A Network-embedding Based Method for Author Disambiguation

Jun Xu, Siqi Shen, Dongsheng Li, Yongquan Fu
College of Computer, National University of Defense Technology (NUDT), Changsha, China
Science and Technology on Parallel and Distributed Laboratory, NUDT, Changsha, China
xujunrt@163.com,{shensiqi,dsli,yongquanf}@nudt.edu.cn

ABSTRACT

Most existing author disambiguation work relies heavily on feature engineering or cannot use multiple paper relationships. In this work, we propose a network-embedding based method for author disambiguation. For each ambiguous name, we construct networks among papers sharing an ambiguous name, and connect papers with multiple relationships (e.g., co-authoring a paper). We focus on maximizing the gap between positive paper edges and negative edges, and propose a graph coarsening technique to learn global information. Further, we design a clustering algorithm which partitions paper representations into disjoint sets such that each set contains all papers of a unique author. Through extensive experiments, we show that our method is significantly better than the state-of-the-art author disambiguation and network-embedding methods.

CCS CONCEPTS

• **Computing methodologies** → **Knowledge representation and reasoning**; *Natural language processing*; Semantic networks;

KEYWORDS

network embedding, author disambiguation

ACM Reference Format:
Jun Xu, Siqi Shen, Dongsheng Li, Yongquan Fu. 2018. A Network-embedding Based Method for Author Disambiguation. In *The 27th ACM International Conference on Information and Knowledge Management (CIKM '18), October 22–26, 2018, Torino, Italy*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3269206.3269272

1 INTRODUCTION

A person cannot be uniquely identified by his or her name. For example, a search query for "Mark Newman" could obtain the pages for a physicist works in the University of Michigan, for a computer scientist who works in the same University, and others. Name disambiguation is of paramount importance in many fields, e.g., law enforcement and bibliometrics science. In this work, we focus on author disambiguation that associates documents (e.g., web pages, papers) to different people who share identical names. Albeit it has received much attention from the research community [18, 24], two significant challenges remain unresolved.

The first challenge is how to represent papers effectively and flexibly by making use of biographical information such as address and organization, and networking information such as citation. Albeit researchers have proposed various feature construction methods [18], most of them use only user-defined heuristics [4, 13] to learn the graph properties from networking information. Moreover, as the online data are becoming more complex and dynamic, feature representation methods are required to be more flexible and extensible for scenarios such as privacy-preserving.

The second challenge is how to determine the assignment of papers to its author. Previously, researchers use supervised methods [11] which require data labeling or use unsupervised methods [13, 24] which require manual input of the number of unique authors with identical name (denoted as K). Author disambiguation methods with little human labor are needed.

To address the first challenge, we propose a flexible and effective network-embedding method to learn paper representations. We model the relationships between papers as a set of undirected graphs, and learn representations from these networks jointly. This method is flexible, as networks formed based on relationships can be added or removed based on data availability. For an ambiguous name, the networks are built among papers associated with the name, rather than a network consisting of all the articles. To learn representations, we optimize the gap between positive and negative edges and coarsen networks to capture global structure information.

To address the second challenge, we propose a clustering algorithm which determines the assignment of papers to authors without manual input for K. It adaptively determines paper assignments based on the results of two clustering algorithms [1, 7].

The contributions of this work are listed as follows.

- We propose a novel network-embedding method (in Sec. 3.2), which can encode multiple types of paper relationships for disambiguation.
- We propose a clustering algorithm (in Sec. 3.3) for name disambiguation which does not require the input of the number of authors with identical names.
- We show through experiments (in Sec. 4) that our method obtains better (7% to 50%) results than state-of-the-art methods. The source code of the proposed method is available at https://github.com/xujunrt/Author-Disambiguation.

The first two authors contributed equally. Siqi Shen is the corresponding author.

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/3269206.3269272

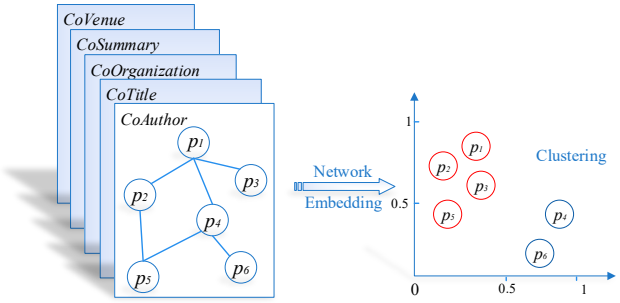

Figure 1: Overview of the disambiguation approach.

2 RELATED WORK

[11] propose a supervised disambiguation method to determine whether a person is the actual author of a paper. [22] use a pairwise factor graph model to improve the results. [18] model the relationships between papers using Hidden Markov Random Field. [13] calculate metric confidence and use the most confident metric to cluster papers. Different from them, we use network-embedding to learn the representations of papers.

DeepWalk [15] and Node2Vec [9] perform random-walk on a network, and uses the word2vec [14] method to learn representations. LINE [20] and GraRep[2] learn high-order proximity of nodes. [3] iteratively merge nodes into higher-level nodes and then use the high-level representations as the initialization value. [8] learn graphs based on meta-path. [17] learn representations based on multi-view of graphs.

The work most similar to us is [24]. It considers only 1 relationship (co-authorship) by building 3 networks and learns representations using PTE. Our work can model multiple relationships and one network is built for each relationship. Then, representations are learned jointly. Besides, ours does not require the input of the number of actual authors.

3 METHODOLOGY

For an author name n, we denote the papers containing the author name n as $\mathcal{P}^n = \{p_1^n, p_2^n, ..., p_m^n\}$, where p_i^n is the ith paper in $\mathcal{P}^n$. For $\mathcal{P}^n$, we construct networks based on paper relationships (Sec. 3.1), and then use network representation learning to learn paper embedding (Sec. 3.2). As the name n is ambiguous, all paper in $\mathcal{P}^n$ are partitioned into disjoint sets (Sec. 3.3). Each set is viewed as all the papers of a unique author. The overview of our approach is depicted in Fig. 1.

3.1 Network Construction

Given all author names $N = \{n_1, n_2, ..., n_m\}$ and all papers $\mathcal{P}$ published by the authors, we want to construct networks based on $\mathcal{P}$. A tempting way is to construct a network consisting of all the papers. However, if $\mathcal{P}$ is huge, then the set of vertices in the network is too large, which causes the network embedding procedure to be ineffective. Therefore, we construct multiple paper-paper networks for each author name n based on $\mathcal{P}^n$, instead of $\mathcal{P}$.

For name n, we create networks connecting papers $\mathcal{P}^n$ based on paper relationships. The following paragraphs will describe the five types of networks: co-author, co-title, co-venue, co-summary, and co-organization. In these networks, V is the vertex set, each vertex v_i represents a paper p_i.

The *co-author network* $G^a = (V, E^a)$, where E^a is the edge set. Given two papers p_i and p_j, and their author list A_i and A_j, the edge weight w_{ij}^a between p_i and p_j is $|A_i \bigcap A_j - n|$. w_{ij}^a is the number of overlapping authors in p_i and p_j (excluding n). To deal with name abbreviation, we adopt the method proposed in [4].

The *co-title network* $G^t = (V, E^t)$ models the relationship between paper titles. The word set used in title of p_i is denoted as t_i (with stop words removed). We transform each word in t_i to its word stem (e.g., networking to network). Then, the edge weight w_{ij}^t between p_i and p_j is the number of words overlapping in t_i and t_j, that is, $w_{ij}^t = |t_i \bigcap t_j|$.

The *co-venue network* $G_v = (V, E^v)$ models the relationship among publication venues (e.g., conference, journal). The *co-summary network* $G_s = (V, E^s)$ models the relationship among paper abstract. The *co-organization network* $G_o = (V, E^o)$ models the relationship among the name of authors' affiliation. These networks are constructed in a similar way as the *co-title network*.

The intuition behind the co-author network is that, if two papers have many co-authors, the probability that the same person writes them is high. Even if they do not have a co-author, but their neighboring papers have many co-authors, these two papers are likely to be written by the same person. Our method is flexible and extensible as networks can be added or removed based on the data availability. For example, if the citation relationship of papers is available, then a citation network could be built.

3.2 Network Embedding

Given the five networks created in Sec. 3.1, the network embedding algorithm generates for each paper p_i (v_i in V) an l-dimensional vector d_i through jointly learning on the networks. The basic idea is that vertices which are close in networks are similar to each other, and thus should be encoded closely in R^l. Because the goal of the embedding procedure is to obtain representations which are useful for author disambiguation, we do not aim to reconstruct the whole network by minimizing the KL-divergent between the probability and the weight of edges. We define the score of an edge e_{ij} connecting vertex v_i and vertex v_j as follows.

$$s(v_i, v_j) = \frac{d_i^T d_j}{\|d_i\| \cdot \|d_j\|} \tag{1}$$

For e_{ij} in E, $s(v_i, v_j)$ is defined as the cosine similarity between the two vertices. As the networks in this work are undirectional, the cosine similarity ensures that $s(v_i, v_j) = s(v_j, v_i)$. Given a vertex k that is not connected to i, we denote the non-existing edge as ne_{ik}, and the set of all non-existing edges as NE. The score of ne_{ik} is defined as the cosine similarity $s(v_i, v_k)$ between v_i and v_k. Then, we model the loss for (i, j, k) triple as:

$$L(i, j, k) = -ln(max(\epsilon, s(v_i, v_j) - s(v_k, v_i))) \tag{2}$$

We set $\epsilon = 0.01$ to avoid the case of dividing by zero. The intuition for Formula (2) is that, the score difference between e_{ij} and ne_{ik} should be large to reflect the edge status. We denote the loss functions of the co-author, the co-title, the co-venue, the

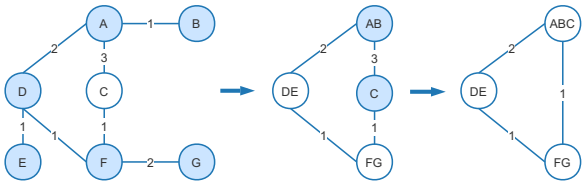

Figure 2: Illustration of the coarsening procedure.

co-summary, the co-organization networks as L^a, L^t, L^v, L^s, L^o, respectively. They are described in formula (3).

$$L^{\mathcal{R}} = \sum_{\substack{e_{ij} \in E^{\mathcal{R}} \\ ne_{ik} \in NE^{\mathcal{R}}}} L(i,j,k), \qquad \mathcal{R} \in \{a,t,v,s,o\} \tag{3}$$

We aim to minimize the final objective function is defined in Formula (4), where $w^{\mathcal{R}}$, $E^{\mathcal{R}}$, $NE^{\mathcal{R}}$ $\mathcal{R} \in \{a,t,v,s,o\}$ are the weights, edges, and non-existing edges of the networks. L_{reg} is the l^2 regulation term to avoid overfitting. For model optimization, we use stochastic gradient descent.

$$OBJ = \sum_{\mathcal{R}} w^{\mathcal{R}} L^{\mathcal{R}} + \lambda L_{reg} \quad \mathcal{R} \in \{a,t,v,s,o\} \tag{4}$$

To reduce the computational overhead, for a network with $|E|$ edges, $T \times |E|$ triples are sampled. The sampling procedure for (i,j,k) is described as follows.

- Positive sampling: the probability of sampling e_{ij} is proportional to the edge weight w_{ij}.
- Negative sampling: for each vertex i's non-neighbor k, the probability of sampling the negative edge ne_{ik} is proportional to $e^{-s(v_k, v_i)}$. Through negative sampling, the more similarity between vertices i and k, the less sampling probability for ne_{ik}. This method is different from DeepWalk which samples vertices based on weights.

To better capture the global properties of networks, we coarsen a network by merging vertices. The procedure is depicted in Figure 2. We randomly select a low-degree vertex v_i and merge it with its high-degree neighbor v_j. The new edge weights are calculated using the same method as in Sec. 3.1. For vertices with an identical degree, the tie is broken by selecting the vertex with a higher-weight edge. The procedure stops when the number of nodes is reduced to 1/3 of the original network.

We first coarsen the most important network (i.e., co-author), then other networks are coarsened in the same way. After that, the node representations in the coarsened networks are learned. Then, these representations are used as initial value for the merged vertices. Finally, the representation learning is also performed on the original networks to obtain the final representations.

3.3 Clustering Algorithm

For each ambiguous author name, the clustering algorithm divides the papers into non-overlapping clusters, each belonging to a distinct author. We find that the number of publications per author is skewed, which indicates that the size of cluster belonging to each author is skewed. We use HDBSCAN [1] and affinity propagation clustering (AP) [7] to perform clustering, as they are effective methods for skewed data and they do not requires the input of the number of clusters.

HDBSCAN prefers to cluster data into a small number of clusters, as it condenses clusters into larger ones. A person with a low publication count may be merged with others. AP tends to produce more clusters, which may treat a person as multiple persons by dividing a person's publications into multiple clusters. To mitigate their drawbacks, we use the SD index [10] to evaluate the results produced by the two methods. The SD value of the clustering results for HDBSCAN and AP are denoted as SD_H and SD_A, respectively. For the final clustering result, the result produced by HDBSCAN is selected if $SD_H \leq SD_A$, otherwise the result produced by AP is selected.

	Arnetminer	DBLP	CiteseerX
# Names	110	679	14
# Authors	1515	1463	468
# Papers	7022	6478	8453

Table 1: Dataset summary

Method	Arnetminer	DBLP	CiteSeerX
Khabsa et al. 2015 [12]	0.584	0.645	0.424
Qian et al. 2015 [16]	0.547	0.681	0.473
Zhang et al. 2016 [23]	0.613	0.723	0.536
Zhang et al. 2017 [24]	0.635	0.742	0.596
DeepWalk 2014 [15]	0.582	0.734	0.482
LINE 2015 [20]	0.609	0.722	0.553
Node2Vec 2016 [9]	0.589	0.685	0.498
PTE 2015 [19]	0.632	0.762	0.578
CANE 2017 [21]	0.624	0.712	0.511
Hin2Vec 2017 [8]	0.616	0.743	0.562
Our	**0.745**	**0.832**	**0.635**

Table 2: The Macro-F1 of the disambiguation results

4 EXPERIMENTAL RESULTS

We conduct experiments on three datasets: Arnetminer, DBLP, and CiteSeerX which are depicted in Table 1. In these datasets, each ambiguous name is a distinct dataset where the disambiguation experiment is conducted. We find that our method is significantly better than all the competing methods.

The representation dimension for all the methods is 40. The parameter setting for our method is: learning rate ($\eta = 0.02$), regularization coefficient ($\lambda = 0.05$), and the initial representations is sampled from [-0.5, 0.5]. The parameter settings for other methods are the same as specified in their publications. All the experiments in this work are conducted with at least 5 repetitions, and the average value of Macro-F1 is reported.

4.1 Competing disambiguation Methods

We compare our method with state-of-the-art disambiguation methods. The author disambiguation methods used in the experiments are listed as follows:

- Khabsa et al. [12] design a pairwise profile similarity function and an improved-DBSCAN method to cluster papers.
- Qian et al. [16] devise a similarity measure for papers, and use hierarchical agglomerative clustering to cluster papers.

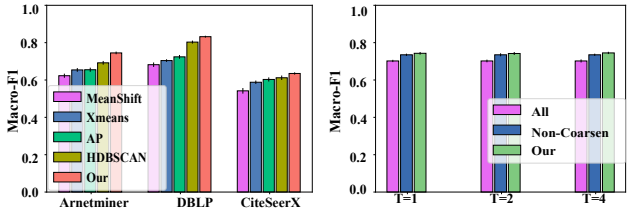

Figure 3: The performance of (Left) different clustering algorithms and (Right) alternative methods.

- Zhang et al. [23] use a Bayesian non-exhaustive classification framework for name disambiguation.
- Zhang et al. [24] use the PTE method to learn paper representations and use a hierarchical agglomerative clustering algorithm to cluster papers.

As it is shown in Table 2, our method **outperforms** all the other methods. For all the datasets, our method is about 7% to 50% better than others, and it is about 7% to 17% better than the second best results.

4.2 Network embedding methods

We evaluate the performance of different network embedding methods. For a fair comparison, the same network construction and clustering method proposed in this work is used. Further, we augment DeepWalk, LINE, and Node2Vec to incorporate heterogeneous networks by concatenating vectors learned from different networks as the final paper representations.

Table 2 summaries the results. Our method is 9% to 31% better than all the others. PTE and Hin2Vec obtain the second and third best performance due to their ability to encode multiple networks directly while the other methods (DeepWalk, LINE, Node2Vec, CANE) cannot. Our method performs better than PTE and Hin2Vec because we focus on the gap between positive and negatives edges, which is suitable for disambiguation task, and our method captures global graph properties via learning on coarsened networks.

4.3 Clustering methods

We evaluate the performance of different clustering algorithms for disambiguation with the proposed network embedding method. The clustering algorithms used for comparison, which does not require the input of the number of clusters, are HDBSCAN, AP, MeanShift [5], and Xmeans [6].

Figure 3 (Left) depicts the results. Our method is about 8% to 30% better than the other clustering methods. Our method outperforms HDBSCAN and AP, and can select better clustering results most of the time. Moreover, it has stable performance (the lowest standard deviation).

4.4 Alternatives and Sensitivity Results

When learning representations, for a network with $|E|$ edges, $T \times |E|$ triples are sampled. We evaluate the impact of T by varying it from 1 to 4. The result for the Artminer dataset is shown in Figure 3 (Right). It can be observed that T does not impact the performance significantly.

For each ambiguous name, we create networks based on papers whose author-list contains the name. An alternative way is to create big networks based on all the papers. The result is shown in Figure 3 (Right) with the legend *All*. Our method is about 6% better than the *All* method for the Arnetminer dataset. We use graph coarsening to learn the global network structure. As shown in Figure 3 (Right), with the coarsened network, the Macro-F1 score of our method is improved by about 2% comparing to Non-Coarsen.

5 CONCLUSION

In this work, we propose a network embedding method to learn the representations of papers for author disambiguation. The method can learn heterogeneous relationships between papers and can be easily adapted to many other scenarios. Furthermore, we propose a clustering algorithm which assigns papers to distinct authors. Through extensive experiments, we show that our method can significantly outperform state-of-the-art methods.

Acknowledgement: This work was partially supported by the National Basic Research Program of China (No. 2014CB340303), the National Science Foundation for Young Scholars of China (No. 61602500, 61502500, 61702529, 61702530, and 61402509).

REFERENCES

[1] Ricardo J. G. B. Campello, Davoud Moulavi, and Jörg Sander. 2013. Density-Based Clustering Based on Hierarchical Density Estimates. In *PAKDD*.
[2] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. GraRep: Learning Graph Representations with Global Structural Information. In *CIKM*.
[3] Haochen Chen, Bryan Perozzi, Yifan Hu, and Steven Skiena. 2018. HARP: Hierarchical Representation Learning for Networks.. In *AAAI*.
[4] Wei Sheng Chin, Yu Chin Juan, and Wei Cheng. 2013. Effective string processing and matching for author disambiguation. In *KDD Cup*.
[5] Dorin Comaniciu and Peter Meer. 2002. Mean shift: a robust approach toward feature space analysis. *PAMI* 24, 5 (2002), 603–619.
[6] Pelleg Dan and Andrew W. Moore. 2000. X-means: Extending K-means with Efficient Estimation of the Number of Clusters. In *ICML*.
[7] Brendan J. Frey and Delbert Dueck. 2007. Clustering by Passing Messages Between Data Points. *Science* 315, 5814 (2007), 972–976.
[8] Tao-Yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. HIN2Vec: Explore Meta-paths in Heterogeneous Information Networks for Representation Learning. In *CIKM*.
[9] A Grover and J Leskovec. 2016. node2vec: Scalable Feature Learning. In *KDD*.
[10] Maria Halkidi, Michalis Vazirgiannis, and Yannis Batistakis. 2000. Quality Scheme Assessment in the Clustering Process. In *PKDD*.
[11] Hui Han, Lee Giles, Hongyuan Zha, Cheng Li, and Kostas Tsioutsiouliklis. 2004. Two supervised learning approaches for name disambiguation in author citations. In *Joint Conference on Digital Libraries (JCDL)*.
[12] Madian Khabsa, Pucktada Treeratpituk, and C. Lee Giles. 2015. Online Person Name Disambiguation with Constraints. In *JCDL*.
[13] Xueqin Lin, Fen Zhu, Bo Peng, and Weiling Li. 2017. A Novel Approach for Author Name Disambiguation Using Ranking Confidence. In *DSFAA*.
[14] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *Computer Science* (2013).
[15] Bryan Perozzi, Rami Alrfou, and Steven Skiena. 2014. DeepWalk: online learning of social representations. *KDD* (2014).
[16] Yanan Qian, Junting Zheng, and Jun Liu. 2015. Author name disambiguation for growing digital libraries. *Information Retrieval Journal* 18, 5 (2015).
[17] Meng Qu, Jian Tang, Jingbo Shang, Xiang Ren, Ming Zhang, and Jiawei Han. 2017. An Attention-based Collaboration Framework for Multi-View Network Representation Learning. In *CIKM*. 1767–1776.
[18] Jie Tang, Alvis C. M. Fong, Bo Wang, and Jing Zhang. 2012. A Unified Probabilistic Framework for Name Disambiguation in Digital Library. *TKDE* 24, 6 (2012).
[19] Jian Tang, Meng Qu, and Qiaozhu Mei. 2015. PTE: Predictive Text Embedding through Large-scale Heterogeneous Text Networks. In *WWW*.
[20] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *KDD*.
[21] Cunchao Tu, Han Liu, Zhiyuan Liu, and Maosong Sun. 2017. CANE: Context-Aware Network Embedding for Relation Modeling. In *ACL*. 1722–1731.
[22] Xuezhi Wang, Jie Tang, Hong Cheng, and Philip S. Yu. 2011. ADANA: Active Name Disambiguation. In *ICDM*.
[23] Baichuan Zhang, Murat Dundar, and Mohammad Al Hasan. 2016. Bayesian Non-Exhaustive Classification A Case Study:Online Name Disambiguation using Temporal Record Streams. In *CIKM*.
[24] Baichuan Zhang and Mohammad Al Hasan. 2017. Name Disambiguation in Anonymized Graphs using Network Embedding. In *CIKM*.