Name Disambiguation by Collective Classification

Zhongxiang Chen, Jiafeng Guo, Yanyan Lan, Lei Cao, and Xueqi Cheng

Institute of Computing Technology, CAS, Beijing, China 100190
{chenzhongxiang,leicao}@software.ict.ac.cn,
{guojiafeng,lanyanyan,cxq}@ict.ac.cn

Abstract. Disambiguating person names in a set of documents (e.g. research papers or Web pages) is a critical problem in many knowledge management applications. The phenomenon of ambiguity will deteriorate the quality of service, such as the scholar searching and expert finding. Despite years of research, this problem remains largely unsolved, where the unknown number of persons with the same name and the information scarcity in documents pose many difficulties and challenges. In this paper, we formalize name disambiguation as a collective classification problem and solve it using a simple yet effective iterative classification algorithm, referred as ICAND (i.e. Iterative Classification Algorithm for Name Disambiguation). Experimental results on researcher dataset show that the proposed approach can significantly outperform several baseline methods.

Keywords: Name Disambiguation, Collective Classification, Digital Library.

1 Introduction

Name ambiguity is a challenging problem in many domains, where one person can be referenced by multiple name variations in different situations or even share the same name with other persons. For example, in digital libraries (like DBLP or CiteSeer), a very common Chinese name "Lei Zhang" can be shared by tens of authors, and may be written in abbreviation like "L. Zhang". The phenomenon of ambiguity will deteriorate the quality of service such as the scholar searching or expert finding. Therefore, it is necessary to study how to solve it effectively. Despite years of research, this problem remains largely unsolved.

Most previous approaches directly take name disambiguation as a clustering problem [4], [7], [18]. Unfortunately, it is unclear how to determine the number of clusters. In [1], Beckkerman and McCallum tried to address this problem by Agglomerative/Conglomerative Double Clustering (A/CDC). However, it still relies on some predefined threshold to control the clustering process, which is unknown in different applications. Tang *et al.* [14] employed Bayesian Information Criterion (BIC) to select the cluster number. However, their approach tends to find a small cluster number, which may fail when the actual number of persons is large [17].

To avoid the decision of the cluster number, some approaches formalize the problem in a classification manner[4], [17]. However, these methods may suffer from the information scarcity problem. Information scarcity refers to the common phenomenon that a single document may have very sparse attributes for disambiguation [3], [12]. Therefore, it would be quite difficult to classify one document or a pair of documents only relying on their intrinsic features.

To address these above problems, in this paper, we introduce a novel classification method for name disambiguation. The motivation of our approach stems from observations on how human beings disambiguate names. When disambiguating person names in a set of documents, people would like to check whether two documents belong to a same person. The disambiguation process usually works in an iterative process: (1) Most obvious and confident pairs would be settled down first; (2) All the information from the predicted results will also help to disambiguate the remaining "difficult" data; (3) The previous decisions may be adjusted along with the process until a stable result is obtained.

Inspired by the above observations, we formalize name disambiguation as a collective classification problem and use an iterative approach to imitate the above process. Specifically, for each test name, we introduce a relational pairwise graph where each node represents a document pair sharing the same name, and there is an edge between two nodes if they share one same document. The task is then to predict whether each pair of documents on the graph belongs to a same person or not. The key idea of the collective classification is that, when predicting the label of a node, one leverages the relational (extrinsic) features from predicted neighbors as well as the intrinsic features of that node. We propose to employ a simple yet effective iterative classification algorithm (ICA) [13] to solve this problem, referred as ICAND (i.e. Iterative Classification Algorithm for Name Disambiguation). In training, a local classifier is learned based on the labeled data. In this work, we develop a novel sampling strategy for better training the local classifier. In testing, an iterative classification process is conducted on the relational pairwise graph by exploiting both intrinsic and relational features.

Our approach enjoys the following merits: (1) The number of distinct persons can be automatically determined after the classification process; (2) It is flexible to incorporate various intrinsic and relational features to help prediction; (3) A collective inference algorithm is employed to exploit dependencies between documents to well address the information scarcity problem.

To verify the effectiveness of our method, we conduct empirical experiments on an academic dataset collected from an online scholar system SocialScholar[1]. The dataset contains 4,429 publications of 75 different author names[2]. The experimental results show that our method can reach a performance of 89.2% (by F1-score), significantly outperforming the baseline methods.

The rest of our paper is organized as follows. Section 2 presents some related work. In section 3, we formalize name disambiguation problem in the relational pairwise graph. In section 4, we give a detailed description of our iterative

[1] `http://soscholar.com`

[2] `http://static.soscholar.com/pub/dataset.html`

approach called ICAND. In section 5, several experiments are conducted to empirically prove the effectiveness of our approach. Finally, we conclude and discuss future work in section 6.

2 Related Work

In this section, we first review techniques developed in literature on name disambiguation, and then review some related work on collective classification.

2.1 Name Disambiguation

Name disambiguation has been studied for several years, and in general previous approaches can be categorized into two folds: clustering, and classification.

In clustering approaches, the problem of name disambiguation is directly formalized as partitioning documents into different clusters, where each cluster corresponds to a distinct person. For example, in [5], Han *et al.* proposed an unsupervised approach using K-way spectral clustering method in which they took use of three citation attributes for name disambiguation. Wang *et al.* [16] investigated an approach for finding atomic clusters to improve the performance of clustering based algorithms. In [7], Huang *et al.* proposed a two steps framework for name disambiguation. They first trained a distance function between papers using LASVM, and then applied the distance function to DBSCAN clustering process to get results.However, most clustering methods need the number of clusters as a preliminary, which is usually not available.

To avoid the decision of the cluster number, several approaches formalize the problem of name disambiguation in a classification manner. For example, Wang *et al.* [17] tried to predict whether two documents belong to a same person based on a pairwise factor graph, and employed active learning to improve disambiguation performance. However, these methods usually classify one document or a pair of documents only relying on their intrinsic features, which may suffer from the information scarcity problem when documents have very sparse attributes. Our approach falls into this classification category, and exploits relational features as well as intrinsic features to alleviate the information scarcity problem.

2.2 Collective Classification

Collective classification is a method for jointly classifying relational data. Collective classification methods employ a collective inference algorithm that exploits dependencies between instances, enabling them to often attain higher accuracies than traditional methods when instances are interrelated [8], [11], [15], [13]. Even though exact methods such as junction trees [6] or variable elimination [2], [19] can be applied for collective inference, these methods may be prohibitively expensive to use in practice. As a consequence, most research in collective classification has been devoted to the development of *approximate inference algorithms*.

There are two primary types of approximate collective inference algorithms, i.e. *local classifier-based methods* and *global formulation-based methods* [13]. For local classifier-based methods, the collective inference is an iterative process where a local classifier predicts labels for each instance using both intrinsic features and relational features (derived from the current label predictions). Two types of most commonly used approximate inference algorithms following this approach are the *iterative classification algorithm* (ICA) and *gibbs sampling.*

3 Problem Formalization

We consider the name disambiguation problem in digital library scenario where the targets are author names. given an author name, we denote papers containing the author name a as $P^a = \{p_1, p_2, \ldots, p_n\}$. Suppose in these papers there are actually K distinct authors $\Pi = \{\pi_1, \pi_2, \ldots, \pi_K\}$ sharing the same name. The task of the problem is to find the number of distinct authors K, and associate each paper $p_i \in P$ to the right author $\pi_k \in \Pi$.

First of all, we view the name disambiguation task in a classification way. That is, we aim to predict whether each pair of papers (p_i, p_j) containing the same name a belongs to a same author or not. If we can correctly predict all the document pairs, we can automatically find the real author number K, and meanwhile associate each paper to the right author.

For formal definition, we first introduce the relational pairwise graph. Given an author name, we denote a relational pairwise graph over the paper collection sharing the author name as $G = (V, E, X, Y)$, where V is a set of nodes with $v_{i,j} \in V$ representing paper pairs (p_i, p_j), E is a set of undirected edges; each node $v_{i,j} \in V$ has a feature vector $\boldsymbol{x}_{i,j} \in X$ which is a concatenation of the intrinsic feature vector $\boldsymbol{x}_{i,j}^{int}$ and relational feature vector $\boldsymbol{x}_{i,j}^{rel}$, and an unknown label $y_{i,j} \in Y$; there is an edge between two nodes if they share a common paper, e.g. node $v_{i,k}$ is connected with node $v_{j,k}$ as they share the paper p_k. An example relational pairwise graph over a list of papers $P = \{p_1, p_2, p_3, p_4\}$ containing a same author name is shown in Fig. 1.

The author name disambiguation task is then formalized as the following collective classification problem. Given a relational pairwise graph $G = (V, E, X, Y)$ for an author name, one needs to predict the label $y_{i,j} \in \{-1, +1\}$ for each node $v_{i,j} \in V$, representing whether the pair of papers (p_i, p_j) belong to a same author $(y_{i,j} = +1)$ or not $(y_{i,j} = -1)$.

4 Our Approach

We propose using an iterative algorithm to solve this collective classification problem. As aforementioned, the key idea of using such an iterative process comes from the observation on how human beings disambiguate names.

When disambiguating author names in a set of papers, people would like to check whether two papers belong to a same author in an iterative process. Those

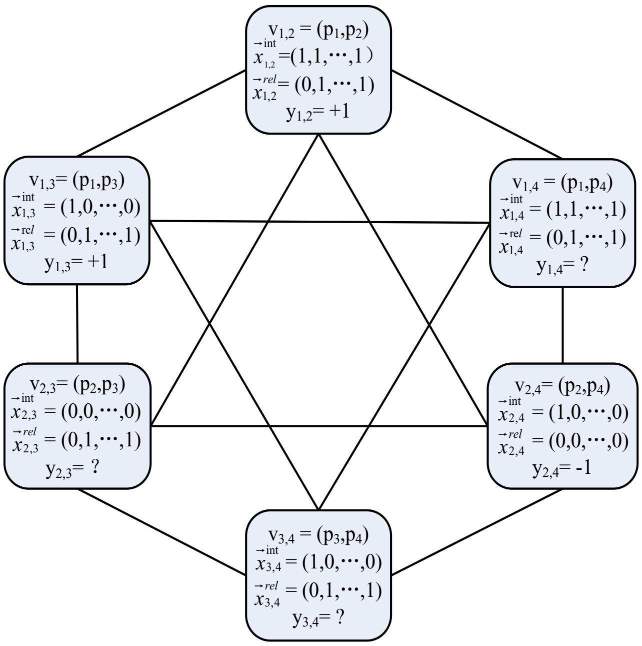

Fig. 1. Relational pairwise graph for 4 papers all sharing an identical author name a

paper pairs with strong signals (e.g. paper p_1 and p_2 appear on a same homepage) would be considered as from a same author with high confidence first, while those with weak and scarce signals (e.g. paper p_3 cites p_2) would be difficult to predict at first and leave for latter decision. Once a pair of papers are taken as from a same author, one will naturally leverage one paper's attributes to enrich the other and help latter prediction.

We employ an iterative classification algorithm, referred as ICAND, to imitate the above process and solve the collective classification problem. The overview of our algorithm is as follows. In the initial step, we predict the label $y_{i,j}$ of each node $v_{i,j} \in V$ using a local classifier f based on intrinsic features $\boldsymbol{x}_{i.j}^{int}$ constructed from the attributes of the paper pairs on $v_{i,j}$. This step is a "bootstrap" step since none of the nodes' labels are known. After the bootstrap step, all labels of the nodes are predicted. Then in the following iteration step, for each node ICAND selects confidently predicted positive neighbors as observed labels, compute the relational features $\boldsymbol{x}_{i.j}^{rel}$ based on these neighbors, and then re-predict the labels using the local classifier based on both intrinsic and relational features. This step iterates until convergence or the iteration count exceeds some predefined threshold. Finally, a post-aggregation step is conducted to generate the paper clusters, each corresponding to a distinct author. The pseudo-code of our algorithm is summarized in Algorithm 1.

Algorithm 1. Iterative classification algorithm for Name disambiguation

```
Input: graph $G = (V, E, X, Y)$ for a given author name where $Y$ is unknown initially, local
       classifier $f$, max iteration steps $T$
Output: paper clusters, each representing a distinct author
1  $step \leftarrow 0$
2  for each node $v_{i,j} \in V$ do
3      $y_{i,j}^{(0)} \leftarrow f\ (\boldsymbol{x}_{i,j}^{(0)})$   \\ bootstrap step
   end
4  repeat
5      for each node $v_{i,j} \in V$ do
6          sort and select confidently predicted neighbors $v_{k,l} \in Neighbor(v_{i,j})$ with
           $y_{k,l}^{(step)} = +1$
7          re-compute the feature vector $\boldsymbol{x}_{i,j}^{(step+1)}$ with the selected neighbors' labels observed
8          $y_{i,j}^{(step+1)} \leftarrow f\ (\boldsymbol{x}_{i,j}^{(step+1)})$
       end
9      $step \leftarrow step + 1$
   until $Y^{(step+1)} = Y^{(step)}$ or $step \geq T$;
10 post-aggregation based on final predictions $Y$ to get paper clusters
11 return paper clusters
```

4.1 Intrinsic and Relational Features

Intrinsic features are static features extracted from the node which describe how likely two papers in that node are from a same author. The value of the feature could either be binary or real value. In this paper, we define 10 features for our disambiguation task, including author-based, venue-based, citation-based, and content-based features. The specific description of each feature is listed in Table 1. As most of the features are quite intuitive, The feature CoConcept may need more explanation. In our work, we build a concept dictionary based on all the key words from papers in the academic dateset.

Table 1. Description of features for a paper pair (p_i, p_j) given the Author Name a

Name	Description	Type
CoAuthor	paper p_i and p_j have at least one same author except a	binary
CoOrganization	the organization of author a in p_i and p_j are the same	binary
CoOrgOccur	the organization of author a in p_i appears in content of p_j, or vice versa	binary
CoHomepage	paper p_i and p_j appear on a same author's homepage	binary
CoVenue	paper p_i and p_j are published on the same journal or conference	binary
CoRefCite	paper p_i and p_j cite or are cited by at least one same paper	binary
Citation	paper p_i cites p_j, or vice versa	binary
TitleSim	title similarity between paper p_i and p_j (cosine similarity based on tf-idf word vector)	real
AbstractSim	abstract similarity between paper p_i and p_j (cosine similarity based on tf-idf word vector)	real
CoConcept	paper p_i and p_j hit at least one same concept in title or abstract	binary

Different from intrinsic features, relational features are dynamic features derived from neighborhood and are re-calculated in each iteration step. As aforementioned, relational features are constructed based on the neighbors with positive labels. Going back to the example shown in Fig. 1, if the paper pair (p_1, p_2)

has already been labeled as positive confidently, we can then leverage the attributes of p_1 to help disambiguate (p_2, p_3), and extract relational features based on papers p_1 and p_3. Note that the specific definition of relational features are exactly the same as that of intrinsic features; in other words, there are 10 relational features.

When there are multiple neighbors with positive labels, we need to use an aggregation operator to generate a fixed-length relational feature vector. Past research has used a variety of aggregation operators such as minimum, maximum, and count [13]. The choice of the aggregation method depends on the specific application and the definition of relational features. In our work, we define a max operator to aggregate the relational features. The purpose of this max operator is to obtain the strongest signals from neighborhoods. Specifically, the l-th relational feature of node $v_{i,j} \in V$ is defined as

$$\boldsymbol{x}_{i.j}^{rel}[l] = \max_{v_{i,k}, v_{k,j} \in S^+} \{\boldsymbol{x}_{i,k}^{int}[l], \boldsymbol{x}_{k,j}^{int}[l]\} \tag{1}$$

where S^+ denotes the set of neighbors of node $v_{i,j}$ with positive labels in prediction.

4.2 Local Classifier

We can use anything ranging from SVM to decision tree as the local classifier. In our work, we choose to use SVM with linear kernel. The local classifier is trained on the labeled ground truth. Specifically, the ground truth dataset for author name disambiguation usually consists of a set of M author names $A = \{a_1, \ldots, a_M\}$, where for each name $a_m \in A$, the number of distinct authors are known and a set of papers containing that name are associated to the right author (i.e. cluster). In the training process, for each name $a_m \in A$, we construct the corresponding relational pairwise graph $G^{a_m} = \{V^{a_m}, E^{a_m}, X^{a_m}, Y^{a_m}\}$, where the features for each node are extracted based on the node attributes and ground truth labels of neighbors. The local classifier is then trained based on the training data $(\boldsymbol{x}_{i,j}^{a_m}, y_{i,j}^{a_m})_{a_m \in A}$.

Note that when calculating the relational features for constructing the training data, there might be multiple strategies on the usage of ground truth labels. In previous work, the local classifier is usually trained on full-label assumption, i.e. for each node, the labels of all the neighbors are available. However, the classifier trained under this strategy may not work well when very few labels can be leveraged in the early stages of the iterative process in testing. Therefore, in this paper, we propose a novel sampling strategy on label usage, i.e. for each node, only partial labels of neighbors are available.In our experiments, we find that the classifier learned under this strategy can achieve better performance.

4.3 Collective Inference

The collective inference algorithm is the central part of a collective classification approach. In our work, we employ a local classifier-based method, i.e. the iterative classification algorithm, to conduct approximate collective inference. This algorithm has been shown to be simple yet effective as compared with other local and global methods [13].

The iterative algorithm exploits positive label predictions to help inference in each iteration, as shown in Algorithm 1. One major issue we need to address is how to select confident label prediction for next inference. In our work, we take some cautious strategies for label selection [9,10]. Specifically, in each iteration, the predicted positive labels are ordered by confidence value (i.e. predicted score) and only top confident labels are used for inference. The proportion of top confident positive labels in usage could be fixed (i.e. using predefined proportion like 10%) or dynamic (e.g. increasing the proportion per iteration from 0%, 10%, ..., up to 100%). In our experiments, we empirically compared different strategies in label selection and show how these strategies affect the performance.

4.4 Post-aggregation Step

For the goal of the name disambiguation task, we need to find out the actual K distinct authors (K is unknown in our case) sharing the same name and associate each paper to the right author (i.e. dividing the papers into K clusters). Therefore, we take a simple post-aggregation step to obtain the final K clusters, i.e. the step 9 in Algorithm 1. The aggregation is in a agglomerative clustering manner. At the beginning, each paper forms a cluster. If a paper pair has been predicted with positive label and the two papers come from two different clusters, then the two clusters are merged together. This process iterates until no further merging can take place. As we can see, the final author number K can be automatically determined after the post-aggregation step.

One thing need to mention in the post-aggregation process is the triplet violation problem. Specifically, suppose both of the pairs (p_1, p_2) and (p_1, p_3) have already been labeled as "+1", the label of pair (p_2, p_3) should be also assigned as "+1", otherwise it will lead to triplet violation problem.

However, since in our iterative classification process there is no constraint to avoid this problem, it is possible that there are triplet violation in the final classification results. Fortunately, such violation can be naturally solved in the agglomerative clustering process described above naturally. Nevertheless, it would be interesting to investigate how to add constraints into the collective classification process to avoid such problem in the future work.

5 Experiments

In this section, we conduct experiments to empirically evaluate the effectiveness and efficiency of our proposed approach.

5.1 Experimental Setting

Dataset. To evaluation our proposed method, we create an academic dataset from SocialScholar system. SocialScholar has collected and combined papers from DBLP, IEEE, ACM and CiteSeer, and formed a publication dataset of $8,014,742$ papers and $24,303,153$ citation relationships. For evaluation, we employed three graduate student in computer science to manually labeled $4,429$ papers for 75 author names. For each author name, a paper containing that name was labeled with a number indicating the actual author. For disagreements in the annotation, we applied majority voting for decision.

Baselines. For evaluation, we consider both clustering and classification methods as our baselines. For the clustering methods, several existing methods for name disambiguation, including hierarchical agglomerative clustering method (HAC), K-Means, and SA-Cluster [20], are taken as baselines. In the first two method, we try to take all features defined in our method. In SA-Cluster method, we consider organization and venue of each paper as attribute features, and simply treat others features as edges. In all these clustering methods, the real number of persons K is preliminarily provided.

For the classification methods, we take the Pairwise Classification (PC) method as the baseline. The PC method can be viewed a simplified version of our ICAND method, which drops the relations between nodes in our relational pairwise graph, and conducts training and prediction only based on intrinsic feature. A same post-aggregation process is employed to get final K Clusters. The PC method also employs SVM with linear kernel as the base classifier.

In our experiments, for the supervised methods where a classifier need to be trained first, we divide the dataset into five folds and conducted five-fold cross validation for evaluation. For our method, we set the max iteration number as 10 since we found that in most cases the iterative process converges quickly.

5.2 Evaluation Measures

We employ the widely used pairwise measures [14], [16,17] to evaluate our approach and compare with baseline methods. The pairwise measures evaluate the performance of disambiguation based on the paper pairs assigned with the same label. For some special author names, there is only one paper corresponds to each distinct author. Since these above measures only consider paper pairs assigned with same label, they cannot well evaluate the results on such names. Therefore,we employ *pairwise_accuracy* as supplementary measure. The definitions of these measures are shown as follows.

$$pairwise\_precison = \frac{\#PairsCorrectlyPredicted2SameAuthor}{\#PairsPredicted2SameAuthor}$$

$$pairwise\_recall = \frac{\#PairsCorrectlyPredicted2SameAuthor}{\#TotalPairs2SameAuthor}$$

$$pairwie\_F1 = \frac{2 * pairwise\_precison * pairwise\_recall}{pairwise\_precison + pairwise\_recall}$$

$$pairwise\_accuracy = \frac{\#PairsCorrectlyPredicted}{\#TotalPairs}$$

5.3 Strategy Analysis

We conduct experiments to study how different strategies used in the training and testing process in ICAND affect the disambiguation performance.

Training Label Selection. Here we first compare the strategy on the usage of ground truth labels in training process, and show how different training strategies affect the final performance. For comparison, two strategies are taken into account. One is the conventional full-label strategy, i.e. when computing relational features for each node, all the ground truth labels of neighbors are available. The other is the proposed sampling strategy, i.e. when computing relational features for each node, only partial ground truth labels of neighbors are available. Specifically, for each node, we vary the proportions of available labels in neighbors from 0% to 100% with step length 10%. For each proportion, we sample the labels of neighbors for each node, compute the relational features as well as intrinsic features, and obtain the corresponding training data. The final training dataset is constructed by merging all the data generated under different label proportions with the duplication removed.

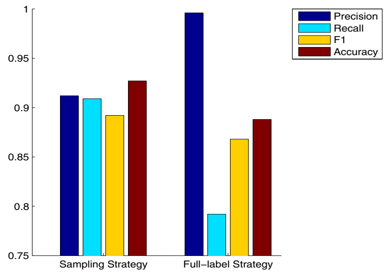

Fig. 2. Performance comparison on name disambiguation using different strategies on training data construction

Table 2. Performance comparison on name disambiguation using different label selection strategies in collective inference

Type		Precision	Recall	F1	Accuracy
	10%	1	0.167	0.253	0.559
	20%	1	0.293	0.425	0.631
	30%	1	0.451	0.596	0.713
	40%	0.997	0.573	0.703	0.781
fixed	50%	0.990	0.660	0.775	0.825
	60%	0.976	0.724	0.816	0.856
	70%	0.968	0.806	0.866	0.898
	80%	0.944	0.852	0.881	0.911
	90%	0.912	0.909	**0.892**	**0.927**
	100%	0.835	0.944	0.853	0.897
dynamic	+5%	0.847	0.941	0.862	0.907
	+10%	0.847	0.942	0.862	0.908

As shown in Fig. 2, the disambiguation performance under the sampling strategy is better than that of full-label strategy in terms of average recall, F1 and accuracy. The average F1 scores are 0.892 and 0.853 under sampling strategy and full-label strategy, respectively. The major reason is that the full-label strategy tends to obtain a strict local classifier with high precision. As a result, the final performance achieve high precision but low recall. In contrast, the sampling strategy builds a more diverse dataset, which captures different scenarios of the classifier may face during the iterative process. Therefore, the local classifier learned under this strategy become more robust with better balance between recall and precision, and thus obtain better disambiguation performance.

Inference Label Selection. We further compare the label selection strategy in the collective inference process. For the fixed proportion strategy, we vary the predefined proportion at different levels (i.e. 10%, 20%, ..., or 100%). For the dynamic proportion strategy, we increase the fraction from 0% to 100% with constant step length (i.e. 5% or 10%).

The disambiguation performance comparison among these label selection strategies is shown in Table 2. We can see that for fixed proportion strategy, a low proportion leads to high precision but low recall in final performance. This is natural since we only trust those very confident positive label predictions. With the increase of the proportion, we obtain higher recall but lower precision gradually. The best performance is achieved at the proportion of 90%. If one simply trust all the positive predictions, the precision become low since there might be many incorrect positive predictions selected in the process making the error propagate. For the dynamic proportion strategy, there is almost no difference between the two different step lengths. However, the precision is not high for dynamic strategy, which might be caused by the selection of incorrect positive predictions when the confidence proportion becomes larger and larger.

In the following experiments, we use the ICAND approach with the local classifier trained under the sampling strategy, and a fixed proportion (i.e. 90%) label selection strategy for the collective inference process.

Table 3. Performance comparison on name disambiguation between different methods

Method	Precision	Recall	F1	Accuracy
HAC	0.838	0.787	0.801	0.859
K-Means	0.763	0.461	0.547	0.694
SA-Cluster	0.669	0.588	0.611	0.766
PC	0.728	0.904	0.720	0.696
ICAND	**0.912**	**0.909**	**0.892**	**0.927**

5.4 Disambiguation Performance

We now compare the disambiguation performance of our approach with the baseline methods. As shown in Table 3, on average, our method can achieve a precision of 91.2%, recall 90.9%, F1 89.2% and accuracy 92.7%. We can see that our approach can clearly outperform all the baseline methods in terms of all the evaluation measures (+9.1% over HAC, +34.5% over K-Means, +28.1 % over SA-Cluster, and +17.2% over PC by F1 score). All the improvements are statistically significant (p-value$<$ 0.01). The results demonstrate that by using the collective inference approach which exploits various intrinsic as well as relational features, we can better address the name disambiguation problem.

6 Conclusion

In this paper, we address name disambiguation by formalizing it as a collective classification problem. We then employ an iterative algorithm to solve this collective classification problem, referred as ICAND. Our approach can automatically determine the number of distinct persons for a given name after the classification process. Moreover, the collective inference algorithm employed in our approach exploits relational features based on label prediction, which can well address the information scarcity problem. We conducted extensive experiments on an academic dataset to demonstrate the effectiveness of our approach.

Acknowledgments. This research work was funded by the National Grand Fundamental Research 973 Program of China under Grant No. 2014CB340401, the National High-tech R&D Program of China under Grant No. 2012AA011003, the National Natural Science Foundation of China under Grant No. 61232010, Grant No. 61173064 and Grant No. 61203298, and the National Key Technology R&D Program of China under Grant No. 2012BAH39B04 and Grant No. 2012BAH39B02.

References

1. Bekkerman, R., McCallum, A.: Disambiguating Web appearances of people in a social network. In: WWW, p. 463. ACM Press (2005)
2. Dechter, R.: Bucket elimination: A unifying framework for probabilistic inference. In: UAI, pp. 211–219 (1996)
3. Fan, X., Wang, J., Pu, X., Zhou, L., Lv, B.: On Graph-Based Name Disambiguation. JDIQ 2(2), 1–23 (2011)
4. Han, H., Giles, L., Zha, H., Li, C., Tsioutsiouliklis, K.: Two supervised learning approaches for name disambiguation in author citations. In: JCDL (2004)
5. Han, H., Zha, H., Giles, C.L.: Name disambiguation in author citations using a K-way spectral clustering method. In: JCDL (2005)
6. Huang, C., Darwiche, A.: Inference in belief networks: A procedural guide. IJAR 11(1), 158 (1994)
7. Huang, J., Ertekin, S., Giles, C.L.: Efficient Name Disambiguation for Large-Scale Databases. In: PKDD, pp. 536–544 (2006)
8. Jensen, D., Neville, J., Gallagher, B.: Why collective inference improves relational classification. In: SIGKDD, pp. 593–598 (2004)
9. McDowell, L., Gupta, K., Aha, D.: Meta-prediction for collective classification. In: Proceedings of the 23rd International FLAIRS Conference (2010)
10. Mcdowell, L.K.: Cautious Collective Classification. JMLR (2009)
11. Neville, J., Jensen, D.: Iterative classification in relational data. In: Proc. AAAI-2000 Workshop on Learning Statistical Models from Relational Data (2000)
12. Qian, Y., Hu, Y., Cui, J., Zheng, Q., Nie, Z.: Combining Machine Learning and Human Judgment in Author Disambiguation. In: CIKM (2011)
13. Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., Eliassi-Rad, T.: Collective classification in network data. AI magazine (2008)
14. Tang, J., Fong, A.C.M., Wang, B., Zhang, J.: A Unified Probabilistic Framework for Name Disambiguation in Digital Library. TKDE 24, 975–987 (2012)
15. Taskar, B., Abbeel, P., Koller, D.: Discriminative probabilistic models for relational data. In: UAI, pp. 485–492 (2002)
16. Wang, F., Li, J., Tang, J., Zhang, J., Wang, K.: Name Disambiguation Using Atomic Clusters. In: WIAM, pp. 357–364. IEEE (July 2008)
17. Wang, X., Tang, J., Cheng, H., Yu, P.S.: ADANA: Active Name Disambiguation. In: ICDE, pp. 794–803. IEEE (December 2011)
18. Yin, X., Han, J., Yu, P.S.: Object Distinction: Distinguishing Objects with Identical Names. In: ICDE, pp. 1242–1246 (2007)
19. Zhang, N.L., Poole, D.: A simple approach to bayesian network computations. In: Canadian Conference on Artificial Intelligence (1994)
20. Zhou, Y., Cheng, H., Yu, J.X.: Graph Clustering Based on Structural / Attribute Similarities. In: VLDB (2009)