A Multi-Level Author Name Disambiguation Algorithm

Siyang Zhang1,3, Xinhua E2,3 and Tian Pan1.

1Beijing University of Posts and Telecommunications, No 10, Xitucheng Road, Haidian District, Beijing, PRC
2Beijing University of Technology, No 100, Pingleyuan, Chaoyang District, Beijing, PRC
3Peng Cheng Laboratory, No. 2, Xingke first street, Nanshan District, Shenzhen, Guangdong, PRC province

Corresponding author: Siyang Zhang (e-mail: zhangyanrui@bupt.edu.cn).

This research was supported by NST20180104-An end-to-end software defined network theory and key technology for differentiated QoS demand; 2018ZX03001019-003-Research and verification of 5G fixed and mobile fusion system based on NFVSDN architecture; 2019RC03-Research on Key Technologies of Network Adaptive Energy Saving Based on Reinforcement Learning.

ABSTRACT With the rapid development of information technology, the name ambiguity problem has become one of the primary issues in the fields of information retrieval, data mining and scientific measurement. Name disambiguation is used to promote computer technology and big data information, which maps virtual relational networks to real social networks to solve the problem that the same name points to multiple entities. At present many literature search platforms launched their respective scholar system, name ambiguity problem will inevitably affect the precision of other information calculations, reduce the credibility of the system and affect the information quality and content quality. Most work deals with this issue by using graph theory and clustering. However, name disambiguation problem is still not well resolved. In this paper, we propose a multi-level name disambiguation algorithm. This algorithm is mainly based on unsupervised algorithm, which combines Hierarchical agglomerative clustering (HAC) and graph theory for disambiguating. Experimental results show that the proposed solution achieves clearly better performance (+17~25% in terms of F1-Measure) than several methods including HAC [2] and Graph [3].

INDEX TERMS Associative processing, Clustering algorithms, Data handling, Informatics

I. INTRODUCTION

According to the National Science Foundation’s “Science & Engineering Indicators” report in 2018 [1], China published 426.165 million academic papers, which is the largest amount in the world, surpassing the United States (408,985) for the first time. Therefore, Chinese papers occupy a large enough share in the world, while the Chinese name repetition issue is very serious. A report about the repetition of Chinese name launched by Tsinghua Big Data Industry Association shows that the top 100 names cover more than 10% of the whole population nationwide. That quite implies a serious repetition condition. It can be seen that Chinese name ambiguity problem is very serious. How to accurately retrieve the required data in this huge data and extract the key information in the paper data for analysis becomes a concern of researchers.

At present, many literature search platforms such as CNKI, Baidu Academic, DBLP, CiteSeer, PubMed, all have launched their own scholar libraries. However, the current problem is that when the scholar library needs to compute the influence of scholars, the number of author’s papers or other information, it is difficult to distinguish the same name scholar accurately. This will inevitably affect the precision of other information calculations. Therefore, how to reduce the impact caused by the phenomenon of name ambiguity, and maximize the effectiveness of the scholar library, become a major concern of researchers. As a result, “Name Disambiguation” began to be proposed, and attracted the attention of a large number of scholars. Name Disambiguation is divide the cluster of articles of the same name authors into several categories, so that the author of the articles in each category is the same person.

Many methods cluster based on HAC [14, 15, 16, 17], but only encode the extracted feature values directly as one-hot vector. We propose a processing method. After processing, the result is improved (+7% in terms of F1) than before processing.

Based on the previous research results, this paper proposes the method of author name disambiguation. This algorithm is mainly based on unsupervised algorithm, which combines hierarchical clustering and graph theory for disambiguating.
Experimental results show that the proposed solution achieves clearly better performance (+17~25% in terms of F1) than several methods including HAC [2] and Graph [3].

The structure of this paper is as follows: In Sect.2, we introduce the related research work of name disambiguation. This part mainly summarizes related work in the past, and the background of the author name disambiguation method proposed in this paper. In Sect.3, we introduce the core of this article including the similarity calculation method of the author name disambiguation and merging procedure. In Sect.4, we describe our experiment and verify the proposed method. In Sect.5, we summarize the method proposed in this paper. This part also propose work to be done in the future.

II. RELATED WORK

Name Disambiguation, also known as Entity Resolution [4], [5], Name Identification [6]. As early as the late 1960s, this issue has received attention [7]. The ambiguity problem of author name often appears in the academic literature library, digital library and other similar systems. The ambiguity data of the author name will reduce the credibility of the system and affect the information quality and content quality [8].

Combined with the multiple attribute characteristics of paper, the method of name disambiguation mainly depends on the degree of dependence on training data, fall into two main categories [8], supervised learning, and unsupervised learning [13, 18].

Supervised learning mainly refers to the method based on probabilistic model. Han et al. [9] proposed a method using the naive Bayesian model and proposed to use the SVM model to solve the name disambiguation problem. Huang et al. [10] used the DBSCAN method. Although this method uses an unsupervised learning algorithm, the similarity between instances in the learning process is calculated using the LASVM model. Supervised method has a high precision rate, but the training of massive data requires a lot of manual labeling, which is time-consuming and labor-intensive, and with the advancement of time, the data is iteratively fast, and the supervised learning has poor portability.

Unsupervised learning mainly refers to graph theory and clustering. The keys of unsupervised learning are how to define the similarity function among the authors of the same name and the choice of clustering methods. A typical method is to take the words around the target entity to form a feature vector, and then use the cosine similarity of the vector to compare and classify the target entity into the closest cluster of entity referential items. As the method proposed by Malin [11], the article is mapped to a vector of the feature space, and then the cosine of the angles of the two article vectors is used as the similarity calculation method. Pedersen et al. [12] improves the precision of the similarity calculation. They adopts SVD decomposition technique decomposes the dimensionality of the text vector space to obtain the shallow semantic features of a given dimension. Evans et al. [13] used the k-means method. Yutao Zhang et.al. [14], Gilles Louppe et.al. [15] and J. Protasiewicz et.al. [16] use HAC for disambiguation.

In recent work, graph-related algorithms have been widely introduced into the name disambiguation task. In addition to the cut graph method, an important aspect is to use the graph to calculate the similarity between the two documents, and to perform appropriate similarity propagation on the graph. Such methods can often greatly improve the performance of the original algorithm. Malin [11] proposed a method for using the social network to perform similarity calculations for the same name. McRae-Spencer and Shadbolt [17] proposed a graph-based approach to eliminate author ambiguity on large-scale citation networks by using self-cited, co-authorships. This method can achieve higher precision, but the recall rate is relatively low. Kang I S et al. [19], Shin et al. [20], Fan et al. [3] used the relationship between authors to solve the problem of ambiguity names.

The biggest advantage of the unsupervised approach is that it does not require a large amount of training data and training time, as opposed to the supervised of the name disambiguation. On large-scale data, unsupervised algorithms are more feasible and scalable than supervised algorithms.

Based on the previous research results, this paper studies the problem of disambiguation combines hierarchical clustering and graph theory for disambiguating. The main contributions of this paper can be summarized as follows:

1. We propose a processing method. After processing, the string similarity calculation result is improved (+10~34% in terms of F-Measure) than before processing
2. Optimize the string matching algorithm (LDE) for name disambiguation, our method outperforms the baseline methods for similarity calculation (+7.7~23.9% in terms of F-Measure).
3. A multi-level name disambiguation algorithm is proposed. It is proved by experiments that our method outperforms the baseline methods for name disambiguation (+17~25% in terms of F-Measure)

III. SIMILARITY CALCULATION METHOD FOR NAME AMBIGUITY AUTHORS

The principles of the algorithm in this paper are as follows: firstly, according to some strong correlation influence factors, the authors of the same name will be merged with higher precision, expand the cardinality of each cluster, and then extract the relationships of the clusters to merge.

**FIGURE 1. The process of atomic clustering**
The Factors Partitioning part in this paper refers to the atomic clustering method proposed by Wang Feng [21] of Tsinghua University. The processing flow is shown in the Fig.1. Each time we extracted the features from the previous merger, the data volume of the corresponding features can be effectively increased, and the precision rate of merger is improved. Each merger is based on the results of the previous merger, therefore each merger must ensure that the precision of the merger is as high as possible, so we should ensure the precision of the next merger, otherwise it will greatly affect the merger.

### Table I
**Author Relation Example**

<table>
<thead>
<tr>
<th>ID:1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author:</strong> Tao Huang; Jiang Liu; Yunyong Zhang</td>
</tr>
<tr>
<td><strong>Organization:</strong> Beijing advanced innovation center for future internet technology; State key laboratory of networking and switching technology, Beijing university of posts and telecommunications; Research institute of China united communications group co. LTD</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID:2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author:</strong> Tao Huang; Ning Yang; Zhijiang Liu; Yunjie Liu</td>
</tr>
<tr>
<td><strong>Organization:</strong> China United Network Communications Group Co., Ltd.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID:3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author:</strong> Yunyong Zhang; Yunjie Liu; Zhijiang Zhang; Guojie Li; Zhongcheng Li</td>
</tr>
<tr>
<td><strong>Organization:</strong> China United Network Communications Group Co., Ltd.</td>
</tr>
</tbody>
</table>

As shown in the Table. I, it is difficult to judge whether the author name “Tao Huang” in ID1 and ID2 refer to the same author. However, when ID2 and ID3 are combined, the number of collaborators of the author “Tao Huang” is expanded, so it becomes easy to confirm that the “Tao Huang” in ID1, ID2 and ID3 is the same person.

The multi-level name disambiguation algorithm is shown in the Fig.2. It is mainly divided into the following steps:

1. Data Processing, perform data processing, and separate the paper data into required data;
2. Factors Partitioning;
3. Node Relationship Matching;

#### A. Problem Formulation

We treat each paper as a node, let \( n \) be a name entity, denoted as \( n \), and the cluster of papers corresponding to the name \( n \) is represented by the cluster \( P_n = \{p_1, p_2, ..., p_n\} \). Each \( p_i \in P_n \) corresponds to a cluster of features \( F_n = \{f_1, f_2, ..., f_k\} \) including organization, author names, subject, etc. The cluster \( R_n = \{r_1, r_2, ..., r_n\} \) corresponding to the classification of name \( n \) in real world. The task of author disambiguation is to find a function \( \phi \) to partition \( P_n \) into a cluster of disjoint clusters,

\[
\phi(P_n) = S_n, \text{where } S_n = \{s_1, s_2, ..., s_n\}
\]

which can make \( S_n \rightarrow R_n \)

#### B. Feature Selection

We extracted some correlation influence factors, such as collaborators, organization, titles, keywords, published journals and so on. We calculated the correlation coefficient between these features and the labels.

We set collaborators and organization as direct influence differentiation factors. Titles, keywords and published journals are set as indirect influence differentiation factors. We find that the title, keywords, and published journals are strongly related to the author’s subject. In order to reduce the amount of calculation and reduce the dimension, we use the subject to represent these three factors as constraints.
C. DATA PROCESSING
Performing pre-processing operations such as integration, cleaning, and de-duplication on the data to obtain initial data. Each piece of data in the initial data is used as an atom. The author information, organization information, keywords and other information in the initial data are extracted and separated to facilitate quick retrieval.

For Chinese databases, the main difficulties in matching organizational information are as follow:

1. The writing format of the organization is different. For example, the organization "Editorial department of journal of aquatic products of Shanghai ocean university", "editorial department of journal of Shanghai ocean university" and "editorial department of journal of Shanghai ocean university" is obvious that the above three organizations. They belong to the same organization, but the writing format is different and the computer cannot match correctly.

2. The authors and the organizations are not one to one corresponding. There are three kinds of the relationship between the author and the organization in the paper data, one-to-one, many-to-one, many-to-many. Among them, the one-to-one and many-to-one situation can determine the relationship between the author and the organization. However, the many-to-many situation cannot be determined.

In order to solve the problems of the different writing styles of same organization and same organization cannot be matched such as "Institute Of Computing Technology Chinese Academy Of Sciences, Beijing, 100080". We first divide multiple organizations according to semicolon, then for each organization we divided them from head to tail using the word such as "laboratory", "center", "school", "university", "company", "college", "system", "instance", "software institute", "bank" and so on. The specific process of the String process is as follow.

Algorithm 1. String processing
Input: N={ni, n2, ..., nk}, Org
Output: Org_name
1: for ni in N:
2:  if Org compile(ni):
3:       sub_Org = Org.index(ni)
4:       Org_name=Org [0:sub_index+len(ni)]
5:   return Org_name
6: else:
7:  Org_name = Org
8: return Org_name

D. FACTOR PARTITIONING
It is much more difficult for human to find the merged errors from the merged data than to merged the clusters that are not properly merged. As a result, we use the strong factors, when the similarity greater than the threshold, assign the same ID to the authors, to ensure the precision of the merged.

The commonly used algorithm to calculate the similarity between two strings is to calculate the editing distance between two strings. We tried to calculate the editing distance between the organizations of the same name author and found that the effect is good. However, there is also such a problem. For example, two corresponding organizations of the same name are the “Beijing University of Posts and Telecommunications” and “Key Laboratory of Universal Wireless Communications, Ministry of Education, (Beijing University of Posts and Telecommunications”.

The problem is how to handle the situation when they belong to the same person in reality, so we optimize the algorithm, we will name the algorithm LDE (Levenshtein Distance Extended) algorithm, and calculate the similarity based on the algorithm for the processed data. The cluster of organization strings X={x1, x2, …, xm} and the cluster of organization strings Y={y1, y2, …, yn}, we construct a relationship matching matrix LD with m+1 columns and n+1, the first column of the matrix represents X, and the first row represents Y:

\[ LD_{(m+1)\times(n+1)} = \{ld_{ij}\} \quad (0 \leq i \leq m, 0 \leq j \leq n) \]

Fill the relationship matching matrix LD according to the following formula:

\[
ld_{ij} = \begin{cases} 
  i, & j = 0 \\
  j, & i = 0 \\
  \min(ld_{i-1,j-1}, ld_{i-1,j}, ld_{i,j-1}) + 1, & i, j > 0, x_i \neq y_j \\
  ld_{i-1,j-1}, & i, j > 0, x_i = y_j 
\end{cases}
\]

After the LD is filled, the element \(d_{mn}\) is the edit distance between X and Y, which is recorded as:

\[ d(X, Y) = d_{mn} \]

The similarity sim(X, Y) is calculated as:

\[
sim(X, Y) = \begin{cases} 
  1, & X \in Y \text{ or } Y \in X \\
  1 - \frac{d(X, Y)}{\max(\text{len}(X), \text{len}(Y))}, & \text{else}
\end{cases}
\]

Where len(X) and len(Y) are the string lengths of the organization x and the organization y.

There is a case where the same organization has multiple authors with the same name. To solve this problem, a constraint Subject is imported, and when they belong to the same subject, they are merged. The specific process of the Factor Partitioning is as follows.
Algorithm 2. Factor Partitioning

Input: O={O1,O2,...,On}, S = {S1,S2,...,Sn}, ID = {ID1,ID2,...,IDn}, X
Output: id
1: i=0; j=i+1; initial cluster N
2: for cluster ni name in N:
3:  for cluster nj name in N:
4:    if sim(ni,nj) > X and si same with sj:
5:      merged IDi and IDj
6:    update id

E. NODE RELATION DIVISION

We consider each cluster that has been merged in the previous step as a node. In the research of the author name disambiguation, the cooperative relationship between nodes has a strong influence on the correct division of nodes [19]. For two nodes with the same name attribute, if they all have a cooperative relationship with another node, the two nodes have greater similarity.

There is a possible that one author works in two different organizations and his or her same collaborator works in different organizations too, so when we try to find their relationship, this collaborators are seen as two different people. In fact, as shown in Table II, we assign an ID to the author of each organization that is matched. Obviously, the two authors names “Da Guo” on the left are the same person, his corresponding collaborator “Zhijiang Zhang” is also the same person.

TABLE II

<table>
<thead>
<tr>
<th>ID</th>
<th>Author</th>
<th>Organization</th>
</tr>
</thead>
<tbody>
<tr>
<td>552456</td>
<td>Da Guo</td>
<td>China United Network Communications Group Co., Ltd.</td>
</tr>
<tr>
<td>612721</td>
<td>Da Guo</td>
<td>China unicom technology department</td>
</tr>
<tr>
<td>551733</td>
<td>Zhijiang Zhang</td>
<td>China United Network Communications Group Co., Ltd.</td>
</tr>
<tr>
<td>612720</td>
<td>Zhijiang Zhang</td>
<td>China unicom technology department</td>
</tr>
</tbody>
</table>

Therefore, we want to make the implicit relationship of nodes. We create graph through co-author between authors. If there is a relationship between the two nodes, there will be a path linking two nodes. We need to find out if there is a path between the two nodes.

We will treat each of the merged clusters in the previous step as a node, denoted as ni, and use the cluster N = {n1, n2, ..., nn}, represents a cluster of identical names. Cn = {c1, c2, ..., cn} represents the author cluster corresponding to the paper collection pi; the partner cluster of the record node ni is ei, which is expressed as:

\[ e_i = P_{n_i} \cdot C_i - n_i = \sum_{i=1}^{k} p_i \cdot c_i - n_i = \{s_1 \cdot c_1 + s_2 \cdot c_2 + \cdots + s_k \cdot c_k - n_i\} \]

Similarly, the cluster of partners of the record node nj is e_j, which is expressed as:

\[ e_j = P_{n_j} \cdot C_j - n_j = \sum_{j=1}^{k} p_j \cdot c_j - n_j = \{s_1 \cdot c_1 + s_2 \cdot c_2 + \cdots + s_k \cdot c_k - n_j\} \]

According to the Jaccard coefficient function, the similarity between the node ni and the node nj is:

\[ \text{sim}(n_i, n_j) = \frac{|e_i \cap e_j|}{|e_i \cup e_j|} \]

There is a case where the author of the same name who has the same collaborator corresponds to two authors in real life. To solve this problem, we import subject as a constraint.

When the similarity of two same name authors is greater than the threshold while they belong to the same subject, two authors of the same name are merged.

IV. EXPERIMENTS

A. DATA SETS

We obtained a large amount of real intellectual property public data, and after data processing, we got 917,701 initial data after removing dirty data. We selected some authors as experimental samples. Table III shows the authors we selected, it includes the author’s name, the number of papers, and the time period for the author to publish the paper. When evaluating the classification results, we use familiar, well-known scholar whose name is prone to the same name as a sample.

We use manual methods to create standard categories. The process is as follows: For each author name in Table III, we retrieve all the papers published by the name in the database. By classifying the authors of the same name by hand, do our best to accurately classify them.
number of papers with authors of the same name that does not need to merge and has not been merged.

We calculated the changes in the precision, recall rate and F-measure of the algorithm similarity when taking different thresholds.

First, we extracted 3050 records, marking the evaluation results obtained by taking different thresholds.

Table IV shows the experimental results, it can be seen that the LDE algorithm achieves the best result when the threshold is 0.75. We use the same method to test the common string similarity calculation algorithms, such as Jaccard, Cosine Similarity, Levenshtein Distance, Eulidean, and select the best performing threshold. At the same time, for these algorithms, we use our processing method as a comparison. We recorded them as Jacca_pro, Cos_pro, Ld_pro, etc.

Table V show that the results of string similarity algorithm under best performing thresholds. We can see that the same algorithm, after using our string processing method, the F1-Measure (+10–34%), and the execution time is greatly reduced. At the same time, we can see that our method outperforms the baselines in terms of F1-Measure (+23.9% over Jacca_pro, +7.7% over Cos_pro, +17.8% over Ld_pro, +12.5% over Euli_pro, relatively). And the execution time is only longer than its underlying algorithm Levenshtein Distance.

In this paper, we considered several baseline methods based on Hierarchical Agglomerative Clustering (HAC) [2] and Graph [3]. For a fair comparison, we use the same feature. Each feature is a one-hot vector. HAC_pro refer to the feature

<table>
<thead>
<tr>
<th>Author</th>
<th>Number</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qiang Guo</td>
<td>138</td>
<td>1985–2018</td>
</tr>
<tr>
<td>Yunjie Liu</td>
<td>14</td>
<td>1994–2017</td>
</tr>
<tr>
<td>Jiang Liu</td>
<td>13</td>
<td>1997–2017</td>
</tr>
<tr>
<td>Changchuan Yin</td>
<td>19</td>
<td>1998–2018</td>
</tr>
<tr>
<td>Yunyong Zhang</td>
<td>69</td>
<td>1988–2017</td>
</tr>
<tr>
<td>Zhijiang Zhang</td>
<td>37</td>
<td>1995–2018</td>
</tr>
<tr>
<td>Tao huang</td>
<td>37</td>
<td>2001–2017</td>
</tr>
<tr>
<td>Jun Guo</td>
<td>61</td>
<td>1988–2017</td>
</tr>
</tbody>
</table>

**TABLE IV**

<table>
<thead>
<tr>
<th>Threshold</th>
<th>Precision</th>
<th>Recall</th>
<th>F-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.44</td>
<td>0.93</td>
<td>0.60</td>
</tr>
<tr>
<td>0.55</td>
<td>0.44</td>
<td>0.93</td>
<td>0.60</td>
</tr>
<tr>
<td>0.6</td>
<td>0.49</td>
<td>0.93</td>
<td>0.64</td>
</tr>
<tr>
<td>0.65</td>
<td>0.49</td>
<td>0.93</td>
<td>0.64</td>
</tr>
<tr>
<td>0.7</td>
<td>0.84</td>
<td>0.93</td>
<td>0.88</td>
</tr>
<tr>
<td>0.75</td>
<td>0.91</td>
<td>0.93</td>
<td>0.92</td>
</tr>
<tr>
<td>0.8</td>
<td>0.91</td>
<td>0.89</td>
<td>0.90</td>
</tr>
<tr>
<td>0.85</td>
<td>1.00</td>
<td>0.83</td>
<td>0.90</td>
</tr>
</tbody>
</table>

**TABLE V**

<table>
<thead>
<tr>
<th></th>
<th>Jacca</th>
<th>Cos</th>
<th>Ld</th>
<th>Euli</th>
<th>Jacca_pro</th>
<th>Cos_pro</th>
<th>Ld_pro</th>
<th>Euli_pro</th>
<th>Lde_pro</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threshold</td>
<td>1</td>
<td>0.4</td>
<td>0.55</td>
<td>2</td>
<td>1</td>
<td>0.45</td>
<td>0.75</td>
<td>2</td>
<td>0.75</td>
</tr>
<tr>
<td>Precision</td>
<td>0.833</td>
<td>0.686</td>
<td>1.000</td>
<td>0.765</td>
<td>1.000</td>
<td>0.848</td>
<td>0.838</td>
<td>0.941</td>
<td>0.915</td>
</tr>
<tr>
<td>Recall</td>
<td>0.217</td>
<td>0.761</td>
<td>0.478</td>
<td>0.565</td>
<td>0.522</td>
<td>0.848</td>
<td>0.674</td>
<td>0.696</td>
<td>0.935</td>
</tr>
<tr>
<td>F-1</td>
<td>0.345</td>
<td>0.722</td>
<td>0.647</td>
<td>0.650</td>
<td>0.686</td>
<td>0.848</td>
<td>0.747</td>
<td>0.800</td>
<td><strong>0.925</strong></td>
</tr>
<tr>
<td>Time(s)</td>
<td>5.0</td>
<td>6.92</td>
<td>14.61</td>
<td>6.93</td>
<td>1.93</td>
<td>2.15</td>
<td>1.43</td>
<td>2.77</td>
<td>1.76</td>
</tr>
</tbody>
</table>
extraction based on our string processing method. Graph only
uses the feature of collaborator for disambiguation.

### TABLE VI
RESULTS OF AUTHOR NAME DISAMBIGUATION

<table>
<thead>
<tr>
<th>Name</th>
<th>HAC</th>
<th>HAC_pro</th>
<th>Graph</th>
<th>Our method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Prec</td>
<td>Rec</td>
<td>F1</td>
<td>Prec</td>
</tr>
<tr>
<td>Qiang Guo</td>
<td>0.55</td>
<td>0.77</td>
<td>0.64</td>
<td>0.80</td>
</tr>
<tr>
<td>Yunjie Liu</td>
<td>0.67</td>
<td>1.00</td>
<td>0.8</td>
<td>0.83</td>
</tr>
<tr>
<td>Jiang Liu</td>
<td>0.75</td>
<td>0.86</td>
<td>0.8</td>
<td>1.00</td>
</tr>
<tr>
<td>Changchuan Yin</td>
<td>0.44</td>
<td>0.89</td>
<td>0.59</td>
<td>1.00</td>
</tr>
<tr>
<td>Yunyong Zhang</td>
<td>1.00</td>
<td>0.99</td>
<td>0.99</td>
<td>0.3</td>
</tr>
<tr>
<td>Zhijiang Zhang</td>
<td>0.13</td>
<td>0.19</td>
<td>0.15</td>
<td>1.00</td>
</tr>
<tr>
<td>Tao huang</td>
<td>0.68</td>
<td>1.00</td>
<td>0.81</td>
<td>0.73</td>
</tr>
<tr>
<td>Jun Guo</td>
<td>0.34</td>
<td>1.00</td>
<td>0.81</td>
<td>0.35</td>
</tr>
<tr>
<td>Average</td>
<td>0.57</td>
<td>0.84</td>
<td>0.70</td>
<td>0.75</td>
</tr>
</tbody>
</table>

Table VI shows the performance of different disambiguation methods on some sampled names. We use precision, recall, F-measure to evaluate our method against the alternative ones. As can be seen from the above table, our method generally has a good recall rate, precision and F-measure. Our method outperforms the baselines in terms of F1-Measure (+25% over HAC, +18% over HAC_pro, +17% over Graph, relatively).

### C. Feature Contribution Analysis
We investigated the contribution of the features for name disambiguation. In particular, we first use organization, followed by adding co-author, and then we add the subject as a constraint. In each step, we evaluate the performance of our method. Fig. 3 shows the average Precision, average Recall, and average F-Measure of our method with different feature combinations. At each step, we observed improvements. We can also see that subject mainly contribute to the improvement of precision, while co-author mainly contribute to the improvement of recall.

### D. Complexity Analysis
In Factors Partitioning, the time complexity is $O(|L_1| + |L_2|)$.

In Node Relationship Matching, the time complexity is $O(n^2)$. In which $n$ is the number of the same name author.

While the time complexity of HAC is $O(n^{2\log n})$ and Graph is $O(n^2)$. Good news is our method can be done offline when precision is the priority concern, and the learning process of the model can be further accelerated by parallel computing technologies.

### V. CONCLUSION AND DISCUSSION
Disambiguation of names in the database is an important task because different people can share the same name. This paper introduces a multi-level author name disambiguation algorithm, which can better solve the same name and optimize the string matching algorithm (LDE) for academic retrieval. In the experiment, we used the Precision, Recall, and F1-Measure to evaluate our method and compare it to other methods. The experimental results show that the method effectively distinguishes the authors of the same name and achieves better results than the baselines methods. However, note that Chinese is one of best suitable languages for this study, since it does not suffer from the matching problem of name coreference. In the future, we plan to map Chinese data with English data to make the data better integrated.
REFERENCES


