Combining Machine Learning and Human Judgment in Author Disambiguation*

Yanan Qian[1], Yunhua Hu[2], Jianling Cui[3], Qinghua Zheng[1], Zaiqing Nie[2],
[1] MOE KLINNS Lab and SKLMS Lab, Xi'an Jiaotong University, Xi'an 710049, P.R.China
[2]Microsoft Research Asia, No. 5 Danling Street, Haidian District, Beijing 100080, P.R China
[3] College of Software, Nankai University, Tianjin 300071, P.R China
ynqian@yahoo.cn, {yuhu, znie}@microsoft.com, jianlingcui@126.com, qhzheng@mail.xjtu.edu.cn

ABSTRACT

Author disambiguation in digital libraries becomes increasingly difficult as the number of publications and consequently the number of ambiguous author names keep growing. The fully automatic author disambiguation approach could not give satisfactory results due to the lack of signals in many cases. Furthermore, human judgment on the basis of automatic algorithms is also not suitable because the automatically disambiguated results are often mixed and not understandable for humans. In this paper, we propose a Labeling Oriented Author Disambiguation approach, called LOAD, to combine machine learning and human judgment together in author disambiguation. LOAD exploits a framework which consists of high precision clustering, high recall clustering, and top dissimilar clusters selection and ranking. In the framework, supervised learning algorithms are used to train the similarity functions between publications and a clustering algorithm is further applied to generate clusters. To validate the effectiveness and efficiency of the proposed LOAD approach, comprehensive experiments are conducted. Comparing to conventional author disambiguation algorithms, the LOAD yields much more accurate results to assist human labeling. Further experiments show that the LOAD approach can save labeling time dramatically.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Digital Libraries; D.2.8 [**Database Management**]: Database applications

General Terms

Algorithms, Experimentation

Keywords

Author Disambiguation, User Contribution, Human Judgment

1. INTRODUCTION

*This work was conducted at Microsoft Research Asia when the first author visited there.

In digital libraries, author disambiguation becomes increasingly difficult as the number of publications continuously grows and so does the number of authors with ambiguous names. Previous research aimed to solve the problem in a fully automatic way. Unfortunately, in current digital libraries, the information about publications is always insufficient, and the rapid increase of digital content also makes the manual creation of metadata highly costly. Fully automatic author disambiguation could not give satisfactory results in this situation. How to leverage limited human efforts to solve the author disambiguation problem becomes more and more important. In this paper, we propose a Labeling Oriented Author Disambiguation approach to solve the problem in a novel and efficient way.

We have been developing Microsoft Academic Search [1] to help scientists and students locate research publications. The publications include papers, books, patents, etc. By integrating information about tens of millions of publications from publishers, CrossRef and the web, we face serious author disambiguation problem. For example, in our system, we find that there are 117 different authors (before 24th Octber, 2010) named 'Lei Zhang'. Due to the incomplete and inconsistent nature of the data, it is impossible for fully automatic disambiguation algorithms to correctly assign the publications to all different 'Lei Zhang's.

A user contribution feature has recently been developed to change the situation. The feature supports users to manually correct the results output by fully automatic disambiguation algorithms. However, we find the manual labeling process is still very time-consuming and tedious. Given that in the large scale digital libraries the number of ambiguous names is huge, it is really not suitable for humans to correct errors made by the automatic algorithm.

In this paper, we propose a novel Labeling Oriented Author Disambiguation approach, called LOAD, to conquer the author disambiguation challenges in construction of digital libraries (or academic search engines) together with users. The key ideas are: 1) Find High Precision Clusters (HPCs) for each author to change the labeling granularity from individual publications to clusters; 2) Cluster the found HPCs into High Recall Clusters (HRCs) to place all publications of one author into the same cluster. Thus the labeling for one author can be limited within a narrow scope; 3) Select the best HPC in each HRC as the starting point to cover different important authors for labeling. Other similar HPCs can be re-ranked for further labeling.

To achieve the goals of the proposed LOAD approach, machine learning algorithms are used. First, we exploit a supervised learning algorithm to get similarity between publications, i.e., judge whether two publications belong to the same author. More specifically, two classifiers for HPC and HRC are trained respectively and various rich features are investigated. On the basis of learned sim-

[1]http://academic.research.microsoft.com

ilarities, a clustering algorithm is further applied to generate final clusters. Comprehensive experiments are conducted to empirically verify the effectiveness of our proposed approach. One public dataset contains about 7,000 publications from DBLP and two internal datasets contains about 6,000 publications are used. Correct authors for all publications were labeled by manual. A typical conventional automatic author disambiguation algorithm is taken as the baseline. The results are obtained by applying the algorithm in the real-world academic search engine in which we do not know which names are ambiguous and which are unambiguous. Our algorithm achieves promising results on ambiguous names while keeps little harm to unambiguous names. For HPC, the experimental results show that it improves precision from 93.72% to more than 99% on average. For HRC, the recall is also improved from 75.43% to 91.85% on average. The results also show that the LOAD can improve the human labeling efficiency by about 10 to 30 times compared to the baseline.

The rest of the paper is organized as follows. Section 2 introduces related work. Section 3 describes motivation of our work. Section 4 explains the proposed approach and Section 5 presents experimental results. Section 6 concludes the paper with remarks while proposing future work.

2. RELATED WORK

Automatic Author Disambiguation for academic publications has gained continuous attention from research community for years. In this section, we will introduce related works of problem formulization, various frameworks and features used in automatic author disambiguation.

Most previous work formulized author disambiguation problem as a statistical learning problem. Han et al. formulated it as a clustering problem and solved it by a model-based k-means algorithm in [4] and by a K-way spectral clustering algorithm in [5]. They also viewed the problem as a classification problem in [6]. The main idea is to train a classifier to identify whether two publications belong to the same author. A graph partition based approach using Quasi-Clique is proposed by [2]. Treeratpituk and Giles [10] also proposed a random forest algorithm.

The framework for author disambiguation has also been studied comprehensively. On et al. [2] proposed a two-step framework which first divides publications into small blocks to reduce computational cost. Several work such as [1] focused on the blocking approaches. Fan et al. [3] proposed the graph-based framework for author disambiguation. However, performance issue in the graph framework is a big problem. Wang et al. [11] lightened the difficulty by changing clustering on papers to clustering on atomic clusters. But for digital libraries with millions of publications, graph based framework is still not the best choice. Huang et al. [7] proposed another two-step framework for large-scale author disambiguation. It learned the similarity function on pair of publications in the first step and performs clustering using the learned similarity in the second step. We adopt a similar framework and the difference is that we improve the framework by splitting its first step into high precision similarity learning and high recall similarity learning.

Features used in author disambiguation are also widely studied. Metadata including automatic extracted data are taken as features in many work [5, 10]. Tan et al. [9] introduced a useful feature named Inverse Host Frequency (IHF) by using search engine for author disambiguation. Song et al. [8] employed PLSA and LDA to get research topics for authors and then disambiguate them. All of the above work did not investigate the features related to high precision classification or high recall classification which are proposed in this paper.

3. MOTIVATION OF LOAD

3.1 Problems in Automatic Author Disambiguation

Fully automatic author disambiguation is infeasible due to the data quality problem and the data sparsity problem.

Data quality problem is very common in large scale digital libraries which often integrate data from different sources such as CrossRef, pdf files, and etc. However, such data often contain errors, conflict with each other, or have many variations. The data quality problem could confuse the automatic algorithms a lot and harm the disambiguation accuracy greatly.

Data sparsity problem means many publications lack necessary metadata information. For example, it is proved that email and affiliation are very useful for finding correct author. However, in our system, only about 10% publications have the email metadata and 23% have affiliation information. Without necessary signals, author disambiguation cannot be fully solved automatically.

3.2 Problems in Human Labeling after Automatic Disambiguation

One possible solution for fully solving the author disambiguation problem is to leverage human judgment to correct the errors of automatic algorithms. To valid the possibility, we perform a case study on authors named ‘Lei Zhang’. The algorithm proposed in [6] is used before human labeling. 117 different authors are split into 141 clusters. Three human labelers were asked to correct the results. Finally they spent 7 hours on average to finish the task. Part (a) in Figure 1 illustrates the situation for human labeling. The human labeling process is very time-consuming for several reasons: First, publications in each cluster are not pure enough and human labelers still need to check them one by one. For example, the most diverse cluster contains publications from 27 different authors. This process is nearly equal to the fully manual labeling. Second, as one author’s publications could be assigned to several different clusters, human labelers need to go through many clusters to find all results. For the ‘Lei Zhang’ from PolyU (Hongkong Polytechnic University), his publications are assigned to 8 different authors. So at least 8 clusters need to be checked one by one to finish the labeling.

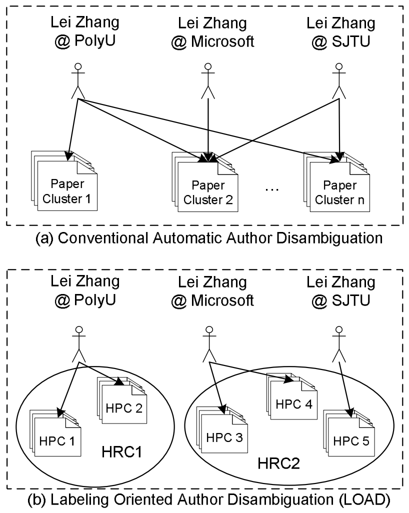

Figure 1: Comparison between Conventional Author Disambiguation and LOAD

From the discussions above, we can find that to fully solve the author disambiguation problem, we need to design an algorithm to assist human labeling efficiently from these perspectives: 1) Most ambiguous cases need to be solved by automatic author disambiguation before human labeling. 2) Human labeling need to be conducted on bigger granularity rather than on each individual publication. 3) Human labeling need to be limited within a narrow range rather than the whole publication list. 4) Human labeling need to be started from some good starting points.

4. OUR APPROACH

In this section, we propose an approach named LOAD for author disambiguation. First we provide an overview of the LOAD and how we formulate it. Then we study a key problem, i.e., the pairwise classification with rich features in LOAD.

4.1 Problem Description and Formulation

4.1.1 Description

We start from an example to show the framework in Figure 2. Step 0 is the initial state which shows several publications authored by ‘Lei Zhang’ at PolyU, ‘Lei Zhang’ at Microsoft, and ‘Lei Zhang’ at SJTU. In Step 1, all publications are clustered into 5 clusters. Notice that each HPC only contains publications of one author here. In Step 2, HPCs are further clustered into several HRCs. Publications of both ‘Lei Zhang’ at Microsoft and ‘Lei Zhang’ at SJTU are placed into the same cluster HRC_2. In Step 3, HPC_1 of ‘Lei Zhang’ at PolyU and HPC_4 of ‘Lei Zhang’ at Microsoft are selected as starting points for labeling. Human labeling starts from the results of Step 3. Once we labeled HPC_1 in HRC_1 as publications of ‘Lei Zhang’ at PolyU, we can easily find his rest publications from HPC_2 in the same HRC. Meanwhile, once we labeled HPC_4 as publications of ‘Lei Zhang’ at Microsoft, HPC_3 is ranked higher for labeling. When all his publications are labeled, we can find that the rest publications in HPC_5 belongs exactly to ‘Lei Zhang’ at SJTU. The output of Step 3 is also shown as Part (b) in Figure 1. Comparing to the confused cases shown as Part (a) in Figure 1, Part(b) is much easier for human labeling. The comparison shows the benefit of our proposed LOAD approach intuitively.

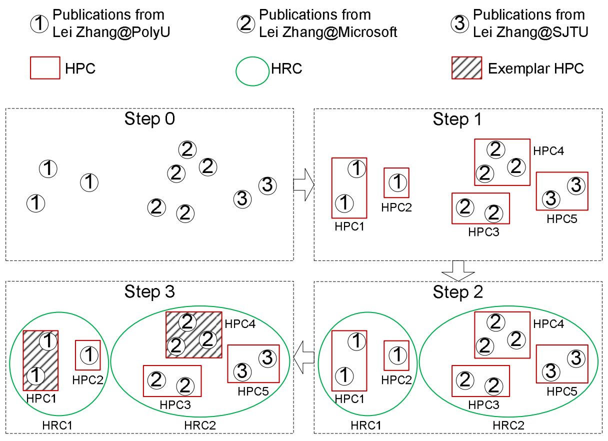
Publications from Lei Zhang@PolyU
Publications from Lei Zhang@Microsoft
Publications from Lei Zhang@SJTU
HPC
HRC
Exemplar HPC
Step 0
Step 1
Step 2
Step 3
HPC1
HPC2
HPC3
HPC4
HPC5
HRC1
HRC2

Figure 2: Working Process of Three Steps Framework

The three-step framework of LOAD is: (1) High Precision Clustering: It tries to place one authors’ publications into as fewer clusters as possible, while tries to keep all publications in each cluster written by the same author. Thus, human labeling can be conducted on clusters rather than on individual publications. (2) High Recall Clustering: It is conducted on the results of HPCs, it is different from HPC in two aspects: First, the goal of HRC is trying to put all publications of one author into the same cluster. Second, each HPC is totally contained in a HRC. (3) Top Dissimilar HPCs Selection and Ranking: after we get the HPCs and HRCs, we can select the best HPC in each HRC as the exemplar clusters for human labeling.

4.1.2 Formulation

Now we formulize the concepts and goals of LOAD in this section. Given a set of publications $X = \{x_1, x_2, \cdots, x_n\}$, each publication x_i has p authors $xa_i = \{xa_{i1}, xa_{i2}, \ldots, xa_{ip}\}$. Using the name blocking method described in [7], publications can be first clustered into blocks according to author name string similarity. So in each name block we only need to consider the disambiguation problem for one author name from each paper. We use $Y = \{y_1, y_2, \ldots, y_n\}$ to indicate the concerned author of related publications in X respectively, $y_i \in xa_i, i \in [1, n]$. Each y_i correspond to one of the real world authors $A = \{a_1, a_2, \ldots, a_h\}$. Then each step in LOAD can be formulized as follows.

High Precision Clustering

Assume n publications $X = \{x_1, x_2, \cdots, x_n\}$ are clustered into l clusters $C = \{c_1, c_2, \cdots, c_l\}$. The goal is to maximize the overall similarity V_{HPC} in Equation (1) while keeping the high precision of each cluster. The c_p and c_q here are HPC clusters.

$$\begin{gathered} V_{HPC} = \sum_{p=1}^{l} \sum_{x_i, x_j \in c_p} S\left(x_i, x_j\right) \\ s.t. \quad \forall p, x_i, x_j \in c_p, \quad y_i = y_j \qquad (1) \\ \forall i, j, p, q \max_{x_i \in c_p, x_j \in c_q, p \neq q} \left\{S(x_i, x_j)\right\} < \min_{x_i, x_j \in c_p} \left\{S(x_i, x_j)\right\} \end{gathered}$$

High Recall Clustering

The goal of step 2 is to place all publications of one author into only one cluster. It can be formulated in the similar way like the first step. We process high recall clustering by maximizing the similarity V_{HRC} in Equation (2) while ensuring the diversity between each cluster. The c_p and c_q here are the HRC clusters.

$$\begin{gathered} V_{HRC} = \sum_{p=1}^{l} \sum_{x_i, x_j \in c_p} S\left(x_i, x_j\right) \\ s.t. \quad \forall p, q, x_i \in c_p, x_j \in c_q, p \neq q, \quad y_i \neq y_j \qquad (2) \\ \forall i, j, p, q \max_{x_i \in c_p, x_j \in c_q, p \neq q} \left\{S(x_i, x_j)\right\} < \min_{x_i, x_j \in c_p} \left\{S(x_i, x_j)\right\} \end{gathered}$$

Top Dissimilar HPCs Selection and Ranking

For all HPCs in one HRC, we pick up the best HPC, i.e., HPC^* as an exemplar cluster for human labeling. We formally define the task in Equation (3). It means the HPCs with bigger size and higher overall inner similarity will have better chances of being selected. These exemplar clusters often corresponds to important authors for they often have more and correlated publications.

$$HPC^* = \arg\max_{c_p} \sum_{x_i, x_j \in c_p, i \neq j} S(x_i, x_j) \qquad (3)$$

Once people select one HPC, the rest HPCs in that HRC can be re-ranked according to the similarity with the labeled HPC. The similarity can be calculated in Equation (4).

$$S(c_p, c_q) = \sum_{x_i \in c_p, x_j \in c_q, p \neq q} S(x_i, x_j) \qquad (4)$$

Through the reranking approach, users need only to check whether top ranked HPC can be merged with labeled HPCs, which further reduces the human labeling efforts.

4.1.3 Learning Algorithms

The object function in Section 4.1.2 is a little difficult to learn, as our goal is to apply the algorithm in the large scale digital library, we need to find a less optimization but more efficient algorithm. In this section, we first analyze the formulations and simplify them into two sub-problems, the pairwise similarity learning problem and the clustering problem. Then we introduce the sub-problems respectively.

Formulations Simplification

We take Equation (1) as example to analysis how to simplify the optimization process. The goal of (1) is to place publications of the same author into the same cluster. At the same time, avoid to place publications of different authors into the same cluster. We propose to simplify the formulation into two steps by combining the approaches proposed by [2, 6, 7]. First, a similarity function between two publications is trained. Then, we can cluster the publications according to the similarity generated by the learned similarity function. Both Equation (1) and (2) can be approximated with these two steps. As explained in Section 4.1.1, HPC and HRC should be trained separately. Section 4.2 will discuss the details.

Pairwise Similarity Learning

Many previous work studied the problem of training a weighted similarity function [4, 5, 6, 7]. Our approach exploits the similar idea. The key problem is to find an adequate function which can represent the similarity/dissimilarity between the publication pairs. In the proposed approach, we view the similarity calculation between publication pairs as a classification problem. First, information about a publication pair is represented in a feature vector $\overrightarrow{x}$. Then a pairwise classification model is trained to judge whether the two publications are similar. We used the linear model showed in Equation (5) to compute the similarity for each paper pair. The weight $\overrightarrow{w}$ and threshold b are trained by SVM model through SVM-light toolkit [2].

$$f(\overrightarrow{x}) = <\overrightarrow{x}, \overrightarrow{w}> - b \tag{5}$$

Clustering

How to cluster publications efficiently in large scale digital libraries is a big challenge. In some previous work [11], the agglomerative clustering algorithm has been proved to be very effective and efficient. So we exploit it as our clustering algorithm. Yes we can also try more powerful clustering algorithms. However, to show the idea of our proposed approach, only this clustering algorithm is tried.

4.2 Pairwise Classification with Rich Features

In this section, we discuss in detail about how we conduct the high precision clustering and high recall clustering with rich features respectively. As explained in Section 4.1.3, the clustering algorithm used in both steps are the same. So we only focus on how to learn two classification models effectively here.

4.2.1 Features for High Precision Classification

Pairwise classification for HPC is to judge whether two publications are authored by the same person, so all features are defined via comparison of a pair of publications. In order to achieve high precision, we drill down features into different value scales. Two types of measures, i.e., SIF (Shared Item Frequency) and IPF (Inverse Publication Frequency) are also defined inspired by the idea of TF and IDF.

Suppose a publication has a list of metadata such as a list of coauthors and a list of references, we define SIF to represent the similarity of two lists I_1 and I_2:

$$SIF(I_1, I_2) = \frac{|I_1 \bigcap I_2|}{\sqrt{|I_1||I_2|}} \tag{6}$$

From another perspective, IPF is defined to discount the importance of common features. Suppose $A = I_1, I_2, \cdots, I_n$ is the set of all feature list, i is a feature that appears in one or several list, IPF_i is defined as follows:

$$IPF(i) = \log \frac{|A|}{|\{I_k : i \in I_k\}|} \tag{7}$$

The IPF to compare two lists is defined as the sum of shared IPF features.

$$IPF(I_1, I_2) = \sum_{i \in I_1 \bigcap I_2} IPF(i) \tag{8}$$

Referred to the above definitions, we adopted the strict comparison and IPF features of the metadata "name", "email", "affiliation" and "homepage" between two authors; also, the SIF and IPF features of "coauthor name", "coauthor email", "coauthor affiliation", "coauthor homepage", "title bigram", "reference" and "download link" are extracted. Besides, a binary feature "self citation" which represents whether one publication cited the other, or vice versa are extracted; the "publishing year" interval between two papers are also considered.

4.2.2 Features for the High Recall Classification

The high precision classifier aims at collecting high confidence signals while the high recall classifier aims at collecting as much signals as possible. The basic high recall features are in accordance with high precision features. Here, we only list the differences.

Email: We split the email into prefix and suffix by '@'. It is because some authors might change the email suffixes but tend to reserve the same prefixes. Besides, the email suffix might reflect the author's organization. When comparing, common email suffixes (e.g. @gmail.com, @yahoo.com) are removed.

Affiliation: The affiliation information is segmented and parsed into three groups: university, department and group name. We will compare these three groups of information separately. Some heuristic rules are defined to segment and parse the variations of affiliation information.

Homepage: Authors from the same organization often share the same domain for their homepage. Thus, 'homepage' is further split into domain and suffix for comparison.

Coauthor related feature: For the coauthor email, affiliation and homepage, similar parsing is performed to obtain more information.

Other features: for all other metadata, IPF features are removed to ensure a high recall.

5. EXPERIMENTS

5.1 Experiment Setup

5.1.1 Data Sets

Two internal data sets and one public data set are used in our experiments. Authors of publications were manually labeled by human experts.

CS data set: The CS(Case Study) data set is created when we conduct the case study for author disambiguation. Publications in this set are highly ambiguous. To avoid the case that the disambiguation algorithm might harm unambiguous authors, we also added some unambiguous authors here.

[2]http://svmlight.joachims.org/

UE data set: As mentions before, our system opens a feature for users to edit and correct author's publication list. All such user edited data are collected as UE(User Edited) data set.

DBLP data set: The DBLP (Digital Bibliography & Library Project) dataset is an open source data set[3]. There are 9,160 labeled publications in it. The publications without any metadata in our system are removed. After the cleaning, 7,434 publications are taken as the data set.

Basic statistics of three data sets are given in Table 1. As one author can have several different name variations, number of authors can be less than the number of names. UE is just the case.

Table 1: Statistical Information of Three Data Sets

Statistics	CS	UE	DBLP	*Total*
# of publications	1,322	4,624	7,434	*13,380*
# of names	54	1,199	739	*1,992*
# of authors	171	1,052	1,267	*2,490*
Avg/Max authors per name	3.96/117	1.01/6	1.71/17	*2.23/117*
Avg/Max names per author	1.25/4	1.15/5	1.00/2	*1.13/4*
Avg/Max publications per authors	7.73/323	4.40/131	5.89/129	*6.01/323*

5.1.2 Evaluation Metrics

As LOAD is to help users to edit the publication list, we evaluate it from several perspectives of user contribution.

First, we evaluate the results by using accuracy.

The evaluation measure is described in Equation (9).

$$Accuracy = \frac{\#\, fully\ correct\ clusters}{\#\, all\ clusters} \tag{9}$$

If one publication in the cluster is incorrect, the whole cluster will be taken as incorrect. For HPC cluster, if all publications in it belong to the same author, the cluster is counted as 'Correct', otherwise 'Incorrect'. For HRC cluster, if all authors in it have no publications in other clusters, this cluster is counted as 'Correct', otherwise 'Incorrect'.

Second, we evaluate the number of publications in HPCs and number of authors in HRCs.

For HPCs, to avoid the case that each HPC contains only one publication, we evaluate the number of publications in them. For HRCs, to avoid the case that all publications are placed into the same HRC, we evaluate the number of authors in them.

5.1.3 Baseline

We take the human labeling after conventional automatic author disambiguation as the baseline. In baseline, we reduce the two steps (clustering of HPCs and HRCs) in LOAD into one step, and then cluster the publications using the same algorithm in LOAD. Moreover, the baseline uses both HPC related and HRC related features. Strictly speaking, the baseline is more powerful than conventional automatic author disambiguation algorithms.

5.2 Evaluation of LOAD

5.2.1 Evaluate HPC and HRC

We first evaluate the accuracy by taking the whole cluster as a base unit. Results are shown in Table 2 and Table 3. Notice that the baseline methods in two tables are actually the same while the results are different because: Table 2 lists the results of finding "whether all publications in the cluster belong to the same author" while Table 3 lists the results of finding "whether all publications of authors can be find in the same cluster".

[3]http://dblp.uni-trier.de/xml/

Table 2: Accuracy Comparison of HPC and Baseline

Method	CS	UE	DBLP	*Average*
HPC	99.80%	100.00%	99.94%	*99.91%*
Baseline	92.95%	95.94%	92.29%	*93.72%*

Table 3: Accuracy Comparison of HRC and Baseline

Method	CS	UE	DBLP	*Average*
HRC	87.80%	95.38%	92.38%	*91.85%*
Baseline	71.88%	80.67%	73.73%	*75.43%*

From the results we can see that the accuracy of both HPC and HRC are greatly improved comparing to the baseline. In three data sets, HPCs archive nearly 100% precision. HRCs also have much better accuracy than baseline.

The average publication number in each cluster for HPC, HRC, and baseline is shown in Table 4. The results of HRC are comparable with those in the baseline. We also compared the average author number in each cluster for HPC, HRC, and baseline. The results are shown in Table 5. Average author number in HRC is 1.43. That is to say, labelers only need to label one or two authors in each HRC.

Table 4: Comparison of Number of Publications in Clusters

Method	CS	UE	DBLP	*Average*
HPC	3.38	2.27	4.22	*3.29*
HRC	8.81	4.39	5.94	*6.38*
Baseline	5.81	3.71	5.47	*5.00*

5.2.2 Evaluate Human Labeling Efficiency

To investigate how much labeling time can be saved through LOAD, we compare it with two other methods, i.e., fully manual labeling and labeling after conventional automatic disambiguation (baseline). For simplicity, the comparison is performed in a publication list with the same author name. We take the total number of publication as n, number of authors as m, reading time for each publication as r, comparing time for each publication as c, d_i is the discount factor in the i^{th} method.

In the first approach, human labelers need to go through all the publications. For each publication, they first need to read the basic information, and then compare the current publication's authorship with previous labeled authors. The reading time can be estimated as $O(r \cdot n)$ and comparing time as $O(c \cdot n \cdot m \cdot d_1)$ here.

In the second approach, as one cluster may contain several authors' publications while an author's publications may be assigned to several different clusters, labelers still need to conduct labeling on individual publications in all clusters. The reading time is still $O(r \cdot n)$ and the comparing time is $O(c \cdot n \cdot m \cdot d_2)$.

In the third approach (LOAD), labelers can view the whole HPC as one unit. Let average size of HPC be n_1, the estimated reading time will be $O(r \cdot n \cdot \frac{1}{n_1})$. Let average size of HRC be n_2, the average HPC number in each HRC becomes $\frac{n_2}{n_1}$. As the comparison can be limited within the HRC, the number of comparison is $(\frac{n_2}{n_1})^2$. Fortunately, HPCs are ranked in LOAD. So each unlabeled HPC need only to be compared with the previous labeled

Table 5: Comparison of Number of Authors in Clusters

Method	CS	UE	DBLP	*Average*
HPC	1.01	1.00	1.02	*1.01*
HRC	1.82	1.21	1.26	*1.43*
Baseline	1.20	1.18	1.11	*1.16*

one. The number of comparison can be further approximated as $\frac{n_2}{n_1} \cdot d_3$. Finally the overall comparing time can be estimated as $O(c \cdot \frac{n}{n_2} \cdot \frac{n_2}{n_1} \cdot d_3) = O(c \cdot n \cdot \frac{1}{n_1} \cdot d_3)$.

Table 6 lists the estimated time for three methods. It also list the real labeling time for 'Lei Zhang'. Compared to the baseline, LOAD improves the label efficiency by about 11 times.

Table 6: Labeling Time Comparison

Approaches	Estimated Time	Labeling Time for 'Lei Zhang'
Manual	$O(r \cdot n + c \cdot n \cdot m \cdot d_1)$	102,000
Baseline	$O(r \cdot n + c \cdot n \cdot m \cdot d_2)$	26,400
LOAD	$O((r \cdot n \cdot \frac{1}{n_1} + c \cdot n \cdot \frac{1}{n_1} \cdot d_3)$	2,460

We also estimate the parameters during the case study to CS data set: $r = 15$, $c = 10$, $d_1 = 0.3$, $d_2 = 0.1$, and $d_3 = 0.3$, $n_1 = 4$, $n_2 = 9$. According to the estimated time, LOAD improves the label efficiency about 30 times compared to the baseline method for 'Lei Zhang'. The gap between estimated time and labeling time is due to that LOAD cannot get 100% HRC accuracy so extra efforts are needed. Anyway, roughly speaking, LOAD can improve the human labeling efficiency about 10 to 30 times.

5.3 Experiments for Pairwise Classification with Rich Features

Pairwise classification is very important for LOAD. We conduct further experiments to validate the effectiveness of the rich features for building the high precision and the high recall classifiers.

The pairwise classification is to judge whether two publications are written by the same author. Thus, publication pairs written by the same author are taken as positive instances, otherwise pairs with the same names but belong to different authors as negative instances. We combine both CS and UE data sets described in Section 5.1 as the new pairwise data set. About 2/3 instances are randomly selected for training and the rest for testing. The total positive pairs are 37,629, negative pairs are 241,178.

The experimental results for pairwise classification are shown in Table 7.

Table 7: Evaluation Results of Pairwise Classification

Parewise classifier	Precision	Recall
HPC	99.92%	99.44%
HRC	97.76%	99.65%

Both pairwise classifiers for HPC and for HRC achieve quite high precision and recall. Although the the error will be amplified with the increase of cluster size, the very accurate results of pairwise classification can help to get the high accuracy for final results for labeling.

6. CONCLUSIONS AND FUTURE WORK

This paper proposes a novel approach named LOAD to solve the author disambiguation problem together with users. The proposed approach follows a three-step framework which is very flexible and scalable for large scale author disambiguation in digital libraries. Comparing to conventional automatic disambiguation algorithms, LOAD can improve disambiguation accuracy greatly and save a lot of labeling time for human labelers.

There are several possible directions of future work. First, it would be interesting to design an iterative process for author disambiguation. Semi-supervised learning algorithms can then be imported to improve the accuracy. Second, we need to enlarge the feature sources to improve the automatic disambiguation accuracy. Also, more directly optimization algorithms can be applied to get better results.

7. ACKNOWLEDGEMENTS

We thank Microsoft Academic Search team for their encouragements and supports. We thank Hang Li and the anonymous reviewers for their comments to this paper.

8. REFERENCES

[1] M. Bilenko, R. J. Mooney, W. W. Cohen, P. D. Ravikumar, and S. E. Fienberg. Adaptive name matching in information integration. *IEEE Intelligent Systems*, 18(5):16–23, 2003.

[2] O. Byung-won, D. Lee, J. Kang, and P. Mitra. Comparative study of name disambiguation problem using a scalable blocking-based framework. In *ACM/IEEE Joint Conference on Digital Libraries*, pages 344–353, 2005.

[3] X. Fan, J. Wang, L. Bing, L. Zhou, and W. Hu. Ghost: an effective graph-based framework for name distinction. In *International Conference on Information and Knowledge Management*, pages 1449–1450, 2008.

[4] H. Han, H. Zha, and C. L. Giles. A model-based k-means algorithm for name disambiguation. In *International Semantic Web Conference*, 2003.

[5] H. Han, H. Zha, and C. L. Giles. Name disambiguation in author citations using a k-way spectral clustering method. In *ACM/IEEE Joint Conference on Digital Libraries*, pages 334–343, 2005.

[6] H. Han, H. Zha, C. Li, K. Tsioutsiouliklis, and C. L. GILES. Two supervised learning approaches for name disambiguation in author citations. In *ACM/IEEE Joint Conference on Digital Libraries*, pages 296–305, 2004.

[7] J. Huang, S. Ertekin, and C. L. Giles. Efficient name disambiguation for large-scale databases. In *Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 536–544, 2006.

[8] Y. Song, J. Huang, I. G. Councill, J. Li, and C. L. Giles. Generative models for name disambiguation. In *World Wide Web Conference Series*, pages 1163–1164, 2007.

[9] Y. F. Tan, M. yen Kan, and D. Lee. Search engine driven author disambiguation. In *ACM/IEEE Joint Conference on Digital Libraries*, pages 314–315, 2006.

[10] P. Treeratpituk and C. L. Giles. Disambiguating authors in academic publications using random forests. In *ACM/IEEE Joint Conference on Digital Libraries*, pages 39–48, 2009.

[11] F. Wang, J. Li, J. Tang, J. Zhang, and K. Wang. Name disambiguation using atomic clusters. In *Proceedings of the 9th International Conference on Web-Age Information Management*, pages 357–364, 2008.