Exploring Graph Based Approaches for Author Name Disambiguation

Chetanya Rastogi*
Stanford University
chetanya@stanford.edu

Prabhat Agarwal*
Stanford University
prabhat8@stanford.edu

Shreya Singh*
Stanford University
ssingh16@stanford.edu

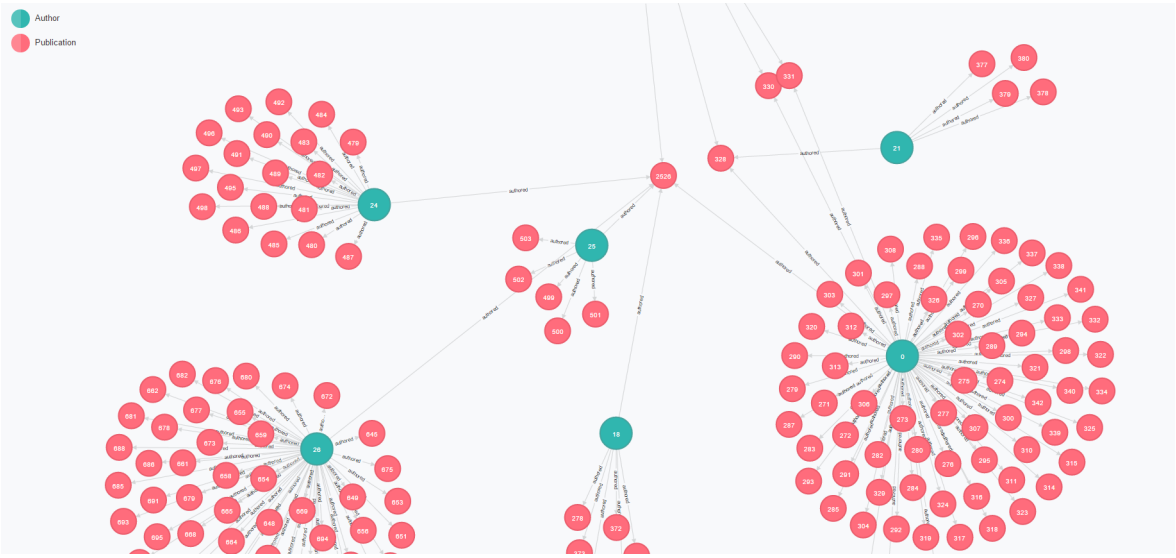

Figure 1: The ever expanding author publication network

ABSTRACT

In many applications, such as scientific literature management, researcher search, social network analysis and etc, Name Disambiguation (aiming at disambiguating WhoIsWho) has been a challenging problem. In addition, the growth of scientific literature makes the problem more difficult and urgent. Although name disambiguation has been extensively studied in academia and industry, the problem has not been solved well due to the clutter of data and the complexity of the same name scenario. In this work, we aim to explore models that can perform the task of name disambiguation using the network structure that is intrinsic to the problem and present an analysis of the models.

1 INTRODUCTION

Online academic search systems (such as Microsoft Academic Graph, Google Scholar, Dblp, and AMiner) have a large amount of research papers, and have become important and popular academic communication and paper search platforms. However, due to the limitations of the paper assignment algorithm, there are many papers assigned to error authors. In addition, these academic platforms are collecting a large number of new papers every day (AMiner has about 130,000,000 author profiles and more than 200,000,000 papers) [15]. Therefore, how to accurately and quickly assign papers to existing author profiles, and maintain the consistency of author profiles is an urgent problem to be solved for current online academic systems and platforms.

In our project, we aim to implement author name disambiguation techniques to disambiguate profiles of authors with similar names and affiliations. We study the problem from a network perspective where researchers communicate with one another by means of their publication. The network is modeled as a bipartite graph containing two types of nodes, viz. author nodes and paper nodes. Each edge in the graph represents an author's contribution to a paper. We believe that this inherent structure will be able to encapsulate much more implicit and intrinsic features that are otherwise impossible to capture using bibliometric data.

2 RELATED WORK

The problem of Author Name Disambiguation has been of interests to researchers for a quite long time. [5] formulates it in the paradigm supervised learning and makes use of the various features associated with a publication, including title, co-authors, conference, etc to make the correctly associate a publication with a specific author by learning the linkage function between the publication and the author. The authors makes use of two different datasets, one from DBLP and the other collected from the web, and test two different classification algorithms on both the datasets. However, the authors do not take into account the implicit network structure that lies in the dataset. [11] furthers the task and provides an extensive study

*All authors contributed equally to this research.

on choosing a minimal subset of features by means of a random forest classifier that can identify the correct author entity linked with a particular publication. The authors also introduce a new dataset called Medline which is particular to the researchers of biomedical science. Once again the authors of these work not only ignore the underlying graph structure but also restrict the work to a particular domain which constraints the problem to a very narrow dataset.

Another important shortcoming of both the above works is that the authors know beforehand of how many clusters they need to identify for a particular name. This challenge is tackled by [10] as they investigate a dynamic approach for estimating the number of people associated with a particular name. They propose a novel approach of framing the problem as a Markov Random Field and try to make use of the underlying graph structure by defining similarity both in terms of the content of the publication as well as the relationship between them in terms of co-authors. Similarly, [15] also talks about learning the cluster size dynamically and quantifying the similarity in terms of the graph structure.

In [8], Ma et al. have proposed a novel AND (Author Name Disambiguation) approach which tries to disambiguate authorship of research papers. As the population is growing, some people will inevitably share some personal features at different levels (like names and affiliations). This poses a huge challenge for many applications like information retrieval and academic network analysis. The dataset used by the authors in this work is the AMiner dataset which is a heterogeneous academic network consisting of multiple entities (i.e. author, paper, venue, topic) as well as relationships (i.e. writing, publishing, collaborating and affiliations). To solve the problem of name disambiguation, the authors propose a metapath channel based heterogeneous network representation learning method (Mech-RL) wherein node embeddings are learned from the whole heterogeneous graph instead of breaking it down to simpler subgraphs.

The node (paper) embeddings are learned at two levels: they are initialized by the textual features (Doc2vec embeddings) and further optimized by relational features (from the metapaths in which they appear). Once, each entity (here paper) is represented through its low dimensional embeddings, the task is reduced to a clustering task where each cluster will contain papers belonging to a unique person. Another thing to be noted is that in this approach, the authors solve the name disambiguation problem without considering the private information of the researchers. The experimental results based on the AMiner dataset show that Mech-RL obtains better results compared to other state-of-the-art author disambiguation methods [9, 13, 14].

3 DATA

3.1 Author Name Disambiguation

We utilise a dataset hosted as a part of a competition called OAG-WhoIsWho Track 1 [1]. The organizers provide three different datasets for training, validation and testing of models but provide the ground truth labels for only the train set. Therefore, to test our models and provide quantitative metrics of our methods we utilise only the training set which we now refer to as the "entire" set.

Task Description: Given a bunch of papers with authors of one same name, the task is to return different clusters of papers. Each cluster has one author, and different clusters have different authors, although they have the same name.

Data Description: The dataset consists of two sets of information: list of publications for same author name and metadata of the publications. The format and fields of the publication metadata is described in Table 1. Additionally the train data contains publications of same author name, clustered by author profile which is the required output of the task. Initial data exploration on the metadata showed that the data is very noisy and has many typos and wrong entries, which makes it non-consumable in the raw form. Therefore, we pre-process and augment the data which is described in the next section.

We run our experiments under supervised as well as unsupervised learning paradigm. To allow for fast and feasible experimentation, we sample 20 names at random from the entire set on which we train and evaluate our methods. For unsupervised methods, we use the complete sampled dataset for training as well as evaluation while for supervised learning methods the sampled dataset is split it into train, validation and test with 15, 2, and 3 names respectively. To verify that the randomly sampled set is a valid placeholder for the entire set, we compare different attributes of the graphs generated by both and see similar distributions. The data summary comparing the statistics of the sampled set with the entire set has been shown in Table 2. The data summary for the train, validation and test sets has been shown in Table 3

3.2 Data pre-processing and summary

3.2.1 Size of the dataset. The number of publications, distinct author names and author profiles is shown in Table 3 and Figure 3 shows the distribution of number of publications across all authors profiles across the sampled dataset. There are on average 103.34 distinct author profiles for each author name in the entire dataset. The distribution of author profile count for author name is shown in Fig. 2.

3.2.2 Conference and Journals. Academic conferences are symposiums which researchers attend to present their findings and hear about the latest work in their field of interest. In Fig. 4, we have illustrated the frequency distribution of top 20 conferences/journals where authors have published their work. We plot the top 20 conferences/journals (by their publication count) on the x-axis and plot the publication count on y-axis. From this data, we can see vividly that conferences and journals in Applied Mechanics/Materials, Applied Physics and Bioinformatics are popular among the authors. This is validated by the keyword frequency distribution graph in Fig. 5 too where we see the top keywords pertaining to topics in these very fields.

3.2.3 Keywords. Effective keywords of an article portray an accurate representation of what an author wants to publish. Many-a-times, in the first glance, we look at the topic, keywords and

Field	Type	Meaning	Example
id	string	PaperID	53e9ab9eb7602d970354a97e
title	string	Paper Title	Data mining: concepts and techniques
authors.name	string	Authors	Jiawei Han
author.org	string	Organization	department of computer science university of illinois at urbana champaign
venue	string	Conference/Journal	Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial
year	int	Publication	2000
keywords	list of strings	Key words	[“data mining”, “structured data”, “world wide web”, “social network”, “relational data”]
abstract	string	Abstract	Our ability to generate...

Table 1: Description of the fields in the paper data

Parameter	Entire Set	Sampled Set
Distinct author names	221	20
Distinct author profiles	22839	1945
# of publications	203184	16788
# of connected components	22105	1927
Largest Connected Component	16339	1048

Table 2: Statistics for Sampled set and the Entire set

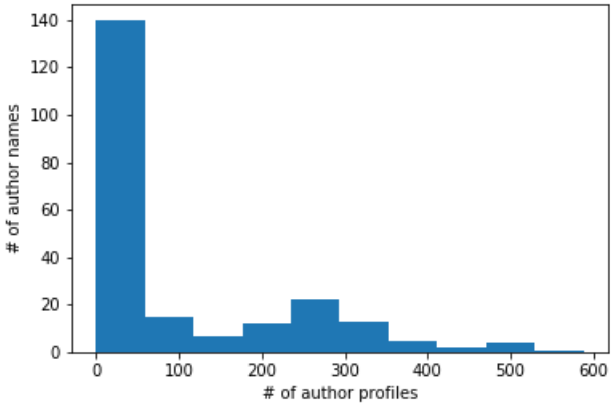

Figure 2: Author name frequency distribution by profile count

Dataset	# of publications	# of author names	# of author profiles	avg. publications per author profile
Train	10966	15	1347	8.14
Validation	1833	2	271	6.76
Test	4055	3	327	12.4

Table 3: Train and validation data summary

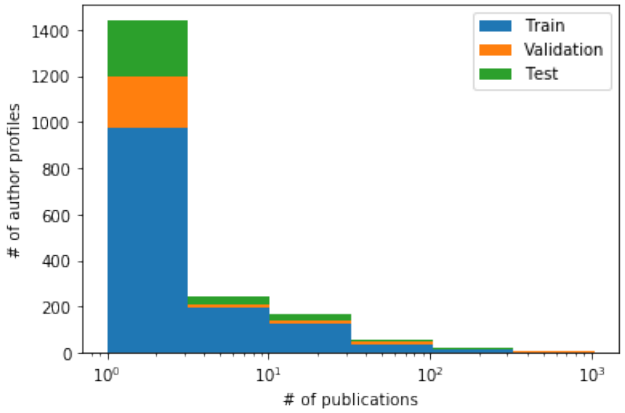

Figure 3: Cluster size distribution(number of publications per author profile)

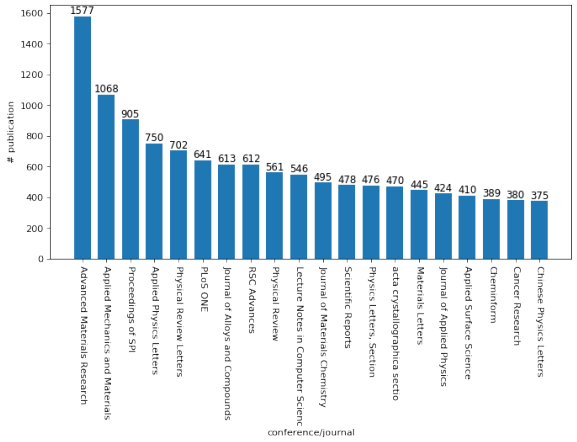

Figure 4: Paper frequency distribution by Conference/Journal

abstract to get an idea about the research context of a publication. Fig. 5 illustrates the frequency distribution of top 20 keywords (by count) in the publications of the training dataset. We plot the top 20 selected keywords (by count) on the x-axis and have their counts plotted on the y-axis. This primarily gives us an idea about the

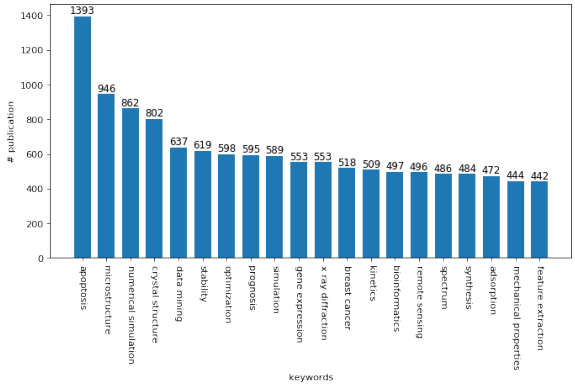

Figure 5: Keyword frequency distribution

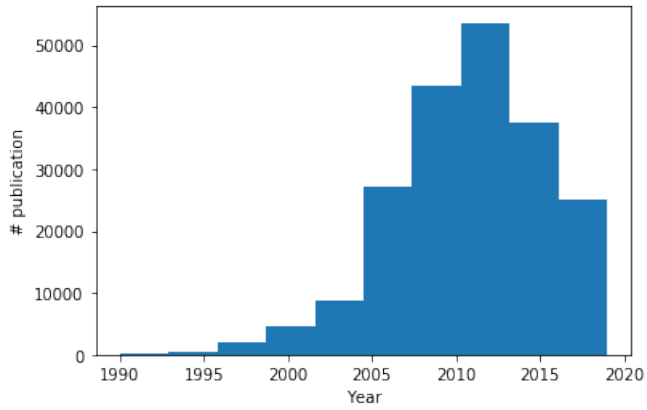

Figure 6: Publication frequency distribution by year

different genres/topics of research where authors have published their work in. It can be seen that many of the publications contain keywords pertaining to the domain of Material research, Applied Mechanics and Bioinformatics.

3.2.4 Year. In Fig. 6, we illustrate the publication frequency distribution by year. We see that our dataset consists of publications throughout the years from 1995 to 2019, with a majority of them being published between 2007-2017. This emphasizes on the recency of the dataset and better robustness to the present scenario. We plot years on the x-axis and the publication count of that year on the y-axis.

3.2.5 Author name and affiliation. In Fig. 7, we illustrate the distribution of author profile count against the count of distinct organizations. More formally, we have recorded the number of author profiles on the y-axis who have been in corresponding number of distinct organizations on the x-axis. The graph shown in Fig. 7 shows that there are many author profiles who have switched across organizations in their career which in turn strengthens the claim that many authors move across different organizations/places to cater to their research interests.

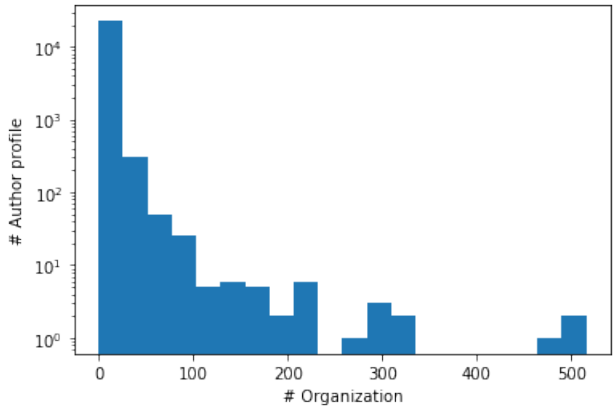

Figure 7: Author profile frequency distribution by organization count

4 METHODS

4.1 Problem Formulation

We formulate the problem of author name disambiguation as finding similarity between nodes in a bipartite graph. Given a set of publications and their respective co-authors, we construct a bipartite graph as shown in Fig 8.

Formally, given a set of publications $\mathcal{P}$, we construct a bipartite graph $\mathcal{G}$ as follows:

$$\mathcal{G} = (U, V, E)$$
$$U = \{a | a \in p.authors \text{ and } p \in \mathcal{P}\}$$
$$V = \mathcal{P}$$
$$E = \{(a, p) | a \in p.authors \text{ and } p \in \mathcal{P}\}$$

Now, we define the task of author name disambiguation as clustering the nodes with same author name in U based on some node similarity function *AUTHOR_SIM*. The clustering algorithm is shown in 1. We analyse the behavior of various node similarity functions *AUTHOR_SIM* based on random walks (section 4.4), node

Algorithm 1: Clustering author-org nodes based on similarity function

Input: $\mathcal{P}$: Set of publications (partitioned by author name) to cluster according to author profiles, *AUTHOR_SIM*: author node similarity function
Output: Cluster of publications according to author profile

```
Clusters = {}
foreach p ∈ P do
    Find cluster in Clusters with greatest similarity s with
      author of interest in p using the similarity function
      AUTHOR_SIM
    If s > θ, add p to the maximum similarity cluster else
      create a new cluster
end
return Clusters
```

embedding (section 4.5), and graph convolution networks (section 4.6).

4.2 Evaluation

The evaluation metric used in the task is macro-averaged F-1 score defined as below:

$$\text{PairwisePrecision} = \frac{\#PairsCorrectlyPredictedToSameAuthor}{\#TotalPairsPredictedToSameAuthor}$$

$$\text{PairwiseRecall} = \frac{\text{\#Pairs Correctly Predicted To Same Author}}{\text{\#Total Pairs To Same Author}}$$

$$\text{Pairwise } F_1 = \frac{2 \times \text{PairwisePrecision} \times \text{PairwiseRecall}}{\text{PairwisePrecision} + \text{PairwiseRecall}}$$

4.3 Text based similarity (Baseline)

We implement two baseline methods and study the performance of our system with respect to them. Both the baselines are described as follows with their performance summarised in Table 4:

(1) **ClusterByName**: For the first baseline we combine all the authors with the same name under a single author profile. Formally, we define the *AUTHOR_SIM* function as follows:

$$AUTHOR\_SIM(a1, a2) \leftarrow a1.name = a2.name$$

This provides a lower bound on any system's overall performance as it is likely that authors with the same name might be the same person (entity) and highly unlikely (though not impossible, since people use different forms of names) that authors with different names represent the same person (entity). As expected, the precision is quite low and the recall of the system is very high in this case.

(2) **ClusterByNameAndOrg**: For the second baseline, we further fine-grained the author profiles with respect to their affiliation and name combined. Due to the highly noisy data, instead of doing a perfect match for organization name we make use of jaro-winkler similarity metric to match organizations.
Formally, we define the *AUTHOR_SIM* function as follows:

$$AUTHOR\_SIM(a1, a2) = a1.name == a2.name \quad \& \quad jaro(a1.org, a2.org) > 0.9$$

Since it is highly unlikely that authors with the same name are affiliated with the same organization, this baseline ensures that we do not cluster different author profiles together but runs a risk of creating multiple profiles for a single author. The performance of the system degraded with this baseline which implies that authors frequently change affiliation over their lifetime which is validated from the author affiliation statistic in Fig. 7.

4.4 Random Walk based similarity

Random walk with restart (RWR) provides a good relevance score between two nodes in a graph, and it has been successfully used in numerous settings, like automatic captioning of images, generalizations to the "connection subgraphs", personalized PageRank, and many more. Hence we use a slightly modified version of the RWR algorithm shown in algorithm 2 to find and merge similar author nodes. In our version, the similar nodes are merged on the go after each random walk so that further iterations can benefit from the results of previous iterations. Formally, we define the *AUTHOR_SIM* function as follows:

Algorithm 2: Random walk based node merging

Input: G: Bipartite graph of authors and publications, α: restart probability, N: max number of epochs, W: Random walk length, T: Threshold of visit count for merge
Output: G with disambiguated nodes merged

```
for epoch ← 1 to N do
    foreach authorNode ∈ G.V do
        visitCount ← {}
        startNode ← authorNode l ← 0
        while l < W do
            if random < alpha then
                authorNode ← startNode
            end
            else
                sample a random neighbor pubNode of
                 authorNode
                sample a random neighbor coAuthorNode
                 of pubNode
                authorNode ← coAuthorNode
            end
            visitCount[authorNode]+ = 1
        end
        Merge startNode with nodes with
         visitCount[node] > T if same is similar.
    end
end
```

$$AUTHOR\_SIM(a1, a2) = a1.name == a2.name \quad \& \quad RWRVisitCount(a1, a2) > 0$$

Unlike the other similarity functions explored, in this method we update the graph online as we find similar nodes using the RWR visit count.

4.5 Transductive embedding based similarity

Node Embeddings have been successful in many graph classification and clustering tasks and hence we explore both transductive and inductive embedding methods to define the author node similarity function. Inductive learning methods are described in next section under the graph convolution methods. In this section, we describe a popular transductive embedding method Node2Vec.

Node2Vec framework learns low-dimensional representations for nodes in a graph by optimizing a neighborhood preserving objective. The objective is flexible, and the algorithm accommodates for various definitions of network neighborhoods by simulating biased random walks. The two main user-defined hyperparameters p and q stand for the return and in-out hyperparameters respectively. The return parameter p controls the probability of the walk staying inwards, revisiting the nodes again (exploitation); whereas

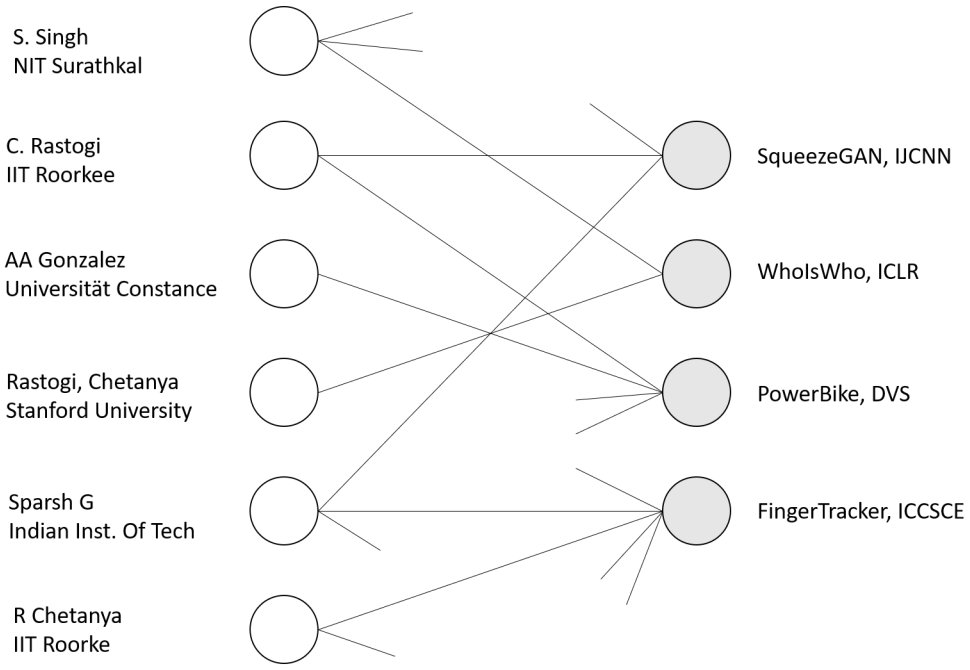

Figure 8: Bipartite graph of author node and publications

the inout parameter q controls the probability of the walk going farther out to explore other nodes (exploration).

In our approach, we run the Node2Vec algorithm on the bipartite graph $\mathcal{G}$ defined in section 4.1. In our setting, we run Node2Vec with the length of the walks set at 10, number of epochs set at 20 and p and q parameters set at 1.
After running the Node2Vec algorithm, we derive the node embeddings *EMB* of all the paper and author nodes of our graph. Then we define the *AUTHOR_SIM* function as follows:

$$AUTHOR\_SIM(a1, a2) = a1.name == a2.name \\ \& \quad cosine(EMB(a1), EMB(a2)) > \theta$$

where θ is a user defined threshold.
The intuition behind using Node2Vec embeddings to express the author and paper nodes is that Node2Vec leverages the inbuilt graphical properties of a graph by running multiple random walks across the nodes. Hence, running Node2Vec on the given Bipartite graph will result in placing similar author (author-organization) nodes together in the walks. The author (author-organization) nodes which occur together in multiple random walks will eventually get very similar embeddings by optimizing over the loss function of Node2Vec. Intuitively, this means that the author ((author-organization)) nodes having similar embeddings might belong to the same author, and hence, should be coalesced together; conditioned on some defined clustering threshold. We tabulate the results from this approach using different values for the threshold in Table 6.

4.6 Graph Neural Networks

Graph neural networks have been very successful in a variety of graph and node classification tasks as they can learn representation of the node incorporating both graph features and node features. Hence we use GNN to learn the *AUTHOR_SIM* function using both supervised and unsupervised setting. The initial node features used in these methods is described below:

4.6.1 Features: To take into account the meta-information of the publications, we make use of the various fields that accurately identify a publication. As mentioned in section 3.2, we analyse all the fields and define the following features that are further used in all the graph neural network based approaches:

- Title: Titles convey a very precise and specific information which is very unique w.r.t each publication. Therefore to incorporate the information contained in the title, we generate 100-dimensional Doc2Vec embeddings [7] which are obtained by training over the entire corpus of titles.
- Abstract: As in the case of a title, abstract also contain crucial information but unlike the title, they are at a higher level and in some sense convey the broader area to which the publication is related. Like titles, we generate 100-dimensional Doc2Vec embeddings [7] which are obtained by training over the entire corpus of abstracts.
- Year: We generate standardized year number for each paper with respect to the starting year number observed in the year distribution of the training corpus. We then use the standardized year number directly as a feature.
- Organization: Inspired by Name2Vec [3], we generate 100-dimensional embeddings of organization using Doc2Vec

where each organization is represented as a document for character bigrams and trained over the whole corpus.

To summarize, generate separate Doc2Vec embeddings [7] for *abstract* and *title* fields, each in a 100-dimensional space. Also to account for the activity of an author in the temporal space, we make use of the *year* field and standardize it w.r.t a starting year. Similarly, we embed the *org* field in a 100-dimensional space using Name2Vec [2]. Also we experimented with two different aggregation methods for combining feature information across nodes. First we projected all the individual features in a latent space and then combined them(sum) whereas in the second method we first combined all the features(concatenate) and then projected the combined feature space to a latent space. In our experiments we noticed that the latter approach (concatenate and project) performed better and hence we report all the results using this method.

4.6.2 Unsupervised Similarity Function. In the unsupervised setting, given the author-publication bipartite graph $\mathcal{G}$, we want to learn embeddings for the nodes such that nodes close in the graph are more similar than those far away. The hypothesis here is that this will lead to node representations such that nodes belonging to same author profile will have similar embeddings when they are in close neighborhood as well as when they are in different components. Formally, given the initial node features $\mathbf{x}$, we calculate the embeddings $\mathbf{z}$ as follows:

$$\mathbf{h}^0 = \mathbf{x}$$
$$\mathbf{h}^l = \text{GNN}(\mathbf{h}^l - 1)$$
$$\mathbf{z} = \mathbf{h}^L$$

We use the GNN described in PinSage [12] using neighborhood sampling to be able to apply this method on large academic graphs. To train the model to learn similar embeddings for nodes in close vicinity and dissimilar embeddings for faraway nodes, we used hinge loss as follows:

$$\mathcal{L} = max(0, \mathbf{z}_{src}\mathbf{z}_{dst} - \mathbf{z}_{src}\mathbf{z}_{dst\_neg} - \delta) \quad (1)$$

Now, we define the *AUTHOR_SIM* function as follows:

$$AUTHOR\_SIM(a1, a2) = a1.name == a2.name \quad \& \quad \mathbf{z}_{a1}\mathbf{z}_{a2} > \theta$$

where θ is a user defined threshold. The results of the model is shown in Table 4.

4.6.3 GNN based Supervised Similarity Function. In the supervised setting, we first create a dataset of pairs of author nodes consisting of pairs which are similar (belonging to the same author profile) and pairs which are dissimilar (belonging to different author profile with same or different name). We then use a Siamese network $\mathcal{F}$ with negative log likelihood loss to learn the weights of the network (shown in 9). We then define the *AUTHOR_SIM* function as follows:

$$AUTHOR\_SIM(a1, a2) = a1.name == a2.name \quad \& \quad \mathcal{F}(a1, a2) > \theta$$

Method	pP	pR	pF
ClusterByName	0.14	1.00	0.25
ClusterByNameAndOrg	0.25	0.12	0.17
RWR-Merge	0.30	0.17	0.22
Node2Vec	0.27	0.11	0.16
Supervised-PinSage	0.22	0.26	0.24
Supervised-GraphSage	0.20	0.27	0.23
Supervised-GCN	0.14	0.72	0.24
Supervised-MLP	0.14	0.28	0.18
Unsupervised-Pinsage	0.12	0.80	0.21

Table 4: Performance of baseline methods

where θ is a user defined threshold. We use different variations of the GNN architecture, results of which is shown in 4.

4.6.4 MLP based Supervised Similarity Function. To explore the effect of using graph features in finding similar nodes, we also train a model with only fully connected layers instead of the GNN layers in the above network. The results of the model is shown in 4.

5 RESULTS AND ANALYSIS

5.1 RWR-Merge

Table 5 shows some of the sample nodes that were correctly identified and merged together by just using the network structure. Since we are only merging the nodes if both the nodes have the same name, this method was supposed to perform better on the pairwise precision metric without any guarantees on recall.

5.1.1 Error Analysis: We expected that this method to have a high pairwise precision but didn't get the desired results as shown in Table 4. On the careful inspection of the merged nodes, we found that the precision suffered because a high number of author nodes didn't had the *org* info associated with them. This might have lead to a wrong initialization at the beginning of the algorithm as two different authors with same name but no *org* were already provided as a single identity to the algorithm. Moreover, as the graph was highly disconnected, the algorithm had no chance of merging two nodes that were split across two or more connected components. This can be seen in the low pairwise recall values.

5.2 Node2Vec

Table 6 tabulates precision, recall and F1 scores for different clustering thresholds used for clustering the Node2Vec vectors, as explained in Section 4.5. The precision, recall and F1 scores are calculated according to the evaluate metrics defined in Section 4.2.

5.2.1 Error Analysis. One of the samples classified correctly by the Node2Vec model (true positive) is for the author name 'Alessandro Giuliani' where two distinct author profiles are identified, the first containing papers with paper ids '5HWAan4P' ('A recursive network approach can identify constitutive regulatory circuits in gene expression data') and 'I7KqbI7a' ('Medical Data Analysis, Third International Symposium, ISMDA 2002, Rome, Italy, October 8-11,

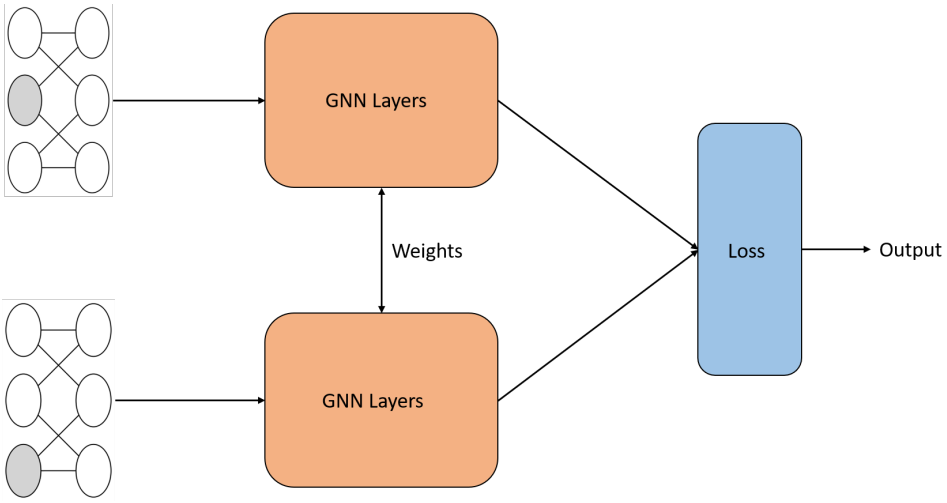

Figure 9: Architecture of network for supervised training

Name	(Name,Org)
m_giffels	(m_giffels, cern) (m_giffels, rwth)
t_dahms	(t_dahms,laboratoire leprinceringuet ecole polytechnique) (t_dahms,cern european organization for nuclear research)
e_yildirim	(e_yildirim,enrico fermi institute) (e_yildirim,desy)
m_k_jha	(m_k_jha,purdue university) (m_k_jha,university of puerto rico)

Table 5: Sample clusters produced by RWR-Merge

2002, Proceedings') and the second containing papers with paper ids 'tznTWpXP' ('Multifractal characterization of protein contact networks') and 'fbcIaJuu' ('A generative model for protein contact networks'). Similarly, among the False positive examples, we have author name 'Yan Liang', for whom we cluster various papers of paperids 'zwzfkKwL' ('Track initiation algorithm based on Hough transform and clustering'), 'rZGsx5cX' ('Space-time linear dispersion codes based on optimal algorithms'), 'rl9MJQHl' ('Coordinative stock management system for permissible storage in VMI pattern') under the same author profile.

5.3 Unsupervised GNN embeddings

The results for unsupervised PinSage algorithm are shown in Table 4. This method had very high pairwise recall at the expense of pairwise precision. The method did extremely well in identifying similar author nodes that were scattered across multiple connected components of the bipartite graph but failed to capture fine-grained distinction between different author nodes with the same name.

Similarity threshold	pP	pR	pF1
0.0	0.14	0.97	0.25
0.5	0.27	0.11	0.16
0.8	0.27	0.11	0.15
0.95	0.26	0.10	0.15

Table 6: Node2Vec evaluation on different clustering thresholds

5.3.1 Error Analysis: The unsupervised version of PinSage overcame the problem of RWR as it defined the similarity function which could assign non-trivial values to any two nodes in the entire graph. Due to this the algorithm successfully identified similar nodes across multiple connected components on the basis of the graph structure(like node degree, egonet, etc). However, the algorithm couldn't discriminate across distinct author profiles due to lack of supervision. This was evident from the fact that the final clustering obtained from this method had 2 clusters for each author name, one with high degree(high publications) and the other group with low degree.

5.4 Supervised GNN/MLP embeddings

We used different GNN architectures to study the variation of performance depending upon the network. We also conducted an ablation study using only FC layers over the node features to study the effect of incorporating features of the neighbors. The results of different experiments in shown in table 4. We expect the MLP to perform poorly when compared to GNN layers and expect that the supervised setting will perform better than the unsupervised model above.

5.4.1 Error Analysis. : Since in the MLP architecture we are training only on author node features, i.e., the embedding of the organization, we expect the results to be similar to the baseline model

where we clustered nodes based on the name and organization and this is indeed the case as can be seen from the table 4.

As compared to the unsupervised GNN embeddings, the supervised architecture is expected to perform better as the labels are directly fed into the system allowing the network that nodes in different components can also be similar and hence bias the network to look more into node features like the abstract and title of the publications. We observe that the different GNN architecture like GCN [6], GraphSage [4], PinSage [12] perform similarly on this task which is an interesting line to explore in the future.

Also, we have also observed that while the embedding similarities are quite skewed in the unsupervised setting rendering the model immune to threshold variation, the performance of the supervised model is dependent on the threshold giving a knob to tune recall and precision as required.

6 CONCLUSION

In this paper, we have thoroughly analysed the dataset hosted as a part of the Open Academic Graph WhoIsWho Track 1 and have implemented various techniques to specifically address the Author Name Disambiguation problem. Formally, we first represent the dataset in a Bipartite graph format containing two types of nodes: author and paper. We then define different flavours of the author similarity function to cluster the author nodes with same author name together. We experiment these different author similarity functions with (1) Text Based similarity (2) Random Walk based similarity (3) Transductive embedding based similarity and (4) Graph Neural Network methods and record results for the same. We conduct extensive quantitative and qualitative analysis of our dataset and graph, run several offline experiments with different combinations of Graph-based approach and author similarity functions and report the results. We observe that random walk based methods have high precision but low recall (as we cluster nodes conservatively) whereas embedding based methods in general have low precision and high recall (due to nodes across components being clustered together).

7 FUTURE WORK

We applied several architectures and learning paradigms to solve the problem of Author name Disambiguation and did a rigorous error analysis on these methods. Based on the results, one straightforward extension is to combine the RWR method along with other supervised learning techniques. This is because RWR can provide with a good starting point by aggregating some nodes which can result in high accuracy and low training time for these networks. Another area to focus is the tuning of the hyper-parameters to achieve a model of optimum performance as the models have shown significant promise in the initial experiments conducted by us.

ACKNOWLEDGMENTS

We would like to thank Michele Catasta for guidance and consistent help throughout the course of the project and the generous google compute credits.

REFERENCES

[1] 2019. OAG-WhoIsWho Track 1. https://www.biendata.com/competition/aminer2019/.

[2] Jeremy Foxcroft, Adrian d'Alessandro, and Luiza Antonie. 2019. Name2Vec: Personal Names Embeddings. In *Advances in Artificial Intelligence*, Marie-Jean Meurs and Frank Rudzicz (Eds.). Springer International Publishing, Cham, 505–510.

[3] Jeremy Foxcroft, Adrian d'Alessandro, and Luiza Antonie. 2019. Name2Vec: Personal Names Embeddings. In *Canadian Conference on Artificial Intelligence*. Springer, 505–510.

[4] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.

[5] Hui Han, Lee Giles, Hongyuan Zha, Cheng Li, and Kostas Tsioutsiouliklis. 2004. Two Supervised Learning Approaches for Name Disambiguation in Author Citations. In *Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL '04)*. ACM, New York, NY, USA, 296–305. https://doi.org/10.1145/996350.996419

[6] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[7] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*. 1188–1196.

[8] Xiao Ma, Ranran Wang, and Yin Zhang. 2019. Author Name Disambiguation in Heterogeneous Academic Networks. In *International Conference on Web Information Systems and Applications*. Springer, 126–137.

[9] Christian Schulz, Amin Mazloumian, Alexander M Petersen, Orion Penner, and Dirk Helbing. 2014. Exploiting citation networks for large-scale author name disambiguation. *EPJ Data Science* 3, 1 (2014), 11.

[10] Jie Tang, Alvis CM Fong, Bo Wang, and Jing Zhang. 2011. A unified probabilistic framework for name disambiguation in digital library. *IEEE Transactions on Knowledge and Data Engineering* 24, 6 (2011), 975–987.

[11] Pucktada Treeratpituk and C Lee Giles. 2009. Disambiguating authors in academic publications using random forests. In *Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*. ACM, 39–48.

[12] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 974–983.

[13] Baichuan Zhang and Mohammad Al Hasan. 2017. Name disambiguation in anonymized graphs using network embedding. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1239–1248.

[14] Baichuan Zhang, Tanay Kumar Saha, and Mohammad Al Hasan. 2014. Name disambiguation from link data in a collaboration graph. In *2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014)*. IEEE, 81–84.

[15] Yutao Zhang, Fanjin Zhang, Peiran Yao, and Jie Tang. 2018. Name Disambiguation in AMiner: Clustering, Maintenance, and Human in the Loop.. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18)*. ACM, New York, NY, USA, 1002–1011. https://doi.org/10.1145/3219819.3219859