ISSN (Online) : 2319 – 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization, Volume 3, Special Issue 1, February 2014

International Conference on Engineering Technology and Science-(ICETS'14)

On 10th & 11th February Organized by

Department of CIVIL, CSE, ECE, EEE, MECHNICAL Engg. and S&H of Muthayammal College of Engineering, Rasipuram, Tamilnadu, India

Annotating Search Results from Web Databases Using Clustering-Based Shifting

Saranya.J[1], SelvaKumar.M[2], Vigneshwaran.S[3], Danessh.M.S[4]

Final year students, B.E-CSE, K.S.Rangasamy College of Technology, Tiruchengode, Tamilnadu, India[1, 2, 3]

Assistant Professor, K.S.Rangasamy College of Technology, Tiruchengode, Tamilnadu, India[4]

ABSTRACT – An increasing number of databases have become web accessible through HTML form-based search interfaces. The data units returned from the underlying database are usually encoded into the result pages dynamically for human browsing. For the encoded data units to be machine processable, which is essential for many applications such as deep web data collection and Internet comparison shopping, they need to be extracted out and assigned meaningful labels. In this paper, we present an automatic annotation approach that first aligns the data units on a result page into different groups such that the data in the same group have the same semantic. Then, for each group we annotate it from different aspects and aggregate the different annotations to predict a final annotation label for it. An annotation wrapper for the search site is automatically constructed and can be used to annotate new result pages from the same web database. Our experiments indicate that the proposed approach is highly effective. So this paper uses data alignment, data annotation, web databases and wrapper generation as the term to provide the user with much better result while they search for the terms.

I. INTRODUCTION

1.1 INTRODUCTION TO WEB MINING

A large portion of the deep web is database based, i.e., for many search engines, data encoded in the returned result pages come from the underlying structured databases. Such type of search engines is often referred as Web databases (WDB). A typical result page returned from a WDB has multiple search result records (SRRs). Each SRR contains multiple data units each of which describes one aspect of a real-world entity. There is a high demand for collecting data of interest from multiple WDBs. For example, once a book comparison shopping system collects multiple result records from different book sites, it needs to determine whether any two SRRs refer to the same book. The ISBNs can be compared to achieve this. If ISBNs are not available, their titles and authors could be compared. The system also needs to list the prices offered by each site. Thus, the system needs to know the semantic of each data unit. Unfortunately, the semantic labels of data units are often not provided in result pages. For instance, no semantic labels for the values of title, author, publisher, etc., are given. Having semantic labels for data units is not only important for the above record linkage task, but also for storing collected SRRs into a database table for later analysis. Early applications require tremendous human efforts to annotate data units manually, which severely limit their scalability. In this paper, we consider how to automatically assign labels to the data units within the SRRs returned from WDBs.

Given a set of SRRs that have been extracted from a result page returned from a WDB, our automatic annotation solution consists of three phases as illustrated in Fig. 1. Let dji denote the data unit belonging to the ith SRR of concept j. The SRRs on a result page can be represented in a table format (Fig. 1a) with each row representing an SRR. Phase 1 is the alignment phase. In this phase, we first identify all data units in the SRRs and then organize them into different groups with each group corresponding to a different concept (e.g., all titles are grouped together). Fig. 1b shows the result of this phase with each column containing data units of the same concept across all SRRs. Grouping data units of the same semantic can help identify the common patterns and features among these data units. These common features are the basis of our annotators. In Phase 2 (the annotation phase), we introduce multiple basic annotators with each exploiting one type of features. Every basic annotator is

used to produce a label for the units within their group holistically, and to determine the most appropriate label.
This paper has the following contributions:

1. While most existing approaches simply assign labels to each HTML text node, we thoroughly analyze the relationships between text nodes and data units. We perform data unit level annotation.
2. We propose a clustering-based shifting technique to align data units into different groups so that the data units inside the same group have the same semantic. Instead of using only the DOM tree or other HTML tag tree structures of the SRRs to align the data units (like most current methods do), our approach also considers other important features shared among data units, such as their data types (DT), data contents (DC), presentation styles (PS), and adjacency (AD) information.

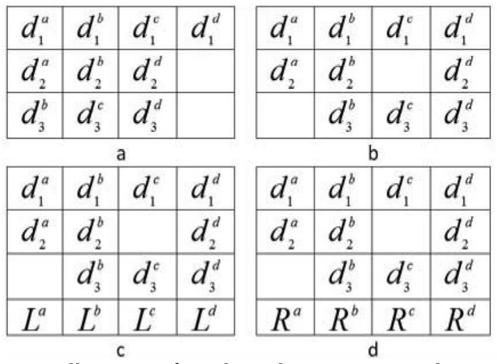


Fig. 1. Illustration of our three-phase annotation solution.

3. We utilize the integrated interface schema (IIS) over multiple WDBs in the same domain to enhance data unit annotation. To the best of our knowledge, we are the first to utilize IIS for annotating SRRs.
4. We employ six basic annotators; each annotator can independently assign labels to data units based on certain features of the data units. We also employ a probabilistic model to combine the results from different annotators into a single label. This model is highly flexible so that the existing basic annotators may be modified and new annotators may be added easily without affecting the operation of other annotators.
5. We construct an annotation wrapper for any given WDB. The wrapper can be applied to efficiently annotating the SRRs retrieved from the same WDB with new queries.

II. EXISTING SYSTEM

In this existing system, a data unit is a piece of text that semantically represents one concept of an entity. It corresponds to the value of a record under an attribute. It is different from a text node which refers to a sequence of text surrounded by a pair of HTML tags. It describes the relationships between text nodes and data units in detail. In this paper, we perform data unit level annotation. There is a high demand for collecting data of interest from multiple WDBs. For example, once a book comparison shopping system collects multiple result records from different book sites, it needs to determine whether any two SRRs refer to the same book.

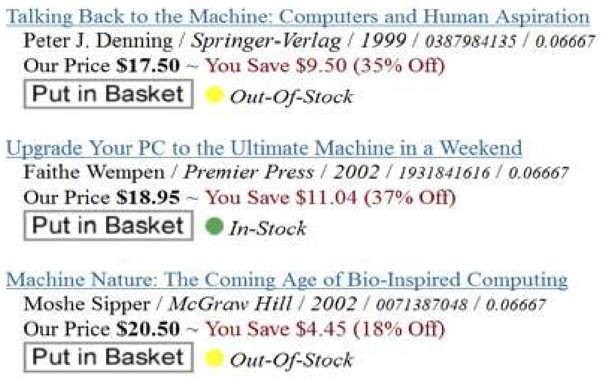
DISADVANTAGES OF EXISTING SYSTEM:

If ISBNs are not available, their titles and authors could be compared. The system also needs to list the prices offered by each site. Thus, the system needs to know the semantic of each data unit. Unfortunately, the semantic labels of data units are often not provided in result pages. For instance, no semantic labels for the values of title, author, publisher, etc., are given. Having semantic labels for data units is not only important for the above record linkage task, but also for storing collected SRRs into a database table.

III. PROPOSED SYSTEM:

In this paper, we consider how to automatically assign labels to the data units within the SRRs returned from WDBs. Given a set of SRRs that have been extracted from a result page returned from a WDB, our automatic annotation solution consists of five annotators:

- Table Annotator (TA)
- Query-Based Annotator (QA)
- Schema Value Annotator (SA)
- Frequency-Based Annotator (FA)
- In-Text Prefix/Suffix Annotator (IA)
- Common Knowledge Annotator (CA)


Talking Back to the Machine: Computers and Human Aspiration
Peter J. Denning / Springer-Verlag / 1999 / 0387984135 / 0.06667
Our Price $17.50 ~ You Save $9.50 (35% Off)
Put in Basket ● Out-Of-Stock

Upgrade Your PC to the Ultimate Machine in a Weekend
Faithe Wempen / Premier Press / 2002 / 1931841616 / 0.06667
Our Price $18.95 ~ You Save $11.04 (37% Off)
Put in Basket ● In-Stock

Machine Nature: The Coming Age of Bio-Inspired Computing
Moshe Sipper / McGraw Hill / 2002 / 0071387048 / 0.06667
Our Price $20.50 ~ You Save $4.45 (18% Off)
Put in Basket ● Out-Of-Stock

Fig. 1. Example search results from Bookpool.com.

1) TABLE ANNOTATOR (TA)

Many WDBs use a table to organize the returned SRRs. In the table, each row represents an SRR. The table header, which indicates the meaning of each column, is usually located at the top of the table. Fig. 6 shows an example of SRRs presented in a table format. Usually, the data units of the same concepts are well aligned with its corresponding column header. This special feature of the table layout can be utilized to annotate the SRRs.

Since the physical position information of each data unit is obtained during SRR extraction, we can utilize the information to associate each data unit with its corresponding header. Our Table Annotator works as follows: First, it identifies all the column headers of the table. Second, for each SRR, it takes a data unit in a cell and selects the column header whose area (determined by coordinates) has the maximum vertical overlap (i.e., based on the x-axis) with the cell. This unit is then assigned with this column header and labeled by the header text A (actually by its corresponding global name gn(A) if gn(A) exists). The remaining data units are processed similarly. In case that the table header is not provided or is not successfully extracted.

2) QUERY-BASED ANNOTATOR (QA)

The basic idea of this annotator is that the returned SRRs from a WDB are always related to the specified query. Specifically, the query terms entered in the search attributes on the local search interface of the WDB will most likely appear in some retrieved SRRs. For example, query term "machine" is submitted through the Title field on the search interface of the WDB and all three titles of the returned SRRs contain this query term. Thus, we can use the name of search field Title to annotate the title values of these SRRs. In general, query terms against an attribute may be entered to a textbox or chosen from a selection list on the local search interface. Our Query-based Annotator works as follows: Given a query with a set of query terms submitted against an attribute A on the local search interface, first find the group that has the largest total occurrences of these query terms and then assign gn(A) as the label to the group. As mentioned, the LIS of a WDB usually does not have all the attributes of the underlying database. As a result, the query-based annotator by itself cannot completely annotate the SRRs.

3) SCHEMA VALUE ANNOTATOR (SA)

Many attributes on a search interface have predefined values on the interface. For example, the attribute Publishers may have a set of predefined values (i.e., publishers) in its selection list. More attributes in the IIS tend to have predefined values and these attributes are likely to have more such values than those in LISs, because when attributes from multiple interfaces are integrated, their values are also combined. Our schema value annotator utilizes the combined value set to perform annotation.

Given a group of data units Gi ¼ fd1; . . . ; dng, the schema value annotator is to discover the best matched attribute to the group from the IIS. Let Aj be an attribute containing a list of values fv1; . . . ; vmg in the IIS. For each data unit dk, this annotator first computes the Cosine similarities between dk and all values in Aj to find the value with the highest similarity. Then, the data fusion function is applied to the similarities for all the data units. More specifically, the annotator sums up the similarities and multiplies the sum by the number of nonzero similarities. This final value is treated as the matching score between Gi and Aj.

The schema value annotator first identifies the attribute Aj that has the highest matching score among all attributes and then uses gn(Aj) to annotate the group Gi. Note that multiplying the above sum by the number of nonzero similarities is to give preference to attributes that have more matches (i.e., having nonzero similarities) over those that have fewer matches. This is found to be very effective in improving the retrieval effectiveness of combination systems in information retrieval.

4) FREQUENCY-BASED ANNOTATOR (FA)

In Fig. 1, "Our Price" appears in the three records and the followed price values are all different in these records. In other words, the adjacent units have different occurrence frequencies. As argued, the data units with the higher frequency are likely to be attribute names, as part of the template program for generating records, while the data units with the lower frequency most probably come from databases as embedded values. Following this argument, "Our Price" can be recognized as the label of the value immediately following it. The phenomenon described in this example is widely observable on result pages returned by many WDBs and our frequency-based annotator is designed to exploit this phenomenon. Consider a group Gi whose data units have a lower frequency. The frequency-based annotator intends to find common preceding units shared by all the data units of the group Gi. This can be easily conducted by following their preceding chains recursively until the encountered data units are different. All found preceding units are concatenated to form the label for the group Gi.

Example 2: In Fig. 1, during the data alignment step, a group is formed for {"$17.50," "$18.95," "$20.50"}. Clearly the data units in this group have different values. These values share the same preceding unit "Our Price," which occurs in all SRRs. Furthermore, "Our Price" does not have preceding data units because it is the first unit in this line. Therefore, the frequency-based annotator will assign label "Our Price" to this group.

5) IN-TEXT PREFIX/SUFFIX ANNOTATOR (IA)

In some cases, a piece of data is encoded with its label to form a single unit without any obvious separator between the label and the value, but it contains both the label and the value. Such nodes may occur in all or multiple SRRs. After data alignment, all such nodes would be aligned together to form a group. For example, in Fig. 1, after alignment, one group may contain three data units, {"You Save $9.50," "You Save $11.04," "You Save $4.45"}.

The in-text prefix/suffix annotator checks whether all data units in the aligned group share the same prefix or suffix. If the same prefix is confirmed and it is not a delimiter, then it is removed from all the data units in the group and is used as the label to annotate values following it. If the same suffix is identified and if the number of data units having the same suffix match the number of data units inside the next group, the suffix is used to annotate the data units inside the next group. In the above example, the label "You save" will be assigned to the group of prices. Any group whose data unit texts are completely identical is not considered by this annotator.

6) COMMON KNOWLEDGE ANNOTATOR (CA)

Some data units on the result page are self-explanatory because of the common knowledge shared by human beings. For example, "in stock" and "out of stock" occur in many SRRs from e-commerce sites. Human users understand that it is about the availability of the product because this is common knowledge. So our common knowledge annotator tries to exploit this situation by using some predefined common concepts.

Each common concept contains a label and a set of patterns or values. For example, a country concept has a label "country" and a set of values such as "U.S.A.," "Canada," and so on. As another example, the e-mail address (assume all lower cases) concept has the pattern ½a-z0 _ 9: %þ__ þ @ð½a-z0 _ 9__ þ n:Þ þ ½a-z_f2; 4g. Given a group of data units from the alignment step, if all the data units match the pattern or value of a concept, the label of this concept is assigned to the data units of this group.

It also uses some conventions to annotate data units. However, it only considers certain patterns. Our Common knowledge annotator considers both patterns and certain value sets such as the set of countries.

It should be pointed out that our common concepts are different from the ontologies that are widely used in some works in Semantic Web (e.g., [6], [11], [12], [16], [26]). First, our common concepts are domain independent. Second, they can be obtained from existing information resources with little additional human effort.

7) COMBINING ANNOTATOR

Our analysis indicates that no single annotator is capable of fully labeling all the data units on different result pages. The applicability of an annotator is the percentage of the attributes to which the annotator can be applied. For example, if out of 10 attributes, four appear in tables, then the applicability of the table annotator is 40 percent. Table 1 shows the average applicability of each basic annotator across all testing domains in our data set. This indicates that the results of different basic annotators should be combined in order to annotate a higher percentage of data units. Moreover, different annotators may produce different labels for a given group of data units. Therefore, we need a method to select the most suitable one for the group.

Our annotators are fairly independent from each other since each exploits an independent feature. Based on this characteristic, we employ a simple probabilistic method to combine different annotators. For a given annotator L, let P(L) be the probability that L is correct in identifying a correct label for a group of data units when L is applicable. P(L) is essentially the success rate of L. Specifically, suppose L is applicable to N cases and among these cases M are annotated correctly, then P(L) ¼ M=N. If k independent annotators Li, i ¼ 1; . . . ; k, identify the same label for a group of data units, then the combined probability that at least one of the annotators is correct is:

$$1-\pi(1-P(L))$$

To obtain the success rate of an annotator, we use the annotator to annotate every result page in a training data set. It can be seen that the table annotator is 100 percent correct when applicable. The query-based annotator also has very high success rate while the schema value annotator is the least accurate. An important issue DeLa did not address is what if multiple heuristics can be applied to a data unit. In our solution, if multiple labels are predicted for a group of data units by different annotators, we compute the combined probability for each label based on the annotators that identified the label, and select the label with the largest combined probability.

ANNOTATION WRAPPER

Once the data units on a result page have been annotated, we use these annotated data units to construct an annotation wrapper for the WDB so that the new SRRs retrieved from the same WDB can be annotated using this wrapper quickly without reapplying the entire annotation process. We now describe our method for constructing such a wrapper below.

Each annotated group of data units corresponds to an attribute in the SRRs. The annotation wrapper is a description of the annotation rules for all the attributes on the result page. After the data unit groups are annotated, they are organized based on the order of its data units in the original SRRs. Consider the ith group Gi. Every SRR has a tag-node sequence like Fig. 1b that consists of only HTML tag names and texts. For each data unit inGi, we scan the sequence both backward and forward to obtain the prefix and suffix of the data unit. The scan stops when an encountered unit is a valid data unit with a meaningful label assigned. Then, we compare the prefixes of all the data units in Gi to obtain the common prefix shared by these data units. Similarly, the common suffix is obtained by comparing all the suffixes of these data units. For example, the data unit for book title has "<FORM><A>" as its prefix and "</A><BR>" as its suffix. If a data unit is generated by splitting from a composite text node, then its prefix and suffix are the same as those of its parent data unit. This wrapper is similar to the LR wrapper. Here, we use prefix as the left delimiter, and suffix as the right delimiter to identify data units. However, the LR wrapper has difficulties to extract data units packed inside composite text nodes due to the fact that there is no HTML tag within a text node. To overcome this limitation, besides the prefix and suffix, we also record the separators used for splitting the composite text node as well as its position index in the split unit vector. Thus, the annotation rule for each attribute consists of five components, expressed as: attributei ¼ <labeli; prefixi; suffixi; separatorsi; unitindexi>. The annotation wrapper for the site is simply a collection of the annotation rules for all the attributes identified on the result page with order corresponding to the ordered data unit groups.

To use the wrapper to annotate a new result page, for each data unit in an SRR, the annotation rules are applied on it one by one based on the order they appear in the wrapper. If this data unit has the same prefix and suffix as specified in the rule, the rule is matched and the unit is labeled with the given label in the rule. If the separators are specified, they are used to split the unit, and labeli is assigned to the unit at the position unitindexi.

EXPERIMENTS

1) Data Sets and Performance Measure

Our experiments are based on 112 WDBs selected from seven domains: book, movie, music, game, job, electronics, and auto. For each WDB, its LIS is constructed automatically using WISEiExtractor. For each domain, WISE-Integrator is used to build the IIS automatically. These collected WDBs are randomly divided into two disjoint groups. The first group contains 22 WDBs and is used for training, and the second group has 90 WDBs and is used for testing. Data set DS1 is formed by obtaining one sample result page from each

ISSN (Online) : 2319 – 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization, Volume 3, Special Issue 1, February 2014

International Conference on Engineering Technology and Science-(ICETS'14)

On 10th & 11th February Organized by

Department of CIVIL, CSE, ECE, EEE, MECHNICAL Engg. and S&H of Muthayammal College of Engineering, Rasipuram, Tamilnadu, India

training site. Two testing data sets DS2 and DS3 are generated by collecting two sample result pages from each testing site using different queries. We note that we largely recollected the result pages from WDBs used.

We have noticed that result pages have become more complex in general as web developers try to make their pages fancier. Each testing data set contains one page from each WDB. Some general terms are manually selected as the query keywords to obtain the sample result pages. The query terms are selected in such a way that they yield result pages with many SRRs from all WDBs of the same domain, for example, "Java" for Title, "James" for Author, etc. The query terms and the form element each query is submitted to are also stored together with each LIS for use by the Query-based Annotator. Data set DS1 is used for learning the weights of the data unit features and clustering threshold T in the alignment step, and determining the success rate of each basic annotator. For each result page in this data set, the data units are manually extracted, aligned in groups, and assigned labels by a human expert. We use a genetic algorithm based method to obtain the best combination of feature weights and clustering threshold T that leads to the best performance over the training data set. DS2 is used to test the performance of our alignment and annotation methods based on the parameter values and statistics obtained from DS1. At the same time, the annotation wrapper for each site will be generated. DS3 is used to test the quality of the generated wrappers. The correctness of data unit extraction, alignment, and annotation is again manually verified by the same human expert for performance evaluation purpose. We adopt the precision and recall measures from information retrieval to evaluate the performance of our methods. For alignment, the precision is defined as the percentage of the correctly aligned data units over all the aligned units by the system; recall is the percentage of the data units that are correctly aligned by the system over all manually aligned data units by the expert. A result data unit is counted as "incorrect" if it is mistakenly extracted (e.g., failed to be split from composite text node). For annotation, the precision is the percentage of the correctly annotated units over all the data units annotated by the system and the recall is the percentage of the data units correctly annotated by the system over all the manually annotated units.

A data unit is said to be correctly annotated if its system-assigned label has the same meaning as its manually assigned label.

2) Experimental Results

The optimal feature weights obtained through our genetic training method (See Section 4) over DS1 are {0.64, 0.81, 1.0, 0.48, 0.56} for SimC, SimP, SimD, SimT, and SimA, respectively, and 0.59 for clustering threshold T. The average alignment precision and recall are converged at about 97 percent. The learning result shows that the data type and the presentation style are the most important features in our alignment method. Then, we apply our annotation method on DS1 to determine the success rate of each annotator.

It shows the performance of our data alignment algorithm for all 90 pages in DS2.The precision and recall for every domain are above 95 percent, and the average precision and recall across all domains are above 98 percent. The performance is consistent with that obtained over the training set. The errors usually happen in the following cases. First, some composite text nodes failed to be split into correct data units when no explicit separators can be identified. For example, the data units in some composite text nodes are separated by blank spaces created by consecutive HTML entities like "&nbsp;" or some formatting HTML tags such as <SPAN>. Second, the data units of the same attribute across different SRRs may sometimes vary a lot in terms of appearance or layout. For example, the promotion price information often has color or font type different from that for the regular price information. Note that in this case, such two price data units have low similarity on content, presentation style, and the tag path.

Even though they share the same data type, the overall similarity between them would still be low. Finally, the decorative tag detection (Step 1 of the alignment algorithm) is not perfect (accuracy about 90 percent), which results in some tags to be falsely detected as decorative tags, leading to incorrect merging of the values of different attributes. We will address these issues in the future. It lists the results of our annotation performance over DS2. We can see that the overall precision and recall are very high, which shows that our annotation method is very effective.Moreover, high precision and recall are achieved for every domain, which indicates that our annotation method is domain independent. We notice that the overall recall is a little bit higher than precision, mostly because some data units are not correctly separated in the alignment step.

We also found that in a few cases some texts are not assigned labels by any of our basic annotators. One reason is that some texts are for cosmetic or navigating purposes. These texts do not represent any attributes of the real-world entity and they are not the labels of any data unit, which belong to our One-To-Nothing relationship type. It is also possible that some of these texts are indeed data units but none of our current basic annotators are applicable to them. For each webpage in DS2, once its SRRs have been annotated, an annotation wrapper is built, and it is applied to the corresponding page in DS3. In terms of processing speed, the time needed to annotate one result page drops from 10-20 seconds without using wrapper to 1-2 seconds using wrapper depending on the complexity of the page. The reason is that wrapper based approach directly extracts all data units specified by the tag path(s) for each attribute and assigns the label specified in the rule to those data units. In contrast, the non-wrapper-based approach needs to go through some time-consuming steps such as result page rendering, data unit similarity matrix computation, etc., for each result page.

In terms of precision and recall, the wrapper-based approach reduces the precision by 2.7 percentage points and recall by 4.6 percentage points, as can be seen from the results. The reason is that our wrapper only used tag path for text node extraction and alignment, and the composite text node splitting is solely based on the data unit position. In the future, we will investigate how to incorporate other features in the wrapper to increase the performance.

Another reason is that in this experiment, we used only one page for each site in DS2 to build the wrapper. As a result, it is possible that some attributes that appear on the page in DS3 do not appear on the training page in DS2 (so they do not appear in the wrapper expression).

For example, some WDBs only allow queries that use only one attribute at a time (these attributes are called exclusive attributes in [15]). In this case, if the query term is based on Title, then the data units for attribute Author may not be correctly identified and annotated because the query-based annotator is the main technique used for attributes that have a textbox on the search interface. One possible solution to remedy this problem is to combine multiple result pages based on different queries to build a more robust wrapper. We also conducted experiments to evaluate the significance of each feature on the performance of our alignment algorithm. For this purpose, we compare the performance when a feature is used with that when it is not used. Each time one feature is selected not to be used, and its weight is proportionally distributed to other features based on the ratios of their weights to the total weight of the used features. The alignment process then uses the new parameters to run on DS2. It shows the results. It can be seen that when any one of these features is not used, both the precision and recall decrease, indicating that all the features in our approach are valid and useful.

We can also see that the data type and the presentation style are the most important features because when they are not used, the precision and recall drop the most (around 28 and 23 percentage points for precision, and 31 and 25 percentage points for recall, respectively). This result is consistent with our training result where the data type and the presentation style have the highest feature weights. The adjacency and tag path feature are less significant comparatively, but without either of them, the precision and recall drop more than 15 percentage points. We use a similar method as the above to evaluate the significance of each basic annotator. Each time, one annotator is removed and the remaining annotators are used to annotate the pages in DS2. It shows the results. It shows that omitting any annotator causes both precision and recall to drop, i.e., every annotator contributes positively to the overall performance. Among the six annotators considered, the query-based annotator and the frequency-based annotator are the most significant. Another observation is that when an annotator is removed, the recall decreases more dramatically than precision. This indicates that each of our annotators is fairly independent in terms of describing the attributes. Each annotator describes one aspect of the attribute which, to a large extent, is not applicable to other annotators.

Finally, we conducted experiments to study the effect of using LISs versus using the IIS in annotation. We run the annotation process on DS2 again but this time, instead of using the IIS built for each domain, we use the LIS of each WDB. The results are shown in Table 5. By comparing the results we can see that using the LIS has a relatively small impact on precision, but significant effect on recall (the overall average recall is reduced by more than 6 percentage points) because of the local interface schema inadequacy problem. And it also shows that using the integrated interface schema can indeed increase the annotation performance. As reported, only three domains (Book, Job, and Car) were used to evaluate DeLa and for each domain, only nine sites were selected. The overall precision and recall of DeLa's annotation method using this data set are around 80 percent. In contrast, the precision and recall of our annotation method for these three domains are well above 95 percent, although different sets of sites for these domains are used in the two works.

IV. CONCLUSION

In this paper, we studied the data annotation problem and proposed a multi-annotator approach to automatically constructing an annotation wrapper for annotating the search result records retrieved from any given web database. This approach consists of six basic annotators and a probabilistic method to combine the basic annotators. Each of these annotators exploits one type of features for annotation and our experimental results show that each of the annotators is useful and they together are capable of generating high quality annotation. A special feature of our method is that, when annotating the results retrieved from a web database, it utilizes both the LIS of the web database and the IIS of multiple web databases in the same domain. We also explained how the use of the IIS can help alleviate the local interface schema inadequacy problem and the inconsistent label problem.

In this paper, we also studied the automatic data alignment problem. Accurate alignment is critical to achieving holistic and accurate annotation. Our method is a clustering based shifting method utilizing richer yet automatically obtainable features. This method is capable of handling a variety of relationships between HTML text nodes and data units, including one-to-one, one-to-many, many-to-one, and one-to-nothing. Our experimental results show that the precision and recall of this method are both above 98 percent. There is still room for improvement in several areas. For example, we need to enhance our method to split composite text node when there are no explicit separators. We would also like to try using different machine learning techniques and using more sample pages from each training site to obtain the feature weights so that we can identify the best technique to the data alignment problem.

REFERENCES

[1] S. Handschuh, S. Staab, and R. Volz, "On Deep Annotation," Proc. 12th Int'l Conf. World Wide Web (WWW), 2003.

[2] S. Handschuh and S. Staab, "Authoring and Annotation of Web Pages in CREAM," Proc. 11th Int'l Conf. World Wide Web (WWW), 2003.

[3] B. He and K. Chang, "Statistical Schema Matching Across Web Query Interfaces," Proc. SIGMOD Int'l Conf. Management of Data, 2003.

[4] H. He, W. Meng, C. Yu, and Z. Wu, "Automatic Integration of Web Search Interfaces with WISE Integrator," VLDB J., vol. 13, no. 3, pp. 256-273, Sept. 2004.

[5] H. He, W. Meng, C. Yu, and Z. Wu, "Constructing Interface Schemas for Search Interfaces of Web Databases," Proc. Web Information Systems Eng. (WISE) Conf., 2005.

[6] J. Heflin and J. Hendler, "Searching the Web with SHOE," Proc. AAAI Workshop, 2000.

[7] L. Kaufman and P. Rousseeuw, Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley & Sons, 1990.

[8] N. Krushmerick, D. Weld, and R. Doorenbos, "Wrapper Induction for Information Extraction," Proc. Int'l Joint Conf. Artificial Intelligence (IJCAI), 1997.

[9] J. Lee, "Analyses of Multiple Evidence Combination," Proc. 20th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval, 1997.

[10] L. Liu, C. Pu, and W. Han, "XWRAP: An XML-Enabled Wrapper Construction System for Web Information Sources," Proc. IEEE 16th Int'l Conf. Data Eng. (ICDE), 2001.