disambiguation" and "collective". So we differentiate the weights of words by the assumption that the more frequently a word appears in titles, the less the word is as an evidence. It is similar of the other two perspectives: venues and coauthor names.

5 COLLECTIVE CLUSTERING

In this section, we first introduce the collective clustering algorithm with speeding-up strategies for scholar name disambiguation. Then we analyze its convergence rate and time and space complexities.

5.1 Atomic Cluster Generation and Author Number Estimation

For scholar name disambiguation, some author references can be easily clustered together. For example, papers "Mining surprising periodic patterns", and "Discovering high order periodic patterns" share the same author names "Jiong Yang", "Wei Wang" and "Philip Yu". It is obvious that these two "Wei Wang" are the same person because they have two identical coauthor names. These two author references form an atomic cluster. Finding the atomic clusters as the bootstrap can reduce the size of the initial network, and improve the efficiency. We use a simple rule-based method to find the atomic clusters, inspired by the above example. Two author references are assigned to the same atomic clusters if they share more than 1 coauthor name. Similar bootstrap strategies are used widely in name disambiguation methods [7, 27, 32].

As mentioned in the solution framework in Section 3, the estimated number of authors for each name is used as the stop condition in collective clustering. Specifically, a name is considered as fully disambiguated if the number of authors of this name reaches the estimated one. Inspired by name ambiguity estimation in the paper [18], we introduce a statistical method, based on the statistics of author names in the bibliography data.

In most cases, names consist of a fixed number of components. For instance, an English name has three parts: first name, middle name and last name, and a Chinese name consists of the first name and last name. We assume that these parts are chosen independently from different multinomial distributions, and the probability of a full author name is the joint probability of its components [18]. Here we use the two-component names as an example to explain the main idea. Given a name n, its first name and last name are denoted by $F(n)$ and $L(n)$, which are independently drawn from multinomial distributions $Multi_F$ and $Multi_L$, respectively. The probability of an author with name n is $Pr(n) = Multi_F(F(n)) \times Multi_L(L(n))$. Then, the number of authors with name n is $\mathbf{k}_n = Pr(n) \sum_{e \in N} \hat{k}_e$, where $\hat{k}_e$ is the number of authors with name e, and $\sum_{e \in N} \hat{k}_e$ is the total number of authors in the bibliography data. Since $\hat{k}_e$ is unknown, we use its estimate $\mathbf{k}_e$ instead.

The parameters of $Multi_F$ are estimated by the maximum likelihood estimation. Specifically, π_f, the probability of a first name f appears, is estimated by $\pi_f = \frac{\sum n \in N, F(n)=f \mathbf{k}_n}{\sum_{e \in N} \mathbf{k}_e}$. So do the parameters in $Multi_L$. We use an EM-like method to calculate $\mathbf{k}$ and parameters in $Multi_F$ and $Multi_L$ iteratively. Specifically, in the beginning, we set $\mathbf{k}_n = 1$ for all names n. At the expectation step, we fix the parameters in $Multi_F$ and $Multi_L$, and update $\mathbf{k}$. It is possible that $\mathbf{k}_n < 1$ or $\mathbf{k}_n > |A_n^{(0)}|$, where $|A_n^{(0)}|$ is the number of atomic author clusters of name n. In this case, we round $\mathbf{k}_n$ to 1 if $\mathbf{k}_n < 1$, and $|A_n^{(0)}|$ for the second case. At the maximization step, we update the parameters in $Multi_F$ and $Multi_L$ with the current $\mathbf{k}$. Expectation and maximization steps are repeated until $\mathbf{k}$ converges.

5.2 Algorithm

Given an initial heterogeneous multipartite network $\mathcal{G}$, which is created directly from the bibliography data with bootstrap, as well as the preprocessing results: $\mathbf{W}^{VV}$, $\mathbf{W}^{TT}$ and $\mathbf{k}$, collective clustering returns the final author-paper matrix, where each author node represents an author entity in the real world, and connects to all and only all its paper nodes.

In collective clustering, disambiguation of one name affects the others by changing the structure of the heterogeneous multipartite network $\mathcal{G}$. In each iteration, we focus on a single name n, instead of directly employing hierarchical clustering methods to merge the author nodes with name n, until the number reaches $\mathbf{k}_n$. We merge the top K pairs with the highest similarity scores. Here we choose K as the half of the difference between the current author number and the estimated one. Formally,

$$K = \lceil \frac{|a(n)| - \mathbf{k}_n}{2} \rceil, \tag{6}$$

where $|a(n)|$ is the number of authors with name n in this iteration. Our framework also supports the other choices of K, and we leave this part as future work. Each name needs to be disambiguated several times until it is considered fully disambiguated, when the number of its authors reaches the estimated number. The final network is the disambiguation result.

Algorithm 1: Collective Clustering

Input: $\mathbf{W}^{AP}$, $\mathbf{W}^{PT}$, $\mathbf{W}^{PV}$, $\mathbf{W}^{VV}$, $\mathbf{W}^{TT}$, $\mathbf{k}$
Output: $\mathbf{W}^{AP}$ of the final network $\mathcal{G}$

1 Use BFS to calculate $\mathbf{W}^{AA}$, $\mathbf{W}^{AA^2}$, $\mathbf{W}^{AT}$, $\mathbf{W}^{AST}$, $\mathbf{W}^{AV}$, $\mathbf{W}^{ASV}$;
2 Initialize an empty queue que;
3 **foreach** *author name* n **do**
4 $a(n) \leftarrow$ the list of authors with name n;
5 **if** $|a(n)| > 1$ **then**
6 que.push(n);
7 **while** que *is not empty* **do**
8 $n \leftarrow que.pop()$;
9 $a(n) \leftarrow$ the list of authors with name n;
10 **if** $|a(n)| \leq \mathbf{k}_n$ **then**
11 **Continue**
12 $K \leftarrow \lceil \frac{|a(n)|-\mathbf{k}_n}{2} \rceil$;
13 Calculate pairwise similarities of authors in $a(n)$ with Eq. (1);
14 $t \leftarrow$ the K-th largest pairwise similarity score;
15 Merge author pairs whose similarity scores are no less than t;
16 Update the author-paper matrix $\mathbf{W}^{AP}$ according to Eq. (7);
17 Update $\mathbf{W}^{AA}$, $\mathbf{W}^{AA^2}$, $\mathbf{W}^{AT}$, $\mathbf{W}^{AST}$, $\mathbf{W}^{AV}$ and $\mathbf{W}^{ASV}$ according to Eq. (8, 9, 10);
18 que.push(n);
19 **Return** $\mathbf{W}^{AP}$.

Observe that it is time-consuming to re-calculate the matrices such as $\mathbf{W}^{AA}$ and $\mathbf{W}^{AA^2}$ in each iteration when network $\mathcal{G}$ is updated for the merging of author nodes, we introduce speeding-up strategies for the computation. We calculate and store those matrices such as $\mathbf{W}^{AA}$ and $\mathbf{W}^{AA^2}$ as a preprocessing step before iterations, and update them inside iterations. Considering the sparsity and dynamics of matrices, we use lists of treemaps to store $\mathbf{W}^{AA}$, $\mathbf{W}^{AA^2}$, $\mathbf{W}^{AT}$, $\mathbf{W}^{AST}$, $\mathbf{W}^{AV}$ and $\mathbf{W}^{ASV}$. Specifically, for each author name, we maintain a list of its author nodes. Each author node contains six treemaps to store the corresponding rows in these metrics, respectively. Considering that the author name is just an attribute attached to the author node, we do not store $\mathbf{W}^{AN}$ and $\mathbf{W}^{AN^2}$, as they can be extracted directly from $\mathbf{W}^{AA}$ and $\mathbf{W}^{AA^2}$, respectively.

We now explain the detail of collective clustering, where Algorithm 1 shows its overall process. In line 1, it uses breadth-first search to calculate all the metrics such as $\mathbf{W}^{AA}$, $\mathbf{W}^{AA^2}$ from the input. In lines 2-6, it uses a queue *que* to store the names to be disambiguated, which is initiated by pushing all names in the bibliography data, except those with only one paper. While *que* is not empty, it iteratively disambiguates names (lines 7–18). It pops out a name from *que*, denoted by n (line 8), and assigns $\mathbf{a}_n$ the list of author nodes with name n (line 9). If the size of $\mathbf{a}_n$ is no larger than $\mathbf{k}_n$, then the name n is believed to be fully disambiguated. In this case, it just continues to deal with the next name. Otherwise, in line 12, it calculates the number of pairs to be merged in this iteration by Equation (6), denoted by K. It then calculates the pairwise author similarity scores in $\mathbf{a}_n$, and finds K-th largest score t by

using a K-size minimum heap (line 14). Then it merges the author pairs whose similarity scores are no less than t, and updates the network $\mathcal{G}$ (equally, the matrix $\mathbf{W}^{AP}$) accordingly (lines 15, 16). It also needs to update $\mathbf{W}^{AA}$ *etc.* (line 17). Then it pushes n into *que*, and waits for the disambiguation results of the remaining names (line 18). After all names have been processed, it finally returns the disambiguation result $\mathbf{W}^{AP}$ (line 19).

5.3 Matrix Updates

In this subsection, we present the updating rules for matrices $\mathbf{W}^{AA}$, $\mathbf{W}^{AA^2}$, $\mathbf{W}^{AT}$, $\mathbf{W}^{AST}$, $\mathbf{W}^{AV}$ and $\mathbf{W}^{ASV}$. Given authors i and j to be merged, without loss of generality, we assume that j is merged to i, and the updated i is denoted as $\hat{i}$.

By the definition, the author-paper matrix $\mathbf{W}^{AP}$ is updated as follows.

$$\hat{\mathbf{W}}^{AP}_{k,p} = \begin{cases} \mathbf{W}^{AP}_{i,p} + \mathbf{W}^{AP}_{j,p} & \text{if } k = \hat{i} \\ \mathbf{W}^{AP}_{k,p} & \text{otherwise} \end{cases} \tag{7}$$

, where a hat denotes the updated matrix.

The author-author matrix $\mathbf{W}^{AA}$ is updated as follows.

$$\hat{\mathbf{W}}^{AA}_{k,l} = \begin{cases} 0 & \text{if } k = l \\ \mathbf{W}^{AA}_{i,l} + \mathbf{W}^{AA}_{j,l} & \text{if } k = \hat{i}, k \neq l \\ \mathbf{W}^{AA}_{k,i} + \mathbf{W}^{AA}_{k,j} & \text{if } l = \hat{i}, k \neq l \\ \mathbf{W}^{AA}_{k,l} & \text{otherwise} \end{cases} \tag{8}$$

Theorem 5.1. *The correctness of Equation (8).*

Proof Sketch: Correctness of Equation (8) can be proved by combining the definition of $\mathbf{W}^{AA}$ and Equation (7). □

Considering that merging of two author nodes incorporates a new 2-hop coauthorship, matrix $\mathbf{W}^{AA^2}$ is updated as follows.

$$\hat{\mathbf{W}}^{AA^2}_{k,l} = \begin{cases} 0 & k = l \\ \mathbf{W}^{AA^2}_{i,l} + \mathbf{W}^{AA^2}_{j,l} & \text{if } k = \hat{i}, k \neq l \\ \mathbf{W}^{AA^2}_{k,i} + \mathbf{W}^{AA^2}_{k,j} & \text{if } l = \hat{i}, k \neq l \\ \mathbf{W}^{AA^2}_{k,l} + \mathbf{W}^{AA}_{i,k} \times \mathbf{W}^{AA}_{j,l} + \mathbf{W}^{AA}_{i,l} \times \mathbf{W}^{AA}_{j,k} & \text{otherwise} \end{cases} \tag{9}$$

Theorem 5.2. *The correctness of Equation (9).*

Proof. We denote the set of valid coauthor paths and valid 2-hop coauthor paths connecting author k and l as $\mathcal{P}_{k,l}$ and $\mathcal{P}^2_{k,l}$, respectively, $a \in p$ if path p contains author a, and use hats to represent the updated sets. Then $\mathbf{W}^{AA}_{k,l} = |\mathcal{P}_{k,l}|$ and $\mathbf{W}^{AA^2}_{k,l} = |\mathcal{P}^2_{k,l}|$.
By the definition of the valid 2-hop coauthor path, if $k = l$, we have $\hat{\mathbf{W}}^{AA^2}_{k,l} = |\hat{\mathcal{P}}^2_{k,l}| = 0$.

Then we consider cases where $k \neq l$.
(1) If $k = \hat{i}$, then

$$\hat{\mathbf{W}}^{AA^2}_{k,l} = |\hat{\mathcal{P}}^2_{\hat{i},l}| = |\mathcal{P}^2_{i,l} \cup \mathcal{P}^2_{j,l}| = |\mathcal{P}^2_{i,l}| + |\mathcal{P}^2_{j,l}| = \mathbf{W}^{AA^2}_{i,l} + \mathbf{W}^{AA^2}_{j,l}.$$

(2) If $l = \hat{i}$, then

$$\hat{\mathbf{W}}^{AA^2}_{k,l} = |\hat{\mathcal{P}}^2_{k,\hat{i}}| = |\mathcal{P}^2_{k,i} \cup \mathcal{P}^2_{k,j}| = |\mathcal{P}^2_{k,i}| + |\mathcal{P}^2_{k,j}| = \mathbf{W}^{AA^2}_{k,i} + \mathbf{W}^{AA^2}_{k,j}.$$

(3) Otherwise

$$\begin{aligned} \hat{\mathbf{W}}^{AA^2}_{k,l} &= |\hat{\mathcal{P}}^2_{k,l}| = |\{p | p \in \hat{\mathcal{P}}^2_{k,l}, \hat{i} \in p\} \cup \{p | p \in \hat{\mathcal{P}}^2_{k,l}, \hat{i} \notin p\}| \\ &= |\{p | p \in \mathcal{P}^2_{k,l}, i \in p\} \cup \{p | p \in \mathcal{P}^2_{k,l}, j \in p\}| \\ &+ |\mathcal{P}_{k,i}| \times |\mathcal{P}_{j,l}| + |\mathcal{P}_{k,j}| \times |\mathcal{P}_{i,l}| + |\{p | p \in \mathcal{P}^2_{k,l}, i, j \notin p\}| \\ &= |\mathcal{P}_{k,i}| \times |\mathcal{P}_{j,l}| + |\mathcal{P}_{k,j}| \times |\mathcal{P}_{i,l}| + |\mathcal{P}^2_{k,l}| \\ &= \mathbf{W}^{AA}_{i,k} \times \mathbf{W}^{AA}_{j,l} + \mathbf{W}^{AA}_{i,l} \times \mathbf{W}^{AA}_{j,k} + \mathbf{W}^{AA^2}_{k,l}. \ \square \end{aligned}$$

Similarly, we update $\hat{\mathbf{W}}^{AV}$, $\hat{\mathbf{W}}^{ASV}$, $\hat{\mathbf{W}}^{AT}$ and $\hat{\mathbf{W}}^{AST}$ by

$$\begin{aligned} \hat{\mathbf{W}}^{AV}_{k,v} &= \begin{cases} \mathbf{W}^{AV}_{i,v} + \mathbf{W}^{AV}_{j,v} & \text{if } k = \hat{i} \\ \mathbf{W}^{AV}_{k,v} & \text{otherwise} \end{cases} \\ \hat{\mathbf{W}}^{ASV}_{k,v} &= \begin{cases} \mathbf{W}^{ASV}_{i,v} + \mathbf{W}^{ASV}_{j,v} & \text{if } k = \hat{i} \\ \mathbf{W}^{ASV}_{k,v} & \text{otherwise} \end{cases} \\ \hat{\mathbf{W}}^{AT}_{k,t} &= \begin{cases} \mathbf{W}^{AT}_{i,t} + \mathbf{W}^{AT}_{j,t} & \text{if } k = \hat{i} \\ \mathbf{W}^{AT}_{k,t} & \text{otherwise} \end{cases} \\ \hat{\mathbf{W}}^{AST}_{k,t} &= \begin{cases} \mathbf{W}^{AST}_{i,t} + \mathbf{W}^{AST}_{j,t} & \text{if } k = \hat{i} \\ \mathbf{W}^{AST}_{k,t} & \text{otherwise} \end{cases} \end{aligned} \tag{10}$$

Theorem 5.3. *The correctness of Equation (10).*

Proof Sketch: Since W^{PV} is static, we can prove the correctness of updating W^{AV} by combining definition of W^{AV} and updating rules of W^{AP}. Considering W^{VV} is also static, Then we can prove the correctness of updating rule for W^{ASV} by definition of W^{ASV} and updating rules of W^{AV}. Similarly, we can prove the correctness of updating rules for W^{AT} and W^{AST}. □

After merging author j to author i, we delete the corresponding rows and columns from the updated matrices.

5.4 Convergence and Complexity Analyses

We denote the set of author nodes in the initial heterogeneous multipartite network as $A^{(0)}$. It is easy to find that the size of A is non-increasing. So it is obvious that collective clustering converges.

We denote the largest number of papers written by the authors with the same name as ℓ, and then prove the bound of the number of iterations as the following.

Theorem 5.4 (Iteration number). *The iteration number of collective clustering is no more than* $|N|(\log(\ell) + 2)$.

Proof. We denote the iteration number as T_N and the number of iterations dealing with name n as T_n. Then $T_N = \sum_{n \in N} T_n$. We also denote the number of authors with name n after the i-th iteration dealing with n as $|A^{(i)}_n|$.

Initially, $|A^{(0)}_n|$ is the number of atomic authors with name n. According to Equation (6), $A^{(i+1)}_n = |A^{(i)}_n| - \lceil \frac{|A^{(i)}_n| - \mathbf{k}_n}{2} \rceil$. Then we have $|A^{(i)}_n| = \lfloor \frac{|A^{(0)}_n| - \mathbf{k}_n}{2^i} \rfloor + \mathbf{k}_n$ and $T_n = \lceil \log(|A^{(0)}_n| - \mathbf{k}_n) \rceil + 1$. Finally, we have

$$\begin{aligned} T_N &= \sum_n T_n = \sum_n (\lceil \log(|A^{(0)}_n| - k_n) \rceil + 1) \leq 2|N| + \sum_n \log(|A^{(0)}_n|) \\ &\leq 2|N| + \sum_n \log(\ell) = |N|(\log(\ell) + 2). \ \square \end{aligned}$$

Then we analyze the time and space complexities of collective clustering. We use the same notations as the proof of Theorem 5.4.

Table 2: Statistics of real-life bibliography datasets

Name	$\|P\|$	$\|T\|$	$\|V\|$	$\|N\|$
AMiner	1,397,240	233,503	16,442	1,062,896
ACM	2,381,719	327,287	273,274	2,002,754
DBLP	3,566,329	251,429	12,486	1,871,439

We assume that each paper has no more than α keywords in its title, and has no more than β authors. The time complexity of creating initial matrices is $O(|A^{(0)}|(\ell^2\beta^2\log(\ell\beta) + \ell\alpha\log(\ell\alpha)))$. Since elements in treemaps are sorted, we only need to traverse the corresponding treemaps to calculate the author similarity, which takes linear time *w.r.t.* the sizes of treemaps. So it takes $O(\ell^2\beta^2 + \ell\alpha)$ time to calculate the similarity of two authors. For each pair of author nodes to be merged, based on Equations (8), (9), and (10), it takes $O(\ell^2\beta\log(\ell\beta) + \ell\alpha\log(\ell\alpha))$ time to update the matrices as well as $\mathcal{G}$. In most cases, a paper contains no more than 10 authors, and no more than 10 keywords. Treating α and β as constants, the time complexities of calculating the similarity of two authors and merging two author nodes are $O(\ell^2)$ and $O(\ell^2\log(\ell))$, respectively. From Theorem 5.4, each name n is disambiguated $\lceil \log(|A_n^{(0)}| - \mathbf{k}_n)\rceil + 1$ times. In the i-th iteration dealing with name n, it takes $O(|A_n^{(i)}|^2\log(|A_n^{(i)}|))$ time to find the K-th largest similarity score. Putting these together, the time complexity of collective clustering is $O(\ell^2\log(\ell)(H + |A^{(0)}|\log(\ell)))$, where $H = \sum_n(|A_n^{(0)}|^2)$ is the number of atomic author pairs sharing the same names.

It takes $O(|A^{(0)}| + |P|(1+\beta+\alpha))$ space to store $\mathcal{G}$ and preprocessing results, and $O(|A^{(0)}|(\ell^2\beta^2 + \ell\alpha))$ space to store the other matrices. By considering α and β as constants, the space complexity is $O(|A^{(0)}|\ell^2)$.

6 EXPERIMENTAL STUDY

In this section, we present an extensive experimental study of NDCC. Using three real datasets, we conduct four sets of experiments to evaluate (1) the effectiveness and efficiency of NDCC versus the state-of-the-art methods CE [7], GHOST [10], CSLR [18] and MIX [17], (2) the effectiveness of NDCC with non-collective clustering, (3) the effectiveness of author number estimation, and (4) the effectiveness of word-word and venue-venue similarities, and the impacts of parameters on accuracy and efficiency.

6.1 Experimental Settings

We first introduce our experimental settings.

Datasets. We use three commonly used real-life datasets AMiner (http://www. aminer.org) [27–29, 32] , ACM (http://dl.acm.org) [32] and DBLP (http://dblp.uni-trier.de) [18] for scholar name disambiguation. Different from previous works which use small size subsets, we build datasets from the whole public available meta-data files directly. Statistics of these datasets are listed in Table 2.

The test set comes from https://aminer.org/disambiguation, which is commonly used in name disambiguation tasks [27, 32]. It contains 6,730 labeled papers of 110 author names. We compare the labelled papers with each dataset, and use their overlapped ones as the corresponding testing dataset. We use the Macro-F1 score to evaluate the effectiveness.

Comparison algorithms. Although NDCC can be easily extended to incorporate other information like homepages and email addresses, the datasets that we use only contain citation information, like many other digital libraries. Thus, we compare NDCC with the following four state-of-the-art methods, which also use citation information only.

(a) CE [7] is a collective entity resolution method for relational data. The similarity function considers both attributes and relational information, and a greedy agglomerative clustering method is used to merge most similar clusters. Moreover, since accuracy results are sensitive to the threshold of clustering, we try different values, and report the best results.

(b) GHOST [10] is a graph-based method employing coauthorship only. Its similarity function considers both quantity and quality (length) of paths, and an affinity propagation clustering method is used to generate clusters of author references of the focused name.

(c) CSLR [18] first groups the author references based on coauthorships to generate initial clusters. Then these clusters are merged by venue-based and title-based similarities.

(d) MIX [17] is a supervised method. Random forests are used to calculate pairwise distances, and DBSCAN is used to group the author references. For effectiveness evaluation, we randomly choose 5 (other) labelled names as the training set for each author name to be disambiguated. For efficiency evaluation, we randomly choose 5 labelled names to train the model, and use the others for testing.

Implementation. In NDCC, the threshold for word-word similarity σ_t is set to 0.75, the threshold for venue-venue similarity σ_v is set to 0.02, and the threshold for using weak evidences θ is set to 20. For the other methods, all parameters are set to their default values. All experiments are conducted on a machine with 2 Intel Xeon E5-2630 2.4GHz CPUs and 64 GB of Memory, running 64-bit windows 7 professional system. Each experiment is repeated 5 times, and the average is reported here.

6.2 Experimental Results

We next present our findings.

Exp-1: Performance comparison with other algorithms. In the first set of experiments, we evaluate the effectiveness and efficiency of NDCC compared with CE, GHOST, CSLR and MIX.

Exp-1.1: Accuracy performance comparison. The accuracy results for all methods in three datasets are shown in Table 3 and Table 4. We rank the author names in the test set based on paper numbers. We list the Macro-F1 scores of the top 10 names with largest number of papers in Table 3. The number of real authors for each name and the corresponding total number of publications are listed in table 5. We also calculate the Macro-F1 scores of top M names listed in Table 4, where $M \in [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]$. For each row, the top performer is highlighted in the bold font.

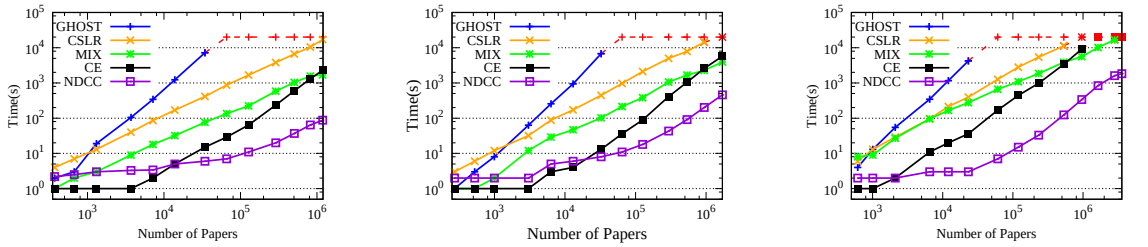
We observe that NDCC consistently performs better than the other methods. NDCC achieves the best performances on 8, 8 and 6 out of 10 names listed in Table 3 in AMiner, ACM and DBLP respectively. With the top 100 names as the testing dataset, NDCC improves the Macro-F1 over (CE, GHOST, CSLR, MIX) by (17.87%, 23.25%,16.65%, 45.39%) on AMiner, (25.36%, 24.26%, 14.16%, 37.46%) on ACM, and (13.11%, 23.31%, 8.47%, 50.37%) on DBLP, on average, respectively. The training data generated by MIX is quite biased,

Table 3: Comparison of accuracy performance with the top 10 names using Macro-F1 scores (%)

Name	AMiner					ACM					DBLP				
	CE	GHOST	CSLR	MIX	NDCC	CE	GHOST	CSLR	MIX	NDCC	CE	GHOST	CSLR	MIX	NDCC
Wen Gao	64.07	46.10	87.32	8.80	**91.77**	90.39	48.85	90.72	8.05	**96.17**	91.87	79.67	95.39	3.76	**96.34**
Lei Wang	41.78	20.30	53.16	39.02	**59.67**	7.75	24.00	23.84	48.67	**55.05**	14.38	15.26	57.17	29.94	**77.36**
David E. Goldberg	85.20	72.95	80.99	7.56	**98.32**	82.92	74.54	91.86	8.10	**98.46**	79.34	73.93	97.11	5.67	**100.0**
Yu Zhang	52.79	30.88	53.16	50.00	**67.20**	10.67	25.19	41.39	48.37	**68.82**	16.56	15.69	**68.17**	33.33	57.30
Jing Zhang	43.71	33.07	**56.32**	54.27	50.31	13.97	22.15	53.01	56.92	**67.35**	20.70	12.69	**63.68**	46.30	61.77
Lei Chen	51.83	36.57	**76.01**	13.10	72.41	53.47	39.38	**78.00**	12.15	72.17	59.53	46.10	**75.14**	8.03	70.10
Yang Wang	36.36	39.96	29.55	18.59	**42.55**	19.37	**36.44**	34.65	20.30	34.94	38.06	**62.82**	43.51	23.63	44.10
Bing Liu	56.32	40.77	60.99	6.31	**62.23**	47.32	49.13	70.33	5.33	**73.62**	61.62	42.68	68.87	4.98	**73.06**
Hao Wang	35.81	41.46	48.11	43.73	**59.38**	13.55	39.40	52.32	62.89	**56.53**	20.64	14.40	48.41	37.85	**57.27**
Gang Chen	43.40	46.52	57.68	46.65	**59.68**	44.69	59.39	50.10	44.63	**62.67**	56.73	46.13	61.05	17.86	**66.13**

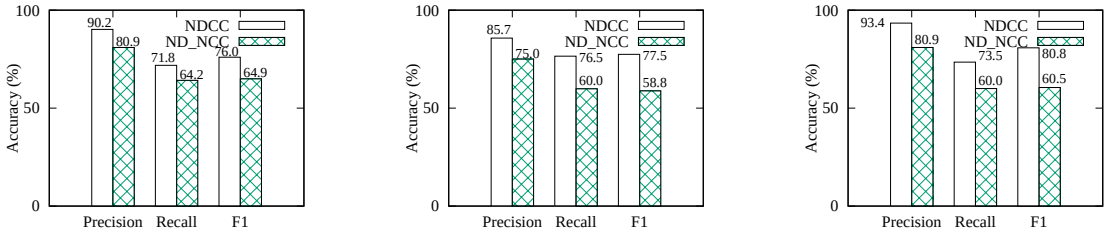
Table 4: Comparison of accuracy performance using Macro-F1 scores (%)

# Top Names	AMiner					ACM					DBLP				
	CE	GHOST	CSLR	MIX	NDCC	CE	GHOST	CSLR	MIX	NDCC	CE	GHOST	CSLR	MIX	NDCC
10	51.13	40.86	60.33	28.80	**66.35**	38.41	41.85	58.62	31.54	**68.58**	45.94	40.94	67.85	21.14	**73.05**
20	48.81	44.79	60.69	24.49	**67.66**	34.10	43.17	56.16	32.08	**71.75**	45.55	42.63	70.13	22.29	**71.88**
30	48.99	50.71	56.78	24.44	**70.75**	35.64	49.06	56.41	31.60	**72.37**	52.60	51.8	71.09	21.88	**77.70**
40	53.04	51.95	57.57	27.85	**70.5**	39.95	51.79	57.63	35.03	**72.14**	56.26	53.32	71.98	25.56	**79.67**
50	52.63	51.94	58.16	27.93	**72.03**	42.66	52.79	58.31	35.20	**74.98**	58.36	54.57	73.92	26.16	**79.88**
60	54.91	51.19	58.36	29.58	**73.04**	46.47	53.25	60.01	35.76	**76.03**	61.36	53.44	74.06	27.27	**79.53**
70	57.31	52.05	58.25	29.99	**74.45**	49.23	52.68	60.22	37.41	**76.90**	63.93	54.58	74.38	28.27	**80.58**
80	58.30	52.44	58.38	30.39	**74.60**	51.06	53.44	61.19	38.58	**77.20**	66.68	56.71	73.90	28.64	**81.00**
90	57.63	52.23	57.88	30.07	**75.51**	51.15	52.78	60.62	38.72	**76.92**	66.93	56.75	73.01	29.48	**80.93**
100	58.04	52.72	59.32	30.58	**75.97**	52.10	53.20	61.81	40.00	**77.46**	67.65	57.45	72.29	30.39	**80.76**



(a) AMiner (b) ACM (c) DBLP

Figure 5: Running time on AMiner, ACM and DBLP *w.r.t.* the network size.



(a) AMiner (b) ACM (c) DBLP

Figure 6: Comparison of accuracy performance between NDCC and ND_NCC

which leads to its low F1 scores. GHOST only uses coauthorships, which explains its unsatisfactory accuracy performance. Besides, 3 and 4-hop coauthorships used in GHOST are weak evidences. The ambiguity of (multi-hop) coauthors further harms the accuracy results. CE neglects the venue-venue and word-word similarities, which leads to its low F1 scores. CSLR is a single name disambiguation method that also considers the similarity between venues and words to alleviate the problem caused by limited information. NDCC outperforms it by (28.07%, 25.32% and 11.71%) on (AMiner, ACM, DBLP), which justifies the advantage of collective clustering.

Table 5: Statistics of the top 10 names in the test set and detected author numbers in three datasets

Name	#Authors	# Publications	AMiner	ACM	DBLP
Wen Gao	10	461	12	12	21
Lei Wang	106	289	91	98	122
David E. Goldberg	3	211	4	4	18
Yu Zhang	65	209	62	87	99
Jing Zhang	76	198	61	74	93
Lei Chen	35	179	39	34	52
Yang Wang	48	177	52	59	68
Bing Liu	16	171	32	29	23
Hao Wang	46	165	48	55	73
Gang Chen	40	163	35	44	57

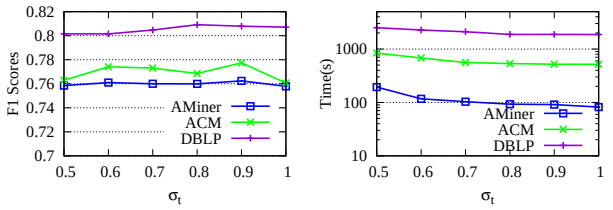
Exp-1.2: Efficiency performance comparison. To evaluate the efficiency of NDCC, we extract several subsets from AMiner (ACM, DBLP) with different sizes by author names. First, we generate several subsets of author names from AMiner (ACM, DBLP), with sizes ranging from 50 to 1M (2M and 1.8M on ACM and DBLP, respectively). To maintain the consistency among these sets, we make sure that smaller sets are subsets of bigger ones. For each set of author names, we extract all papers written by authors in this set to generate the corresponding subset. In this way, we make sure that the generated subsets are dense. The results of running time on AMiner, ACM, DBLP are reported in Figure 5(a), 5(b) and 5(c), respectively. The dotted lines denote that the corresponding running time is beyond 6 hours or the methods run out of memory.

The results tell us that NDCC is more efficient than the other four methods. (a) NDCC is (18, 195, 19) times faster than (CE, CSLR and MIX) on AMiner, (15, 8) times faster than (CE, MIX) on ACM, 10 times faster than MIX on DBLP, on average, respectively. (b) While GHOST on (AMiner, ACM, DBLP), CSLR on (ACM, DBLP) and CE on DBLP could not finish in 6 hours, NDCC could finish on (AMiner, ACM, DBLP) in (98, 543, 2106) seconds, respectively.

Exp-2: Insight of effectiveness. In the second set of experiments, we report an insight of effectiveness of NDCC. We compare NDCC to its non-collective clustering method, denoted as ND_NCC. Their difference lies in that we neglect the ambiguity of the other names for ND_NCC. Specifically, like traditional author name disambiguation methods, ND_NCC disambiguates author names one-by-one. Each name corresponds to a single author, except the name being disambiguated. We use the same author similarity measurement and clustering strategy for NDCC and ND_NCC. With top 100 author names, we compare these two methods in three datasets. The comparison results are shown in Figure 6.

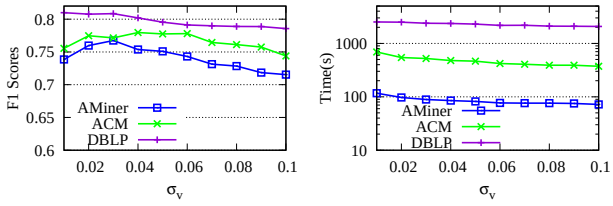
The figures show that disambiguation of author names affects each other. By treating these names separately and independently, the disambiguation results of ND_NCC are not satisfactory. On the other hand, by considering their relations and disambiguate all names collectively, NDCC improves accuracy dramatically. Specifically, NDCC improves the Macro-F1 over NDCC by (11.1%, 18.7% and 20.3%) on (AMiner, ACM, and DBLP), respectively.

Exp-3: Effectiveness of estimating author numbers. In the third set of experiments, we evaluate the effectiveness of NDCC of estimating author numbers. We note that the test dataset does not cover all authors in the datasets, and only part of authors are labelled. For example, there are over 120 authors with name "Lei Wang" in DBLP, but only 106 of them are labelled. Thus, We cannot evaluate the estimation method directly by comparing the estimated author numbers with labelled authors. Instead, given an author name, we compare the number of clusters of the labelled papers with the number of labelled authors to verify the effectiveness of the proposed estimating method. We list the results of the top 10 names in the test set in Table 5. We find that, in most cases, our method achieves reasonable estimating results: $\frac{\#detected\ \ Authors}{\#labelled\ \ Authors} \in (0.5, 2)$. Besides, the numbers of detected authors are usually larger than the true values. The reason is that, authors may change their affiliations and research interests at the same time. In this case, it is hard for name disambiguation methods to tell whether papers published in two periods are written by the same person with limited information.



(a) Accuracy (b) Efficiency

Figure 7: Accuracy and efficiency *w.r.t.* σ_t.



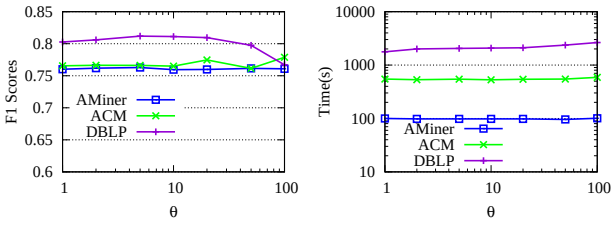
(a) Accuracy (b) Efficiency

Figure 8: Accuracy and efficiency *w.r.t.* σ_v.

Exp-4: Impacts of parameters. In the last set of experiments, we evaluate the effectiveness of including word-word and venue-venue similarity, as well as the impacts of parameters on accuracy and efficiency of NDCC. Parameter σ_t controls the number of non-zero elements in $\mathbf{W}^{TT}$, which is the number of similar word pairs. Similarly, σ_v determines the number of similar venue pairs , and θ is the parameter determining whether to use multi-coauthorship and multi-coauthor names as features.

Exp-4.1: Impacts of σ_t. To evaluate the impacts of word-word similarity, we vary σ_t from 0.5 to 1(*0.5, 0.6, 0.7, 0.8, 0.9, 1*), and fix other parameters to their default values. With σ_t increasing, fewer similar pairs of words are taken into consideration. $\sigma_t = 1$ means that we dismiss the word-word similarity. The accuracy and running time results of NDCC with respect to different σ_t in AMiner, ACM and DBLP are plotted in Figure 7(a) and Figure 7(b), respectively.

The results show that (a) including word-word similarity can increase the accuracy performance of NDCC. Specifically, it improves F1 scores up to (0.59%, 2.21%, 0.26%) on (AMiner, ACM, DBLP), respectively, (b) small σ_t, such as 0.5, which means words pairs with



(a) Accuracy (b) Efficiency

Figure 9: Accuracy and efficiency *w.r.t.* θ.

low similarities are also considered, may decrease the accuracy results, (c) NDCC achieves relatively high accuracy in a wide range of σ_t, (d) the running time decreases with σ_t increasing, because larger σ_t reduces the number of nonzero elements of $\mathbf{W}^{TT}$.

Exp-4.2: Impacts of σ_v. To evaluate the impacts of venue-venue similarity, we vary σ_v from 0.01 to 0.1 by step 0.01, and fix other parameters to their default values. Since most similarity scores of venue pairs are located in the range of (0, 0.1), we just range σ_v up to 0.1. The accuracy and running time results of NDCC with respect to different σ_v on AMiner, ACM and DBLP are plotted in Figure 8(a) and Figure 8(b), respectively.

The results tell us that (a) the usage of venue similarity can improve the accuracy performance significantly. Specifically, it improves the F1 scores up to (7.28%, 4.58%, 3.09%) on (AMiner, ACM, DBLP), respectively, (b) when σ_v is small, venue pairs with low similar scores are involved, which may decrease the accuracy results, (c) NDCC achieves relatively high F1 scores in a wide range of σ_v, (d) the running time goes down with σ_v increasing, for higher threshold means that less similar venue pairs are considered.

Exp-4.3: Impacts of θ. To evaluate the impacts of θ, we vary θ from 1 to 100 (1, 2, 5, 10, 20, 50, 100), and fix other parameters to their default values. The accuracy results and running time of NDCC with different θ on AMiner, ACM and DBLP are plotted in Figure 9.

The results tell us that (a) NDCC achieves relatively high F1 scores in a wide range of θ, (b) when θ is very large, for example, 100, multi-hop coauthorships and author- (multi-hop coauthor name) relationships are used in similarity calculation of high ambiguous names, which may impair the accuracy performance. At the same time, it leads to more running time.

Summary. From these tests, we find the followings.

(1) Our approach NDCC is effective for scholar name disambiguation. Macro-F1 scores of NDCC are consistently higher than the compared methods in all datasets.

(2) Our approach NDCC is also very efficient. With speeding up strategies, NDCC could finish on DBLP, which contains over 3 million papers, within an hour.

(3) By considering the ambiguity problem of coauthor names and dealing with entire names collectively, NDCC can improve the accuracy over non-collective method ND_NCC significantly,

(4) The author numbers detected by NDCC are reasonable.

(5) Strategies dealing with sparsity improve the accuracy performance. Specifically, incorporating word-word similarity and venue-venue similarity improves the F1 scores by (0.59%, 2.21%, 0.26%) and (7.28%, 4.58%, 3.09%) on (AMiner, ACM, DBLP), respectively. Besides, NDCC introduces thresholds to word-word similarity, venue-venue similarity and author similarity measurement for the sake of practicability and flexibility of real-life applications. We have experimentally shown that NDCC is robust to these parameters.

7 RELATED WORK

In general, existing methods for scholar name disambiguation can be divided into two classes: supervised [4, 14–17, 30, 33, 34] and unsupervised [7, 9, 10, 18, 23, 24, 27, 29, 32, 35, 36]. Supervised methods use human-labelled data to train a classifier, *e.g.,* SVM [33] and random forests [16, 17, 30], which is then used to assign publications to different author entities. However, labelling data is time-consuming and impractical when the bibliography data is large. Unsupervised methods use clustering, *e.g.,* agglomerative clustering [9, 18, 35], affinity propagation [10] and Markov clustering [36] or topic modeling [23, 24] to divide the set of author references into different subsets. Our work belongs to the second category.

There are three kinds of evidences that are commonly explored by disambiguation methods [11]: citation information [7, 33], web information [18] and implicit evidence [23, 24]. Citation information is extracted directly from citation records, including author names, title, venue, publication year. Our work only uses citation information, and is applicable to most digital libraries, as they typically contain citation information. It is also known that the usage of new evidences, *e.g.,* wiki [18], abstracts [27, 32], and homepages [32], usually improves the disambiguation performance. These methods are orthogonal to our method, and can be combined to further improve the performance of our method.

Most existing name disambiguation methods are designed to tackle single name ambiguity and dismiss their connections. Different from these work, our approach focuses on scholar name disambiguation in a collective way. There are also some collective entity resolution methods that can be used to solve multiple name disambiguation problem [6, 7, 13, 22]. However, they are not designed for scholar name disambiguation, as they mainly aim to deal with duplication problems in relational databases caused by different forms of the same names. Most of them need another clean knowledge base (KB) [13, 22], which is unavailable in most cases. [7] is a collective entity resolution method without a KB. However, it needs to store all pairs of similar author references as well as their similarity scores in a single queue, and its high space complexity keeps it away from large-scale data analytics.

8 CONCLUSIONS

Considering the connections of scholar names, we have proposed a collective approach NDCC to scholar name disambiguation. We have developed a novel metric to determine the author similarity by assembling the similarities of four features (coauthors, venues, titles and coauthor names). To deal with the sparsity problem, we have also introduced word-word and venue-venue similarities. As is shown in the experimental study, NDCC is both effective and efficient for scholar name disambiguation.

A couple of topics need further investigation. First, we are to combine new evidences to further improve the performance of our method. Second, we are to study NDCC under a dynamic scenario.

REFERENCES

[1] DBLP, 2017. *http://dblp.uni-trier.de/*.
[2] Google Scholar, 2017. *https://scholar.google.com/*.
[3] Microsoft academic search, 2017. *http://academic.research.microsoft.com/*.
[4] Mehmet Ali Abdulhayoglu and Bart Thijs. Use of researchgate and google cse for author name disambiguation. *Scientometrics*, 111(3):1965–1985, 2017.
[5] Ander Barrena, Aitor Soroa, and Eneko Agirre. Alleviating poor context with background knowledge for named entity disambiguation. In *ACL*, 2016.
[6] Indrajit Bhattacharya and Lise Getoor. A latent dirichlet model for unsupervised entity resolution. In *SDM*, 2006.
[7] Indrajit Bhattacharya and Lise Getoor. Collective entity resolution in relational data. *TKDD*, 1(1):5, 2007.
[8] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media", 2009.
[9] Lei Cen, Eduard C Dragut, Luo Si, and Mourad Ouzzani. Author disambiguation by hierarchical agglomerative clustering with adaptive stopping criterion. In *SIGIR*, 2013.
[10] Xiaoming Fan, Jianyong Wang, Xu Pu, Lizhu Zhou, and Bing Lv. On graph-based name disambiguation. *JDIQ*, 2(2):10, 2011.
[11] Anderson A Ferreira, Marcos André Gonçalves, and Alberto HF Laender. A brief survey of automatic methods for author name disambiguation. *SIGMOD Record*, 41(2):15–26, 2012.
[12] Andrea Glaser and Jonas Kuhn. Named entity disambiguation for little known referents: a topic-based approach. In *COLING*, 2016.
[13] Amir Globerson, Nevena Lazic, Soumen Chakrabarti, Amarnag Subramanya, Michael Ringaard, and Fernando Pereira. Collective entity resolution with multi-focal attention. In *ACL*, 2016.
[14] Donghong Han, Siqi Liu, Yachao Hu, Bin Wang, and Yongjiao Sun. Elm-based name disambiguation in bibliography. *WWWJ*, 18(2):253–263, 2015.
[15] Jian Huang, Seyda Ertekin, C Lee Giles, et al. Efficient name disambiguation for large-scale databases. In *PKDD*, 2006.
[16] Madian Khabsa, Pucktada Treeratpituk, and C Lee Giles. Large scale author name disambiguation in digital libraries. In *IEEE Big Data*, 2014.
[17] Madian Khabsa, Pucktada Treeratpituk, and C Lee Giles. Online person name disambiguation with constraints. In *JCDL*, 2015.
[18] Shaohua Li, Gao Cong, and Chunyan Miao. Author name disambiguation using a new categorical distribution similarity. *Machine learning and knowledge discovery in databases*, pages 569–584, 2012.
[19] Xiang Li, Yao Wu, Martin Ester, Ben Kao, Xin Wang, and Yudian Zheng. Semi-supervised clustering in attributed heterogeneous information networks. In *WWW*, 2017.
[20] Shuai Ma, Chen Gong, Renjun Hu, Dongsheng Luo, Chunming Hu, and Jinpeng Huai. Query independent scholarly article ranking. In *ICDE*, 2018.
[21] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv:1301.3781*, 2013.
[22] Wei Shen, Jiawei Han, and Jianyong Wang. A probabilistic model for linking named entities in web text with heterogeneous information networks. In *SIGMOD*, 2014.
[23] Liangcai Shu, Bo Long, and Weiyi Meng. A latent topic model for complete entity resolution. In *ICDE*, 2009.
[24] Yang Song, Jian Huang, Isaac G Councill, Jia Li, and C Lee Giles. Efficient topic-based unsupervised name disambiguation. In *JCDL*, 2007.
[25] Yizhou Sun and Jiawei Han. Mining heterogeneous information networks: principles and methodologies. *Synthesis Lectures on Data Mining and Knowledge Discovery*, 3(2):1–159, 2012.
[26] Michael J Swain and Dana H Ballard. Color indexing. *International journal of computer vision*, 7(1):11–32, 1991.
[27] Jie Tang, A. C. Fong, Bo Wang, and Jing Zhang. A unified probabilistic framework for name disambiguation in digital library. *TKDE*, 24(6):975–987, 2012.
[28] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: extraction and mining of academic social networks. In *KDD*, 2008.
[29] Jie Tang, Jing Zhang, Duo Zhang, and Juanzi Li. A unified framework for name disambiguation. In *WWW*, 2008.
[30] Pucktada Treeratpituk and C Lee Giles. Disambiguating authors in academic publications using random forests. In *JCDL*, 2009.
[31] Fang Wang, Wei Wu, Zhoujun Li, and Ming Zhou. Named entity disambiguation for questions in community question answering. *Knowledge-Based Systems*, 126:68–77, 2017.
[32] Xuezhi Wang, Jie Tang, Hong Cheng, and S Yu Philip. Adana: Active name disambiguation. In *ICDM*, 2011.
[33] Xiaoxin Yin, Jiawei Han, and S Yu Philip. Object distinction: Distinguishing objects with identical names. In *ICDE*, 2007.
[34] Baichuan Zhang, Murat Dundar, and Mohammad Al Hasan. Bayesian non-exhaustive classification a case study: Online name disambiguation using temporal record streams. In *CIKM*, 2016.
[35] Baichuan Zhang and Mohammad Al Hasan. Name entity disambiguation in anonymized graphs using link analysis: A network embedding based solution. *arXiv*, 2017.
[36] Baichuan Zhang, Tanay Kumar Saha, and Mohammad Al Hasan. Name disambiguation from link data in a collaboration graph. In *ASONAM*, 2014.
[37] Yizhou Zhang, Yun Xiong, Xiangnan Kong, Shanshan Li, Jinhong Mi, and Yangyong Zhu. Deep collective classification in heterogeneous information networks. In *WWW*, 2018.
[38] Yutao Zhang, Fanjin Zhang, Peiran Yao, and Jie Tang. Name disambiguation in aminer: Clustering, maintenance, and human in the loop. In *KDD*, 2018.