Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2018) 26: 554 – 569

doi:10.3906/elk-1702-293

Resolving namesakes using the author's social network

Ijaz HUSSAIN*, Sohail ASGHAR
Department of Computer Science, Faculty of Information & Technology, COMSATS Institute of Information Technology, Islamabad, Pakistan

Received: 21.02.2017 • **Accepted/Published Online:** 12.11.2017 • **Final Version:** 26.01.2018

Abstract: Author name ambiguity may occur when multiple authors share the same name or different name variations of a single author exist. This degrades search results and correct attributions in bibliographic databases. Existing solutions require either the actual number of ambiguous authors or extra information that is collected from the Web. However, in many scenarios, obtaining such auxiliary information is not possible or requires much extra effort. An effective and scalable method, ASONET, is proposed that uses graph community detection algorithms and graph operations to disambiguate namesakes. The citation dataset is preprocessed and ambiguous author blocks are formed. A graph structural clustering, gSkeletonClu, is applied to identify hubs, outliers, and clusters of nodes in a coauthor's graph. Namesakes are resolved by splitting these clusters across the hub if their feature vector similarity is less than a predefined threshold. ASONET utilizes only coauthors and titles that are surely available in all bibliographic databases. To validate the ASONET performance, experiments are performed on two real-world datasets of Arnetminer and DBLP. The results confirm that ASONET is scalable and outperforms baselines.

Key words: Author name disambiguation, namesakes, graph structural clustering, community detection

1. Introduction

Bibliographic databases conserve bibliographic citations and provide several services, such as receiving items associated with a particular author, multiple searches, browsing personalization, and building communities with certain educational fields [1]. One of the main challenges highlighted by Lee et al. in [1] is to have high-quality contents in bibliographic databases come from several sources of errors. Among these sources of errors, researchers paid great attention to ambiguous author names due to their nontrivial solution [1–3]. The systems that resolve author name ambiguity from publications are called author name disambiguation (AND) systems. Author name ambiguity occurs when multiple authors share the same name. In this case, it is difficult to be certain about the accuracy of retrieved results. This phenomenon of citation merger, the case of two or more persons having the same name (e.g., "Wei Wang"), is also known as mixed citations [3].

Generally, ambiguity in author names is resolved by means of different publication attributes such as coauthors, title, abstract, venue, and publication year [1,3,4]. However, each bibliographic database does not provide all these attributes; they provide only limited information and manual annotation is not possible on such a large scale. Furthermore, in recent years, large amounts of publication data are being created and accepted by bibliographic databases that make the name ambiguity problem even more severe than it was in the past [1].

When we search for an author named "Wei Wang" in DBLP (a renowned bibliographic database), we get

*Correspondence: ijazhussain7979@hotmail.com

86 authors having the same name. The same situation occurs in almost all major bibliographic databases [3]. This example is representative of the magnitude of the mixed citation problem that motivated us. Moreover, DBLP and other bibliographic databases are frequently not informative enough in their metadata and lack pieces of important information such as an author's affiliation, e-mail address, medical subject heading (present only in PubMed), and publication references [4].

Several AND methods have been proposed in the literature [2–25]. These methods have improved the situation somewhat, but there is still a need for improvement in current solutions. Supervised methods [2,5,6,20–21,25] require much clean and representative data for the training of the model and give poor results when a model is trained on noisy data or nonrepresentative data [1,3]. Most unsupervised AND methods assume that a number of ambiguous authors/clusters "K" are known in advance [4,7,10,12]. Some techniques are not scalable when the number of ambiguous authors increases [2,3,16]. Some require extra information from the Web or user feedback for disambiguation [7,16].

In this paper, ASONET ("resolving namesakes using the Author's SOcial NETwork") is proposed to resolve namesakes, as shown in Figure 1. ASONET is a graph-based method in which a citation dataset is preprocessed, blocks of the ambiguous author are created, and the coauthor's graph is constructed using the preprocessed citation data. Then gSkeletonClu is used to identify hubs, outliers, and clusters of vertices (communities) from the coauthor's graph [26]. ASONET resolves namesakes by splitting these communities across the hub nodes if their title feature vector similarity is less than a predefined threshold and using graph operations. However, distinct from existing techniques, we exploit the community detection algorithm in combination with graph operations to disambiguate namesakes. To the best of our knowledge, the ASONET algorithm is the first that uses a gSkeletonClu community detection algorithm to disambiguate namesakes.

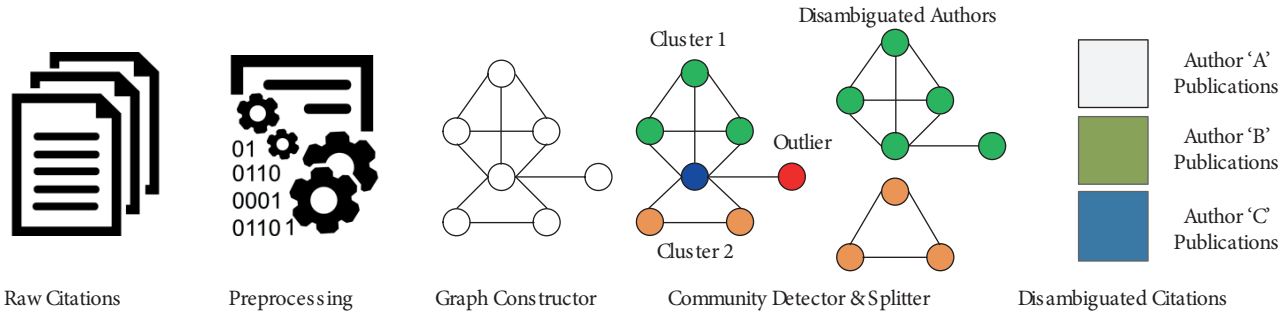


Figure 1. ASONET architecture.

ASONET's performance is evaluated by comparing it with three unsupervised graph-based AND methods using Arnetminer and DBLP datasets [3,4,7].

2. Related work

According to Ferreira et al. [1], AND methods can be categorized as author assignment methods [2,5,6,9,20,21] and author grouping methods [3,4,7,8,11,12,18,24].

Onodera et al. proposed a methodology for targeting the author from false homonym authors [13]. They used publication features such as affiliation, title, and coauthors for discriminating the oeuvre of an author. Zhu et al. proposed a hybrid name disambiguation framework that not only used the traditional information (coauthors) but also Web page genre information [14]. This framework consists of two main steps: Web page

genre identification and a reclustering model. In Web page genre identification, the returned pages are classified as either home page of an author or not. Those records found at the author's homepage belong to him and are disambiguated. The remaining ambiguous records are disambiguated using coauthors' information. Some records that are not disambiguated from coauthors or from the Web information are sent to the reclustering model. They then build a graph "G" using all citation records, in which each vertex represents a citation record and each edge denotes the same coauthor's relationship. If there are many links present between two vertices, then they are considered to be related/close to each other. They transfer this graph into a similarity matrix by using a multidimensional scaling algorithm. This algorithm detects similarities among objects. They constructed many two-dimensional matrices for coauthorship and the topic relationship and calculated the distance between two vertices with the help of the Euclidean metric. If the distance between citations is less than a specific threshold, then they are considered to be by the same author. This method is rather slow as it crawls results from the author's home page, identifies their genre, and requires more computations. One other issue with this method is how to identify that this is an author's home page or not and it is a nontrivial task. Dempster–Shafer theory (DST) was fused with Shannon entropy (SE) for author name disambiguation in [12]. In the first step, high-level features such as Affiliation, Venue, ContentSim, Coauthor, Citation, and Webcorrelation and their correlation similarities are calculated. In the next step, these features are fused using DST and SE. On the basis of this information, belief and plausibility of each author are calculated. Then they obtain a matrix of pairwise correlations of papers. Each entry in this matrix is linked to a belief function and a plausibility function. In the end, they apply a DST-based hierarchical agglomerative clustering (DSHAC) algorithm for author disambiguation. In the process of clustering they use three different convergence conditions for the clustering algorithm: a preset number of clusters, number of available pieces of evidence, and distance between clusters. This method has lower accuracy than other unsupervised methods. The dataset used is relatively small and specialized so the method may not work well on other datasets and thus needs to be tailored to different situations. In the clustering process, the main hindrance of this method is where to stop the clustering process. The authors use three convergence conditions: the preset number of clusters, number of pieces of available evidence, and distance between clusters. However, in real-world settings these three conditions are not feasible. Furthermore, this method suffers from overfitting and optimistic accuracy estimates.

Carvalho et al. proposed INDi as a good solution for the existing cleaned/disambiguated DLs in [15]. INDi utilizes similarity among bibliographic records and groups the new records for authors with similar citation records in the DL or new authors when the similarity evidence is not strong enough. Some particular heuristics are used for checking whether references of new citation records belong to preexisting authors of the DL or if they belong to new ones (i.e. authors without citation records in the DL), avoiding running the disambiguation process in the entire DL. They run simulations on the BDBComp collection and synthetic DSs are used to assess the effectiveness of their method. They compare it with a state-of-the-art unsupervised method. Their experiments show gains of up to 19% when compared to a state-of-the-art method without the cost of having to disambiguate the whole DL at each new load (as done by supervised methods) or the need for any training (as done by supervised methods). This is an incremental method that disambiguates only new entries. Liu et al. in [9] used a three-step clustering framework for name disambiguation. In the first step, they obtained clusters based on common coauthors. Titles were then used to make bigger clusters from the first step's fragmented clusters. Finally, they fused clusters based on venues. An ethnicity-sensitive method that mainly comprises three parts was presented in [15]. In the first part, phonetic-based blocking for similar author signatures was done. A supervised machine learning-based linkage function was used that exploited the ethnicity-sensitive information. Finally, hierarchical agglomerative clustering was done based on a distance between two pairs of

publications' linkage functions. In [16], the authors proposed an algorithm that used both Web correlation and author correlation-based approaches to measure similarities between publications.

ADANA (Active nAme Disambiguation for the Name Ambiguity problem) was proposed in [7]. In this method, the authors modeled a pairwise factor graph that can be used to integrate several features and user feedback into a unified model. They defined three types of feature functions, i.e. document pair, correlation, and constraint-based features. In document pair feature functions, they found known relationships from publications. In the correlation feature function, they found some hidden features with the help of known functions, and in the constraint-based feature functions the user was involved in finding the unknown features. Lastly, they exploited active selection of the user corrections in an interactive mode to improve the disambiguation performance after some preliminary clustering results. Some additional information was used for disambiguation, such as affiliation and references, which is not present in every bibliographic database.

Li et al. in [18] proposed a categorical set similarity measure to disambiguate namesakes. They used the author's preference for specific venues with the help of a categorical distribution and calculated the probability to estimate the true authors when two sets of authors belong to the same distribution. They used this probability as the similarity measure for finding that either of two sets belong to the same author. Vishnyakova presented a journal descriptor indexing (JDI) tool to resolve ambiguous authors [19]. It utilized titles, abstracts, and MeSH terms as an input to the JDI and it returned the journal descriptors and semantic types as its output. This information was used as a feature for the supervised model. This model can only be used on MEDLINE, not on other bibliographic databases. Tran et al. in [20] suggested a deep neural network-based approach to automatically learn the weights of the features and disambiguate the authors. Determining optimal number of hidden layers, data representations for the first layer, and number of units is a complex task and requires skill and experience. Incremental disambiguation methods were proposed by Qian et al. and Santana et al. that can be used on an already disambiguated database [22,23]. Torvik et al. presented a name disambiguation method that used much auxiliary information such as shared title words, journal name, coauthors, medical subject headings, language, e-mail, affiliations, and author name features, which are not available in many bibliographic databases [24].

In contrast to previous AND techniques, ASONET requires no costly training data, it uses only coauthors and titles information, and it does not need to set a priori the number of ambiguous author clusters. A comparison between related methods with ASONET is done in Table 1.

3. Proposed methodology of ASONET

We implemented the gSekeletonClu algorithm [26] in Python using the Networx package to detect hubs, outliers, and clusters (communities) in the coauthor's graph, as given in the ASONET algorithm. More details of gSkeletonClu can be found in [26].

3.1. Preprocessing of dataset

The raw citations are preprocessed and split into authors, titles, and venues terms. In blocking, ambiguous authors are grouped in candidate classes using a predefined similarity threshold. The fragment comparison method is used for this purpose because it proved to be very effective in some earlier studies [4,6]. The blocking stage is very crucial as it affects the computations in the later stages of name disambiguation. The blocking stage returns 'b' number of blocks if given 'a' number of authors.

The computation complexity is O(a^2) for 'a' authors if we do not use blocking, whereas blocking

Table 1. Comparison between different related methods.

Ref.	Authors	Scalable	Methodology	Dataset	Features
7 ADANA	Known	No	User feedback-based AND	Publication, Web page, news stories	Citation, Coauthor, Covenue, Coaffiliation, Coaffoccur, Titlesim, Cohomepage
12 DSHAC	Uses known/unknown	No	Dempster–Shafer theory, Shannon entropy	DBLP	Coauthor, Affiliation, Venue, Content similarity, Citations, Web correlations
14 MMC	Unknown	Yes	Logistic regression	Subset of WoS	Coauthor, Affiliation, Citation relations, Titles, Cocitations, Year, Affiliation country
15 INDi	Known	Yes	Multidimensional scaling	DBLP	Uses Web genre identification-based graph clustering
16 Modified MNDF	Unknown	No	Heuristic	BDBComp, synthetic	Author, Coauthors, Title, and Venue
17 ASE	Known	No	Web, authorship correlations	DBLP	Uses additional information from Web
20 DNN	Known	No	Deep neural network	Vietnamese	Author name, Affiliation, Coauthor
21 JDS	Known	No	Journal descriptor, semantics	MEDLINE	Titles, Abstracts, and Medical Subject Heading
25 NB/SVM	Unknown	No	Similarity-based clustering	MEDLINE	Shared title words, Journal, Coauthors, MeSH, Language, e-mail, Affiliations, Authors
Proposed	Unknown	Yes	Graph structural clustering, feature vector	Arnetminer, DBLP	Coauthors, Title

considerably reduces computational complexity to $O(B|S|)$, where 'B' is the number of blocks and 'S' is the average size of blocks. The similarity threshold controls the precision and recall of blocking. If we set the similarity threshold too high, then recall is very low, but it creates very pure blocks. On the other hand, if we set it very low, then high false positives are produced. For the selection of the threshold, we sampled names and set this threshold to 0.7, which produces the optimal blocking of ambiguous names.

3.2. Graph construction

After blocking, each citation is converted into a set of 'n' authors in a graph $G = (V,E)$. Authors are extracted from the citation dataset and each distinct author a_i is mapped to a vertex $v \in V$. Then, for each citation, if there are 'a' number of authors then there are . number of edges created in the graph 'G'. In this way, the coauthor's graph of all the citations in the dataset is formed. A small working example of the citation dataset,

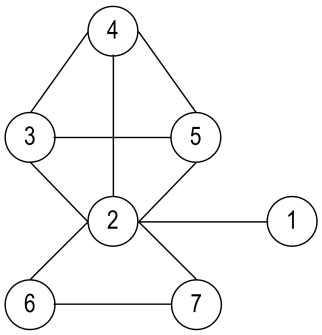
vertices, edges, and constructed graph is shown in Figure 2. There are seven authors (nodes) and ten edges that are created from six citations. "Wei Wang" is a node that is common to five citations and "M. Rehan" is present only in citation 4.

Citation No.	Bibliographic Citations Data
1	Wei Wang, S. Huang: Disambiguating Author Names using spectral clustering: KDD, 2015: page 29-38
2	Wei Wang, C. Chen : Graph Pattern Mining--A review: OIR, 2016: page 124-140
3	Wei Wang, Ajay Gupta, S. Huang : Non-linear Control for Hot Air Blower System: JUCS, 2012: page 78-87
4	Ajay Gupta, S. Huang, M.Rehan: Community Detection based Graph Author Name Disambiguation Framework: KIR, 2014: page 329-339
5	Wei Wang, M. Miller, Phillips S. Yu: POSHI:HeArt failure Prediction Tool: INMIC, 2016: page 295-308
6	C. Chen, Wei Wang: Graph utilization in Author Name Ambiguity Problem: JUCS, 2014: page 319-329

(a) An example of small citations data set

Vertex Id	Vertex Name	Publications
1	C. Chen	P2, P6
2	Wei Wang	P1-P3, P5-P6
3	S. Huang	P1, P3-P4
4	M. Rehan	P4
5	Ajay Gupta	P3-P4
6	M. Miller	P5
7	Phillips S. Yu	P5

(b) Information of nodes in the graph



(c) Graph constructed from the excerpt of bibliographic citations

Figure 2. Coauthor's graph of Wei Wang sample citations.

3.3. Detecting author's social network

The ASONET algorithm detects the namesakes using the gSkeletonClu community detection algorithm [26] and then resolves these namesakes by applying graph operations. The use of gSkeletonClu is done here for four reasons: 1) It detects outliers that are not possible with existing graph-based methods. 2) It makes the algorithm scalable as its computational complexity is much lower than that of other competing community detection methods, as given in Section 1. 3) In contrast to other popular community detection algorithms, it is useful to identify overlapping communities in the coauthor's graph. 4) There is no need to tune threshold parameters; it automatically computes these parameters. In graph-based structural clustering algorithms, two parameters are usually required to be chosen, ε and μ, where $\varepsilon \in \Re$ and $\mu \in \mathrm{N}$. A threshold ε is applied to the computed structural similarity when assigning cluster membership. Core vertices are a special class of vertices that have a minimum of μ neighbors with a structural similarity that is above a specified threshold. Typical values for μ and ε are 2 and 0.7. From core vertices, we grow the similar group of clusters. In this way, the parameters μ and ε determine the clustering of networks. In gSkeletonClu, the optimal values of both these parameters are calculated automatically. For further details, interested readers can refer to the work of Huang et al. in [26].

ASONET assumes that different namesakes have different communities in an academic social circle and different namesakes seldom work in the same institution or community [3,4]. Thus, they belong to different author communities. A community is generated from each citation in the coauthor's graph and thus each community denotes the coauthorship for each citation that is the smallest social circle in the academic domain. For example, in 'citation 4' there are three authors, 3, 4, and 5. Due to this citation the upper three nodes and their relationships are constructed as shown in Figure 2c and when we add more citations to this graph it becomes wider and bigger. Finally, when we construct the graph of all sample citations in Figure 2a, it looks as shown in Figure 2. With the help of the coauthor graph, it is possible to infer the social circle of an author from one's coauthors by finding shared communities.

3.4. Feature vector construction

After removing stop words and stemming those words, we construct feature vectors of titles. ASONET assumes that results for an author linked with multiple social circles contain mixed information for several authors if the similarity between feature vectors of two clusters is less than 0.2. This threshold is chosen empirically by discretizing the search space and then varying its value in small increments of 0.01 in the range of 0.1 to 0.3 and finding the optimal value of it. It is worthwhile to mention here that this step is needed only once for all datasets. ASONET splits that node and its information into several different nodes that are present in nonoverlapping communities.

Feature vector example: Suppose we have two titles of papers T1 and T2.

T1: Disambiguating homonyms using graph-based community detection algorithms.

T2: Graph semantic similarity for author name disambiguation.

After removing stop words and stemming, we obtain the following titles, T1' and T2'.

T1': disambiguate homonym graph community detect algorithm

T2': graph semantic similar author name disambiguate

Let us have a look at the total vocabulary now: disambiguate, homonym, graph, community, detect, algorithm, semantic, similar, author, name.

Vocabulary size is 10 words, so the vector will have 10 dimensions.

Let us fit our title documents into vectors:

D1 : [1, 1, 1, 1, 1, 1, 0, 0, 0, 0]

D2 : [1, 0, 1, 0, 0, 0, 1, 1, 1, 1]

There are two words in common between the two title vectors of the papers out of a total 10 words, so their title similarity is 0.2.

3.5. ASONET algorithm

In the coauthor's graph, an edge represents a coauthor's relationship between the two nodes (authors). The ASONET algorithm (Algorithm 1) is based on gSkeletonClu, a structural clustering algorithm for networks, to detect different nonoverlapping communities in the coauthor's graph [26]. In this algorithm, when we use gSkeletonClu on a coauthor's graph, it outputs communities (clusters of nodes), hub nodes, and outliers.

The community consists of a set of nodes from the coauthor's graph. The node that is involved in many social circles in the coauthor's graph is called a hub node. The hub node is the potential homonym that needs disambiguation. Outliers are nodes that only contribute one or a limited number of publications with the hub node and have no other coauthors. In the example coauthor's graph, detected communities, hub, and outlier

Algorithm 1 ASONET algorithm

Data: *Coauthors Graph (CG)*
Result: *Disambiguated Graph (DG)*
begin
Authors ← *PreprocessData (CitationsData)*
CG ← *BuildGraph (Authors)*
[ClustersList, Hub, Outliers] ← *applygSkeletonClu (CG)*
HubCitations ← *get.Citations (Hub)*
FV_{Hub} ← *get.FV (Hub)*
k, l ← *0*
for *cluster* ∈ *ClustersList* **do**
    clusterCitations ← *0*
    FV_C ← *get.FV (Cluster)*
    FV Similarity ← *sim* (FV_C, FV_{Hub})
    if *FV Similarity < 0.2* **then**
        for *vertex* ∈ *cluster* **do**
        clusterCitations ← *clusterCitations U getCitations (vertex)*
        end
        NS_k*Citations* ← *HubCitations I clusterCitations*
        NS_k*Name* ← *getName (Hub)*
        NS_k*Id* ← *getLength (CG)*
        CG.InsertNewNode (NS_k*Id*, NS_k*Name*, NS_k*Citations*)
        UpdateGraph (CG, outlier)
        HubCitations ← *HubCitations* \ *clusterCitations*
        k ← *k + 1*
    else
        cluster ← *cluster + 1*
    end
end
for outlier ∈ outliers **do**
    outlierCitations ← *0*
    FV_O ← *get.FV (outlier)*
    FV Similarity ← *sim* (FV_O, FV_{Hub})
    if *FV Similarity < 0.2* **then**
    O_l*Citations* ← *HubCitations I outlierCitations*
    O_l*Name* ← *getName (Hub)*
    O_l*Id* ← *getLength (CG)*
    CG.InsertNewNode (O_l*Id*, O_l*Name*, O_l*Citations*)
    UpdateGraph (CG, outlier)
    HubCitations ← *HubCitations* \
    outlierCitations l ← *l + 1*
    else
    outlier ← *outlier + 1*
    end
end
end

can be seen in Figure 3. There are two clusters of nodes (communities), one hub, and one outlier. The first and second clusters consist of nodes <3, 4, 5> and <6, 7>, respectively. The hub is node <2> and the outlier is node <1>. A hub node that has multiple nonoverlapping communities contains mixed information for several namesakes. ASONET splits the information about each author on the node containing the namesakes along the nonoverlapping communities if its feature vector similarity is less than 0.2. This part starts from line 7 of the ASONET algorithm, where the hub node publications list is retrieved. Similarly, publications of all cluster nodes (community) are retrieved and saved in a list (lines 13–15). Now, for each homonym, the intersection of hub publications and community publications is found. A new node is created in the coauthorship graph for each community detected that has the same name as that of the hub node and has an identity one more than in the current nodes of the coauthorship graph. This newly created node has the publication list that is found at line 16. Likewise, for all communities detected in the coauthor's graph, new nodes are created. The coauthor's graph and hub node publications are updated on each new node insertion into the graph. The same procedure is repeated for all outliers, as is done in the case of all communities (lines 27–41). This whole process is pictorially shown in Figure 4, where hub node <2> is split into two new nodes, <2A, 2B>, as there are two clusters of nodes (communities) that have feature vector similarity less than 0.2. In this way, the namesake's problem is solved using the community detection algorithm.

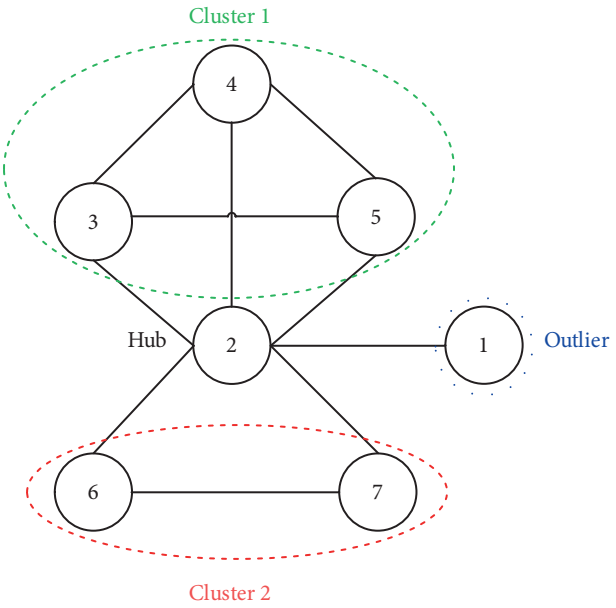


Figure 3. Example of detected communities, hub, and outlier in Wei Wang's sample. graph after clustering.

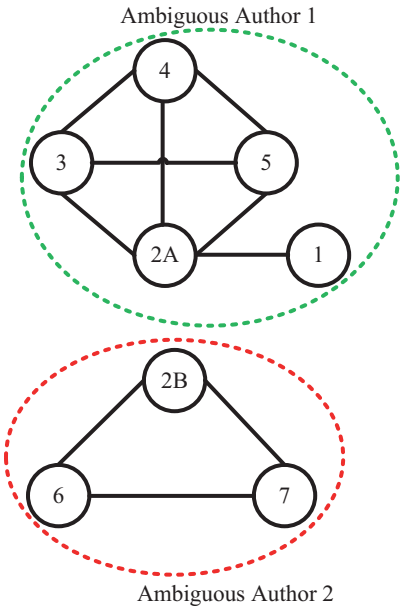


Figure 4. An example of namesake resolution.

3.6. ASONET complexity analysis

The initial stage of ASONET is to create ambiguous author blocks that can be created in O (B|S|), where 'B' is the number of blocks and 'S' is the average size of the blocks. According to Huang et al., graph "G (V, E)" can be constructed in O(|V|+|E|) and graph structural clustering is done in O(|E|) [26]. Text feature vectors and other graph comparison are done only in small blocks found in the initial stage of ASONET. These stages take negligible time as compared to other stages. In the above three steps, the blocking step dominates the computational complexity. Hence, ASONET has an overall time complexity of O(B|S|). GHOST, presented

by Fan et al. in [4], used valid path-based similarity between two nodes in the order of $O((|R^2|) \times (|V| + |E|))$, where V is the vertices, E is the edges, and R is the set of names to resolve. ADANA, proposed by Wang et al. in [7], used a pairwise factor graph that was also intractable due to the calculation of the normalization factor, which made their method complexity exponential to the number of nodes in the graph. Similarly, GFAD, proposed by Shin et al. in [3], used Johnson's simple cycle enumeration algorithm [27] that has a complexity of O((V + E)(C + 1)) with V for vertices, E for edges, and C for elementary cycles in the graph. A system is said to be scalable if it is applied to a larger dataset and gives statistically the same results. For instance, a system designed for Arnetminer (small dataset) is applied to DBLP (a large dataset), and if it gives approximately the same results, then it is said to be scalable. These complexities of all the techniques are summarized in the Table 2.

Table 2. A comparison between baseline techniques.

Reference	Technique	Complexity
[3]	GFAD	O((V+E)(C+1))
[4]	ADANA	Intractable
[7]	GHOST	$O((\|R^2\|) \times (\|V\| + \|E\|))$
Proposed	ASONET	O(B|S|)

4. Performance evaluation of ASONET

In this section, the performance of ASONET is compared with baseline methods using Arnetminer and DBLP datasets in the form of clustering evaluation metrics.

4.1. Datasets

To evaluate the ASONET performance, the first dataset is Arnetminer, whose details are given in Table 3. This dataset was originally created by Wang et al. [7] and has been used in many previous studies with slight variations [3,4,7]. We also choose a small subset of this dataset that is composed of 950 citation records associated with 344 distinct authors belonging to 39 ambiguous groups, with approximately 2.76 citation records per author, as shown in Table 3.

Currently, the DBLP lists more than 1500 journals and 5000 conferences and workshop series in computer science. The second collection is taken from DBLP that is composed of 8442 citation records associated with 480 distinct authors belonging to 14 ambiguous groups, which means an average of approximately 18 citation records per author, as shown in Table 4. The original version of this collection was created by Han et al. in 2004, and, with slight variations, it has been used in performance evaluation of various author disambiguation methods [2,3,5,6,17]. Han et al. created this collection by collecting bibliographic citation records from DBLP and authors' homepages [25]. After that, they transformed author names to abbreviated forms consisting of the first name's initial and the last name and clustered the bibliographic citation records into ambiguous groups, each of which corresponds to the authors with the same abbreviated name. This dataset is now considered as a de facto benchmark of AND.

4.2. Evaluation metrics

We used average cluster purity (ACP), average author purity (AAP), K-metric, pairwise precision (PP), pairwise recall (PR), pairwise-F1 (PF1), cluster precision (CP), cluster recall (CR), and cluster-F1 (CF1) to measure the effectiveness of ASONET [3,6].

Table 3. Details of Arnetminer dataset.

Name	Ath.	Rec.	Name	Ath.	Rec	Name	Aut.	Rec
Ajay Gupta	8	31	B. Wilkinson	1	17	Bob Johnson	3	4
Bin Zhu	15	45	Cheng Chang	5	23	M Lang	4	16
Charles Smith	4	7	Paul Wang	6	15	Fei Su	4	37
Michael Siege	6	50	David Cooper	7	18	Robert Allen	8	21
Gang Luo	8	42	S. Huang	15	16	Hui Fang	8	42
J. Guo	9	12	Hui Yu	20	30	Xiaoyan Li	6	31
Yan Tang	11	31	Lei Fang	7	17	Ping Zhou	17	33
X Wang	14	33	Yue Zhao	9	36	Lei Jin	6	16
Paul Brown	8	20	Peter Phillips	3	13	T Taylor	3	4
David Nelson	10	15	F. Wang	14	15	Z. Wang	35	43
Hong Xie	6	10	J. Yin	7	18	J Wang	5	35
John Hale	3	36	Kuo Zhang	4	16	M. Rahman	7	15
Michael Smith	16	27	R Taylor	11	27	Young Park	8	17
Ath.: Number of distinct authors, Rec.: citation records associated with that author.								

Table 4. Details of DBLP dataset.

S. no.	Name	No. of authors	Citation records
1	A. Gupta	26	577
2	A. Kumar	14	244
3	C. Chen	61	800
4	D. Johnson	15	368
5	J. Lee	100	1417
6	J. Martin	16	112
7	J. Robinson	12	171
8	J. Smith	31	927
9	K. Tanaka	10	280
10	M. Brown	13	153
11	M. Jones	13	259
12	M. Miller	12	412
13	S. Lee	86	1458
14	Y. Chen	71	1264
Total		480	8442

ACP, AAP, and K-metric are expressed in Eq. (1), where n_i is the number of elements in cluster i, n_j is the number of elements in cluster j, n_{ij} is the number of elements belonging to both clusters i and j, N denotes the size of the citations in the collection, J is the number of gold-standard reference clusters manually generated, and I is the number of clusters automatically generated by ASONET.

$$ACP = \frac{1}{N}\sum_{i=1}^{I}\sum_{j=1}^{J}\frac{n_{ij}^2}{n_i}, AAP = \frac{1}{N}\sum_{i=1}^{I}\sum_{j=1}^{J}\frac{n_{ij}^2}{n_j}, K = \sqrt{ACP * AAP} \quad (1)$$

The PP, PR, and PF1 measures are expressed in Eq. (2), where $C(n;i)$ denotes the number of combinations of i elements from n elements. Other parameters including i, j, n_i, and n_{ij} are defined as before in Eq. (1).

$$PP = \frac{\sum_{r=1}^{R}\sum_{s=1}^{S} C(n_{rs},2)}{\sum_{r=1}^{R} C(n_r,2)}, PR = \frac{\sum_{r=1}^{R}\sum_{s=1}^{S} C(n_{rs},2)}{\sum_{s=1}^{S} C(n_s,2)}, PF1 = \frac{2(PP*PR)}{PP+PR} \tag{2}$$

CF1 is the harmonic mean of CP and CR, where CP is the fraction of the generated clusters that are equal to the reference clusters and CR is the fraction of correctly retrieved clusters from the reference clusters. The CP, CR, and CF1 measures are given in Eq. (3).

$$CP = \frac{I \cap J}{I}, CP = \frac{I \cap J}{J}, CF1 = \frac{2(CP*CR)}{CP+CR} \tag{3}$$

4.3. Baseline methods

Three unsupervised author name disambiguation methods were used as baseline methods to compare with ASONET [3,4,7]; details of all these methods are given in the related works.

4.4. Experiments and comparisons

Table 5 shows the average ACP, AAP, K-metric, PP, PR, PF1, CP, CR, and CF1 of ASONET. In general ACP and PP of ASONET in the majority of cases are approximately equal to 100%. In Figure 5, ASONET performance is compared with three baseline methods, GHOST, ADANA, and GFAD, using Arnetminer. It shows overall better results than these baseline methods and achieves values 12%, 8%, and 5% higher for K-metric; 32.1%, 27.6%, and 4.2% higher for CF1; and 3.7%, 6.3%, and 1.2% higher for PF-1 as compared to GHOST, ADANA, and GFAD, respectively. Similarly, in Figure 6, ASONET performance is compared with three baseline methods, GHOST, ADANA, and GFAD, using DBLP. It also shows overall better results than baseline methods and achieves values 14%, 10%, and 7% higher for K-metric; 27.8%, 26.4%, and 10.3% higher for CF1; and 4.9%, 8.6%, and 6.2% higher for PF-1 as compared to GHOST, ADANA, and GFAD, respectively.

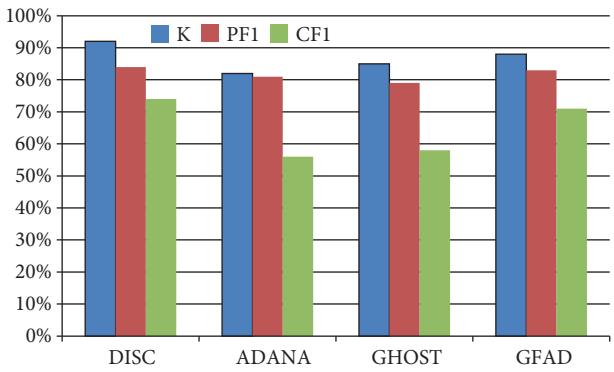


Figure 5. Average K-metric, PF1, and CF1 of the ASONET, ADANA, GHOST, and GFAD on Arnetminer.

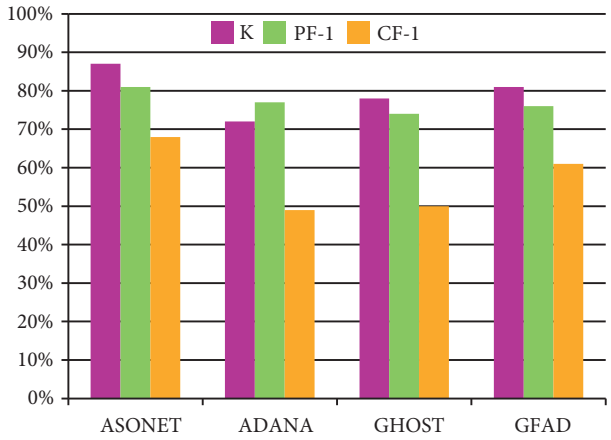


Figure 6. Average K-metric, PF1, and CF1 of the ASONET, ADANA, GHOST, and GFAD on DBLP.

Table 5. Performance evaluation of ASONET for ambiguous author group of Arnetminer.

Name	ACP	AAP	K	PP	PR	PF1	CP	CR	CF1
Ajay Gupta	1.00	0.64	0.80	1.00	0.59	0.74	0.31	0.62	0.42
Barry Wilkinson	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Bin Zhu	1.00	0.85	0.92	1.00	0.72	0.83	0.68	0.87	0.76
Bob Johnson	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Charles Smith	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Cheng Chang	1.00	0.79	0.89	1.00	0.70	0.82	0.38	0.60	0.46
David Cooper	1.00	0.92	0.96	1.00	0.90	0.95	0.75	0.86	0.80
David Nelson	1.00	0.93	0.97	1.00	0.88	0.93	0.82	0.90	0.86
F. Wang	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Fei Su	0.95	0.71	0.83	0.98	0.77	0.86	0.25	0.50	0.33
Gang Luo	1.00	0.95	0.98	1.00	0.93	0.96	0.78	0.88	0.82
Hong Xie	1.00	0.87	0.93	1.00	0.50	0.67	0.75	0.86	0.80
Hui Fang	1.00	0.70	0.83	1.00	0.55	0.71	0.42	0.62	0.50
Hui Yu	1.00	0.97	0.98	1.00	0.95	0.97	0.91	0.95	0.93
J. Yin	0.95	0.52	0.70	1.00	0.20	0.33	0.91	0.95	0.93
J. Guo	1.00	0.92	0.96	1.00	0.67	0.80	0.82	0.90	0.86
Jianping Wang	1.00	0.63	0.80	1.00	0.56	0.72	0.50	0.80	0.62
John Hale	1.00	0.52	0.72	1.00	0.49	0.66	0.11	0.33	0.17
Kuo Zhang	1.00	0.91	0.95	1.00	0.91	0.95	0.60	0.75	0.67
Lei Fang	1.00	0.90	0.95	1.00	0.77	0.87	0.75	0.86	0.80
Lei Jin	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
M. Rahman	1.00	0.79	0.89	1.00	0.43	0.60	0.67	0.86	0.75
Michael Lang	1.00	0.73	0.86	1.00	0.56	0.72	0.50	0.75	0.60
Michael Siege	1.00	0.83	0.91	1.00	0.83	0.91	0.36	0.67	0.47
Michael Smith	1.00	0.76	0.87	1.00	0.35	0.52	0.62	0.81	0.70
Paul Brown	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Paul Wang	1.00	0.91	0.95	1.00	0.90	0.95	0.71	0.83	0.77
Peter Phillips	1.00	0.77	0.88	1.00	0.74	0.85	0.20	0.33	0.25
Ping Zhou	1.00	0.97	0.98	1.00	0.98	0.99	0.89	0.94	0.91
Richard Taylor	1.00	0.73	0.86	1.00	0.49	0.65	0.64	0.82	0.72
Robert Allen	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Thomas Taylor	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Xiaoming Wang	0.97	0.86	0.92	1.00	0.94	0.97	0.62	0.71	0.67
Xiaoyan Li	1.00	0.94	0.97	1.00	0.92	0.96	0.71	0.83	0.77
Yan Tang	1.00	0.85	0.92	1.00	0.80	0.89	0.64	0.82	0.72
Young Park	1.00	0.85	0.92	1.00	0.77	0.87	0.70	0.88	0.78
Yue Zhou	1.00	0.66	0.81	1.00	0.52	0.68	0.46	0.67	0.55
Z. Wang	0.93	0.91	0.92	0.57	0.40	0.50	0.78	0.80	0.79
Average	0.99	0.85	0.92	0.99	0.76	0.84	0.69	0.82	0.74

ASONET performance is better as compared to baseline methods with both the clean and noisy datasets because GHOST, ADANA, and GFAD are unable to detect outlier authors that only share one coauthor with the hub node whereas ASONET can identify these authors.

ASONET and GFAD are automatically able to find hidden numbers of ambiguous authors "K". Complete details of the ground truth clusters, ASONET-generated clusters, and GFAD-generated clusters of the Arnetminer dataset are shown in Figure 7. Using Arnetminer, 77.5% of clusters are within the range (Ground Truth Clusters ± 3), whereas only 66.7% of the clusters of GFAD are within this range. ASONET performance is better than all baseline methods, even though GHOST and ADANA used the predefined number of clusters "K".

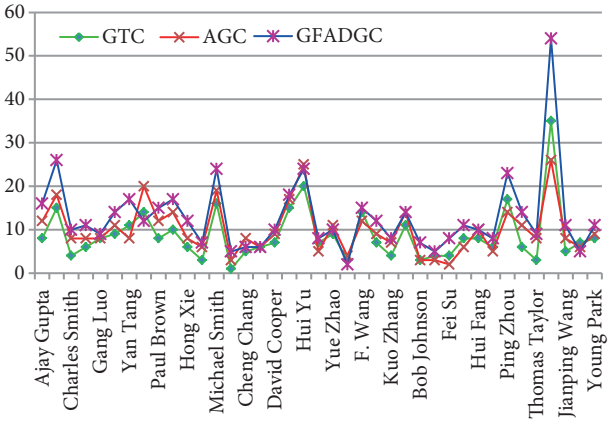


Figure 7. Comparison among ground truth clusters (GTC), ASONET-generated clusters (AGC), and GFAD-generated clusters (GFADGC).

To show the scalability of ASONET, comparison of running time of all methods on Arnetminer and DBLP is shown in Table 6. All experiments were performed on a personal computer with Intel Core i5-5200U CPU @ 2.20 GHz and 8 gigabytes of memory. GFAD is the slowest and ASONET is the fastest among all the baselines. GFAD most time-consuming stage is the cycle finding in the coauthor's graph.

Table 6. Summary of all methods running time on both datasets (seconds).

Dataset	ASONET	ADANA	GHOST	GFAD
Arnetminer	22	86	27	257
DBLP	41	158	109	875

ASONET is unable to identify when two different authors with the same name have different coauthors with the same name, which are called very ambiguous authors in AND literature [1]. However, analysis of both datasets shows that only 0.31% of authors are very ambiguous authors, particularly some Asians names (Chinese and Korean).

5. Conclusions and future work

Graph structural clustering-based community detection algorithms are not used, to the best of our knowledge, in author name disambiguation algorithms. In this paper, we have proposed ASONET, which solves namesakes using only coauthors and titles, which are surely present in all bibliographic datasets. ASONET is a graph-based

algorithm that requires neither costly training data nor a priori hidden information about how many ambiguous authors there are nor any web searches, and no expert knowledge is required. ASONET performance was tested on two Arnetminer datasets and it showed better results than baseline methods. ASONET delivered effective performance in terms of cluster metrics and scalable solution to an author name ambiguity problem. In the future, we plan to analyze self-citations and affiliation of authors (if available) to disambiguate very ambiguous authors.

References

[1] Ferreira AA, Gonçalves MA, Laender AH. A brief survey of automatic methods for author name disambiguation. SIGMOD Rec 2012; 41: 15-26.

[2] Tang J, Fong AC, Wang B, Zhang J. A unified probabilistic framework for name disambiguation in digital library. IEEE T Knowl Data En 2012; 24: 975-987.

[3] Shin D, Kim T, Choi J, Kim J. Author name disambiguation using a graph model with node splitting and merging based on bibliographic information. Scientometrics 2014; 100: 15-50.

[4] Fan X, Wang J, Pu X, Zhou L, Lv B. On graph-based name disambiguation. ACM J Data Inf Qual 2011; 2: 10.

[5] Han D, Liu S, Hu Y, Wang B, Sun Y. ELM-based name disambiguation in bibliography. World Wide Web 2015; 18: 253-263.

[6] Ferreira AA, Veloso A, Goncalves MA, Laender AH. Self-training author name disambiguation for information scarce scenarios. J Assoc Inf Sci Tech 2014; 65: 1257-1278.

[7] Wang X, Tang J, Cheng H, Philip SY. Adana: Active name disambiguation. In: 2011 IEEE 11th International Conference on Data Mining; 11 December 2011. New York, NY, USA: IEEE. pp. 794-803.

[8] Liu Y, Li W, Huang Z, Fang Q. A fast method based on multiple clustering for name disambiguation in bibliographic citations. J Assoc Inf Sci Tech 2015; 66: 634-644.

[9] Liu W, Islamaj DR, Kim S, Comeau DC, Kim W, Yeganova L, Lu Z, Wilbur WJ. Author name disambiguation for PubMed. J Assoc Inf Sci Tech 2014; 65: 765-781.

[10] Levin M, Krawczyk S, Bethard S, Jurafsky D. Citation-based bootstrapping for large-scale author disambiguation. J Am Soc Inform Sci 2012; 63: 1030-1047.

[11] Kang IS, Na SH, Lee S, Jung H, Kim P, Sung WK, Lee JH. On co-authorship for author disambiguation. Inform Process Manag 2009; 45: 84-97.

[12] Wu H, Li B, Pei Y, He J. Unsupervised author disambiguation using Dempster-Shafer theory. Scientometrics 2014; 101: 1955-1972.

[13] Onodera N, Iwasawa M, Midorikawa N, Yoshikane F, Amano K, Ootani Y, Kodama T, Kiyama Y, Tsunoda H, Yamazaki S. A method for eliminating articles by homonymous authors from the large number of articles retrieved by author search. J Am Soc Inf Sci Tec 2011; 62: 677-690.

[14] Zhu J, Yang Y, Xie Q, Wang L, Hassan SU. Robust hybrid name disambiguation framework for large databases. Scientometrics 2014; 98: 2255-2274.

[15] DeCarvalho AP, Ferreira AA, Laender AH, Gonçalves MA. Incremental unsupervised name disambiguation in cleaned digital libraries. Journal of Information and Data Management 2011; 2: 289.

16 Peng HT, Lu CY, Hsu W, Ho JM. Disambiguating authors in citations on the web and authorship correlations. Expert Syst Appl 2012; 39: 10521-10532.

[16] Tang L, Walsh JP. Bibliometric fingerprints: name disambiguation based on approximate structure equivalence of cognitive maps. Scientometrics 2010; 84: 763-784.

[17] Li S, Cong G, Miao C. Author name disambiguation using a new categorical distribution similarity. Lect Notes Artif Int 2012; 7523: 569-584.

[18] Vishnyakova D. Author name disambiguation in MEDLINE based on journal descriptors and semantic types. In: BioTxtM; 2016. pp. 134-142.

[19] Tran HN, Huynh T, Do T. Author name disambiguation by using deep neural network. arXiv preprint 2015; 1502.08030.

[20] Song M, Kim EHJ, Kim HJ. Exploring author name disambiguation on PubMed-scale. J Informetr 2015; 9: 924-941.

[21] Qian Y, Zheng Q, Sakai T, Ye J, Liu J. Dynamic author name disambiguation for growing digital libraries. Inform Retrieval J 2015; 18: 379-412.

[22] Santana AF, Gonçalves MA, Laender AH, Ferreira AA. Incremental author name disambiguation by exploiting domain-specific heuristics. J Assoc Inf Sci Tech 2017; 68: 931-945.

[23] Torvik VI, Smalheiser NR. Author name disambiguation in MEDLINE. ACM T Knowl Discov D 2009; 3: 11.

[24] Han H, Giles L, Zha H, Li C, Tsioutsiouliklis K. Two supervised learning approaches for name disambiguation in author citations. In: Proceedings of 2004 Joint ACM/IEEE Conference on DL; 2004. New York, NY, USA: IEEE. pp. 296-305.

[25] Huang J, Sun H, Song Q, Deng H, Han J. Revealing density-based clustering structure from the core-connected tree of a network. IEEE T Knowl Data En 2013; 25: 1876-1887.

[26] Johnson DB. Finding all the elementary circuits of a directed graph. SIAM J Comput 1975; 4: 77-84.

[27] Cota RG, Ferreira AA, Nascimento C, Gonçalves MA, Laender AH. An unsupervised heuristic-based hierarchical method for name disambiguation in bibliographic citations. J Assoc Inf Sci Tech 2010; 61: 1853-1870.