2010 Sixth International Conference on Semantics, Knowledge and Grids

A Holistic Solution for Duplicate Entity Identification in Deep Web Data Integration

Wei Liu [1,2], Xiaofeng Meng [3]

[1] *Institute of Computer Science and Technology, Peking University, Beijing 100871, China*

[2] *Institute of Scientific and Technical Information of China*

[1] gue1976@gmail.com

[3] *School of Information, Renmin University of China, Beijing 100872, China*

[3] xfmeng@ruc.edu.cn

Abstract—**The proliferation of deep Web offers users a great opportunity to search high-quality information from Web. As a necessary step in deep Web data integration, the goal of duplicate entity identification is to discover the duplicate records from the integrated Web databases for further applications(e.g. price- comparison services). However, most of existing works address this issue only between two data sources, which are not practical to deep Web data integration systems. That is, one duplicate entity matcher trained over two specific Web databases cannot be applied to other Web databases. In addition, the cost of preparing the training set for *n* Web databases is C_n^2 times higher than that for two Web databases. In this paper, we propose a holistic solution to address the new challenges posed by deep Web, whose goal is to build one duplicate entity matcher over multiple Web databases. The extensive experiments on two domains show that the proposed solution is highly effective for deep Web data integration.**

I. INTRODUCTION

With the proliferation of deep Web, a flood of high-quality information(usually in form of structured records) can be accessed through online Web databases. The recent statistics[1] reveal that there are more than 450,000 Web databases in the current Web. These Web databases offer web services, RSS feeds, even provide APIs to allow data to be exchanged between them. Deep Web data integration aims to combine Web information to provide users a unified view.

At present, many issues in the field of deep Web data integration, such as interface integration[2][3] and Web data extraction[4,5], have been widely studied. However, as a necessary step, identifying the duplicate entities(records) from multiple Web databases has not received due attention yet. In one domain(book, music, computer, etc.), there are often a large proportion of duplicate entities across Web databases, so it is necessary to identify them for further applications, such as de-duplication or price-comparison services. Due to the heterogeneity of Web databases, the duplicate entities usually exist in various inconsistent presentations. In this paper, we study the duplicate entity identification problem in the context of deep Web data integration.

Until now, there are already lots of research works to address this issue, but most of them only focus on the two-data-source situation. However, when facing the lots of Web databases, they have to build C_n^2 matchers, where *n* is the number of Web databases. This makes the unaffordable costs for both preparing training set and building the duplicate identification matcher. In most existing deep Web data integration systems, the duplicate entity matchers are manually built under small scale and static integration scenarios. In contrast, in large scale deep Web data integration scenarios, this process needs to be as automatic as possible and scalable to large quantities of Web databases. We will review previous works in Section VI.

In this paper, we propose a domain-level solution to address the challenges posed by deep Web, which means, the trained matcher can identify the duplicate entities over multiple one-domain Web databases. The intuition behind our solution is that, given a domain, the number of attributes is convergent, and further, each attribute plays a definite role on the duplicate entity identification problem. In another word, the importance (or weight) of an attribute is actually domain-dependent. For example, in Book domain, "title" is always more important than "publisher" to determine whether two book records refer to the same book. our solution consists of the following three main steps.

Semi-automatic training set generation: in previous works, the training set(matched record pairs) was prepared through manually, which is unpractical when facing lots of *WDB*s(Web databases for short). Thus, a semi-automatic method is proposed to generate the training set automatically, which can significantly reduce the labelling cost.

Attribute weight training: in order to weigh the importance of the similarity of each attribute reasonably, we propose an iterative training approach to learn the attribute weights of. The basic idea is that, under the ideal weights, the similarity of any two matched records must be larger than that of any two unmatched records.

In summary, the contributions of this paper are:

- As our problem, it is first time to probe the duplicate entity identification problem in the context of deep Web data integration, where lots of Web databases pose new challenges to this issue.
- As our insight on the observation, we discover the attributes in a same domain play definite roles on the problem of duplicate entity identification, which makes it possible to build one matcher over multiple Web databases in one domain.
- As our solution, we propose a holistic solution on

978-0-7695-4189-1/10 $26.00 © 2010 IEEE
DOI 10.1109/SKG.2010.38

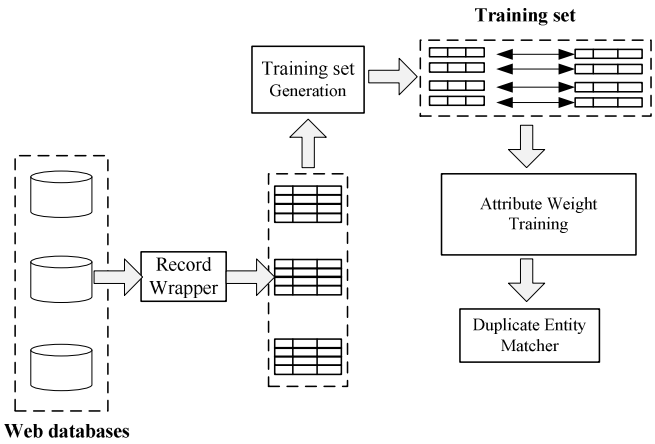


Fig. 1. Solution Overview

duplicate entity identification under the context of deep Web data integration. Our experiments show the promise of this solution.

The rest of paper is organized as follows. In section 2, we present the overview of our solution. Section 3 proposes a semi-automatic method to produce the training set. Section 4 proposes a novel approach of attribute weight learning. An experimental evaluation for our approach is shown in Section 5. In Section 6, we talk about the related works. Section 7 discusses several further opportunities and then concludes the paper.

II. SOLUTION OVERVIEW

In this paper we proposed a practical domain-level solution to address the problem of duplicate entity identification under the context of deep Web data integration. Figure 1 shows the overview of our solution. The input is the records in different *WDB*s. The output is a set of record pairs, where each pair denotes two duplicate records. [14] for Web data item extraction have been proposed and confirmed to achieve satisfactory accuracy.

There are three primary components in our solution. Their functions are introduced briefly as follows.

- **Record Wrapper:** In general, the records in Web databases are embedded in web pages when users submit queries. The function of this component is automatically extracting the structured records from web pages at attribute level.
- **Semi-automatic training set generation:** This component aims at semi-automatically obtaining enough duplicate records as the training set from the records wrappered from *WDB*s. Each training sample refers to two duplicate records.
- **Attribute weight training:** Each attribute is assign an appropriate weight by the component by employing our proposed iterative training-based approach, and two thresholds T_1 and T_2 ($T_1 > T_2$) are also learned.
- **Duplicate entity matcher**: Given two inputted records, their similarity can be computed with the weights of the shared attributes, and further, the two records can be determined whether being duplicates using the thresholds.

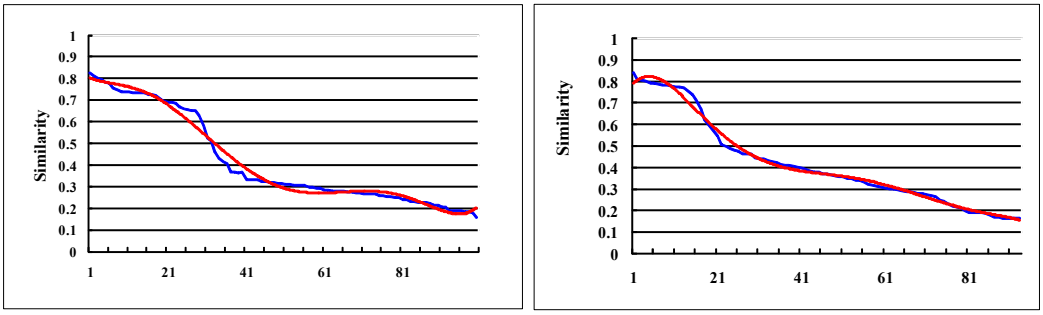
Wrapper belongs to the research field of web data extraction has been widely studied, and many automatic approaches have been proposed to address this issue. The idea of duplicate entity matcher is rather simple and direct. So no more discussions are for them in this paper due to the limitation of paper length. The rest of this paper will focus on the underlying techniques of the two components training set generation and attribute weight training.

III. SEMI-AUTOMATIC TRAINING SET GENERATION

In previous approaches, the training set was always prepared manually in advance. That is, domain experts label some record pairs to be duplicates or not as the training set. Unfortunately, lots of Web databases makes manually labelling training set time-consuming and error-prone. Obviously, the labelling cost for n Web databases is C_n^2 times of that for 2 Web databases. In this section, a semi-automatic method is introduced to generate the training set for n Web databases.

Instinctively, if two records from different *WDB*s are determined to be matched (i.e. they are duplicates), they often share more texts than the unmatched ones. So a naive approach is to regard each record as a short text document, and determine whether two records are duplicates by comparing text similarity, such as *tf-idf* function. But obviously the accuracy is not satisfying and not stable. We have evaluated this naive approach on two domains(book, computer), and the accuracies are only about 83% and 47% respectively. We check the results and divide all matched record pairs into correct ones and wrong record ones. The correct record pairs refer to the duplicates in fact, while the wrong record pairs are not. If ranking all matched record pairs by their *tf-idf* similarity in descending order, we find most correct record pairs congregates in the head, while most wrong ones are in the tail. This phenomenon motivates us to obtain training set(the right ones) from the head.

Figure 2 shows the curves of the record pair sequences for two domains. Through farther analysing, we find that: if the total matched record pairs are enough(say, more than 100), two distinct inflexions divide the whole curve into three segments(head segment, body segment, and tail segment).



(a) Book domain (b) Computer domain

Fig. 2 The relationship of the ith record pair and its similarity

Most correct ones locate at the head segment, and most wrong ones locate at the tail segment. The body segment is mainly the mixture of correct ones and wrong ones. So it is feasible to regard the record pairs in the head segment as the training set. And the problem is how to find the head segment in the curve.

In order to detect the first arc accurately, we resort to a mathematic mean which consists of two steps: curve fitting and curvature computation. In the first step, given a sequence of similarity values, point them in a two-dimensional reference frame, where *y* axes denotes the similarity value and *x* axes denotes the similarity ranking result. The least squares fitting method is applied for curve fitting (the red curve in Figure 2). The least squares fitting method is a very popular mathematic method of fitting data, and so its technique detail is not discussed here anymore. In the second step, the curvature for each similarity value in the curve is computed, and the similarity with the maximum downward curvature is located. Then this similarity value in the curve is we want to locate. One training set is obtained for every two *WDB*s. Suppose *n WDB*s, totally C_n^2 training sets have to be generated. The final training set is the sum of these training sets.

However, the training set is often not perfect, which means several wrong matched data record pairs may mix in. If their experiments are based on the noisy training set, the accuracy will be far away from what they reported in their experiments. So a quick one-pass checking for the training set is needed to get rid of the wrong matched data record pairs. Though this is manual, the cost is obviously far less than the traditional way. Intuitively, to guarantee the quality of the training set, the size of the unlabeled training set should be large enough. The related experiments will be given in Section 6 to guide us to leverage this problem.

IV. ATTRIBUTE WEIGHT TRAINING

In this part, we study the problem of training the appropriate domain-level attribute weights using the training set obtained from Web databases. As a result, the trained attribute weights can be applied for any two Web database in this domain.

A. Preliminaries

An iterative training mechanism is proposed to this task, and we call it IBITA (Inequalities Based Iterative Training Approach) in this paper. Figure 3 shows its architecture. IBITA starts with two *R* sets from WDB_A and WDB_B. Without loss of generality, we suppose that there are *m* attribute among *n WDB*s. For each record pair $<R^i, R^j>$, we define the record similarity as follows.

Definition 5.1. (**Record Similarity**) The similarity of R^i and R^j is the weighted sum of the similarities of the shared attributes. Correspondingly, weight w_k ($1<= k<=m$) is assigned to the corresponding attribute am^k to show its contribution to the similarity of R^i and R^j. Formally, record similarity is denoted below:

$$S(R^i, R^j) = \sum_{k=1}^{m} w_k \times S(am^k) \quad (1)$$

Using weight vector $<w_1, w_2 \ldots\ldots w_m>$(*WV* for short), we can measure the similarity of any record pair $< R^i, R^j >$ as a real number larger than 0. The ideal weights vector is hoped to make all the matched record pairs and non-matched record pairs take on a distinct bipolar distribution when projecting their similarities on the axis as shown in Figure 4. The bipolar distribution requires all those matched record pairs (denoted as circles) to locate at the starboard of the axis, while all those non-matched record pairs (denoted as rectangles) to locate at the larboard of the axis. We would like the optimal weight vector ($WV_{optimal}$) which makes the bipolar distribution on the axis most distinct, that is, bring the largest distance of matched and non-matched record pairs marked on Figure 4. Meanwhile, two thresholds are also needed to determine each record pair to be "matched", "non-matched", or "possibly matched".

Suppose the training set contains *n* matched record pairs, where each pair describes one same entity. We use $< R^i, R^i >$ to denote the matched record pair, and use $< R^i, R^j >(i \neq j)$ to denote the non-matched pair.

B. Inequalities-based Metrics

By observing the bipolar distribution shown in Figure 4, we find that *WV* needs to be adjusted to meet the following condition: The similarity of a matched pair is greater than the similarity of a non-matched pair. Formally, the similarity of *n* uniquely matched pair $< R^i, R^i >$ should be greater than any of the $n*(n-1)$ non-matched pairs $< R^i, R^k >$ ($j \neq k$). Therefore, a group of $n * n * (n - 1)$ inequalities can be correspondingly obtained as follows:

$$\{S(R^i, R^i) \geq S(R^j, R^k)\} \quad 1 \leq i, j, k \leq n, j \neq k \qquad \textbf{(2)}$$

Totally $n*(n-1)$ inequalities are generated. We try to find $WV_{optimal}$ from the solution space of Inequalities 2. Intuitively,

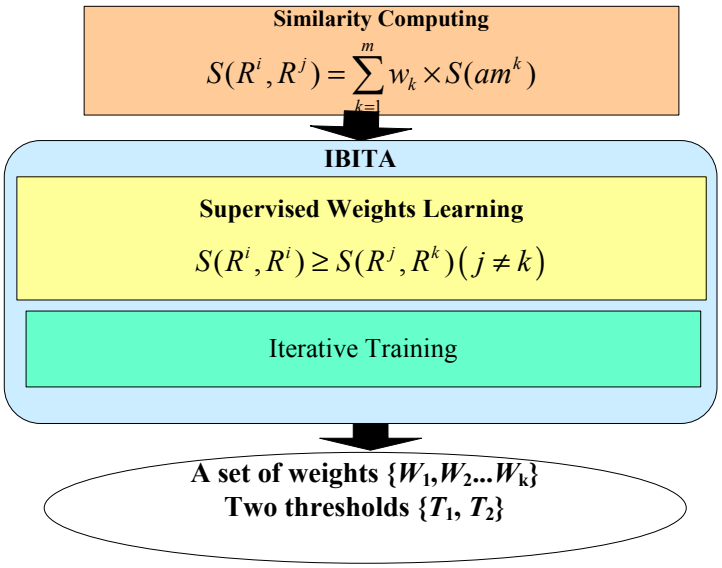


Fig. 3. General IBITA architecture

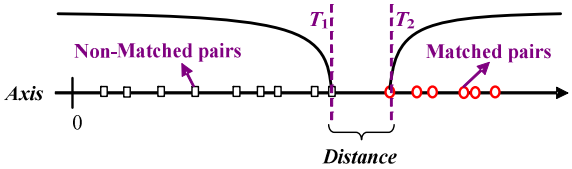


Fig. 4. The ideal bipolar distribution

we have to solve these n * (n - 1) inequalities, a right (not optimal) WV can be got. However, the exponential growth of the number of inequalities is too costly for real applications. So we use the following subset of these inequalities instead of Inequa. (2):

$$\{S(R^i, R^i) \geq S(R^i, R^j)\} \quad 1 \leq i, j \leq n, i \neq j \qquad \textbf{(3)}$$

For any WV in the solution space of Inequa. (3), there will be two possibilities: the WV satisfies Inequa. (2), or not satisfies Inequa. (2). In another word, not all the WVs of Inequa. (3) can make the n matched record pairs and n * (n - 1) non-matched record pairs a bipolar distribution as we wanted (see Figure 5(a)). Some WVs may lead to the cross-region situation shown in Figure 5(b), where it is still guaranteed on each axis, the matched record pair is closer to the starboard than all the non-matched record pairs. The cross-region situation means not all the similarities of n matched record pairs are larger than the similarities of all n*(n–1) non-matched record pairs. This cross-region situation is thus caused where the n matched record pairs in training data set cannot be divided into matched or non-matched. As we can see from Figure 5(b), the similarity of the non-matched record pair $< R^2 , R^y >$ (y >= 2) is larger than the similarity of the matched record pair $< R^1 , R^1 >$. The confusion in this situation can be described as that: if the similarity of the new record pair falls into the cross-region formed by T_1 and T_2, the system will not be able to judge whether this two records represent the same entity due to the ambiguity they have. So what we need to do next is to try to obtain a WV in the solution space of Inequa. (2) using Inequa. (3).

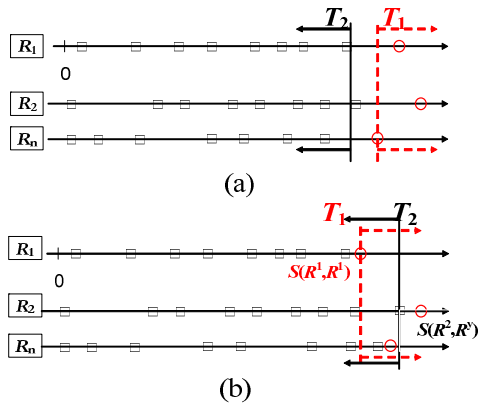


Fig. 5. Ideal situation and Cross-region situation

C. Iterative Training

Given a WV in the solution space of Inequa. (3), the similarity of $< R^i , R^j >$ in the training set can be derived as the weighted sum of the similarities of all attributes. In the training set containing n matched record pairs there are n * n record similarities being computed, each of which corresponds to one random combination of R^i ($1 \leq i \leq n$) and R^j ($1 \leq j \leq n$). We project these n^2 record similarities to n axes and try to iteratively analyse different similarity distributions on the axes caused by different WVs in order to find $WV_{optimal}$.

For each R^i($1 \leq i \leq n$) we build an axis, and n similarities are projected on the axis as shown in Figure 5. The similarities of R^i are located in the ith axis. The circle denotes matched record pair $<R^i, R^i>$ which are closest to the starboard of the

axes, while the small rectangles denote non-matched record pairs $<R^i, R^j>$ $(i \neq j)$.

Given a *WV*, the minimum similarity of all *n* matched pairs is regarded as a threshold T_1 (dashed line in Figure 5) and the maximum similarity of all *n* * (*n* - 1) non-matched pairs is regarded as a threshold T_2 (real line in Figure 5). Formally, we denote them as the following form:

$$\begin{cases} T_1 = min\{S(R^i, R^i)_{WV}\} \\ T_2 = max\{S(R^i, R^j)_{WV}\} \end{cases} \quad (i \neq j)$$

where $S(R^i, R^j)_{WV}$ is the similarity of R^i and R^j being computed with current *WV*.

If $T_2 < T_1$, we can guarantee the similarity of $< R^i, R^i >$ is larger than the similarity of $< R^i, R^j >$ $(i \neq j)$. So the ideal situation is $T_2 < T_1$, and cross-region situation is $T_1 < T_2$.

There are two main steps in the implement of this component which tries to obtain $WV_{optimal}$ staring at an arbitrary *WV* in the solution space of Inequa. (3). The first step is to obtain a *WV* satisfying Inequa. (2) from the *WV* of Inequa. (3), and the second step is to obtain $WV_{optimal}$ from a *WV* of Inequa. (2).

Step 1 *WV* of Inequalities 3→ *WV* of Inequalities 2

At the beginning, a *WV* is got by solving Inequa. (3), and further T_1 and T_2 are got. If $T_2 < T_1$, this means this *WV* satisfies Inequa. (2), and the next step is activated. Otherwise, the *WV* caused the cross-region situation, just like Fig. 5 (b). The goal of this step is to obtain a *WV* of Inequa. (2) using the *WV* of Inequa. (3). Next, for $T_1 < T_2$, it is represented in the following form:

$$min\{S(R^i, R^i)_{WV}\} < max\{S(R^i, R^j)_{WV}\} (i \neq j) \quad (4)$$

Then Inequa. (5) is formed by appending Inequa. (4) to Inequa. (3), and *WV'* is obtained by solving Inequa. (5). The left of this step is repeating the above process until the *WV* satisfies Inequa. (2).

The main idea of this step is to iteratively append the inequalities which do not satisfy Inequa. (2) to Inequa. (3) until a *WV* satisfying Inequa. (2) is got. In another word, the solution space continues shrinking during the process and a *WV* in the solution space of Inequa. (3) has more probability to be in the solution space of Inequa. (2). Actually, there is more than one inequality which does not satisfy Inequa. (2), but only one inequality (Inequa. (4)) is appended every iteration, due to the consideration of efficiency improvement. In practical, the iteration process is less than 4 times averagely.

Step 2 *WV* of Inequalities 2→ $WV_{optimal}$

This process starts at a *WV* of Inequa. (2). The current *WV* can guarantee the similarity of any matched record pair is larger than that of any non-matched record pair in the training set. In order to reach high accuracy, we need get $WV_{optimal}$ which can make the matched record pairs and non-matched record pairs the most distinct bipolar distribution. In another word, $WV_{optimal}$ can make the distance of T_1 and T_2 (i.e. T_1 - T_2) reach the maximum.

In order to make the description concisely and without confusion, we use Inequa. (4) to denote all the inequalities appended to Inequa. (3). Suppose Inequa. (5) is Inequa. (3) and the inequalities appended to Inequa. (3) in the first step. So Inequa. (5) is denoted as the following:

$$\begin{cases} S(R^i, R^i) - S(R^i, R^j) \geq 0\} \quad (1 \leq i, j \leq n, j \neq i) \\ max\{S(R^i, R^j)_{WV}\} - min\{S(R^i, R^i)_{WV}\} > 0 (j \neq i) \end{cases} \quad (5)$$

Initially, the zeros in the right side of inequalities is replaced by T_1-T_2, and the new inequalities (e.g. Inequa. (6)) are denoted as the following:

$$\begin{cases} S(R^i, R^i) - S(R^i, R^j) \geq 0 \quad (1 \leq i, j \leq n, i \neq j) \\ \max\{S(R^i, R^j)_{WV}\} - \min\{S(R^i, R^i)_{WV}\} > T_1 - T_2 \ (i \neq j) \end{cases} \quad (6)$$

WV' is got by solving Inequa. (6), and further T'_1 and T'_2 are got. Then $T'_1 - T'_2$ replaces T_1 - T_2 in Inequa. (6), and the above process is repeated until $(T'_1 - T'_2)$ - $(T_1$ - $T_2) < \sigma$, σ is set in advance, and the smaller σ is, the current *WV* is closer to $WV_{optimal}$. In practice, σ is set to be 0.12.

Till now, for any number of *WDBs* in one domain, IBITA can ultimately bring to us an optimum group of quantified weights $WV_{optimal}$ and two stabilized thresholds T_1 and T_2. Then it is easy to compute the similarity for any two records (R^i, R^j) from different web databases using $WV_{optimal}$. Via comparing the similarity value with T_1 and T_2, we can easily determine they are matched or not. If the similarity of the record pair falls into the possibly matched region(i.e. $T_2 \leq S(R^i, R^j) \leq T_1$), it needs to be manually checked.

V. EXPERIMENTS

A prototype system, DWDEI(Deep-web Duplicate Entity Identifier), has been implemented based on our solution. We evaluate this system over the real Web databases on two popular domains. The test bed and the evaluation measures are introduced first. Then, a series of experiments are conducted for evaluation.

A. Data Set

TABLE 1: WEB DATABASES IN THE EXPERIMENTS

ID	Web database	Description
1	*Amazon*	www.amazon.com/□textbooks/□
2	*Bookpool*	www.bookpool.com/□
3	*Blackwell*	www3.interscience.wiley.com/browse/BOOK□
4	*ClassBook*	www.classbook.com/□
5	*Bookbyte*	www.bookbyte.com/□

(a) Book Domain

ID	Web database	Description
1	*Superwarehouse*	http://www.superwarehouse.com/□
2	*Amazon*	http://www.amazon.com/Computers□
3	*CNET*	http://reviews.cnet.com/desktop-computers/□
4	*Computers4sure*	www.computers4sure.com/□
5	*Bookbyte*	www.pcconnection.com/□

(b) Computer Domain

The test bed for evaluation is the Web databases on book and computer domains. For each domain, we select 5 popular web sites as the Web databases. Table 1 lists these Web databases. The reason that we select these *Web databases as the test bed is there are enough*

duplicate records among them. Therefore, there are totally C_5^2 Web database pairs in each domain. And we use 5 of them to produce the training set to learn the attribute weights and two thresholds in this domain. For each Web database pair, we submit 6 queries. For each query, we would select top 100 records from the query results. Both the training set and testing set are coming from the returned query results.

The characteristics of our data set can be concluded as follows. (1) For each submit query, the returned query results from 2 paired *WDBs* shared a large proportion of overlapping entities. (2) The scale of our data set is quite large that thetotal number of ∇⌉⌋≀∇⌈ ⊣⟩∇∫ has achieved more than 800 for each pair of *WDBs*. (3) The submitted queries are completely different, which guarantees that there is almost no overlap between the query results. All those features of our data set ensure the objectivity of our experimental results.

B. Evaluation Measures

Four criteria are defined to evaluate the effectiveness of our solution, which are listed below.

$$\text{Precision } M = \frac{|PredictedMP \cap ActualMP|}{|PredictedMP|}$$

$$\text{Precision } N = \frac{|PredictedNP \cap ActualNP|}{|PredictedNP|}$$

$$\text{Uncertainty} = \frac{|UncertainP|}{|PredictedMP + PredictedNP + UncertainP|}$$

$$\text{Precision } T = \frac{|PredictedMP \cap ActualMP| + |PredictedNP \cap ActualNP|}{|PredictedMP + PredictedNP + UncertainP|}$$

where ActualMP is the set of real matched record pairs in the testing set and PredicatedMP is the set of matched record pairs identified by DWDEI. Similarly, ActualNP is the set of real non-matched record pairs in the testing set and PredicatedNP is the set of non-matched record pairs identified by DWDEI. In addition, UncertainP denotes the set of record pairs that cannot be determined by DWDEI.. Those uncertain pairs need to be further manually checked.

C. Evaluation of semi-automatic training set generation

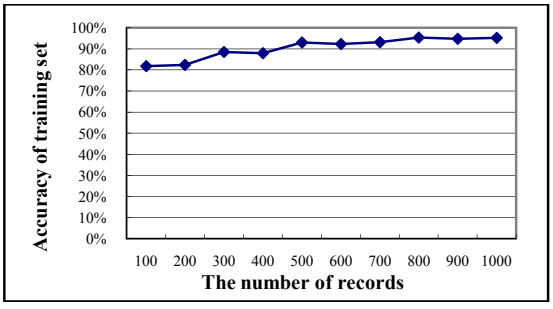


Fig. 6. The Experimental results of semi-automatic training set generation

In this part, we conduct the experiment to evaluate the effectiveness of the component semi-automatic training set generation. It is obvious that the performance of our solution depends greatly on the quality of the training set. In our solution, we wrapper the records from the result pages through submitting some popular queries. X axis in Figure 6 refers to the number of records wrappered from each result page.

As it can be seen from Figure 6, the accuracy tend to be stable as the number as the number of records increases. The accuracy is convergent at about 95% when the number of record pairs is larger than 800. Hence, we recommend more than 800 records are wrappered from each Web database to generate the training set in practice. Though manual checking is needed to pick out the 5% errors, the cost is far less than the traditional way because identifying two records are duplicates or not is much easier than finding the duplicate for one record from a large number of ones.

D. Overall Performance

TABLE II. WEIGHTS FOR BOOK DOMAIN

	Title	Author	Publisher	Date	Price	Edition
Weight	0.34	022	0.13	0.12	0.09	0.06

TABLE III. WEIGHTS FOR COMPUTER DOMAIN

	Model	CPU	Monitor	RAM	HD	CD
Weight	0.28	0.09	0.07	0.06	0.04	0.02

TABLE IV. EXPERIMENTAL RESULTS FOR BOOK DOMAIN

	PrecisionM	PrecisionN	Uncertainty	PrecisionT
P12	0.989	1	0.020	0.994
P13	0.985	0.956	0.036	0.937
P14	0.965	0.983	0.009	0.970
P15	0.916	1	0.024	0.955
P23	0.96	0.986	0.024	0.970
P24	0.928	0.905	0.044	0.941
P25	0.977	0.991	0.009	0.985
P34	0.896	0.946	0.014	0.931
P35	0.979	0.948	0.022	0.966
P45	0.919	0.974	0.022	0.944
AVG	0.951	0.968	0.022	0.959

TABLE V. EXPERIMENTAL RESULTS FOR COMPUTER DOMAIN

	PrecisionM	PrecisionN	Uncertainty	PrecisionT
P12	0.858	0.997	0.026	0.914
P13	0.811	0.952	0.033	0.882
P14	0.954	0.895	0.012	0.923
P15	1	0.863	0.029	0.940
P23	0.876	0.813	0	0.848
P24	0.834	0.776	0.019	0.828
P25	0.819	0.848	0.017	0.836
P34	0.849	0.958	0.013	0.893
P35	0.943	0.919	0.007	0.931
P45	0.978	0.950	0.014	0.969
AVG	0.892	0.897	0.017	0.896

For each domain, totally 10(i.e. C_5^2) Web database pairs are produced. We use Pij to denote a specific Web database pair.

For example, P24 refers to the 2nd web database and the 4th web database. Web database pairs P12, P13, P14, P15 and P23 are used to learn the optimal attribute weights $WV_{optimal}$ and the thresholds T_1 and T_2. The test bed consists of two parts: (a) the record pairs from P12, P13, P14, P15 and P23; (b) the record pairs from P24, P25, P34, P35 and P45. Table 2 and Table 3 show the normalized attribute weights for Book domain and Computer domain respectively. Due to the space limitation, only top 6 frequent attributes are listed.

Table 4 and Table 5 show the accuracy of DWDEI for book domain and computer domain respectively. As it can be seen from Table 4 and Table 5, our experimental results reveal 3 features of DWDEI: (1) Our solution performs well on all the four measures. This shows that our solution is highly effective. (2) The measure UncertaintyP are extremely low (AVG 2.2% on book domain and AVG 1.7% on computer domain), which greatly reduces the manual intervention. (3) The small decrease in performance on several Web database pairs strongly indicates that our approach is very robust, considering the facts that record pairs are from completely new Web database pairs.

In addition, we also observe that the performance on book domain is a little better than that on computer domain. The main reason for this phenomenon is that, the value ranges of computer attributes are often small, so it is more difficult to differentiate the matched record pairs and non-matched record pairs.

E. Performance comparison with previous related works using Cora dataset

To compare with the works in this field, we also conduct the experiment on the popular Cora dataset. Cora dataset is the standard in the duplicate entity identification community and is frequently used as the test bed for evaluation. Cora contains 2191 5-field citations to 305 computer science papers. The goal of this experiment is two-fold. First, *Cora* is often as the test bed in the related works. The experiments can be used for the performance comparison between our approach and previous works which also carried out their experiments on it. Second, we believe DWDEI can also be applied to unacquainted Web databases.

TABLE VI. EXPERIMENTAL RESULTS ON CORA DATA SET

Precision	*Recall*	*F-measure*
97.6%	92.7%	95.1%

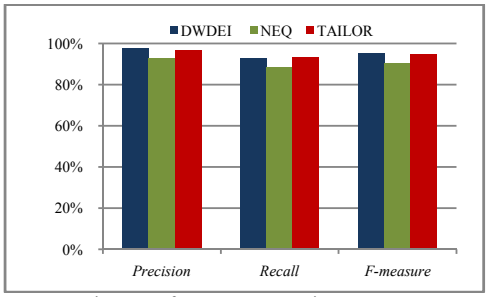


Fig. 7. Performance comparison among

Since only one threshold is learned in the previous works, we use the mean of as the threshold. We compare DWDEI with two recent related works [35] and [36]. From the experimental results shown in Table 6 and the performance comparison on F-measure shown in Figure 7, two conclusions can be made. First, the performance of DWDEI on *Cora* is very good on both the three traditional measures. Second, the performance on *Cora* is a little better than those reported by the related works. The experiments indicate that DWDEI based on our solution takes on the domain-level character. That is, DWDEI can still achieve a satisfactory performance among new Web databases(the training set is not generated from them) in this domain when enough important attributes are covered.

VI. RELATED WORK

The goal of duplicate entity identification is to identify the duplicated records in the same or different databases that refer to the same real world entity, even if the records are not identical. It is well known that the duplicate entity identification problem have been studied for decades. This first work can be seen in [10], which is proposed by Fellegi-Sunter in the late 1950s and 1960s. A recent survey[34] has been given to summary the research works in this field.

In this Section, we give a more detailed category for the works on duplicate entity identification according their techniques, and present primary representative works for each class.

A large number of works and solutions have been proposed to address this challenging problem. These works mainly focused on the entity identification problem from two aspects: attribute similarity comparison and duplicate records detection. Attribute similarity comparison produces a vector of similarity scores corresponding to attribute pair comparison result; with this similarity vector, each record pair is classified as a match or non-match using different classification techniques. Attribute similarity comparison often use some string distance metrics. Edit distance[22], as a generic metric, can be considered as the first metric for string comparison. The following proposed metrics, such as affine gap distance[23] and Jaro distance[24], etc, define different penalties for the position of gap or the string order, which can be applied to some special situations, such as person name and address. For example, affine gap distance can work well when matching strings that have been truncated or shortened, while Jaro distance allows for better local alignment of the strings. However, all of them cannot address the situation due to various representations which is very common across multiple Web databases.

For identifying record pairs as matching or non-matching, there are several class of solutions [20]: rule-based methods that use matching rules given by human experts; supervised learning techniques which use labelled examples to learn rules or train a probabilistic model such as Bayesian network, SVM, a decision tree and so on; unsupervised learning techniques that can label the matched records from the training data automatically; distance-based methods which avoid the

need for training data by defining a distance metric and an appropriate matching threshold. Actually, the matchers they generated can only work well over two Web databases.

Recently, the value of additional information for duplicated entity identification has been recognized by researchers, such as semantic relationships or mappings. The rich information present in the associations between references is exploited for reference reconciliation [16]. [19] described a source conscious compiler for entity resolution which exploits information about data sources and employs multiple matching solution to improve matching accuracy. Moma matching system uses a library of matching algorithm and the combination of their results to improve match quality [21]. But it is an overhead problem to build and maintains the library of matching algorithm and selects the suitable algorithm.

Overall, there are two significant differences between our work and the previous works. First, most of previous works only deal with entity identification problem in the specific data sources, so it is hard to produce one robust matcher to cover multiple Web databases. Second, most previous works prepare their training sets in the manual way, which is impractical when facing lots of Web databases. Instead, we propose an automatic approach to generate the flawed training set, and only a quick one-pass check is needed to pick up the errors. This can reduce the labelling cost significantly.

VII. CONCLUSIONS AND FUTURE WORKS

In this paper, we study the problem of duplicate entity identification for deep Web data integration. We first give an observation to the attributes in one domain and hypothesize that their roles are definite or domain-dependent. Then, we propose a holistic approach to address this problem, which includes training set achieving, attribute mapping, and attribute weight assigning. In the experiments, we choose two representative domains (book and computer) to evaluate our approach, and the experimental results prove its accuracy is satisfying in practical. In the future, we will expand the scale of our from two aspects: (a) increase the number of Web databases in each domain; (b) extend our experiments to more important domains, such as automobile, movie and research papers, etc.

References

[1] Chang K. C., He B., Li C., Patel M., Zhang Z.. Structured Databases on the Web: Observations and Implications. SIGMOD Record 33(3): 61-70 (2004).

[2] Dragut E. C., Wu W., Sistla A. P.: Merging Source Query Interfaces on Web Databases: ICDE 2006: 46

[3] Wu W., Doan A., Yu C. T.: WebIQ: Learning from the Web to Match Deep-Web Query Interfaces. ICDE 2006: 44

[4] Zhao H., Meng W., Wu Z., V. Raghavan: Fully automatic wrapper generation for search engines. WWW 2005: 66-75s

[5] Liu B., Grossman R. L., Zhai Y.: Mining data records in Web pages. KDD 2003: 601-606 [6]

[6] Tejada S., Knoblock C. A., Minton S.: Learning domain-independent string transformation weights for high accuracy object identification. KDD 2002: 350-359

[7] Sarawagi S., Bhamidipaty A.: Interactive de-duplication using active learning. KDD 2002: 269-278

[8] Zhang J., Ling T. W., Bruckner R. M., Liu H.: PC-Filter: A Robust Filtering Technique for Du-plicate Record Detection in Large Databases. DEXA 2004: 486-496.

[9] Newcombe H. B., Kennedy J. M., Axford S.J.: Automatic linkage of vital records. Science, 130(3381):954-959, 1959.

[10] Fellegi I. P. and Sunter A. B.: A theory for record linkage. Journal of the American Statistical Association, 64(328):1183-1210, 1969.

[11] Cochinwala M., Kurien V., Lalk G.: Improving generalization with active learning. Information Sciences, 137(1-4):1-15, 2001.

[12] Bilenko M., Mooney R. J., Cohen W. W.: Adaptive name matching in information integration. IEEE Intelligent Systems, 18(5):16-23, 2003.

[13] He B., Chang K. C.-C.: Making holistic schema matching robust: an ensemble approach. KDD 2005: 429-438

[14] Newcombe H. B., Kennedy J. M., Axford S.J., and James A.P.: Automatic linkage of vital records. Science, 130(3381):954-959, October 1959.

[15] Jaro M. A.. Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida. Journal of the American Statistical Association, 84(406):414-420, June 1989.

[16] Dong X., Halevy A., and Madhavan J.. Reference reconciliation in complex information spaces. SIGMOD 2005, 85-96.

[17] Winkler W. E.. Improved decision rules in the felligi-sunter model of record linkage. Technical Report Statistical Research Report Series RR93/12, U.S. Bureau of the Census, Washington, D.C., 1993.

[18] Cochinwala M., Kurien V., Lalk G.: Improving generalization with active learning. Information Sciences, 137(1-4):1-15, September 2001.

[19] Shen W., DeRose P., Vu L., Doan A., Ramakrishnan R.. Source-aware Entity Matching: A Compositional Approach. ICDE 2007, 196-205.

[20] N. Koudas, S. Sarawagi, and D. Srivastava. Record linkage: Similarity measures and algorithms (tutorial). SIGMOD 2006, 802-803.

[21] A. Thor, E. Rahm. MOMA - A Mapping-based Object Matching System. CIDR 2007, 247-258.

[22] Levenshtein V.I.. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Doklady Akademii Nauk SSSR, vol. 163, no. 4, pp. 845-848, 1965, original in Russiantranslation in Soviet Physics Doklady, vol. 10, no. 8, pp. 707-710, 1966.

[23] Waterman M.S., Smith T.F., and Beyer W.A.. Some Biological Sequence Metrics. Advances in Math., vol. 20, no. 4, pp. 367-387, 1976.

[24] M.A. Jaro. Unimatch: A Record Linkage System: Users Manual. technical report, US Bureau of the Census, Washington, D.C., 1976.

[25] Sarawagi S., Bhamidipaty A.. Interactive deduplication using active learning. KDD 2002: 269-278

[26] Tejada S., Knoblock C. A., Minton S.. Learning domain-independent string transformation weights for high accuracy object identification. KDD 2002.

[27] Ananthakrishna R., Chaudhuri S., Ganti V.. Eliminating fuzzy duplicates in data warehouses. VLDB 2002

[28] Guha S., N. Koudas, A. Marathe. Merging the results of approximate match operations. VLDB 2004: 636-647.

[29] Chaudhuri S., Ganti V., Motwani R.. Robust identification of fuzzy duplicates. ICDE 2005: 865-876.

[30] Wang Y. R., Madnick S. E.. The inter-database instance identification problem in integrating autonomous systems. ICDE 1989: 46-55.

[31] Hernandez M. A., Stolfo S. J.. Real-world data is dirty: Data cleaning and the merge/purge problem. Data Mining and Knowledge Discovery, 2(1):9-37, January 1998

[32] Galhardas H., Florescu D., Shasha D.. Declarative data cleaning: Language, model, and algorithms. VLDB 2001: 371-380.

[33] Zhang Z., He B., Chang K. C.-C.: Light-weight Domain-based Form Assistant: Querying Web Databases On the Fly. VLDB 2005: 97-108

[34] Elmagarmid A. K., Ipeirotis P. G., Verykios V. S.: Duplicate Record Detection: A Survey. IEEE Trans. Knowl. Data Eng. 19(1): 1-16 (2007)

[35] Elfeky M., Verykios V., and Elmagarmid A.. TAILOR: A record linkage toolbox. ICDE 2002, 17–28.

[36] Arasu A., Ré C., Suciu D.: Large-Scale Deduplication with Constraints Using Dedupalog. ICDE 2009: 952-963