2011 11th IEEE International Conference on Data Mining

ADANA: Active Name Disambiguation

Xuezhi Wang*†, Jie Tang*, Hong Cheng‡, Philip S. Yu§
†*Department of Computer Science and Technology, Tsinghua University, China*
**Computer Science Department, Carnegie Mellon University, USA*
‡*Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, China*
§*Department of Computer Science, University of Illinois at Chicago, USA*
xuezhiw@cs.cmu.edu, jietang@tsinghua.edu.cn, hcheng@se.cuhk.edu.hk, psyu@cs.uic.edu

***Abstract*—Name ambiguity has long been viewed as a challenging problem in many applications, such as scientific literature management, people search, and social network analysis. When we search a person name in these systems, many documents (e.g., papers, web pages) containing that person's name may be returned. It is hard to determine which documents are about the person we care about. Although much research has been conducted, the problem remains largely unsolved, especially with the rapid growth of the people information available on the Web.**

In this paper, we try to study this problem from a new perspective and propose an ADANA method for disambiguating person names via active user interactions. In ADANA, we first introduce a pairwise factor graph (PFG) model for person name disambiguation. The model is flexible and can be easily extended by incorporating various features. Based on the PFG model, we propose an active name disambiguation algorithm, aiming to improve the disambiguation performance by maximizing the utility of the user's correction. Experimental results on three different genres of data sets show that with only a few user corrections, the error rate of name disambiguation can be reduced to 3.1%. A real system has been developed based on the proposed method and is available online.

***Keywords*-Name Disambiguation, Social Network Analysis, Digital Library, Active Learning**

I. Introduction

Name disambiguation, also known as entity resolution, web appearance disambiguation, name identification, and object distinction, has long been viewed as a challenging problem in many domains. Despite slight differences, the general task of name disambiguation is to associate documents (or web pages) to different people who share an identical name. Considerable research has been conducted to deal with this problem, for example [1], [2], [3], [4], [5], [6]. However, the problem remains largely unsolved. On the contrary, it becomes more critical due to the fact that information of more and more real-world people are getting online. It is estimated that merely in United States, the 300 most common male names are used by more than 114 million people.

The first challenge still remaining in the name disambiguation problem is: given a person name and a collection of documents containing that person name, how to determine the number of distinct persons who share the identical name? Bekkerman and McCallum [5] propose using Agglomerative/Conglomerative Double Clustering (A/CDC) to solve this problem. However, it still needs a predefined threshold to control the disambiguation process, which may be very different in different applications. Tang et al. [6] employ Bayesian Information Criterion (BIC) as the criterion to estimate the number of persons K. The idea is to enumerate and find a number K that could maximize an objective function based on Bayesian Information Criterion. However, the approach tends to find a small number; thus when the actual number of persons is large, the approach fails to find the accurate number. Another challenge to name disambiguation is that the data are becoming more and more complex and dynamic. This requires a name disambiguation algorithm to be extendable and flexible for different scenarios. However, most of existing methods are designed for particular tasks and not easy to be extended. As a result, the best performance (accuracy) achieved by the state-of-the-art algorithms is still under 90%. The result is unsatisfactory and invariably contains a number of errors.

Now, the success of many online social networks offers the opportunity for users to provide feedbacks to search/mining systems, as well an unprecedented chance to learn from the users with machine learning algorithms. It is promising to design an interactive interface for a disambiguating system to acquire feedbacks from users. However, we should be aware that the interactive process might be tedious, error-prone, and time-consuming. For example, in the scientific literature management, an author may have hundreds of publications. The user may soon become tired, if she/he is asked to *carefully* go through all her/his publications and those of the others to validate the disambiguation results. Ideally, it is desirable that the disambiguation system can *actively* select only a few potentially wrong disambiguation results to query the user, instead of *passively* waiting for user inputs. The problem is referred to as *active name disambiguation*.

In this paper, we try to systematically investigate the problem of active name disambiguation with the following contributions:

- We precisely define the problem of active name disambiguation and propose a method called ADANA

1550-4786/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDM.2011.19

to actively disambiguate person names. In particular, we present a pairwise factor graph model, which can automatically determine the number of distinct names in contrast to most conventional models, and can be dynamically refined during the active learning in an efficient manner using a novel concept of atomic clusters.
- To make optimal use of user interaction, we present an interactive disambiguation framework. An influence-maximization based active selection strategy has been devised to actively select potentially wrong but most useful disambiguation results to query the user. It is shown to be very effective over alternative strategies on reducing the number of interactions and improving the accuracy on disambiguation.
- We conducted experiments on three data sets: Publication (a publication data set), CALO (a web page data set), and News Stories (a news page data set). Experimental results show that the proposed ADANA method can achieve better performance for name disambiguation than several existing methods. Experiments also demonstrate that with only a few (2-5) active user interactions, our method can reach a performance of 96.9% (by F1-score) for name disambiguation, significantly outperforming the baseline methods.

Organization. Section II formulates the problem and Section III describes the data preparation. Section IV presents a pairwise factor graph model for name disambiguation, and Section V introduces our proposed algorithms for active name disambiguation by learning from user interaction. We present experimental results that validate the effectiveness of our methodology in Section VI. Finally, Section VII discusses related work and Section VIII concludes the paper.

II. Preliminaries

In this section, we first give several definitions and then present a formal definition of the problem.

Given a person name a, let D^a denote a collection of documents (e.g., papers, web pages, or news stories), which contain the person name a, i.e., $D^a = \{d_1^a, d_2^a, \cdots, d_N^a\}$. Assume that each document can be associated with d attributes, then we can define an $N \times d$ attribute matrix X^a, in which each row corresponds to a document, each column an attribute, and an element X_{ij}^a is the j^{th} attribute value of document d_i^a. For simplicity, if there is no ambiguity in the following explanation, we remove the superscript a in the notations.

The attributes can be defined differently in different applications. For example, to disambiguate author names in a publication data set, the attributes mainly include coauthorships, paper title, publication year, and publication venue; while to disambiguate person names in web pages, the attributes mainly include text words and hyperlinks contained in the web pages. Now we can describe the problem of name disambiguation as follows:

Problem 1: **Name Disambiguation.** For a given person name a, a collection of N associated documents D^a and the corresponding $N \times d$ attribute matrix X^a, our goal is (1) to find how many distinct persons (L) the documents should belong to, and (2) to cluster the N documents into L groups (persons) with high accuracy.

Please note that both subtasks are non-trivial. Most existing algorithms fail to tackle the first subtask, by assuming that the number L of persons is provided by the user, which is obviously impractical in real applications.

Another important objective of our work is to study how to actively leverage user interaction to help name disambiguation. In particular, we aim to answer the following questions: (1) Given the initial results by a disambiguation algorithm, which results should we select to query the user? (2) When the user provides feedbacks (corrections) on the results, how could we leverage the feedbacks to refine the disambiguation model? In general, the user can have different types of interactions with the disambiguation system, for example, telling the system which clusters she/he likes and which clusters she/he does not like. However, the open interactive solution makes it intractable to design a feasible active name disambiguation algorithm. Thus in this paper, we confine ourselves to the pairwise document query, i.e., to query whether two documents belong to the same cluster or not. Based on these concepts, we give a precise definition of the problem of active name disambiguation:

Problem 2: **Active Name Disambiguation.** Given a person name a, a set of associated documents D^a with the attribute matrix X^a, and initial disambiguation results $\mathcal{G} = \{G_1, \cdots, G_L\}$, where each cluster $G_k \subset D^a$ consists of a subset of documents in D^a, the goal of active name disambiguation is to refine the disambiguation model by selecting a number of document pairs $\{(d_i, d_j)_k\}$ to query the user whether the two documents should belong to the same cluster or not.

The objective of active name disambiguation is to achieve a maximal improvement on the accuracy of disambiguation after a predefined number of user interactions. The problem is very different from existing work on name disambiguation. Most existing works such as [2] and [4], try to improve the performance of automatic name disambiguation. However, they do not consider user interaction and the obtained performance is still unsatisfactory. Davis et al. [7] have developed an interactive system for disambiguation. However, they only provide a tool for users to browse and correct the disambiguation results, but do not consider how to make optimal use of user interaction. In this paper, our ultimate goal is to provide a practical and highly accurate framework to solve the name disambiguation problem with user interaction. To the best of our knowledge, there is no real system that exploits the active user interaction for name disambiguation.

Table I
FEATURES FOR A PAIR OF DOCUMENTS (d_i^a AND d_j^a).

Name	Description
Citation	doc d_i cites d_j in the reference, or vice versa.
CoAuthor	d_i and d_j share at least one coauthor (except a)
CoVenue	d_i and d_j are published at the same venue
CoAffiliation	the affiliations of author a in d_i and d_j are the same
CoAffOccur	the affiliation of author a in d_j appears in the content of document d_i, or vice versa
TitleSim	similarity between titles of d_i and d_j
Homepage	doc d_i and d_j appear on the same homepage

III. DATA PREPARATION

Data Set We performed experiments on three different genres of real-world data sets: Publication (a publication data set from [8][1]), CALO (a web page data set from [5]), and News Stories (a data set from the top two (Business and U.S. Politics) of the ten MSNBC news categories on January 2, 2007 [9]). Detailed statistics of the three data sets will be given in the experimental section. In this section, we use the publication data set as the example to explain how we collect the data set and how we define the attributes (features).

For the purpose of problem analysis and performance evaluation, we collected 6,730 published papers of 100 author names. 22 human annotators performed disambiguation on this data set. A specification was created to guide the annotation process and each paper was labeled by at least three annotators. Each paper is associated with a number indicating the cluster that it has been assigned to. The annotation was carried out based on the publication list on the authors' homepages, the affiliation and email addresses in the PDF files. There are a few extreme cases (less than 1%) that even human cannot judge which person (cluster) a paper belongs to. For such cases, we assigned this paper to an "other" cluster. For disagreements in the annotation, we had more annotators to double-check and finally conducted "majority voting".

A. Feature Definition

In the publication data set, each paper is associated with a set of attributes: coauthors, title, publication venue, publication year, references, paper content, and affiliations. As our approach always tries to deal with a pair of documents instead of a single one, we take each pair of documents as the basic unit in our algorithm framework and define the following features for a document pair: Citation, CoAuthor, CoVenue, CoAffiliation, CoAffOccur, TitleSim, CoHomepage, as summarized in Table I, and detailed as follows.

Citation It is possible that an author cites his own paper, but seldom cites a paper authored by another author with the same name. Thus, if document d_i^a cites document d_j^a, then we define the value of the direct Citation feature for the two documents as 1, otherwise 0. In the CALO data set, if web page d_i^a has a hyperlink to web page d_j^a, we say that the two web pages have a direct Citation relation. While one may argue that direct citation occurs less frequently, we also define an indirect Citation feature, which illustrates the fact that a researcher may cite papers of his coauthors (e.g., his advisor). In particular, if document d_i^a cites a document with author b and another document d_j^a is coauthored by b, then we say that d_i^a and d_j^a have an indirect Citation relation, i.e., the value of the indirect Citation feature for the two documents is 1.

CoAuthor Each document can be coauthored by multiple authors. If two documents d_i and d_j have a same coauthor, except author a, then the value of the CoAuthor feature between the two documents is defined as 1, otherwise 0. In the CALO and News Stories data sets, we consider co-occurrence of the person names in the documents.

CoVenue Persons with the same name may work in different fields, thus publishing in different venues. For example in the publication data set, if two documents are published at the same conference or journal, the CoVenue value of the two documents is defined as 1, otherwise 0. The CoVenue feature contains a lot of noise. We refine the feature by considering only "uncommon" venues (the number of published papers at the venue is less than a threshold).

CoAffiliation In some documents, but not all, we may have the affiliation information. If a same affiliation appears in both documents d_i and d_j, then the value of the CoAffiliation feature of the two documents is defined as 1, otherwise 0. In the publication data, the affiliation information can be extracted from the metadata of each paper. Please note that the affiliation itself may have the ambiguity problem. Thus we use a university name dictionary and heuristics to normalize the affiliation names before generating this feature.

CoAffOccur In practice, the available affiliation information in the documents (papers) is very limited. Statistics show that only 45% of the papers in the publication data set have the affiliation information in their metadata information. On the other hand, such information is usually hidden in the content. For example, the content of a paper usually contains the names of coauthors and their affiliation information on the top of the first page. Thus the feature is defined as whether the contents of two documents contain a same affiliation. To avoid noise and improve the efficiency, we consider only the first 500 words of each document.

TitleSim This feature evaluates the keyword-based similarity of titles between two documents. In particular, we use the vector space model to represent each title (with TFIDF as the value of each word), and then use cosine similarity between two titles as the feature value.

CoHomepage If two documents (papers) appear together

[1]http://arnetminer.org

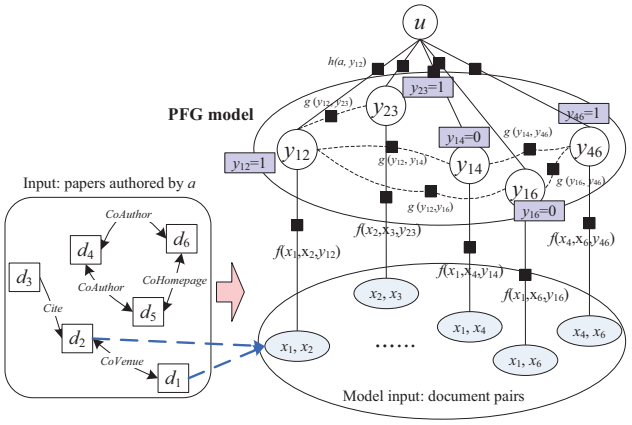

Figure 1. Graphical representation of the pairwise factor graph model. $\{d_1, \ldots, d_6\}$ are six documents with various relations (features); $\{(\mathbf{x}_1, \mathbf{x}_2), \cdots, (\mathbf{x}_5, \mathbf{x}_6)\}$ are observable variables defined based on the input documents; $\{y_{12}, \cdots, y_{56}\}$ are hidden variables defined for all pairs of documents, with each element representing whether the corresponding pair of documents should belong to the same cluster or not; $f(.)$ represents a feature function defined for each pair of documents, $g(.)$ represents a correlation (or constraint) function defined over the hidden variables and $h(a, y_{ij})$ denotes a constraint function defined on y_{ij}.

on a person's homepage, then it is very likely that the two documents (papers) belong to the same person. We use a classification-based method to find the homepage of a person. Specifically, we first use Google to find relevant web pages to a given person name, and then use the classification model to identify whether a returned web page is a homepage or not. The classification model is trained using SVM-light with a labeled training data set. The precision of the classification model is 91.39% and the recall is 78.3%. Our preliminary experiments show that about 67% homepages can be found by this approach, and the results show that this feature is very useful and can significantly improve the disambiguation performance.

One thing we need to mention is that not all of these attributes are available in the data set. For example, in the publication data set, there are only 31.2% papers having the corresponding content information (PDF files), 47% papers having the references, 45% papers having the authors' affiliations, and 67% author names having corresponding homepages. Also the attribute values may contain noise.

IV. Name Disambiguation via Pairwise Factor Graph Model

In this paper, we formalize the problem of name disambiguation in a pairwise factor graph (PFG) model. The basic idea is to associate each pair of documents (e.g., d_i and d_j) with a hidden variable y_{ij}, representing whether these two documents should be assigned to the same cluster ($y_{ij} = 1$) or not ($y_{ij} = 0$). For example, Figure 1 shows a simple example of a PFG. The input is a person name a and the collection of documents $D^a = \{d_1, d_2, \cdots, d_6\}$. We generate a set of document pairs $X = \{(\mathbf{x}_1, \mathbf{x}_2), \cdots, (\mathbf{x}_5, \mathbf{x}_6)\}$, which forms the observable variables of the PFG model. The hidden variables $Y = \{y_{12}, \cdots, y_{56}\}$ are defined corresponding to the observable variables. A pairwise factor graph model is then constructed based on this formulation. Typically, we hope that a model can best fit (reconstruct) the observation data (observable variables), equivalently, a configuration of Y can maximize a defined objective function. In general, we can define the objective function as the generative likelihood of the hidden variables Y given all the observable data X:

$$p(Y|X) = \frac{1}{Z}\exp\{\sum_{i\neq j}\sum_{k} w_k f_k(\mathbf{x}_i, \mathbf{x}_j, y_{ij}) + \sum_{e_{ij}^{jk}\in E} \mu\ g(y_{ij}, y_{jk}) + \sum_{l} \alpha_l h_l(a, y_{ij})\} \quad (1)$$

where $f_k(\mathbf{x}_i, \mathbf{x}_j, y_{ij})$ is a feature function defined for the pair of documents (d_i, d_j); e_{ij}^{jk} is an edge on the PFG graph connecting y_{ij} and y_{jk}; basically, we define a correlation feature function between two observable variables if they share one document, e.g., $(\mathbf{x}_1, \mathbf{x}_2)$ and $(\mathbf{x}_2, \mathbf{x}_3)$; $h_l(a, y_{ij})$ is a constraint function defined over y_{ij}; w_k, μ, and α_l are weights of the corresponding feature functions. The objective function has three types of feature functions:

- **Document-pair feature function** $f_k(\mathbf{x}_i, \mathbf{x}_j, y_{ij})$. It represents features defined for each pair of documents (d_i, d_j).
- **Correlation feature function** $g(y_{ij}, y_{jk})$. It denotes the correlation defined between y_{ij} and y_{jk}.
- **Constraint feature function** $h(a, y_{ij})$. It denotes constraint defined on y_{ij}.

The feature functions can be instantiated in different ways for different applications. In Section III, we use the publication data as the example to introduce how we define document-pair features. For most features, we consider them as binary features. For example, $f(x_{12} = \text{“}KDD\text{”}, x_{22} = \text{“}KDD\text{”}, y_{12} = 1)$ represents if both documents (papers) d_1 and d_2's venues are "KDD" and they are assigned to the same cluster, then the feature value is 1, otherwise 0. For TitleSim, we define the value of the feature as real value. Correlation feature function is useful for modeling to guarantee the logic consistency. For example, when $y_{12} = 1$ and $y_{23} = 1$, we should have $y_{13} = 1$ as well. $h(a, y_{ij})$ is a constraint feature function. It is used to incorporate the user feedbacks into the graph model. We see such a formulation elegantly combines all the different types of features and the user feedbacks into a unified model.

Inference Now, our problem is how to solve the objective function (Eq. 1), in particular, how to learn the parameters $\theta = (\{w_k\}, \{\mu\}, \{\alpha_l\})$ and the unknown hidden variables Y. However, without any constraints (e.g., user feedbacks or labeled training data), it is infeasible to learn the parameters θ and the hidden variables Y together. The general idea

Algorithm 1 The MH-based learning algorithm for PFG.

Input: number of iterations
Output: learned configuration for Y

```
Initialize all θ = ({w_k}, {μ}, {α_l}) as 1
Initialize all hidden variables Y = {y_ij} with y_ij = 0
repeat
    % sample a new configuration Y' based on q(Y'|Y)
    Y' ← q(Y'|Y)
    τ ∼ min(p(Y'|X,θ) / p(Y|X,θ), 1)
    toss a coin s according to a Bernoulli(τ, (1 − τ))
    if (s = 1) then
        % accept the new configuration Y'
        Y ← Y'
    end if
until convergence
return Y;
```

here is that when there is no constraint, we simply set the weights of all feature functions as 1 and aim to learn the hidden variables Y. When the user provides feedbacks to the system, we update the weights of feature functions.

It is still intractable to do exact inference in such a probabilistic graphical model. The intrinsic difficulty is to calculate the normalization factor Z, which sums up all possible configurations of Y. This makes the complexity exponential to the number of nodes in the graph. Several methods have been proposed to address this problem, such as Junction Tree [10] and Belief Propagation [11], to obtain an exact solution. In this paper, we use a sampling-based Metropolis-Hastings (MH) algorithm [12], a particular Markov-chain Monte Carlo method to achieve approximate inference. The advantage of the MH algorithm is that it can derive a global gradient update for each parameter (or hidden variable), thus can obtain better performance.

The learning algorithm is summarized in Algorithm 1. In each iteration of the learning algorithm, by the Metropolis-Hastings algorithm, we first sample a new configuration Y' conditioned on Y according to a proposal distribution $q(Y'|Y)$, which is defined over all possible configuration space $\mathcal{Y}$. The algorithm accepts the new configuration with an acceptance ratio τ. The proposal distribution $q(Y'|Y)$ can be defined in different ways. In this paper, we adopt the adaptive proposal distribution [13] with our attribute matrix X. We pick a cluster c_i randomly and use the attribute matrix to determine the probability ζ that all documents in the cluster are coreferent. So with probability ζ the cluster is selected for a merge, and by computing the probability distribution ρ_i over c_i being coreferent with each other cluster, we can draw a cluster $c_j \sim \rho_i$ to merge with c_i. Also with probability $1-\zeta$ cluster c_i is selected for a split, and by randomly selecting two documents d_a and d_b in c_i and placing them in separate clusters c_a and c_b, we can compute a Bernoulli distribution for each remaining document and decide whether to assign them to c_a or c_b.

V. Active Name Disambiguation

Given the initial results by a disambiguation algorithm (e.g., by the PFG model), which disambiguation results should we select to query the user? In particular, in our problem, which document pairs $\{(d_i, d_j)_k\}$ should we select to query? Moreover, when the user provides feedbacks (corrections) on the disambiguation results, how to efficiently and effectively refine the disambiguation (PFG) model? The first problem is referred to as active selection and the latter is called model refinement.

The goal of active selection is to select top K document pairs to query the user. Recently, the problem of active learning has seen a growing interest. For example, in [14], the authors focus on how to acquire the labels for a few nodes at inference time to improve the accuracy; in [15], an algorithm is proposed to effectively exploit the links between network data and the interactions between the local and collective aspects of a classifier to improve the accuracy of learning from fewer labeled examples. There are several basic requirements for the active selection: (a) the K document pairs are the most uncertain ones by the disambiguation algorithm; (b) feedbacks on the K document pairs can be used to help correct the other disambiguation results; (c) the active selection should be efficient.

A straightforward solution to the active selection problem is to select the most uncertain results by the disambiguation algorithm.

Uncertainty-based Active Selection (UB) According to Eq. 1, we could have the probability of two documents belonging to the same cluster, i.e.,

$$p(y_{ij}=1|\mathbf{x}_i,\mathbf{x}_j,\theta)=\frac{1}{Z_1}\exp\{\sum_k w_k f_k(\mathbf{x}_i,\mathbf{x}_j,y_{ij}=1)\} \quad (2)$$

If $p(y_{ij}=1|\mathbf{x}_i,\mathbf{x}_j,\theta)=0.5$, we say that the disambiguation model is the most uncertain about the document pair (d_i,d_j). $p(y_{ij}=1|.)=1.0$ and $p(y_{ij}=1|.)=0.0$ respectively denote that the model is confident in that the two documents d_i and d_j should be clustered together and should not be clustered. Based on the probability, we define a confidence score for a document pair (d_i,d_j) as

$$conf(d_i,d_j)=|p(y_{ij}=1|\mathbf{x}_i,\mathbf{x}_j,\theta)-0.5| \quad (3)$$

An uncertain document pair will have a low confidence score. Therefore, we select the K most uncertain document pairs with the lowest confidence scores according to Eq. 3.

Influence Maximization-based Active Selection (IM) The uncertainty-based method considers only the local information between two documents, but ignores the global information. We further propose a method which considers influence between documents. Intuitively, when a user corrects a document pair, it is expected that the correction can be propagated to the neighborhood pairs to further correct other

potential errors. To this end, we present a variation of the linear threshold model [16].

We define the document pair (d_i, d_j) as the node and e_{ij}^{jk} as the edge in the linear threshold model. (d_i, d_j) is called active when y_{ij} is determined. In this model there is a threshold which represents the confidence score for the document pair (d_i, d_j). When a pair is activated by the user, the gained score will be propagated to the other nodes in the PFG model uniformly. For example, one document pair node (d_i, d_j) has a confidence score 0.1. When the user specifies that the two documents should be assigned to the same person, i.e., $conf(d_i, d_j) = 0.5$, thus the gained score is $0.5 - 0.1 = 0.4$. Suppose in the PFG model, the node (d_i, d_j) has three neighborhood nodes, then each neighborhood node will receive a propagated score $\frac{0.4}{3}$. Further, we define an uncertain threshold (usually defined as the average uncertain score of all nodes). If the confidence score of a node is above the threshold, we call it active node, otherwise inactive node. Now, the problem is to choose the node which can maximally activate the other nodes by propagating its gained score. The problem is obviously NP-hard, and the proof and the approximate greedy algorithm are given in [16]. In this paper, we consider a simplified version of the greedy algorithm, which is validated to be effective and more efficient. Specifically, in each iteration, we select a document pair node with

$$\max_{y_{ij}} \sum_{e_{ij}^{jk} \in E} \mu \; g(y_{ij}, y_{jk}) \tag{4}$$

Model Refinement The objective of this step is to update the disambiguation model based on the user feedbacks. As for the PFG model, the task is cast as updating the parameters θ by maximizing a conditional probability $p(Y|X)$ (Eq. 1). Regarding the model refinement, we further have the constraint function derived from the user feedbacks. Here we adopt the SampleRank [13] algorithm to update the parameters $\theta \in \{w_k, \mu, \alpha_l\}$.

By collecting user data and manually checking its correctness, we can obtain the probability λ that user feedback is correct (toward better cluster partition). Thus with probability λ the new configuration (proposed by user) is preferred, indicated by scoring metric $S(y', y) > 0$, and with probability $1 - \lambda$ we have $S(y', y) < 0$. Then we update the parameter according to:

$$\theta = \begin{cases} \theta - \eta \cdot \phi_{y',y} & \text{if } S(y',y) < 0 \text{ and } M(y',y) > 0 \\ \theta + \eta \cdot \phi_{y',y} & \text{if } S(y',y) > 0 \text{ and } M(y',y) \leq 0 \end{cases} \tag{5}$$

where η is the learning rate and $M(y', y) = \phi_{y',y} \cdot \theta$ is the unnormalized log probability ratio according to the Metropolis-Hastings Model.

Improving Efficiency by Atomic Cluster In practice, however, in both the above selection criteria, we need to enumerate all possible document pairs, which is obviously time-consuming and infeasible for an online system (the disambiguation model is updated dynamically based on user feedbacks and the selection should also be dynamically updated according to the disambiguation model). We thus present an atomic cluster-based method to improve the efficiency. The idea is to first cluster the documents into atomic clusters. Each atomic cluster is a cluster of documents in which documents are closely connected (e.g., the probability $p(y_{ij} = 1|.) > threshold$). Documents with similarity less than the threshold will be assigned to disjoint atomic clusters. We explain the atomic cluster using an example from the publication data set: if two documents d_i^a and d_j^a have two or more common authors except a, we group the two documents into an atomic cluster. In this way, our active selection is performed on the cluster level instead of the document level. This could significantly improve the efficiency. Correspondingly, the selection criteria (Eqs. 3 and 4) are defined based on the atomic cluster by aggregating the scores of all document pairs in each atomic cluster. Algorithm 2 shows the atomic cluster generation process.

Algorithm 2 Atomic cluster generation.

Input: A list of publications which all share same author name
Output: A list of atomic clusters

```
Sort the publications in the descending order of their published year.
Create an empty atomic cluster list c = {Φ};
for each publication d_i^a in D^a do
    for each cluster c_j in the atomic cluster list c do
        if Bias-Classifier(d_i^a, c_j) > 0 then
            Assign the publication d_i^a into the cluster c_j;
        end if
    end for
    if no cluster assigned with publication d_i^a then
        Create a new cluster c_k with d_i^a and add c_k to the list c;
    end if
end for
```

As the bias-classifier, we use an adapted AdaboostM1 [17], [18]. In particular, we aim to find atomic clusters with high precision (but not necessarily high recall). In our setting, we want to minimize the number of false positives while tolerating the possible false negatives (thus called bias classifier). In order to bias the classifier, we introduce a related notion of asymmetric loss Asymmetric Loss [19]:

$$ALoss = \begin{cases} \sqrt{k} & \text{if } y_{ij} = 0 \text{ and } C(d_i, d_j) = 1 \\ \frac{1}{\sqrt{k}} & \text{if } y_{ij} = 1 \text{ and } C(d_i, d_j) = 0 \\ 0 & \texttt{otherwise} \end{cases} \tag{6}$$

where false positives cost k times more than false negatives, and $C(d_i, d_j) = 1$ indicates that two documents d_i and d_j are in the same atomic cluster and $C(d_i, d_j) = 0$ indicates they are not.

Our final clustering process stems from observations on how human beings disambiguate publications: (1) People would like to start from the most recent paper, when there

Table II
STATISTICS OF THE THREE DATA SETS.

Data Set	#Names	#Persons	#Documents
Publication	100	1,382	6,730
CALO	12	187	1,085
News Stories	380	755	20

is no ambiguity problem because there is only one paper; (2) By going on checking more published papers, people always would like to adopt the "easy first" and "high confidence first" strategy, that is, people first cluster the papers that she/he is more certain about; (3) For the rest papers that might be somehow difficult, people would adopt some compromising method. Our algorithm matches the three points above. We sort the papers by their published years, find atomic clusters, and employ the PFG model to obtain clustering results.

VI. EXPERIMENTS

In this section, we present experimental results of the proposed ADANA to evaluate its effectiveness. All data sets, codes, and related tools are publicly available. [2]

A. Experimental Setting

Data Sets We perform our experiments on three different genres of real-world data sets: Publication, CALO, and News Stories. Statistics of the three data sets are listed in Table II and more details are available online.

- Publication [8]. The Arnetminer system has collected about 1,300,000 publication papers from DBLP, 450,223 papers from IEEE, 1,343,442 papers and 3,687,675 citation relationships from ACM. By combining all the papers and removing papers with incomplete information, we finally have a publication data set of 1,632,442 papers and 3,021,489 citation relationships. For evaluation, we manually labeled 6,730 papers for 100 author names.
- CALO [5]. It contains a labeled data set of 1,085 Web pages for 12 person names. The data set is the email directory of one participant (i.e., Melinda Gervasio) of the CALO project. The 12 names appear in headers of messages in the email directory. On average, each person name corresponds to about 15 different persons. The task is to associate the emails to different persons.
- News Stories [9]. It consists of 755 ambiguous entities appearing in 20 news pages. The task is to cluster the ambiguous entities into different groups.

Evaluation Measures and Baseline Methods We evaluate the proposed method in terms of Precision, Recall, and F1 score. On the publication data set, we compare our method with several existing methods for name disambiguation, including SA-Cluster [20], DISTINCT [4], HAC [3], and CONSTRAINT [21]. In the SA-Cluster method, we use coauthor relationship as the edge, and all the other relationships as the attribute features. In DISTINCT and CONSTRAINT, we use all the information to calculate similarity between papers. In HAC, we search each citation in Google Engine and weight by its IHF. On the CALO data set, we compare with the LS+A/CDC method in [5]. On the News Stories data set, we compare with the baseline method and the proposed method in [9]. In the rest of this section, we will first present experimental results of name disambiguation by different methods on all three data sets and then focus on the detailed analysis and active name disambiguation using the publication data set.

[2]http://www.arnetminer.org/disambiguation

B. Results of Name Disambiguation

As all the comparison methods require the number of persons sharing the same name, we use the actual person number in the labeled training data as the input. Our method does not need the number as input.

Results on Publication Data Set Figure 2 shows the average disambiguation performance on the publication data set by the different methods in terms of pairwise precision, recall and F1-score. We see that our method clearly outperforms all the comparison methods. On average, our method achieves a precision of 95.4%, recall 85.6%, and F1-score 89.2%. Table III lists details of the experiments on each of the 100 author names. In general, our method can achieve good performance (>85%). For a few cases such as J. Guo and Rafael Alonso, even human cannot make a good disambiguation. The results also indicate that automatic name disambiguation is still insufficient, even with the various features, thus user interaction is necessary. In addition, our method can accurately find the number of persons. Due to space limitation, we do not list the results here. Interested readers please refer to the online page.[2]

Here we perform analysis to evaluate the contribution of different features defined in our method. We first rank the individual features by their performance, then add those features one by one in the order of their disambiguating power. In particular, we first use CoAuthor (A), followed by adding CoAffiliation (O), and then Homepage (H), TitleSim (T), Citation (C), CoVenue (J), and CoAffOccur. In each step, we evaluate the performance of our method. Figure 3 shows the average pairwise precision, recall and F1-score of our method with different feature combinations. By adding a new feature, our method clearly improves the disambiguating performance, which indicates that our method works well by integrating the different features for name disambiguation and each defined feature in our method contributes improvement in the performance.

Results on CALO Data Set Table IV lists the results of our method and the LS+A/CDC method in [5] on the CALO data set. We see that our approach outperforms the LS+A/CDC method, with 1.2% improvement on F1-score.

Table III
RESULTS FOR THE 100 AUTHOR NAMES ON THE PUBLICATION DATA SET.

Name	Rec.	Prec.	F1	Name	Rec.	Prec.	F1	Name	Rec.	Prec.	F1	Name	Rec.	Prec.	F1
Michael Smith	0.79	1.0	0.88	Philip J. Smith	0.84	0.85	0.85	Yoshio Tanaka	0.86	1.0	0.92	Yang Yu	0.84	1.0	0.91
Jose M. Garcia	0.95	1.0	0.97	John F. McDonald	0.88	1.0	0.94	Z. Wang	0.9	0.75	0.82	Yue Zhao	0.98	0.92	0.95
Lu Liu	0.9	0.89	0.9	Jing Zhang	0.84	0.71	0.77	David Cooper	0.9	1.0	0.95	John Collins	1.0	1.0	1.0
Wen Gao	0.97	0.98	0.97	Fan Wang	0.94	1.0	0.97	F. Wang	1.0	1.0	1.0	Keith Edwards	0.41	1.0	0.58
Hui Fang	0.98	0.98	0.98	Paul Wang	0.86	1.0	0.92	Alok Gupta	0.9	1.0	0.95	Hui Yu	0.95	0.83	0.88
Qiang shen	1.0	1.0	1.0	Kai Tang	0.87	1.0	0.93	Ping Zhou	0.82	1.0	0.9	Yan Tang	0.93	1.0	0.97
Peter Phillips	0.74	1.0	0.85	Wei Xu	0.82	0.99	0.9	Michael Lang	1.0	1.0	1.0	Manuel Silva	0.97	1.0	0.98
Charles Smith	1.0	1.0	1.0	Thomas Zimmermann	0.91	1.0	0.95	Yu Zhang	0.81	0.76	0.79	Kuo Zhang	0.82	0.87	0.84
Thomas Meyer	0.72	1.0	0.84	William H. Hsu	0.88	1.0	0.94	Frank Mueller	0.92	1.0	0.96	Gang Chen	0.55	0.72	0.63
Xiaoming Wang	0.88	1.0	0.94	Eric Martin	1.0	1.0	1.0	Kai Zhang	0.83	0.89	0.86	Fei Su	1.0	1.0	1.0
Paul Brown	0.73	1.0	0.85	Jie Tang	1.0	1.0	1.0	Feng Liu	0.57	0.55	0.56	Robert Schreiber	0.77	1.0	0.87
Satoshi Kobayashi	0.76	1.0	0.86	Lei Jin	1.0	1.0	1.0	R. Balasubramanian	0.69	1.0	0.82	David Jensen	0.92	0.96	0.94
Thomas Wolf	0.87	1.0	0.93	Li Shen	0.79	0.99	0.88	Hao Wang	0.78	1.0	0.88	Robert Allen	0.87	1.0	0.93
Steve King	0.46	1.0	0.63	Lei Chen	0.87	0.96	0.91	Koichi Furukawa	0.92	1.0	0.96	Thomas Tran	1.0	1.0	1.0
Thomas Hermann	0.71	0.98	0.83	J. Guo	0.67	0.4	0.5	John Hale	0.83	1.0	0.91	Jie Yu	1.0	1.0	1.0
Yun Wang	0.71	1.0	0.83	Ji Zhang	0.96	0.78	0.86	Mark Davis	0.99	1.0	0.99	David Brown	1.0	1.0	1.0
Cheng Chang	0.65	1.0	0.79	Gang Luo	0.94	0.98	0.96	Xiaoyan Li	1.0	1.0	1.0	Bin Li	0.95	0.55	0.69
Bing Liu	0.97	0.97	0.97	R. Ramesh	0.69	1.0	0.82	Jianping Wang	1.0	0.89	0.94	Barry Wilkinson	1.0	1.0	1.0
David E. Goldberg	0.99	1.0	1.0	Feng Pan	0.58	0.99	0.73	David Nelson	0.72	1.0	0.84	Lei Fang	1.0	1.0	1.0
Rakesh Kumar	0.94	1.0	0.97	Thomas D. Taylor	1.0	1.0	1.0	Jeffrey Parsons	0.93	1.0	0.97	Richard Taylor	0.7	0.98	0.82
Jim Gray	0.86	1.0	0.92	Juan Carlos Lopez	0.94	1.0	0.97	Sanjay Jain	0.98	1.0	0.99	Ajay Gupta	0.56	1.0	0.72
David Levine	0.95	0.91	0.93	Shu lin	0.82	1.0	0.9	Michael Siegel	0.87	1.0	0.93	S. Huang	1.0	1.0	1.0
Bin Zhu	0.81	0.98	0.88	Young Park	0.84	1.0	0.91	Yi Deng	1.0	0.9	0.95	Daniel Massey	1.0	1.0	1.0
Bob Johnson	0.4	1.0	0.57	Michael Wagner	0.63	1.0	0.77	Ning Zhang	0.91	1.0	0.95	David C. Wilson	0.93	1.0	0.96
Wei Wang	0.94	0.68	0.79	Yong Chen	0.72	0.86	0.78	Rafael Alonso	0.38	1.0	0.55	Bin Yu	0.93	0.92	0.93

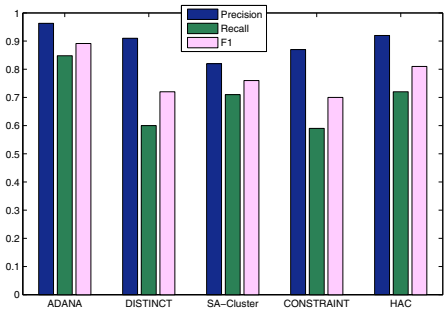

Figure 2. Performance (F1-score) of the comparison methods.

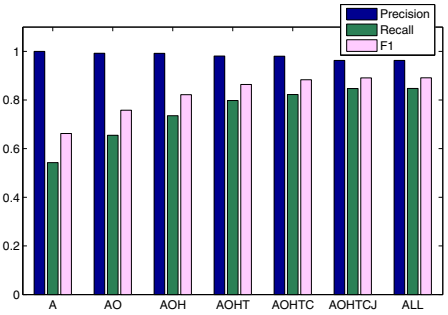

Figure 3. Feature contribution analysis.

Results on News Stories Data Set Table V lists the results of our method on the News Stories data set. The accuracy of the two comparison methods are obtained from [9]. We see that our approach clearly outperforms both the baseline method and the proposed method in [9], with improvement on accuracy from 51.7% (baseline) and 91.4% (proposed in [9]) to 97.3% in our approach (our method also achieves a recall of 91.1%, precision 82.4%, F1-score 84.8%)[3].

Table IV
RESULT ON THE CALO DATA SET.

Method	Recall	Precision	F1-score
LS+A/CDC [5]	0.745	0.869	0.803
Our Approach	0.761	0.878	0.815

Table V
RESULT ON THE NEWS STORIES DATA SET.

Method	Baseline in [9]	Approach in [9]	Our Approach
Accuracy	0.517	0.914	0.973

C. Results of Active Name Disambiguation

As the CALO only contains 12 persons and the News Stories only has 20 news pages, for the active name disambiguation, we focus on the Publication data set. In particular, we compare the results of the proposed algorithm for active name disambiguation and name disambiguation with random selection (randomly selecting a pair of documents to query the user for feedbacks). We also evaluate the effectiveness of the model refinement. Table VI shows the results of active name disambiguation. Figure 4 shows the variation of F1-score with the number of queries. We see that only with 5 active interactions by our approach (with IM), we can achieve a performance of 95.9% in terms of PairwiseF1 score; with 10 active interactions, we can obtain a performance of 96.9%,

[3]Here we only list the score of accuracy as it is the only metric presented in [9]. We have confirmed that with the authors of [9].

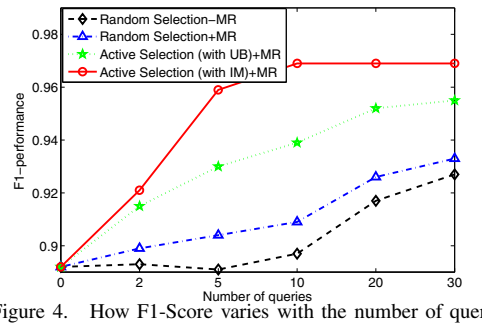

Figure 4. How F1-Score varies with the number of queries.

which is much better than the random selection strategy. With 5 active interactions, the performance gain of our approach (6.7% with IM) is more than 500% of the random selection strategy (1.2%).

We can also see that the strategy of influence maximization-based active selection performs better than the uncertainty-based strategy. With 5 active interactions, the performance gain obtained by the influence maximization-based strategy is 6.7% against 3.8% by the uncertainty-based strategy. This indicates that a selection strategy using only local information is insufficient and a strategy by considering both local and global information is necessary.

VII. Related Work

Generally speaking, existing methods for name disambiguation mainly fall into three categories: supervised-based, unsupervised-based, and constraint-based. The supervised-based approach (e.g., [1]) tries to learn a classification model for each author name from the human labeled data. Then the learned model is used to predict the author assignment of each paper. In the unsupervised-based approach (e.g., [2], [4]), clustering algorithms or topic models are employed to find paper partitions. Papers in different partitions are assigned to different persons. The constraint-based approach also utilizes the clustering algorithms. The difference is that user-provided constraints are used to guide the clustering towards better data partitioning (e.g., [22], [21]).

Recently, a few efforts have been made to combine the graphical information or external information to help name disambiguation. For example, McRae-Spencer and Shadbolt [23] present a graph-based approach to author disambiguation on large-scale citation networks by using self-citation and coauthor relationships. The approach can achieve a high precision but a relatively low recall. Bunescu and Pasca [24] propose a disambiguation method based on SVM kernel by exploiting the high coverage and rich structure of the knowledge encoded in an online encyclopedia. Yu et al. [25] have developed supervised approaches to identify the full forms of ambiguous abbreviations within the context they appear. In [3], a search engine based on clustering method is proposed. It represents the features of each citation as relevant URLs from search engine and weights it by its IHFs. Chen et al. [26] study how to combine the different disambiguation approaches and propose an entity resolution ensemble framework, which combines the results of multiple base-level entity resolution systems into a single solution to improve the accuracy of entity resolution. Whang et al. [27] propose an iterative blocking framework where the resolution results of blocks are reflected to subsequently processed blocks. Wick et al. [28] propose a unified approach for schema matching, coreference and canonicalization. However, most existing methods ignore user interaction. The disambiguation performances of these methods vary between 70%-88%. With the increase of the complexity of the data set, the performance of these methods degrades quickly (e.g., when we directly apply the methods to our data sets).

A few algorithms also try to consider user interaction, such as [7] which provides an interactive system. The system allows the user to locate the occurrences of named entities within a given text. However, it only allows the user to correct the mistaken disambiguation she/he found, but does not consider how to maximize the user guidance (such as correction propagation). Some other works also consider the large-scale issue in name disambiguation. [9] presents a large-scale system for the recognition and semantic disambiguation of named entities based on information extracted from Wikipedia and Web search results. However, in a large-scale system, it is highly infeasible for the user to manually check the disambiguation results. It is extremely necessary to have a mechanism that can actively select a small number of results to query the user so as to maximally improve the disambiguation accuracy.

Also there are works of name disambiguation with active learning like [29], which proposed a supervised online active selection support vector machine algorithm (LASVM) and used active sample selection by choosing the most informative sample which should be the one closest to the hyperplane. However, they only validate the approach on a small publication data set with 10 author names and 3,335 publications. In this paper, we propose a flexible pairwise factor graph model, which can be easily extended to different applications, and has been validated on the publication data, web page data, and the news story data.

VIII. Conclusion

In this paper, we study the problem of active name disambiguation. We propose a pairwise factor graph (PFG) model, which is able to incorporate various types of features as well as user feedbacks into a unified model. Based on the PFG model, we propose an active name disambiguation algorithm to improve the disambiguation performance by actively select and query the user with document pairs. Experimental results show that on all the three different genres of data sets our proposed method clearly outperforms the existing methods. Experiments also show that with only a few user corrections, the error rate of name disambiguation

Table VI
RESULT OF ACTIVE NAME DISAMBIGUATION. MR: MODEL REFINEMENT; UB: UNCERTAINTY-BASED ACTIVE SELECTION; IM: INFLUENCE MAXIMIZATION-BASED ACTIVE SELECTION.

Method	Random Selection-MR			Random Selection+MR			Active Selection (with UB)+MR			Active Selection (with IM)+MR		
#Query	Recall	Precision	F1-score	Recall	Precision	F1-score	Recall	Precision	F1-score	Recall	Precision	F1-score
0	0.856	0.954	0.892	0.856	0.954	0.892	0.856	0.954	0.892	0.856	0.954	0.892
2	0.857	0.954	0.893	0.867	0.953	0.899	0.896	0.953	0.915	0.892	0.955	0.921
5	0.855	0.954	0.891	0.873	0.953	0.904	0.922	0.952	0.930	0.976	0.953	0.959
10	0.863	0.956	0.897	0.885	0.951	0.909	0.937	0.953	0.939	0.994	0.952	0.969
20	0.889	0.963	0.917	0.905	0.959	0.926	0.958	0.953	0.952	0.996	0.951	0.969
30	0.903	0.964	0.927	0.915	0.961	0.933	0.965	0.953	0.955	0.997	0.951	0.969

can be reduced to 3.1%. A real system has been developed based on the proposed method and is available online.

Acknowledgements Jie Tang and Xuezhi Wang are supported by the Natural Science Foundation of China (No. 61073073) and Chinese National Key Foundation Research (No. 60933013, No.61035004). Hong Cheng was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (No. CUHK 411310). Philip Yu is supported in part by US NSF through grant IIS-0905215, and DBI-0960443.

REFERENCES

[1] H. Han, C. L. Giles, H. Zha, C. Li, and K. Tsioutsiouliklis, "Two supervised learning approaches for name disambiguation in author citations," in *JCDL'04*, 2004, pp. 296–305.

[2] H. Han, H. Zha, and C. L. Giles, "Name disambiguation in author citations using a k-way spectral clustering method," in *JCDL'05*, 2005, pp. 334–343.

[3] Y. F. Tan, M.-Y. Kan, and D. Lee, "Search engine driven author disambiguation," in *JCDL'06*, 2006, pp. 314–315.

[4] X. Yin, J. Han, and P. S. Yu, "Object distinction: Distinguishing objects with identical names," in *ICDE'07*, 2007, pp. 1242–1246.

[5] R. Bekkerman and A. McCallum, "Disambiguating web appearances of people in a social network," in *WWW'05*, 2005, pp. 463–470.

[6] J. Tang, A. C. Fong, B. Wang, and J. Zhang, "A unified probabilistic framework for name disambiguation in digital library," *IEEE Transactions on Knowledge and Data Engineering*, vol. 99, no. PrePrints, 2011.

[7] P. T. Davis, D. K. Elson, and J. L. Klavans, "Methods for precise named entity matching in digital collections," in *JCDL'03*, 2003, pp. 125–127.

[8] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: Extraction and mining of academic social networks," in *KDD'08*, 2008, pp. 990–998.

[9] S. Cucerzan, "Large-scale named entity disambiguation based on wikipedia data," in *EMNLP'07*, 2007, pp. 708–716.

[10] W. Wiegerinck, "Variational approximations between mean field theory and the junction tree algorithm," in *UAI'00*, 2000, pp. 626–633.

[11] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Generalized belief propagation," in *NIPS'01*, 2001, pp. 689–695.

[12] S. Chib and E. Greenberg, "Understanding the metropolis-chastings algorithm," *American Statistician*, vol. 49, no. 4, p. 327C335, 1995.

[13] M. Wick and A. McCallum, "Advances in learning and inference for partition-wise models of coreference resolution," in *University of Massachusetts Technical Report, UM-CS-2009-028*, 2009.

[14] M. Bilgic and L. Getoor, "Active inference for collective classification," in *AAAI'10*, 2010.

[15] M. Bilgic, L. Mihalkova, and L. Getoor, "Active learning for networked data," in *ICML'10*, 2010.

[16] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," in *KDD'03*, 2003.

[17] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," in *EuroCOLT'95*. London, UK: Springer-Verlag, 1995, pp. 23–37.

[18] Y. Freund and R. Schapire, "Experiments with a new boosting algorithm," in *ICML'96*, 1996, pp. 148–156.

[19] P. A. Viola and M. J. Jones, "Fast and robust classification using asymmetric adaboost and a detector cascade," in *NIPS'01*, 2001, pp. 1311–1318.

[20] Y. Zhou, H. Cheng, and J. X. Yu, "Graph clustering based on structural/attribute similarities," *Proc. VLDB Endow.*, vol. 2, no. 1, pp. 718–729, 2009.

[21] D. Zhang, J. Tang, and J. Li, "A constraint-based probabilistic framework for name disambiguation," in *CIKM'07*, 2007, pp. 1019–1022.

[22] S. Basu, M. Bilenko, and R. J. Mooney, "A probabilistic framework for semi-supervised clustering," in *KDD'04*, 2004, pp. 59–68.

[23] D. M. McRae-Spencer and N. R. Shadbolt, "Also by the same author: Aktiveauthor, a citation graph approach to name disambiguation," in *JCDL'06*, 2006, pp. 53–54.

[24] R. C. Bunescu and M. Pasca, "Using encyclopedic knowledge for named entity disambiguation," in *EACL'06*, 2006, pp. –1–1.

[25] H. Yu, W. Kim, V. Hatzivassiloglou, and J. Wilbur, "A large scale, corpus-based approach for automatically disambiguating biomedical abbreviations," *ACM TOIS*, vol. 24, no. 3, pp. 380–404, 2006.

[26] Z. Chen, D. V. Kalashnikov, and S. Mehrotra, "Adaptive graphical approach to entity resolution," in *JCDL'07*, 2007, pp. 204–213.

[27] S. E. Whang, D. Menestrina, G. Koutrika, M. Theobald, and H. Garcia-Molina, "Entity resolution with iterative blocking," in *SIGMOD'09*, 2009, pp. 219–232.

[28] M. L. Wick, K. Rohanimanesh, K. Schultz, and A. McCallum, "A unified approach for schema matching, coreference and canonicalization," in *KDD'08*, 2008, pp. 722–730.

[29] J. Huang, S. Ertekin, and C. L. Giles, "Efficient name disambiguation for large-scale databases," in *PKDD'2006*, 2006.