Name Disambiguation Using Web Connection

Yiming Lu[1*], Zaiqing Nie[†], Taoyuan Cheng[2*], Ying Gao[2*], Ji-Rong Wen[†]

[†]Microsoft Research Asia, Beijing, China

[1]University of California, Irvine, U.S.A.

[2]Renmin University of China, Beijing, China

[†]{znie, jrwen}@microsoft.com, [1]yimingl@uci.edu, [2]{chengty,gaoying2}@ruc.edu.cn

ABSTRACT

Name disambiguation is an important challenge in data cleaning. In this paper, we focus on the problem that multiple real-world objects (e.g., authors, actors) in a dataset share the same name. We show that Web corpora can be exploited to significantly improve the accuracy (i.e. precision and recall) of name disambiguation. We introduce a novel approach called WebNaD (Web-based Name Disambiguation) to effectively measure and use the Web connection between different object appearances of the same name in the local dataset. Our empirical study done in the context of Libra, an academic search engine that indexes 1 million papers, shows the effectiveness of our approach.

INTRODUCTION

Name disambiguation is one of the major challenges when integrating data from multiple data sources. It usually has such a problem: an object type has few other attributes other than its name, and targets to distinguish among two or more objects which share the same name. For example, when you search for the publications of "Lei Zhang", a very common Chinese name, in DBLP [1], different author objects are mixed in the same Webpage (see Figure 1).

This problem relies more on additional information besides the attribute values, and is challenging especially when there are few attributes. In this paper, we focus on solving this problem.

Motivation example: Our major motivation comes from our Libra academic search engine (http://libra.msra.cn). Libra is an object-level search engine and those who use it can search various types of objects such as papers, authors, conferences, and journals. The object metadata and their relationship information is extracted from papers, Webpages, and feed data from publishers. It is relatively easy to integrate papers, conferences, and journals, since they have enough differentiating attribute values. However, for authors, few other attributes are available except for names. How to disambiguate authors with the same name becomes a challenging problem.

Usually there are multiple types of objects and rich information about the relationships among these objects in an object-level search engine like Libra. In Figure 2 we show the relationships among different types of objects in Libra. This relationship information can be modeled as an entity-relationship graph. Some recent work, such [2] [3], proposes to exploit the connection via relationship in the graph for name disambiguation. Their main idea is that if two identical names in different contexts refer to the same object, they are more likely to be strongly connected in the entity-relationship graph. For example, if two "Lei Zhang"s refer to the same person, it is very likely that they share some coauthors, references, or are indirectly related by a chain of relationships.

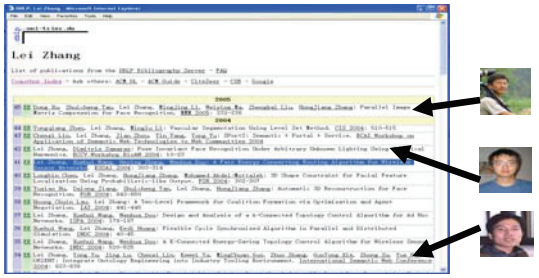

Figure 1. Three "Lei Zhang" are found in DBLP.

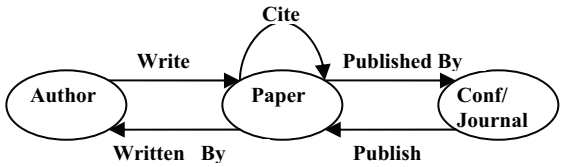

Figure 2. The relationships among author, paper, conference/journal objects

The success of the existing connection-based approaches relies on the sufficiency of relationship information. However, in real applications, many relationships may be missing in the local dataset. For example, it always happens that some citation links between papers are missing in Libra because of the limitation of reference extraction techniques. Therefore, an approach that leverages more information besides the connection evidence in the local dataset (i.e. local connection) is desired.

The Web is a rich and ubiquitous data source implicitly containing various kinds of real-world objects and their corresponding relationship information. Based on this fact, we explore ways to utilize object information on the Web to improve the accuracy of name disambiguation.

One basic assumption behind our method is that, for different object appearances with identical names that refer to the same object, their contexts should be strongly connected with each other on the Web. We call such connections *Web Connections*. In this paper, we propose an approach called *WebNaD* (Web based Name Disambiguation), which exploits not only the local connection, but also the Web connection for name disambiguation.

We measure the Web connection based on the co-occurrence of the contexts in a Website (or Webpage). The co-occurrence information could be easily obtained by sending the context information as queries to a search engine (e.g., Google, MSN Search). However, to measure the Web connection strength is not

trivial, since co-occurrences in different Websites usually indicate quite different connection strengths. Even in the same Website, different co-occurrences need to be discriminated (e.g., co-occurrences in the same Webpage usually indicate stronger connection than those in different Webpages.).

To leverage the Web connection for name disambiguation, *WebNaD* extends existing local connection-based approaches by combining both local and Web connections. Our empirical study of *WebNaD* in the context of Libra shows that *WebNaD* performs much better than the local connection-based approaches.

The main contributions of our papers are as follows:

1. We propose a novel name disambiguation approach that exploits Web connection evidence in addition to local connection;
2. We propose an effective measure of the Web connection between different object appearances based on their Website co-occurrences, which could be efficiently obtained by using a Web index or even a search engine;
3. An systematic experiment is studied in the context of our paper search engine — Libra.

The rest of paper is organized as follows. In the next section, we will provide an overview of our problem and *WebNaD* approach; in Section 3, we describe the measure of Web connection; in Section 4, we will summarize the whole framework of *WebNaD*; in Section 5, we will show our experiment results in the context of Libra; Section 6 is a description of related work, and Section 7 is a conclusion.

OVERVIEW

Problem Definition

There are usually multiple types of objects and corresponding relationships in Libra. However, what we get from our extraction programs (or from structured data feeds) are usually *appearances* (i.e. sets of attribute values) of real-world objects. Each appearance contains only partial information about an object and usually has some representative information to denote the object (e.g., the name of an author, the title of a paper, etc.). Such information can be seen as a general **name** of the corresponding appearance.

For some object types, such as paper, conference and journal, names are discriminative enough and each name will very likely refer to a single object. However, for some other object types, such as author, simply grouping appearances sharing the same name will falsely merge information of different objects (false positives). For those object types, we need to first detect the ambiguous names (i.e., the names referring to multiple objects), and then do the disambiguation for those names.

For all the appearances sharing a certain name, the goal of name disambiguation is to partition them so that each partition corresponds to an object in the real world, while different partitions correspond to different objects. Putting any two appearances in the same partition is called as a **match**.

We evaluate the quality of name disambiguation by precision and recall. For a certain name, precision measures the percentage of the correctly detected appearance matches over all the appearance matches; and recall measures the percentage of correctly detected appearance matches over all the appearance pairs actually referring to the same object.

Overview of WebNaD Approach

There are several possible ways to detect Web connection. For instance, we can crawl the Web to construct a Web graph, and if some hyperlinks (directly or indirectly) connect Webpages containing the contexts of two appearances, the hyperlink chain can be considered as Web connection. Another way is to use sophisticated text mining methods for the content of Webpages to judge if the contexts of two appearances really correlate. However, these methods are either very costly, or are complicated, especially given that a normal user usually have only the search engine (or Web index) at hand. In this paper, we propose a relatively practical way to detect Web connection by easily utilizing the search function of the conventional search engines. We define the *Web occurrence* as search results when using some distinctive information of that object as a query to Web index or search engine, and define *context-object* as the directly related objects of an appearance. For example, if papers P_1 and P_2 are authored by one "Lei Zhang", we can say they are the context-objects of the two appearances of "Lei Zhang".

Two objects co-occurring in the same Webpage/Website can be seen as both related to the same Webpage/Website editor (i.e. a person or an organization) in a certain way, so we can use them as the Web connections. Intuitively, the *Website co-occurrence* (i.e. the co-occurrence in the same Website) is a more general version of the *Webpage co-occurrence* (i.e. the co-occurrence in the same Webpage), and will very likely result in higher recall.

WEB CONNECTION CALCULATION

In this section, we discuss how to measure the Web connection between two context-objects based on their Web occurrences. As mentioned in the overview section, Website co-occurrences will produce higher recall, and Webpage co-occurrences can be treated as specific cases of Website co-occurrences, so we focus on how to effectively measure Web connection based on Website co-occurrences in this section. We first discuss how to calculate the site-level Web connection (i.e. the Web connection at a given site) (Section 3.1). Then, we introduce how to calculate the Web connection strength by combining of site-level connections at different sites (Section 3.2 and 3.3). After that, we discuss how to deal with the practical issues in using Web connection (Section 3.4).

Site-level Web Connection Calculation

A natural way to measure the site-level connection between the context-objects o_1 and o_2 in a certain site s is as follows:

$$c_s(o_1,o_2)=\begin{cases}1, \text{if } o_1 \text{ and } o_2 \text{ co-occur in site s}\\ 0, \text{otherwise}\end{cases}$$

However, this measurement is imprecise in terms of all the co-occurrences at a certain site having the same connection strength.

A simple example is that co-occurrence in a page tends to show stronger connection than two occurrences in different pages.

To discriminate different co-occurrences in a given site, we propose to take into account the URL distance between the two Web occurrences in the Website.

The logical organization of a Website can be seen as a hierarchy, and the URL of a Webpage is an indicator of its logical position in the hierarchy structure. A URL is split by slashes into several layers, and each layer can be considered the name of a domain, a directory or a file. The editor of the Website usually tends to put similar Webpages as near as possible in the hierarchy structure, therefore we assume that with more layer matches in two URLs, a stronger connection can be indicated. For instance, in our "Lei Zhang" example, the co-occurrences whose URLs share the prefix http://wotan.liu.edu/docis/dbl/acmmmm/ indicates stronger connection evidence than the co-occurrences whose URLs only share http://wotan.liu.edu/docis/dbl/, because the former one may shows that two "Lei Zhang"s are both related to the ACM Multimedia Conference, a much smaller community than the whole computer science community.

Therefore, we can measure the distance between u_1 and u_2 with layer matches, and use the total number of layers of u_1 and u_2 for normalization. The calculation is shown as follows:

$$Sim(u_1,u_2) = \frac{2l_{matched}}{(l_{layer1} + l_{layer2})}$$

where

- $Sim(u_1,u_2)$ is the similarity of u_1 and u_2 based on URL layers, and indicates the distance of u_1 and u_2 in the hierarchy structure of a Website;
- l_{layer1}, l_{layer2} are the length of layers of u_1 and u_2;
- $l_{matched}$ is the length of matched layers of u_1 and u_2.

Given the similarity based on URLs, we can define a site-level connection between the context-objects o_1 and o_2 at site s as follows:

$c_s(o_1,o_2) = \underset{\forall u_j,\forall u_k}{Max}(Sim(u_j,u_k))$, which satisfy that o_1 occurs in page u_j, o_2 occurs in page u_k, and u_j, u_k are both in site s.

Obviously, if o_1 and o_2 co-occur in the same Webpage, we can get maximum site-level connection at that site (i.e. the URL distance is equal to 1).

Some sites may use a nearly flat schema with numerical identifiers, such as http://portal.acm.org, and URL distance seems not a good indicator of similarity in such case. However, we argue that it could be interpreted that all the occurrence pairs are with the same similarity in the site editor's eye. In the next two subsections, we will introduce two novel ways of combining the site-level connections by weighting the Websites unequally.

Linear Combination of Site-level Web Connections

To combine the site-level connections at different sites, it should be noticed that not all sites play the same role for name disambiguation. In our "Lei Zhang" example, we may find two Website co-occurrences of papers. One is http://research.microsoft.com/, the Website of Microsoft Research, while the other one is DBLP [1]. Although the two sites both represent an academic community, the former co-occurrence usually provide stronger connection for a match. This is because the former one shows that two "Lei Zhang"s are probably both the members of Microsoft, a relatively small community, while the latter one only shows that they both work in the computer science domain.

Intuitively, for a given object type, the fewer objects one site contains, the more connection evidence can be indicated if two context-objects co-occur in it. This shares a similar idea as the IDF in the information retrieval field. Thus, we introduce the concept of *coverage* of a site:

Definition 3.1. Given a certain type of object set C, the coverage score of each site S is:

$$coverage(S) = \frac{|\{x \mid x \in C, x \text{ occurs in } S\}|}{|\{x \mid x \in C\}|}$$

With this concept, given n Websites, a straightforward way to combine site-level connection between O_1 and O_2 of different sites is a linear combination using the inverse coverage score of each site, as follows:

$$Conn(O_1,O_2) = \sum_{s=1}^{n} c_s w_s$$

where

- $Conn(O_1,O_2)$ represents the Web connection score of context-objects O_1 and O_2;
- c_s is the site-level connection between O_1 and O_2 of site s;
- w_s indicates the inverse coverage of site s, $w_s = \frac{1}{coverage(s)}$.

A Learning Approach to Web Connection Calculation

The linear combination approach in the above section is problematic: for the purpose of name disambiguation, the relative importance of each site is usually domain-specific and not exactly proportional to the inverse coverage score. For example, in our dataset, www.cs.cmu.edu has much more coverage than www.springerlink.com, but it is obvious that co-occurrence in the publications of CMU provides stronger connection than the co-occurrence in the paper collection of springerLink, an online paper search portal. Thus simply using the inverse coverage score for combination may not be reasonable for the name disambiguation purpose. A more adaptive combination could be learned if a set of pairs, each of which are labeled as either a match or non-match, are available.

Actually, the essential in using site-level connection for name disambiguation is to determine whether the Website co-occurrences indicate strong enough connection to match the corresponding appearance pair. This is essentially a binary classification problem. If we use a feature vector $p^{(O_1,O_2)}$, in which each dimension is the Web connection between the context-objects O_1 and O_2 in one of the sites, to represent a context-object pair, the task can be formalized as follows:

$$p^{(O_1,O_2)} = \langle c_s \rangle \to 1,-1$$

where c_s is the site-level connection between O_1 and O_2 at site s.

For this binary classification task, we choose support vector machine (SVM) [4] for this classification task because of its learning capability from limited training sets of high-dimensional data and resilience of noise.

SVM is based on the idea of structural risk minimization rather than empirical risk minimization. It maps the input vectors to a high-dimensional space, and applies a hyperplane which leaves the maximum margin between two classes. Given a labeled training data set $D = \{X_i, y_i\}_{i=1}^{t}$, where $y_i \in \{1,-1\}$, the corresponding decision function of SVM has the following form:

$$f(X) = sign\left(\sum_{i=1}^{t} \alpha_i y_i K(X_i, X) - b\right)$$

where K is the kernel function. The typical kernel functions include polynomial kernel, Gaussian RBF (radial basis function) kernel, sigmoid kernel.

For name disambiguation problem, the prediction score of the decision function can be seen as the confidence whether the corresponding appearance pair are of the same object. Thus, if a set of training data labeled as either match or non-match is available, the Web connection between O_1 and O_2 can be measured by the prediction score, as follows:

$$Conn(O_1, O_2) = f(P^{(O_1,O_2)})$$

This learned Web connection function will give more weight to those sites in which the co-occurrence indicates stronger evidence for match.

Practical Considerations

There are still some issues in computing the site-level Web connection. First, when only a search engine is given, all the Web sites are "hidden" behind the search interface, and cannot be obtained beforehand. Second, even if all the Web sites are known beforehand (e.g., we have already indexed the Web corpora locally), there are usually billions of Web sites in Web corpora, and to train a connection function for such a high dimension is nearly out of the question in practical.

To deal with these issues, we first present some useful observations on the Web corpora according to a study in Libra.

First of all, for a given type of objects, the coverage distribution of each site usually obeys a power-law distribution. By the study on the distribution of around 1 million papers of Libra in the Web corpora (using the paper titles to get the Web occurrences of paper objects), as illustrated in Figure 3, only a few sites are the ones with high coverage; while the "massive many" are the sites which rarely contain papers. The discovered sites with relatively higher coverage are not surprising: many of them are the Websites which provide paper search service or the Websites of prestigious research organizations.

Second, although co-occurrences of two context-objects in different sites are of different importance for name disambiguation and are not necessarily proportional to the inverse coverage score, we find that co-occurrences of context-objects in sites with extremely low coverage score (i.e., the sites whose coverage scores are below a certain threshold) always provide strong enough evidence for a correct match. For example, if two papers "Emergent Semantics from Folksonomies: A Quantitative Study" and "Integrating Web Services into Ontology-based Web Portal" can both be found in http://apex.sjtu.edu.cn/, a site of a research lab focusing on knowledge management, then the "Lei Zhang" of those two papers will most likely refer to the same person. For the purpose of convenience, we call these sites as small hubs, while the ones with high coverage score are called as big hubs.

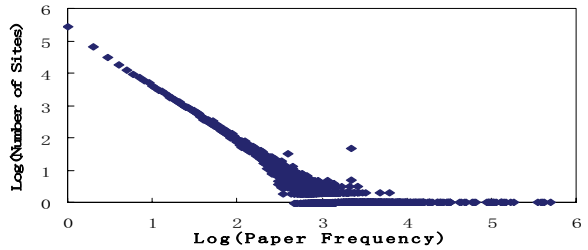

Figure 3. Power Law Distribution of Paper Frequencies of Websites.

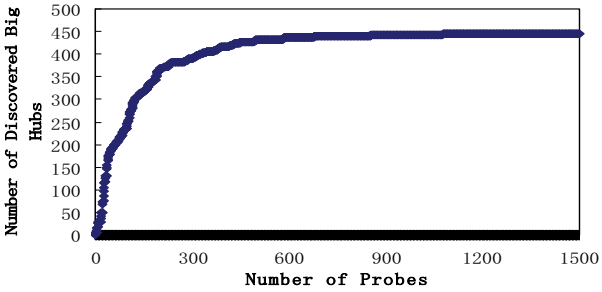

Figure 4. The Growth of Big hub (>1%) Numbers by Randomly Probing

Based on the prior observations, we can make the following assumption for name disambiguation:

Assumption 3.1: *For two appearances sharing a given name A_1, A_2, if their context-objects are found to co-occur in one of the small hubs (whose coverage scores are below a certain threshold), A_1, A_2 will most likely refer to the same object.*

The assumption above is always valid if the threshold is set conservative enough.

Third, it is feasible to discover the big hubs for a certain type with few times of probing. Figure 4 shows the growth of the number of newly discovered big hubs (i.e. Websites with coverage scores greater than 1%) when we probe paper titles to Web corpora. We observe that the newly discovered big hubs tends to converge

after only around 600 probes of papers, while there are around 1 million papers in our dataset. Note that the sample papers for probing are randomly selected, and there is no bias in it.

Given the above three observations, we can determine an appearance pair as referring to the same object once their context-objects are found in a small hub. The adaptive site-level connection function described in the above subsections is only for the co-occurrences in big hubs. As shown in the above observation, the number of big hubs is relatively limited, so it becomes feasible to train an adaptive connection function which gives a suitable weight to the co-occurrence in each big hub.

To determine the coverage threshold for small hubs, we can simply use the co-occurrence in small hubs for name disambiguation in the labeled sample dataset. With different coverage threshold setting, we observe the precision result. Generally, the higher the coverage threshold is, the lower the precision will be. To make Assumption 3.1 stand, we usually set a relatively conservative threshold.

To get the big hub sets for an adaptive Web connection function, we only need to discover them by a few times of probing of context-objects.

WebNaD Approach

The whole process of our *WebNaD* approach composes three phases: Probing, Training, Name Disambiguation, as shown in figure 5.

During the Probing phase, we obtained the corresponding big hubs by probing the context-objects. During the Training phase, given the big hub set obtained in Probing phase, an adaptive combination for site-level connection at big hubs is learned. During the Disambiguation for a certain name, any appearance pair X_1, X_2 will be detected as a match once they satisfy one of the following conditions:

- The connection strength in local data set is above a pre-defined threshold (Local Connection);
- One context-object pair is found to co-occur in a certain small hub (Small Hub Connection Rule);
- The site-level connection at big hubs of one context-object pair is above a pre-defined threshold (Big Hub Connection Model);

Finally, to complete the disambiguation phase, we compute the transitive closure of the appearance matches based on the transitive assumption: if X_1 and X_2 refer to the same object, and X_2 and X_3 refer to the same object, then X_1 and X_3 should also refer to the same object even if their connection strength is not strong enough.

EXPERIMENTS

Experimental Setup

Libra currently has indexed more than 1 million papers and their corresponding authors and conferences/journals, crawled from several Web datasets. For authors, author appearances extracted from many data sources only contains name information, and name disambiguation is one of the major challenges in integration.

We use 9,330 distinct author names out of 651,228 for disambiguation. From those names, 120 distinct author names (9,559 appearances in total) of 201 different authors are labeled. Around 2/3 appearances are randomly selected for training and 1/3 are left for testing.

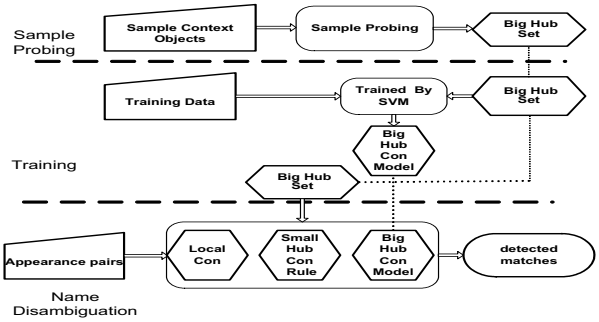

Figure 5. The *WebNaD* Procedure

We use Delivered Current Model [5] to measure the local connection evidence, and use paper's title as the query for search engine to get the Web occurrence.

In our experiment, we measure the overall performance with precision and recall of the match, as defined in the overview section, and use $F-score$ to harmonize them as the following:

$$F-score = \frac{2 * prec * recall}{prec + recall}$$

Study on Web Connection Measures

The Small Hub Threshold Setting: With different small hub threshold settings (coverage threshold), we observe the number of big hubs and the accuracy (i.e., the precision and recall). Table 1 shows the result. As we might have predicted, with a lower small hub threshold, more big hubs can be obtained, and higher precision can be reached, while the recall becomes lower. How to determine the threshold is a trade off. However, as we can see, when the threshold exceeds 1%, the precision for using Web connection in small hubs has a sharp decrease. A reasonable small hub threshold setting in our experiment should not exceed 1%. In the following experiments, the small hub threshold settings are set to 1%.

Web connection at Big Hubs: To study the Web connection at big hubs, we have compared four different measures:

- "*Learnable+URL*": Learnable combination considering the distance of URL layers. This is the calculation of site-level Web connection at big hubs in *WebNaD*.
- "*Learnable*": Learnable combination simply based on the co-occurrence at site level without considering the distance of URL layers.
- "*Linear+URL*": Simply linear combination of distance of URL layers, using the inverse coverage score as the weight.
- "*Linear*": Simply linear combination of co-occurrence at site level, using the inverse coverage scores as the weight.

The kernel used in SVM is the RBF kernel, and we just use the site-level connection in big hubs for author name disambiguation.

Table 1. Accuracy by only using Web connection in small hubs and the number of big hubs with different small hub threshold settings

Threshold(%)	Precision	Recall	# of Big Hubs
0.1	0.994	0.350	6363
0.2	0.935	0.432	2852
0.5	0.918	0.568	1009
1	0.906	0.703	448
2	0.625	0.767	199
5	0.603	0.801	72
10	0.588	0.876	42

Table 2. Maximum $F-score$ of four Web connection measures at big hubs.

Measures	F-score
Learnable+URL	**0.773**
Learnable	0.675
Linear+URL	0.621
Linear	0.622

Table 2 shows the optimal $F-score$ of the four measures. Note that the relatively lower $F-score$ is due to our ignoring the Web connection in small hubs and the local connection in this experiment. Obviously, the "*Learnable+URL*" by far outperformed other three measures, because of the learnable relative importance of each site and consideration of URL layer distance.

Study of WebNaD

Comparison with Previous Approaches: Table 3 summarizes the overall performance of different approaches for name disambiguation: *WebNaD*, *WebNaD*(page), Name Match, and the local connection-based approach[2]. Among them, the Name Match approach is the integration that ignores the disambiguation (i.e. simply grouping the appearances sharing the same name, and obviously the recall is 1). *WebNaD*(Page) uses page-level connection instead of site-level connection in the disambiguation. For the local connection-based approach, we show two results with different threshold setting: Local (1) is the result when approximately reaching the optimal $F-score$; Local (2) is the result when the precision is similar to the optimal result of the *WebNaD*.

Due to the insufficiency of relationship information in the local dataset, we can see that the optimal result of the local connection-based approach is even worse than the result of Name Match. From the comparison of Local (2) and *WebNaD*, we can see that by leveraging the Web connection, the recall improves a lot (from 0.254 to 0.892), while not sacrificing much in precision. As we can predict, the *WebNaD* results are much higher in recall than those of *WebNaD*(Page).

RELATED WORK

The traditional work on name disambiguation is usually based on the string similarity of the attribute value [6][7][8][9]. These approaches couldn't work for scenarios where there are few attribute values available for disambiguation.

Recently, the relationship information among different types of objects in a local dataset has started to be exploited for name disambiguation, such as [2][3]. The limitation of these approaches is that they rely too much on the completeness of relationship information, and will probably result in low recall.

In contrast, *WebNaD* exploits the local connection and the rich Web connection, and results in high recall.

Table 3. Accuracy comparison of different approaches

Approach	Precision	Recall	F-score
Name Match	0.436	1	0.607
Local (1)	0.668	0.396	0.497
Local (2)	0.902	0.254	0.397
WebNaD	0.878	0.892	**0.885**
WebNaD(Page)	0.889	0.636	0.741

CONCLUSION AND FUTURE WORK

In this paper, we study the name disambiguation problem in the scenario that there are few attribute values and one name may correspond to multiple objects. We show that the Web corpora can be exploited to significantly improve the accuracy of name disambiguation. We introduce a novel approach called *WebNaD* to effectively measure and use the Web connection for name disambiguation.

REFERENCES

[1] DBLP. http://dblp.uni-trier.de/

[2] Z. Chen, D.V. Kalashnikov, and S. Mehrotra. *Exploiting relationships for object consolidation*. In ACM IQIS, 2005.

[3] X. Dong, A. Halevy, and J. Madhavan. *Reference econciliation in Complex Information Spaces*. In Proc. Of SIGMOD, 2005.

[4] V. Vapnik. *Principles of risk minimization for learning theory*. Advances in Neural Information Processing Systems 3, pages 831-838. Morgan Kaufmann, 1992.

[5] C. Faloutsos, K. McCurley, and A. Tomkins. *Fast discovery of connection subgraphs*. In Proc. Of SIGKDD, 2004.

[6] W. E. Winkler. *The state of record linkage and current research problems*. Technical report, Statistical Research Division, U.S. Bureau of the Census, Washington, DC, 1999.

[7] A. Monge, and C. Elkan. *An efficient domain independent algorithm for detecting approximately duplicate dataset records*. In DMKD, 1997.

[8] A. McCallum, K. Nigam, and L. H. Ungar. *Efficient Clustering of High-Dimensional Data Sets with Application to Reference Matching*. In Proc. Of SIGKDD, 2000.

[9] L. Jin, C. Li, and S. Mehrotra. Efficient record linkage in large data sets. In DASFAA 2003.