Name Disambiguation Using Atomic Clusters

Feng Wang, Juanzi Li, Jie Tang, Jing Zhang, and Kehong Wang
Department of Computer Science and Technology, Tsinghua University
East Main Building Room 10-201,Tsinghua University, Beijing
100084, China
{wangfeng,ljz,tangjie,zhangjing,wkh}@keg.cs.tsinghua.edu.cn

Abstract

Name ambiguity is a critical problem in many applications, in particular in the online bibliography systems, such as DBLP and CiteSeer. Previously, several clustering based methods have been proposed although, the problem still presents to be a big challenge for both research and industry communities. In this paper, we present a complementary study to the problem from another point of view. We propose an approach of finding atomic clusters to improve the performance of existing clustering-based methods. We conducted experiments on a dataset from a real-world system: Arnetminer.org. Experiments results show that significant improvements can be obtained by using the proposed atomic clusters finding approach (about +8% and +27% improvements depending on different clustering methods).

1. Introduction

Name ambiguity is a real-world problem that one name possibly refers to more than one actual persons. It is a critical problem in many applications, such as: expert finding, people connection finding, and information integration.

Specifically, in scientific bibliography systems, the name disambiguation problem can be formalized as: given a list of publications with all sharing an identical author name but might actually referring to different persons, the task then is to assign the publications to some different clusters, each of which contains publications written by the same person.

By viewing all publications as vectors of multiple dimensions and each publication as a point in the multiple-dimensional space, we can obtain a straightforward solution by using clustering methods to deal with this problem. Unfortunately, such simple solution seems not work well on this task. Our experiments also confirm this (cf. Section 6). The reason might be that publication data in the multiple-dimensional space are quite sparse and thus it is hard to find the correct centres of clusters and further hard to cluster each publication correctly.

Several research efforts have been placed to the problem. For example, [11] defines three types of features and applies a K-way spectral clustering algorithm for name disambiguation. [30] proposes an approach called DISTINCT by combining two similarity measures: set resemble of neighbor tuples and random walk probability. [31] proposes a semi-supervised framework to the problem based on hidden Markov random fields. We consider this problem from another angle. Our basic idea is to find the atomic clusters and then use the atomic finding results to help improve the traditional clustering algorithms. The atomic cluster means that publications in the same atomic cluster must be correctly grouped (high precision) but might be further grouped in the process of clustering (possible low recall). Intuition behind this idea is that we can use atomic cluster finding results to ease the problem, so as to improve the performance of clustering methods for name disambiguation.

Therefore, questions arising here include: 1) how to accurately find the atomic clusters, as it will be critical to the following clustering process? 2) how to combine the found atomic clusters into the traditional clustering algorithms?

In this paper, we first present the concept of atomic cluster. We propose using a bias classification method for finding the atomic clusters for a disambiguation problem. Features were defined for the classifier. Moreover, we examine the contributions of different kinds of features in training the classifier. Then, we describe how to integrate the atomic clusters finding results into the traditional algorithms. Specifically, we integrate the atomic clusters into two state-of-the-art clustering algorithms: Hierarchical clustering and K-means. We have found that those clustering methods can benefit a lot from our method.

We applied the obtained disambiguation process, consisting of atomic clusters finding and combination of the finding results into existing clustering algorithms, to a dataset collected from Arnetminer.org. The data set contains 1,250 publications of 16 different author names in total. Our experiment results show that

using atomic clusters can significantly improve the performance of disambiguation by Hierarchical clustering and K-means clustering (about 8% and 27% in terms of F1-measure respectively).

Our contributions in this paper include: 1) proposal of an atomic cluster finding method; 2) combination of the atomic cluster method into the traditional clustering algorithms; and 3) empirically verification of the proposed methods.

The rest of this paper is organized as follow: Section 2 reviews the related work. Section 3 describes the overview of our approach of using atomic clusters to improve the performances of two kinds of basic clustering methods. Section 4 proposes our method for finding atomic clusters. Section 5 presents two implementations of name disambiguation based on the atomic cluster finding results. Experimental results are presented in section 6. Finally, we conclude the paper in section 7.

2. Related work

Several approaches have been proposed for name disambiguation in different domains, for example, disambiguation on Citations [10] [11] [26] [30], Web Pages [1] [21], Email data [22], Internet Movie Database [19] and so on.

[11] proposed an unsupervised learning approach using K-way spectral clustering method. They studied two types of feature weight, the usual "TFIDF" and the normalized "TF" ("NTF"). Then, they calculate a Gram matrix for each name dataset and apply K way spectral clustering algorithm to the Gram matrix to get the result. In [26], a search engine based clustering method is proposed. They represent the features of each citation as relevant URLs from search engine and weighted it by its IHFs. Then they compute the pair wise similarity of two citations using cosine similarity and perform a hierarchical agglomerative clustering to derive the final clusters.

Two supervised methods are proposed in [10]. This paper investigates two supervised learning approaches to disambiguate authors in the citations. One approach uses the Naive Bayes probability model, a generative model to capture all authors' writing patterns using only positive training citations; the other uses Support Vector Machines(SVMs) and the vector space representation of citations, a discriminative model learning from both positive and negative training citations the distinction between different authors' citations. Therefore, when there is a new citation, both two models will predict whether it belongs to a certain author. However, one drawback of supervised methods is its scalability when using features like this. It is also impractical to train thousands of models for all individuals in a large Digital Library. See also [6].

[31] proposes a constraint-based probabilistic model for semi-supervised name disambiguation. They formalize the name disambiguation problem in a constraint based probabilistic framework using Hidden Markov Random Fields. Then define six types of constraints and employ EM algorithm to learn different distance metric for different persons.

Some other methods have also been proposed based on graphical theory. For example, Chen et al. [4] propose a graphical approach for entity resolution. They construct and analyze the entity-relationship graph constructed for the dataset and propose a method to measures the degree of interconnectedness between nodes in the graph, which is verified to be effective to improve the quality of entity resolution. Yin et al. [30] propose an approach called DISTINCT by combining two similarity measures: set resemble of neighbor tuples and random walk probability. They propose a method to weight different types of links in the graph. [24] adapt the multi-level graph partition technique to solve the large-scale name disambiguation problem.

On the other dataset, [1] tries to distinguish web pages to different individuals with the same name. It extends the problem from one person to a group of people who are related to each other, and identifies all their Web presence simultaneously. They present two unsupervised frameworks: one was based on link structure of the Web pages; the other used Agglomerative/Conglomerative Double Clustering method. [22] solves the name disambiguation problem in email data. They extend similarity metrics for emails embedded in graphs via a lazy graph walk. The re-ranking schemes based on the graph-walk similarity measures are demonstrated to work well. In [19], they propose a method which employs a less strict similarity measures by using random walks between ambiguous observations on a global social network. This similarity can provide more robust disambiguation capability rather than exact one.

3. Our approach overview

In this section, we first give an overview of our approach and then introduce the details of each step in the approach.

Our preliminary experiments and analysis show that one difficulty in name disambiguation is that data points (publications) are quite sparse in the feature space. Some publications might be located far away from the target cluster centres, which makes it difficult to obtain good performance by directly using the tradition clustering methods. By further analyzing the data, we have found that there are strong relationships between some points. For example, two papers have the same co-author and were published at the same conference. If such kind of points can get together as

atomic clusters, they will feed the clustering algorithm a better initial condition. By the atomic cluster, we mean a cluster of publications that have strong relationships/similarity with each other and will not be split in the following clustering process. Based on this observation, we propose an approach of atomic cluster finding for name disambiguation.

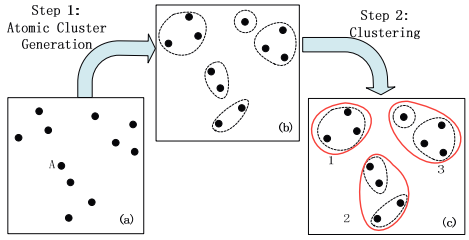

Fig. 1 Processing flow in our disambiguation approach. (a) Original Points (b) Atomic Clusters (c) Disambiguation results
Note: each point represents a publication.

Fig. 1 shows the processing flow of our approach for name disambiguation. The process consists of two steps: atomic cluster finding and disambiguation.

In the first step, we intend to find publications which have 'strong' relationships/similarities (which can be shown from (a) to (b) in Fig. 1). We output the strongly related points as atomic clusters for the later processing. To determine how strongly two publications are related to each other, we utilize a bias classification based method.

In the second step, we perform the disambiguation based on the atomic cluster finding results (as shown from (b) to (c) in Fig. 1). We take atomic clusters as the input for a clustering algorithm (e.g., Hierarchical clustering or K-means clustering) and output the final clustering results as the disambiguation results.

4. Atomic cluster finding

In this step, the main purpose is to use some features which contain information that can strongly indicates whether the publications are written by the same person. In our approach, we use a supervised classifier to generate the atomic clusters. The algorithm used in this step is described in Algorithm 1.

The input of the algorithm is a list of publications having the author name we are going to disambiguate. We sort all publications by their publish year (for the publications which publish in the same year, we simply use the alphabetical order), and then feed them to the algorithm one by one. The key in the algorithm is that a bias-classifier is used to determine if a publication should be grouped into an existing atomic cluster or a new cluster. We call it bias-classifier because we want to have a classifier that has high precision, which means that if a publication is assigned to cluster c_j, it must be highly possibly correct. Otherwise, we prefer to create a new cluster. In this way, we can have a high probability to keep the atomicity for each cluster.

Algorithm I. Algorithm of atomic cluster generation

Input:
A list of publications which all share same author name
Output:
A list of atomic clusters
Initialization:
Sort the publications by the order of increment of their publisher year. Create an empty atomic cluster list **c**={Ø}.
Iteration:
1. For each publication p_i, in $\{p_1, p_2, \ldots, p_m\}$ 2. For each cluster c_j in the atomic cluster list **c** //use a classifier to predict if paper p_i should be grouped into c_j. 3. If Bias-Classifier(p_i, c_j)>0 then 4. assign the publication into the cluster c_j; 5. End If 6. End For 7. If no cluster assigned with publication p_i; 8. create a new cluster c_k with p_i and add c_k to the list **c**; 9. End If 10. End For

The intuition behind the algorithm stems from observations on how human beings disambiguate publications: (1) people would like to start from the first published paper, when there is no ambiguity problem because there is only one paper; (2) by going on checking more published papers, people always would like to adapt the "easy first" and "high-confidence first" strategy, that is people first cluster the papers that she/he has high confidence with; (3) for the rest papers that might be somehow difficult, people would adapt some compromising method. Our algorithm matches with the three points above. We sort the papers by their published year (1), find atomic clusters (2), and employ traditional clustering algorithm to find the final disambiguation results (3).

Correspondingly, we see that the bias-classifier here play a role as a human being finds the paper clusters of "high-confidence". The classification based method consists of two stages: training and prediction. We use human labeled data to train a bias-classifier and use the classifier for the later prediction.

4.1. Bias classifier

The difference in the classifier from conventional classification problem is that we view a pair of paper-cluster (p, c) as an instance. Features are defined based on relationships of each pair. If the prediction result for a pair by the classifier is positive, we say that the paper should be grouped into the atomic cluster c.

As the classifier, we use AdaboostM1 [8] [9] with several adaptations. In our system, we adapt a variation of AdaBoost as a bias classifier, in that we want to find

atomic clusters with high precision (but not necessary high recall). Specifically, in the learning algorithm, each weak classifier is defined as a single layer perceptron. (We selected perceptron due to its easy implementation and that its precision and recall can be easily balanced.) AdaBoost minimizes a quantity related to the classification error. In our setting, we want to minimize the number of false positive with tolerating the possible false negative. In order to bias the classifier, we introduce a related notion of asymmetric loss Asy_Loss [29] :

$$Asy\_Loss = \begin{cases} \sqrt{k}, & \text{if } y_i = 0 \text{ and } C(x_i) = 1 \\ \frac{1}{\sqrt{k}}, & \text{if } y_i = 1 \text{ and } C(x_i) = 0 \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

where $C(x_i)$ is the class assigned by the boosted classifier. In this way, by tuning the value of k, a negative point classified as a positive can be penalized much more than a positive point as a negative. (In our experiments, we empirically set it as 1.5.)

The advantage of using AdaBoost and perceptron as a training scheme of classifier is that we can easily tune the parameters to make the training process bias to high precision.

4.2. Features

In total, we define three types of features. All of the feature information can be easily obtained from online digital libraries such as ArnetMiner.org, DBLP, and CiteSeer. All features are defined for each pair of paper and cluster. The features are defined based on relationships of the pairs (p, c), which makes them quite general and can be used for any disambiguation task. For simplifying the description, we give several notations first.

Let the principle author denotes the author name to be disambiguated. Let p denotes a paper, c denotes a cluster of papers, a denotes an author, function $a(p)$ denotes a set of authors of paper p excluding the principle author, i.e. $\{a_1, a_2, \ldots\}$ (likewise for $a(c)$), $N_c(a_i)$ denotes the occurring time of a_i in the cluster c. Similarly, we define $w(p)$ to represent the set of words in paper p and $N_c(w_j)$ to represent the occurring time of word w_j in the cluster c.

4.2.1 Co-author features

Co-author relationship is very important information for name disambiguation. Given a pair of paper and cluster, we define three types of co-author features.

The first feature is defined as $|a(c) \cap a(p)|$, representing the number of co-occurring authors between the paper and the cluster.

The second feature is defined as $\max_{a_i \in a(p) \cap a(c)} N_c(a_i)$, which essentially represents how frequent an co-author a_i in paper p occurs in the cluster c.

The other feature is defined as $\frac{\max_{a_i \in a(p) \cap a(c)} N_c(a_i)}{\max_{a_i \in a(c)} N_c(a_i)}$, a smoothingly normalized form of the second feature.

For example, for disambiguating the name "Michael", we have a paper with co-authors "Tom, Jerry, Jack" and a cluster with co-authors "Tom (2), Jerry (3), Alice (5)" (the number in the round brackets denotes the number of the co-author appearing in the cluster). Then the value of the first feature is 2 (|{"Tom", "Jerry"}|); the value of the second feature is 3 (N_c("Jerry")); and the value of the third feature is 0.6.

4.2.2. Title features

We define three similar title features as those of author features. The first feature is $|w(c) \cap w(p)|$; the second feature is $\max_{w_j \in w(p) \cap w(c)} N_c(w_i)$; and the third feature is $\frac{\max_{w_j \in w(p) \cap w(c)} N_c(w_j)}{\max_{w_j \in w(c)} N_c(w_j)}$.

4.2.3. Published venue features

The feature represents the similarity between the published venue of the paper and the set of published venues in the paper cluster. The similarity is defined as:

$$sim(p,c) = \begin{cases} 1.0, & if\ (v(p) \in v(c)) \\ \max_{v_i \in v(c)} sim(v_p, v_i), & otherwise \end{cases} \quad (2)$$

where $v(p)$ is the published venue of paper p and $v(c)$ is the set of published venues in cluster c; $sim(v_p,v_i)$ denotes the similarity between two published venues.

For calculating the similarity $sim(v_p,v_i)$, we first generate a vector by viewing all authors who published papers at v_p or v_i as features (the authors are obtained from Arnetminer.org), and then compute the cosine similarity of the two vectors.

5. Disambiguation

Based on the atomic clusters generated from the previous step, we can apply any kind of clustering method to generate the final clustering result, which is viewed as the disambiguation result. In this paper, we employ two clustering methods: Hierarchical clustering and K-means clustering method. More advanced clustering methods might be helpful for improving the disambiguation performance. For further work, we will try more advanced clustering methods. In this work we focus on investigating if/how the atomic clustering idea can help the disambiguation task.

We make several adaptations in the original clustering methods for utilizing the results of atomic cluster finding.

5.1. Hierarchical clustering with atomic clusters

In this paper, we use the agglomerative Hierarchical clustering method which runs in a bottom-up process. The algorithm starts by placing each original point into a single cluster. In each of iteration, the clustering

algorithm calculates distance between each two clusters, and merges the pair of clusters into a larger one which has the smallest distance of all the pairs. The process terminates when obtaining a pre-defined number of clusters.

We apply the Hierarchical clustering method on the atomic clusters generated from the previous step and set the real author number as the cluster number. In essence, we replace the initializing points of single publication with the result of atomic clusters finding. Then in each iteration, we calculate the similarity between two of clusters c_1 and c_2 using Equation (3), and combine the two atomic clusters which have the highest similarity.

$$Sim\_Cluster(c_1, c_2) = \frac{\sum_{p_1 \in c_1, p_2 \in c_2} Sim\_Pub(p_1, p_2)}{|c_1| \cdot |c_2|} \quad (3)$$

In Equation (3), p_1 and p_2 are the publications respectively belong to c_1 and c_2. For the similarity between two publications, we employ the vector space model to represent each publication and utilize the cosine measurement to calculate the similarity. Finally, the obtained clustering results will be taken as the disambiguation results.

5.2. K-means clustering with atomic clusters

K-means is another state-of-the-art algorithm for the clustering problem. Given the expect number of cluster k, the algorithm proceeds as follows: first, it selects k atomic cluster as "initial points" according some strategy, with each of them representing the centre of a cluster. Then in iterations, the algorithm alternately performs: (1) assignment of the rest atomic cluster to its nearest cluster according to the distance from the cluster centre; (2) calculation of the new cluster centre based on the mean of all points assigned to each cluster. This process continues until convergence (no point changes its assignment).

Differing from the traditional K-means, in iterations, we always view the atomic cluster as an inseparable "point" as in the traditional method. Thus, the assignments and distance calculation are based on atomic cluster. Finally, again we can obtain k clusters, which are viewed as the disambiguation results.

5.3. Features

We define features for the clustering algorithm. For both of the two clustering algorithms, we define each word occurring in paper titles, paper authors, and paper published venues as a binary feature.

6. Experiment

6.1. Data sets

To evaluate our proposed method, we created a dataset from the Arnetminer system (http://arnetminer.org). This dataset includes 16 real person names with their published 1,250 papers. For these names, some only have a few persons. For example "Jie Tang" and "Kuo Zhang" only represent two different persons respectively, and "Chang Cheng" three. However, some other names seem to be popular. For example, there are 16 different "Gang Wu" and 22 different persons named "Yi Li". Table 1 and Table 2 show statistics of the data set.

Table 1. Statistics of the data set

Name	*#Pub*	*#Per*	*Name*	*#Pub*	*#Per*
Ajay Gupta	26	4	Yi Li	42	21
Jing Zhang	54	25	Bing Liu	130	11
Rakesh Kumar	61	5	Bin Yu	66	12
Lei Wang	109	36	Jie Tang	21	2
Kuo Zhang	6	2	Wei Wang	305	80
Michael Wagner	44	12	Gang Wu	40	16
Jim Smith	33	5	Wen Gao	286	4
Cheng Chang	12	3	Hui Fang	15	3

Table 2.Detailed statistics of two author names

Name	*Affiliation*	*#Public-ations*
Wen Gao (286/4)	Chinese Academy of Sciences	70
	The College of William and Mary in Virginia, USA	2
	Thomson Inc.Princeton	1
	Purdue Univ.	1
Jing Zhang (54/25)	Shanghai Jiao Tong Univ.	6
	Alabama Univ.	8
	Univ. of California, Davis	4
	Carnegie Mellon Univ.	5

Five human annotators conducted annotation for disambiguating the names. A spec was created to guide the annotation process. Each paper is labelled with a number indicating the actual person. The labeling work was carried out based on the publication lists on the authors' homepages, affiliations and email addresses on Web databases (e.g., ACM Digital Library). For further disagreements in the annotation, we conducted "majority voting".From the Table 1 and Table 2, we can see that the distribution of actual person for each name is extremely unbalanced. For example, there are in total 286 papers authored by "Wen Gao" with 282 of them authored by Prof. Wen Gao from Institute of Computing at Chinese Academy of Science. Only four papers are authored by the other three "Wen Gao". This also reveals that the problem is some kind different from the traditional clustering problem and the existing methods may not be effective on it.

6.2. Evaluation measures

We use pair-wise measures to evaluate our name disambiguation method and compare the results with baseline methods. The pair-wise measures are based on the traditional information retrieval measures, adapted for evaluating disambiguation by considering the same-assigned pairs. The definitions of these measures are shown as follows:

$$Pairwise\_Precision = \frac{\#PairsCorrectlyPredictedToSameAuthor}{\#TotalPairsPredictedToSameAuthor}$$

$$Pairwise\_Recall = \frac{\#PairsCorrectlyPredictedToSameAuthor}{\#TotalPairsToSameAuthor}$$

$$Pairwise\_F1-measure = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

6.3. Experimental design

We used the Hierarchical clustering and K-means clustering without atomic cluster finding as the baselines. We employed the clustering algorithms implemented in Weka 3.5.6 [1]in our experiments.

As the atomic cluster finding needs training data. We used the cross validation for evaluation. We divided the dataset into four sets, and in each time, chose three of them as training set and the rest as the test set. On each test data, we compared the results obtained by the two clustering methods with and without atomic clusters finding. In the experiment, we empirically set the real author number as termination condition of the clustering iterations.

6.4. Experimental result

6.4.1. Hierarchical clustering + atomic clusters

We evaluated the performances of using hierarchical clustering for name disambiguation with (AHC) and without (HC) atomic clusters. Figure 2 shows the experimental results. We see that our atomic cluster based approach indeed help improve performances on most of the names against the baseline method. Averagely, the improvement is 8% in terms of pairwise-F1-measure.

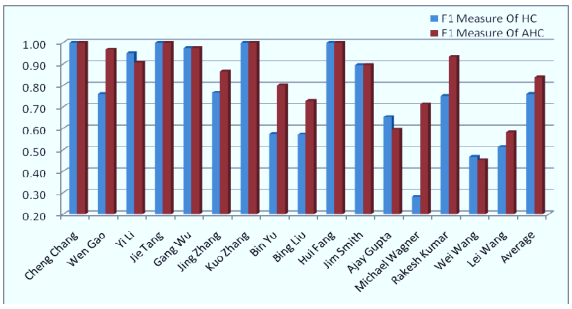

Fig. 2 Performances of using hierarchical clustering for name disambiguation with (AHC) and without (HC) atomic clusters.

We can also see that on some names, the improvements are limit while on some other names, the improvements are significant. By further analysing the results, we have found that generally when the number of real persons for a name is small (e.g., 2-5), the naive hierarchical clustering seems to perform well such as, "Jie Tang", "Cheng Chang", and "Hui Fang". However, when number of real persons is large (e.g., >10), the naive hierarchical clustering seems to not be able to accurately capture the distribution of the papers. The found has a straightforward explanation: with the number of real persons increasing, the border of each cluster also becomes more blurry, which makes the naive hierarchical clustering algorithm difficult to distinguish the clusters from each other. The atomic cluster finding idea can be viewed as a method of conversion of the distribution space of papers. In the new space, the clusters would be easier to be separated.

Finally, we have to say in a few cases, e.g., "Wei Wang", even with the atomic clustering results, it is still difficult to obtain a good disambiguation result. Several "Wei Wang" work on the same research topics such as "data mining" and have the same co-author, even work in a same university.

6.4.2. K-means clustering + atomic clusters

We evaluated the performance of using K-means clustering for disambiguation with and without atomic clusters. Figure 3 shows the experimental results. For K-means with atomic clusters, we have tried different strategies for selecting the initial centers. For the first one (we called K-means(R)), we use random selection. For the second one (we called K-means(H)), we use as the initial centers the K atomic clusters that have the averagely largest distance from all other clusters. For the third one (we called K-means(L)), we use as the initial centers the K atomic clusters that have the averagely smallest distance from all other clusters. For the random selection strategy, we have tried 100 times of random selection and give the average performance.

As we can see from the Figure 3, all the different strategies of using the atomic clustering results improve the performance of K-means more than 20%.

It can be also seen that the improvements on different names are different. According to our analysis, we have found the improvements depend on the distribution of features for each name. Naive K-means is based on the assumption that the data is generated from a gauss distribution. This implies that if the data distribution of a name (e.g., "Cheng Chang") can fit a gauss distribution well, our atomic finding method may have limited contributions to the final results. Otherwise, the atomic finding essentially convert the clustering problem from the data points space to a atomic cluster space that would be more fit to the gauss distribution.

6.4.3. Distribution analysis

Figure 4 shows three typical feature distributions in our data sets. The graphs were generated using a dimension reduction method described in [3]. With this method, the high dimension points can be described in two dimensions, and the distance between each two points is similar to the distance in original space. The three typical distributions are: (1) publications of

[1] http://www.cs.waikato.ac.nz/ml/weka/

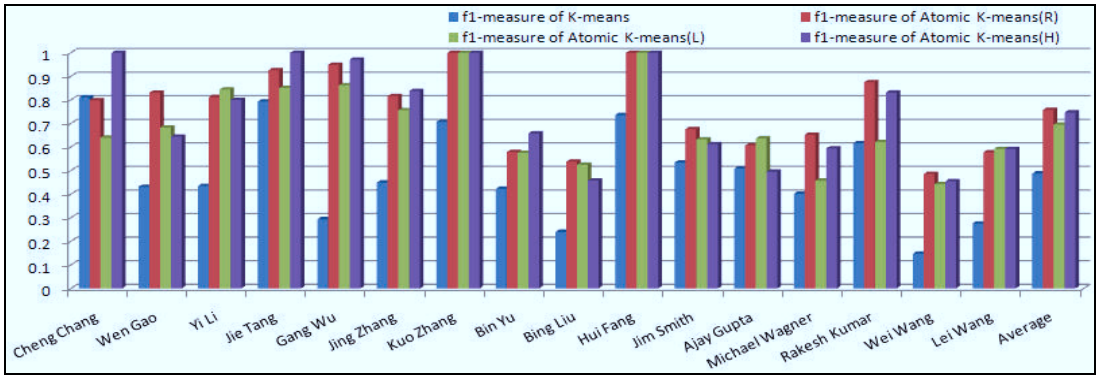

Fig.3 Performances of using K-means clustering for name disambiguation with (Atomic K-means) and without (K-means) atomic clusters.

different persons are clearly separated ("Hui Fang", in Figure 4 (a)). Name disambiguation on this kind of data can be solved pretty well by both naive hierarchical clustering method and our approaches; (2) publications are mixed together, however, there is a dominate author who writes most of the paper (e.g., "Bing Liu", in Figure 4 (b)); our approach of hierarchical clustering with atomic clusters can achieve 75% of average recall; and (3) publications of different authors are mixed ("Jing Zhang" & "Yi Li", in Figure 4 (c) and (d)). Our method with atomic clusters can achieve a performance of 80% in terms of F1-measure. There are also some other cases with more complex distributions, for example "Wei Wang" and "Lei Wang". In such cases, even with atomic clusters, both Hierarchical clustering and K-means cannot obtain satisfactory results.

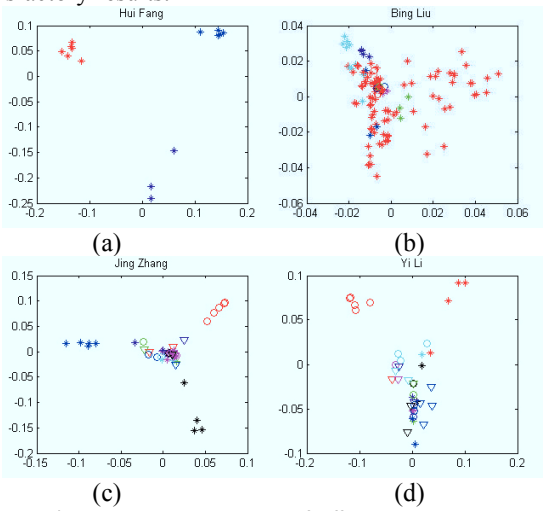

Fig.4 Publications distribution of different person names in dimension reduction space

We also track the process of algorithm. Take Bing Liu as an example, 51 distinct sets of publication can be get according to different set of associated co-authors. But finally, there are only 11 different persons named Bing Liu in the dataset. It shows that although author relationship feature can achieve high precision, recall is too low in many name set. The second step of clustering can boost recall of final result.

6.4.4. Atomic cluster finding

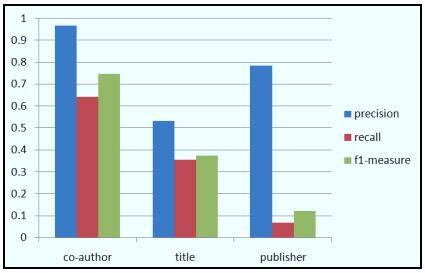

Fig.5 Performances of using different features

We conducted an experiment to evaluate the performance of the atomic cluster finding. We tested the approach with different types of features. Figure 5 shows the performances with different features. We see that averagely the co-author features result in the best performance in terms of both precision and recall. But sometimes, the title features seem to useful. For example, for "Bing Liu", the precision and recall of using co-author features are "1.00" and "0.39", respectively; while using title features, the precision and recall are "0.63" and "0.59". The title features can greatly improve the recall in this case. With the publisher features only, we can achieve a high precision but very low recall.

7. Conclusions

In this paper we have investigated the problem of name disambiguation. We have proposed an approach of using atomic clusters to improve the performance of name disambiguation, and presented two implementations by extending two basic clustering methods. We applied the approach to a real-world dataset from ArnetMiner.org. Experimental results show that our approach of using atomic clusters

finding for name disambiguation can performs significantly better than baseline clustering methods. Experiments also unveil that co-author features are the most important features when generating the atomic clusters.

8. Acknowledgement

The work is supported by the National Natural Science Foundation of China (90604025, 60703059), Chinese National Key Foundation Research and Development Plan (2007CB310803), and Chinese Young Faculty Research Funding (20070003093).

. References

[1] R. Bekkerman and A. McCallum. Disambiguating Web Appearances of People in a Social Network, In Proc. of WWW'2005, pp. 463-470, ACM Press, 2005.

[2] I. Bhattacharya, L. Getoor. Deduplication and Group Detection using Links. In Proc. of Link KDD'04, USA, 2004.

[3] D. Cai, X. He, and J. Han. Spectral Regression for Dimensionality Reduction. Department of Computer Science Technical Report No. 2856, University of Illinois at Urbana-Champaign.

[4] Z. Chen, D. V. Kalashnikov, and S. Mehrotra. Adaptive Graphical Approach to Entity Resolution. In Proc. of JCDL'2007, pp. 204-213.

[5] D. Cohn, R. Caruana, and A. McCallum. Semi-supervised Clustering with User Feedback. Technical Report TR2003-1892, Cornell University, 2003.

[6] E. Elmacioglu, Y. Tan, S. Yan, M. Kan, D. Lee. PSNUS: Web People Name Disambiguation by Simple Clustering with Rich Feature. In Proc. Of 4th Int'l Workshop on Semantic Evaluation (SemEval), 2007.

[7] M. Ester, R. Ge, B.J. Gao, Z. Hu, and B. Ben-Moshe. Joint Cluster Analysis of Attribute Data and Relationship Data: the Connected K-center Problem. In Proc. of SDM'2006.

[8] Y. Freund, R. Schapire, A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting, In Proc. of EuroCOLT '95, pages 23-37, 1995.

[9] Y. Freund, R. Schapire, Experiments with a New Boosting Algorithm, In Proc. of ICML'1996.

[10] H. Han, L. Giles, H. Zha, C. Li, and K. Tsioutsiouliklis. Two Supervised Learning Approaches for Name Disambiguation in Author Citations. In Proc. of JCDL'2004, USA, 2004, 296 – 305.

[11] H. Han, H. Zha, and C.L. Giles. Name Disambiguation in Author Citations using a K-way Spectral Clustering Method. In Proc. of JCDL'2005, USA, 2005, 334 – 343

[12] H. Han, W. Xu, H. Zha, C. Giles. A Hierarchical Naïve Bayes Mixture Model for Name Disambiguation in Author Citations. In Proc. of SAC'05, 2005.

[13] J. Han, M. Kamber. Data Mining, Concepts and Techniques. Second Edition, 2006.

[14] J. Huang, S. Ertekin, C. Giles. Efficient Name Disambiguation for Large-Scale Databases. In Proc. of PKDD, 2006.

[15] J. Huang, S. Ertekin, C. Giles. Fast Author Name Disambiguation in CiteSeer". IST Technical Report No. 0019, the Pennsylvania State University, 2006

[16] L. Kaufman and P. Rousseeuw. Finding Groups in Data: An Introduction to Cluster Analysis. New York: Wiley, 1990.

[17] D. Lee, B. On, J. Kang, S. Park. Effective and Scalable Solutions for Mixed and Split Citation Problems in Digital Libraries. In Proc. of IQIS, 2005.

[18] X. Li, P. Morie, and D. Roth. Identification and Tracing of Ambiguous Names: Discriminative and Generative Approaches, In Proc. of AAAI'2004, pp. 419-424

[19] B. Malin, E. Airoldi, K. Carley. A Network Analysis Model for Disambiguation of Names in Lists, In Proc. of SIAM Workshop, 2005.

[20] B. Malin. Unsupervised Name Disambiguation via Social Network Similarity. In Proc. of SIAM SDM Workshop, 2005.

[21] G. Mann and D. Yarowsky. Unsupervised personal name disambiguation. In Proc of CoNLL. Canada. 2003.

[22] E. Minkov, W.W. Cohen, and A.Y. Ng. Contextual Search and Name Disambiguation in Email using Graphs. In Proc. of SIGIR'2006, USA, 2006, 27-34.

[23] B. On, D. Lee, J. Kang and P. Mitra, Comparative Study of Name Disambiguation Problem using a Scalable Blocking-based Framework, In Proc. of JCDL'05, 2005.

[24] B. On, D. Lee. Scalable Name Disambiguation using Multi-level Graph Partition. In Proc. of SIAM Int'l Conf. on Data Mining (SDM), 2007.

[25] Y. Song, J. Huang, et al. Generative Models for Name Disambiguation. In Proc. of WWW, 2007.

[26] Y. F. Tan, M. Kan, and D. Lee. Search Engine Driven Author Disambiguation. In Proc. of JCDL'2006, USA, June 2006, 314-315.

[27] J. Tang, D. Zhang, and L. Yao. Social Network Extraction of Academic Researchers. In Proc. of ICDM'2007. pp. 292-301.

[28] P. Viola, M. Jones, Rapid Object Detection using a Boosted Cascade of Simple Features, In Proc. of CVPR, 2001.

[29] P. Viola, M. Jones, Fast and Robust Classification using Asymmetric AdaBoost and a Detector Cascade. In NIPS, 2002

[30] X. Yin, J. Han and Philip S. Yu, Object Distinction: Distinguishing Object with Identical Names, in Proc. of ICDE'07, Turkey, 2007.

[31] D. Zhang, J. Tang, J. Li, and K. Wang. A Constraint-Based Probabilistic Framework for Name Disambiguation. In Proc. of CIKM'2007. pp. 1019-1022