Toward Entity Retrieval over Structured and Text Data

Mayssam Sayyadian, Azadeh Shakery, AnHai Doan, ChengXiang Zhai
Department of Computer Science, University of Illinois at Urbana-Champaign
{sayyadia, shakery, anhai, czhai}@cs.uiuc.edu

ABSTRACT

Many real-world applications increasingly involve both structured data and text. Hence, managing both in an efficient and integrated manner has received much attention from both the IR and database communities. To date, however, little research has been devoted to semantic issues in the integration of text and data. In this paper we introduced a problem in this realm: *entity retrieval.* Given data fragments that describe various aspects of a real-world entity, find all other data fragments as well as text documents that describe that same entity. As such, entity retrieval is a novel retrieval problem, which differs from both regular text retrieval and database search in that it explicitly requires matching information at the *semantic* level; matching syntactically as done in the current search engines and relational databases would be inherently non-optimal. We define entity retrieval and conduct a case study of retrieving information about a researcher from both the Web and a bibliographic database (DBLP). We propose several methods for exploiting the structured information in the database to improve entity retrieval over the text collection. Specifically, we present a query expansion mechanism based on extracted information from structured data. Experiment results show that selectively using more structured information to expand the text query improves entity retrieval performance on text. We conclude the paper with future research directions for entity retrieval.

Keywords

Entity Retrieval, Semantic Retrieval, Query Expansion, Information Retrieval, Disambiguating Search Results

1. INTRODUCTION

Traditionally, text and structured data have been managed in different ways, resulting in two related but separate fields: information retrieval (IR) and management of structured data, as exemplified by relational database systems.

However, the boundary between the two fields has become much blurred in the past few years. For example, database researchers have been actively studying how to incorporate text search facilities into relational database systems, while IR researchers have investigated exploiting structure over the data (e.g., XML structure) to perform more expressive keyword queries (see the related work section). This trend is motivated largely by a fundamental need to manage text and structured data in an integrated way, given the proliferation of applications that involve *both* kinds of data. In such applications, users seek all information that can help solving a problem, regardless of whether the information comes from text or structured data. Hence, it is important to develop techniques that enable the efficient discovery of information from both text and structured data.

This paper aims to develop such techniques. It focuses on *entity retrieval*, the problem of finding missing information about a *real-world entity* (e.g., person, course, or place) from both text and structured data, given some initial information about the entity. Examples include finding information about a researcher from the World-Wide Web and a set of bibliographic databases, given a paper written by that researcher, or finding information about a product of a business competitor, from online discussion groups and a purchase database.

As described, entity retrieval (or ER for short) plays a fundamental role in many information seeking contexts, but has received little attention. Furthermore, current IR and database techniques do not appear to be well suited to this problem. IR techniques has been optimized largely for retrieving documents on *general topics* (e.g., "data mining"). When applied to retrieving documents about a particular real-world entity, say person, they often return documents about multiple persons that are mixed together: a highly undesirable situation. Current relational database techniques also do not fare well on this problem, as they can only perform *syntactic matching* and thus often return information about multiple real-world entities.

It is also worth mentioning that the notion of relevance in text retrieval is *subjectively* defined; the criterion is in the user's mind. But in ER, the notion of relevance has an *objective* definition. For example, we may simply use a person's name as a query to search over text. But this only works when the name is not ambiguous. Similarly, querying a database with a person's name may return tuples that describe a different entity who happens to have the same name as our target entity. Thus to optimize ER results, we must infer entity-level semantics, that is, decide if two data fragments refer to the same real-world entity. This is a major reason on why ER is a challenging problem.

WIRD'04, the first Workshop on the Integration of Information Retrieval and Databases (WIRD'04), Sheffield, United Kingdom
© 2004 the author/owner

Clearly, we can study the ER problem given (1) only structured data, (2) only text data, and (3) a combination of text and structured data. In practice, all three settings arise very often. In [12] we have carried out a detailed study of setting (1). We are currently studying settings (2) and (3).

This paper focuses on setting (3), and examines a specific problem in this setting, namely, *given both text and structured data, can we exploit structured data to achieve better ER over text, compared to ER over text alone?* We first give an informal definition of this problem, then propose several solutions. The key insight behind our solutions is that we can selectively expand the ER query over text with keywords obtained from the given structured data.

The key challenge in realizing the above idea is to discover which parts of structured data should be considered, and which keywords to "pull" from those parts (to be used to augment the query). We note that many variations of query expansion have been considered in the IR community, but few if any works have considered query expansion using structured data.

We then evaluate our solutions on a real-world setting, in which we retrieve information about a researcher from both the World-Wide Web and DBLP, a bibliographic database. Our preliminary experimental results show that selectively using more structured information to expand the text query improves the ER performance on text.

The rest of the paper is organized as follows. We define the ER problem in Section 2. In Section 3 we present several ER methods that can exploit structured information to improve ER over text. We discuss experiment results in Section 4. Finally, we review related work in Section 5 and discuss future research directions in Section 6.

2. THE ENTITY RETRIEVAL PROBLEM

Informally, given a collection C of text documents, a relational database T, and an ER query that specifies the description of a particular real-world entity called *target entity*, the ER problem is to find documents in C and data fragments in T that describes the entity. For example, we may want to retrieve information about a particular researcher from both a bibliographic database (e.g., DBLP) and a text collection (e.g., web pages) given some information about the researcher (e.g., the researcher's name and affiliation as well as his home page).

An ER query consists of two parts: a constraint and a template. The constraint specifies whatever information a user already knows about the target entity. This information typically includes some attribute values in the database and some documents in the text collection. For example, a query may specify the following: (1) the name of the target researcher is "*John Smith*", (2) he has published a paper titled "*A study of entity retrieval*" in the conference *SIGIR 2004*, (3) the text of his home page. The first two specify the values of three attributes in a bibliographic database – author, title, and conference. The last one gives one example of text documents. The target entity is a person.

The template of an ER query specifies what information the user wants returned about the target entity. To continue with the above example, the user may specify that he or she want to retrieve only all papers that the target researcher has written as well as any course that is reading any of these papers. In this case, most of the papers may come from the bibliographic database, while information about courses (that read these papers) may come from the World-Wide Web.

While we could have defined ER on only a relational database or only a text collection, it is interesting to consider ER over both text and relational data. This way it is possible to leverage useful context information from different types of data to improve retrieval accuracy. Indeed, in this paper, we will show that the performance of ER over a pure text collection can be improved by exploiting the structured data related to the target entity in a relational database. As a preliminary study, we will consider a simplified ER problem, though we will also touch on the general ER problem and the major research issues.

2.1 The Simplified ER Problem:

Let $E = \{e_1, ..., e_{|E|}\}$ be a set of real world entities. Let T be a relational table with attributes $A = \{A_1, ..., A_k\}$, such that each tuple in T describes some aspects of entities in E. Let $C = \{d_1, ..., d_n\}$ be a set of documents. A user who is interested in entity e_i would pose a query Q which contains some or all of the following information:

1. A basic description of e_i (e.g., name of a researcher),
2. Structured context information $c_1 = v_1, c_2 = v_2, \ldots, c_k = v_k$, where $c_j \in A$ is an attribute, and v_j is the corresponding value.
3. Unstructured context information in the form of a set of example text documents that are known to be about the target entity.
4. A template that is specified as a set of attributes in A, or in a more expressive language (e.g., SQL) over the schema of table T, to identify the types of information the user would like to obtain about the target entity e_i.

Note that both (2) and (3) can be regarded as providing context information for (1).

We further simplify the above problem by focusing only on ER over text. However, the ideas that we offer here also carry over to the setting of ER over structured data.

3. ER METHODS

Because of the need for semantic-level matching, ER is in general a quite challenging task. A straightforward matching at the syntactic level (e.g., by retrieving researchers with the exact same name) is clearly insufficient to achieve good performance. Intuitively, the ER task involves two subtasks: understanding the target entity referred to by a query and deciding whether a data fragment is about the same entity.

Unfortunately, we generally have no *unique* representation of a real world entity, which makes this task difficult. The lack of such a unique representation is obvious in text information where the primary representation is words describing an entity. It is also true in relational data, for example, when two people have the same name. It can be argued that entities in relational databases have unique identifiers such as primary key, but again when considering different relations and different mentions of a real-world entity we have no unique representations to identify the entity. Another complication is that there may be variations of spelling or wording for the same attribute value. For example, a name may be abbreviated. The challenge is thus to infer whether an information item is about the same entity referred to by a query based on the generally non-unique descriptions of various kinds of entities.

Since we have two different types of data, a straightforward strategy is to separate ER over text from ER over structured data. However, a complete separation would deprive us of the opportunity to leverage different types of data to enrich our representation of the target entity. Indeed, a main point we make in this paper is to advocate methods that take advantage of *all* information available to us, and in particular, to leverage structured information to improve ER over text.

We now present four different methods for ER over text, corresponding to different types of queries and/or different levels of sophistication in leveraging the related structured data. We assume that we have available a basic text retrieval method that can match any text query with the documents in the text collection and generate a relevance score for each document. The text documents can then be ranked according to their relevance scores. The retrieval results are obtained by applying some pre-determined score threshold cutoff. Currently this threshold is manually set and tuning it is the subject of future work. There are many choices of text retrieval methods (e.g., [35, 9]), but this is not our focus. Instead, we are interested in different ways for constructing a regular text query based on the ER query, especially on how we may exploit structured information to construct a potentially better text query.

The first, also the simplest method, is to use as the text query only the text description of the target entity in the ER query. This method, called Text Only, does not exploit any structured information, and will be used as our baseline method.

The second method uses immediate and highly reliable context information from the database to expand the Text Only query, and is referred to as Add Immediate Structure. The third method (Add All Structures) uses more structured information than the second method, and expands the Text Only query with all structured information that it found to be describing the target entity.

The last method attempts to further improve the quality of the query in Add All Structures, by *selectively* using only the most relevant attribute values to expand the Text Only query, and will be referred to as Add Selective Structures. We now describe these methods in detail.

3.1 Text Only

The Text Only method ignores any structured information in the query, and simply uses the text description of the target entity as the text query for searching the text collection. For example, when the target entity is a person, the person's name can be used directly as the query.

This method only uses the minimum amount of information in the query, and will clearly not perform well when the description is non-discriminative. For example, when the name of a researcher is ambiguous, using only the name may return documents about a different person who happens to have the same name.

When the user can provide some structured information about the target entity, it is intuitively better to enrich our query with such extra structured information. For example, if a user knows one paper published by a researcher, the title and/or the conference information of this paper can then be exploited to expand the query. Note that, when searching information about a researcher, even if the name is not ambiguous, such an expanded query can still be expected to perform better than Text Only, since the additional structured information can be expected to help refine the text query and thus improve the retrieval accuracy. This strategy leads to the method Add Immediate Structure.

3.2 Add Immediate Structure

This method expands the Text Only query with the immediate (one record), but highly reliable (e.g., provided by the user or automatically obtained when the name is not ambiguous) structured information. To obtain the expanded query, we simply treat the database record (e.g., the title, coauthors, and the conference name of the known publication) as additional text and append it to the original text query. More sophisticated expansion methods are possible. For example, we can incorporate the record text with different weights. But we leave this as our future work. This is also true for all other methods that we describe below.

3.3 Add All Structures

Since in general we can expect the retrieval performance to improve as we provide more relevant context to expand the query, we can expand the query with as much structured information as possible. That is, we go further than just adding the most reliable database record(s) and expand the query with any related structured information (about the target entity) that we can obtain from the database. This method is called Add All Structures and would require doing ER over the database.

All the additional structured information is then appended to the original text query. In the case of searching information about a researcher, this corresponds to using the papers published by the researcher that we can obtain from the bibliographic database to expand the query. Once again, this is the simplest expansion method, and more sophisticated methods that involve weighting the structured information differently than the original text may be more appropriate as we may have noise in the obtained structured information.

3.4 Add Selective Structures

While ADD ALL STRUCTURES allows us to use the "maximum" amount of relevant structured information to expand the text query, not the fields in the extracted database records are equally useful. Indeed, some fields (e.g., the publication dates) may be quite distracting, especially if they are not weighted appropriately. Thus one idea to further exploit the database schema is to *selectively* add only those highly relevant attribute values. This is the ADD SELECTIVE STRUCTURES method. To do so, we can expand the query with only the values of the *most relevant attribute*, which is to be selected with an attribute selection method. So devising a good method for selecting more informative, valuable, and disambiguating attributes from the structured data is very important.

One such method we devise and use is to score each attribute in the database schema based on its frequency in the top a few text documents retrieved by previous methods. A method for calculating the effectiveness of each attribute and selecting the best to use in ADD SELECTIVE STRUCTURES is as follows:

- W = *set of documents returned by* TEXT ONLY *method.*
- D = *The first* $\frac{size(W)}{4}$ *documents in* W
- *For each attribute* i *and all its values in the database schema;* $a_{i,j}$, *let* $s_{i,j}$ = *the number of occurrence of* $a_{i,j}$ *in* D
- *let the effectiveness score of attribute* i *be:* $score_i = \sum_j s_{i,j}$
- *normalize* $score_i$ *according to the size of* D *and number of values of* a_i
- BEST ATTRIBUTE $= argmax_i\{score_i\}$

The assumption is that an attribute can be expected to be more useful if it occurs more frequently in the top text documents. Once again, we concatenate the selected attribute values with the original text query to generate an expanded query.

4. EMPIRICAL EVALUATION

We evaluate our solutions using a bibliographic database (DBLP) and a collection of Web pages (constructed using Google). The ER task is to retrieve information about a researcher from both the database and the Web-page collection, given the researcher's name and some citation information from the bibliographic database. Our goal is to study whether the performance of ER over Web pages can be improved as we exploit more structured information.

4.1 Data Set and Evaluation Measures

Data Set: Our structured data source is the DBLP bibliography database at $www.informatik.uni-trier.de/\ ley/db$. DBLP lists more than 460,000 articles and has become a very popular bibliography search engine for the computer science research community. Our test bed consists of 11 researchers from DBLP whose names are ambiguous. Out of these researchers, 36% of the researchers have totally ambiguous names, meaning that when searching the Web with their names, we come across different real world people (entities) with completely identical names. For example "Amit Singhal" (one of our queries) appears to match two people – one is a researcher in IR, while the other is a researcher in system architecture. 45% of the names are partially ambiguous, meaning that there are entities with similar names differing only in the middle name, or initials. For example "Richard Cox" and "Richard V. Cox" are two researchers in this group.

Our text collection is constructed as follows. We use the name of each of the 11 researchers as a simple keyword query to search with Google, and take the top 100 Web pages as a "working set". We combine these working sets for all the 11 researchers to form our text collection. Since we have deliberately chosen ambiguous names, a working set generally has documents about distinct entities that share the same name, thus our text collection is adequate for discriminating different entity retrieval methods.

In order to measure the ER accuracy, we create a gold standard by manually reviewing these results and judging whether they are relevant to the corresponding entity query (i.e., whether they are truly about the target researcher). In this way, we can create a judgment file similar to those used in standard IR evaluation [37]. Note that although our judgments are not on the entire Web, they are still very useful for comparing different methods.

Evaluation Measures: We use precision, recall, and the $F1$ measure to evaluate the ER results. Given a set of result pages for a researcher query, precision is the percentage of pages retrieved that are relevant (i.e., truly about the target researcher), and the recall is the fraction of all relevant pages in our working set that are returned as results. $F1$ is a combination of precision and recall, and will be our primary measure when comparing different methods. It is given by $F1 = \frac{2PR}{P+R}$, where P and R are precision and recall respectively.

4.2 Implementation of ER Methods

Our general experiment procedure is to use one of our methods to construct a text query, which is then used to search the text collection with a standard IR method. We use the Lemur toolkit [26] to index our web text collection and perform text retrieval using the popular vector-space retrieval model [32] with BM25 TF weighting [30]. We used the default parameter seating provided in Lemur. Since we use a constant score threshold for cutoff, it is important to make the scores of all documents for all queries comparable. We thus normalize all the similarity scores by dividing the scores by the maximum score given by the top-ranked document. The cutoff value here is set manually and we will further look into it in future.

The implementation of the TEXT ONLY method is trivial, as it simply involves taking the name of the researcher as the text query. The ADD IMMEDIATE STRUCTURE method generates a text query by manually picking one citation that belongs to the target researcher from DBLP and concatenating the text in all fields with the researcher's name. This simulates a query posed by a user who has some limited additional information about the target researcher.

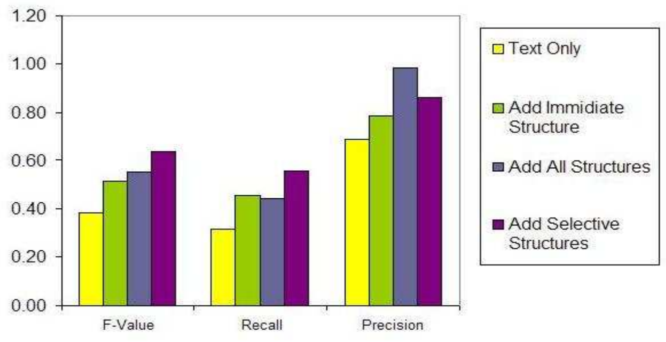

Figure 1: Effect of adding structured information to the ER query.

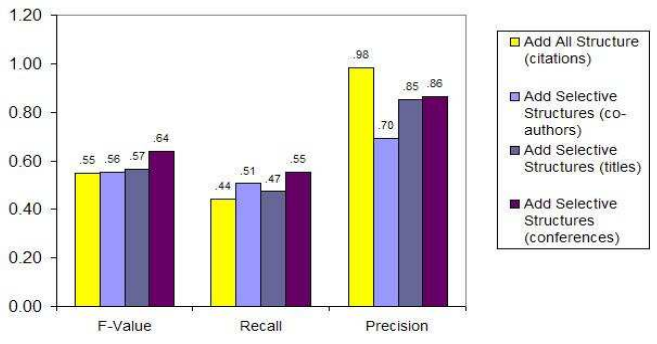

Figure 2: Comparing the effectiveness of using different attributes to expand the ER query.

The Add All Structures method assumes that we can perform ER over the database with reasonable accuracy, a task that we have studied in [12]. The method takes all papers that are published by the target researcher and concatenate all of them as text with the researcher's name to generate an expanded text query. We implement this method by exploiting the DBLP web server to find relevant citations to the target researcher. Specifically, we search DBLP with the researcher's name and manually choose all the papers that belong to the target researcher. In this way, all the chosen papers would be relevant, and thus the Add All Structures method is quite similar to relevance feedback in pure text retrieval.

The Add Selective Structures method is implemented in almost exactly the same way as Add All Structures with only one difference: while the Add All Structures method adds all the text information from a relevant paper, which includes coauthors, titles, dates, and conferences, the Add Selective Structures method only adds all the information in the values of the most highly scored attribute, which is the conference name. We score each attribute of the database schema by a formula based on normalized frequency of the values of the corresponding attribute in the top few web pages from previous text retrieval results, and only add the text values of the attribute with the highest score.

4.3 Result Analysis

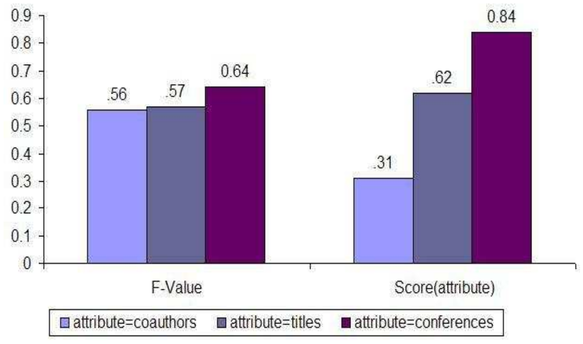

Figure 3: Correlation between the attribute scores and the F-Values of using them.

4.3.1 *Comparison of the four proposed methods*

In Figure 1, we compare the four proposed methods in terms of all three measures. The baseline is the Text Only method which uses only the researcher's name as the query. As expected, Add Immediate Structure performs much better than the baseline by all three measures, indicating that the added paper provides very useful context to focus the results on the target person entity.

As we use more structured information in the Add All Structures method, we see that we can further improve the precision and F1 value, though the recall decreases slightly. The increase of precision and decrease in recall suggest that the number of pages returned is reduced. This may be caused by a dramatic increase in the similarity score of a few web pages (e.g., the publication page of the researcher); with the same relative score threshold, the effect is a decrease in the number of pages that can pass the threshold.

Finally, by comparing Add All Structures with Add Selective Structures, we see that selectively using the values from the most effective attribute to expand the query helps improve F1 and recall, though the precision is decreased. This means that the number of pages returned is likely increased when using only the selected attribute values, and the reason may be the top score is relatively low compared with using all the attribute values.

Overall, the F1 value consistently increases as we selectively use more structured information. It is interesting to note that Add Selective Structures is better than Add Immediate Structure by all three measures, whereas Add All Structures has a lower recall than Add Immediate Structure.

This suggests that selectively adding the values from the most effective attributes is a more robust way of expanding the query than using all the attribute values. Intuitively, this also makes sense, as the selected attribute, which is the conference name, is indeed a good discriminator for identifying pages about the target researcher; other fields, such as co-authors and titles may introduce noise.

4.3.2 *Effectiveness of Attribute Selection*

Add Selective Structures uses a frequency-based scoring formula to select the best attribute for query expansion, which is shown to perform better than using all the attributes. In Figure 2, we look into the possibilities of using different attributes, and compare the results from using the co-authors, paper titles, and conference names, respectively. We see that the conference name attribute is indeed the best for query expansion.

To further analyze the effectiveness of our attribute scoring method, we plot the scores of the three attributes together with their corresponding F1 values in Figure 3. We see that there is a positive correlation between the score of an attribute and the F1 value of using the attribute for query expansion, indicating that our attribute scoring method is effective.

5. RELATED WORK

The topic of integrating structured data and text has received much attention recently, in both the database and IR communities. Works have focused on keyword search over structured data [21, 19, 1], adding text search extensions to database query languages (e.g., XQuery) [3, 14], integrating querying database, web, and XML [17], translating relational querying into fuzzy-search IR style querying and ranking relational queries [2, 7], and also combing structure indices and inverted ones [28]. Several works use structured queries to query meta-data, and use meta-data to improve local search [22, 25].

A key commonality underlying the above works is that they integrate text and structured data at the *syntactic* level. In contrast, our work is at a more *semantic* level, focusing on gluing together disparate data fragments that belong to the same real-world entity.

Entity retrieval is related to the entity matching, *a.k.a.* tuple matching, problem, which has been studied extensively in the database and AI communities (e.g. [36, 8, 24, 39, 5, 23, 4, 33, 18, 20, 16, 29]). But the two problems differ in important aspects. Entity matching decides if two given tuples refer to the same real-world entity, whereas entity retrieval retrieves all tuples (or data fragments in general) that belong to a single entity. As such, ER is a more general problem, and can leverage existing entity matching techniques.

There is also a huge amount of work on XML retrieval where semi-structured queries are supported (e.g., the recent ACM SIGIR workshops on XML retrieval [10, 27] and the INEX workshop on evaluating XML retrieval [15]). While XML document collections also involve both text and structures, they are already "integrated" in some sense. Our problem deals explicitly with integrating a *separate* text collection with a relational database. Moreover, while most of the work on XML retrieval focuses on designing a query language that would allow a user to constrain keyword matching to certain XML tag path as well as the selection of the right information segment to retrieve, we emphasize leveraging structured information to refine a text query and improve entity retrieval performance on text.

The work [34] performs IR over documents that contain both free text and semantically enriched markup. However, it does not address the specific issue of entity retrieval. Our idea of exploiting structured data to refine a text query is related to query expansion and relevance feedback commonly used in information retrieval [31, 13, 35]. It differs from relevance feedback in that we use relevant *structured* information and exploit structures to selectively choose text fragments for query expansion, while the standard relevance feedback uses text documents to expand the query.

Our work is also related to the pseudo feedback technique in IR [11, 6, 38], which exploits likely relevant information in an unsupervised way, but our methods are applied to structured data, rather than the unstructured text documents, and we make a distinction between different attributes and further select most useful attributes. There are also some works on TREC web track task of topic distillation which involves finding relevant home pages, given a broad query (topic). This task could be similar to our work in case we would have some ambiguous topics, which is not the case in topic distillation task.

6. CONCLUSIONS AND FUTURE WORK

Many real-world applications increasingly involve both structured data and text. Hence, managing both in an efficient and integrated manner has become a critical topic that has received much attention. To date, however, little attention has been paid to semantic issues in the integration of text and data.

In this paper we introduced an issue that belongs to the above realm: ER. Given data fragments that describe various aspects of a real-world entity, find all other data fragments as well as text documents that describe that same entity. As such, ER arises in numerous information management applications. It addresses a semantic aspect of integration of text and data.

We then considered how ER over text documents can be significantly aided by the availability of structured data. We developed several solutions and empirically evaluate them on the problem of retrieving information related to researchers from Web documents, by leveraging the structured data of DBLP. The experimental results show the promise of our approach.

As a new retrieval problem, ER poses many interesting and challenging research questions: (1) What is an appropriate query language for ER? A simple combination of a SQL-like query language with an IR-style keyword query is inadequate for a user to express all known information about a target entity. (2) What is an appropriate formal retrieval framework for entity retrieval? Presumably, such a retrieval framework should include, at the minimum, a model of entities, which is missing in all current retrieval frameworks. (3) What are the best strategies and methods for ER? We have shown that leveraging context clues from different data types is clearly beneficial. But the methods we have experimented with are heuristic. It is necessary to develop a more principled ER method that can exploit useful information from *all* types of data, including both text and structured data. One attractive strategy is to perform "mutual feedback", i.e., using the most reliable relevant information from the structured data to improving ranking of text documents, and at the same time, using reliable text documents to help select the most relevant structured data fragments.

Currently we are extending the methods studied in this paper to incorporate ER techniques we have recently developed over relational data. Our long-term goal is to develop a formal model and a general solution for ER. We are also developing novel techniques (e.g., those that exploit the link structure of the Web) to improve retrieval accuracy and exploring applications of ER in information integration.

Acknowledgments: We thank the anonymous reviewers for invaluable feedback on the paper.

7. REFERENCES

[1] S. Agrawal, S. Chaudhuri, and G. Das. DBXplorer: A system for keyword-based search over relational databases. In *Proc. of International Conf. on Data Engineering (ICDE)*, 2002.

[2] S. Agrawal, S. Chaudhuri, G. Das, and A. Gionis. Automated ranking of database query results. In *Proc. of Conf. on Innovative Data Systems Research (CIDR)*, 2003.

[3] S. AmerYahia, C. Botev, and J. Shanmugasundaram. A full-text search extension to XQuery. In *Proc. of WWW Conf.*, 2004.

[4] R. Ananthakrishna, S. Chaudhuri, and V. Ganti. Eliminating fuzzy duplicates in data warehouses. In *Proc. of Int. Conf. on Very Large Databases (VLDB)*, 2002.

[5] M. Bilenko and R. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *Proc. of the ACM SIGKDD Conf. on Knowledge Discovery and Data Mining (KDD)*, 2003.

[6] C. Buckley. Automatic query expansion using SMART: Trec-3. In D. Harman, editor, *Overview of the Third Text Retrieval Conference (TREC-3)*, pages 69–80, 1995. NIST Special Publication 500-225.

[7] S. Chaudhuri, G. Das, V. Hristidis, and G. Weikum. Probabilistic ranking of database query results. In *Proc. of Conf. on Very Large Databases (VLDB)*, 2004.

[8] W. Cohen. Integration of heterogeneous databases without common domains using queries based on textual similarity. In *Proc. of then ACM Conf. of Special Interest Group on Management of Data (SIGMOD)*, 1998.

[9] W. B. Croft and J. Lafferty, editors. *Language Modeling and Information Retrieval.* Kluwer Academic Publishers, 2003.

[10] A. S. D. Carmel, Y. Maarek. XML and information retrieval: a SIGIR 2000 workshop. *ACM SIGIR Forum*, 34(1):31–36, 2000.

[11] D. A. Evans and R. G. Lefferts. Design and evaluation of the CLARIT TREC-2 system. In *Proc. of the Second Text REtrieval Conference (TREC-2)*, 1994.

[12] H. Fang, R. Sinha, W. Wu, A. Doan, and C. Zhai. Entity retrieval over structured data. Technical report, University of Illinois at Urbana-Champaign, Department of Computer Science, Jan. 2004.

[13] N. Fuhr and C. Buckley. A probabilistic learning approach for document indexing. *ACM Transactions on Information Systems*, 9(3):223–248, 1991.

[14] N. Fuhr and K. Grossjohann. XIRQL: A query language for information retrieval in XML documents. In *Proc. SIGIR-2001.*

[15] N. Fuhr and M. Lalmas. Report on the INEX 2003 workshop. *ACM SIGIR Forum*, 38(1):46–51, 2004.

[16] H. Galhardas, D. Florescu, D. Shasha, and E. Simon. An extensible framework for data cleaning. In *Proc. of Int. Conf. on Data Engineering (ICDE)*, 2000.

[17] R. Goldman and J. Widom. WSQ/DSQ: A practical approach for combined querying of databases and the web. In *Proc. of the ACM Conf. of Special Interest Group on Management of Data (SIGMOD)*, 2000.

[18] L. Gravano, P. Ipeirotis, N. Koudas, and D. Srivastava. Text join for data cleansing and integration in an RDBMS. In *Proc. of Int. Conf. on Data Engineering (ICDE)*, 2003.

[19] L. Guo, F. Shao, C. Botev, and J. Shanmugasundaram. XRANK: Ranked keyword search over XML documents. In *Proc. of the ACM Conf. of Special Interest Group on Management of Data (SIGMOD), year=2003.*

[20] M. Hernandez and S. Stolfo. The merge/purge problem for large databases. In *Proc. of then ACM Conf. of Special Interest Group on Management of Data (SIGMOD)*, 1995.

[21] V. Hristidis, L. Gravano, and Y. Papakonstantinou. Efficient IR-style keyword search over relational databases. In *Proc. of the 29th Conf. on Very Large Data Bases (VLDB)*, 2003.

[22] B. Kelly. Using metadata to improve local searching. *Exploit Interactive*, 2000. http://www.exploit-lib.org/issue5/metadata/.

[23] S. Lawrence, K. Bollacker, and C. L. Giles. Autonomous citation matching. In *Proc. of the 3rd Int. Conf. on Autonomous Agents*, 1999.

[24] A. McCallum, K. Nigam, and L. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proc. 6th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 2000.

[25] J. Milstead and S. Feldman. Metadata: Cataloging by any other name. *Online Magazine*, pages 24–31, Jan/Feb 1999.

[26] P. Ogilvie and J. Callan. Experiments using the lemur toolkit. In *Proceedings of the 2001 TREC conference*, 2002.

[27] N. F. R. Baeza-Yates and Y. Maarek. Second edition of the xml and information retrieval workshop. *ACM SIGIR Forum*, 36(2), 2002.

[28] J. F. N. Raghav Kaushik, Rajasekar Krishnamurthy and R. Ramakrishnan. On the integration of structure indexes and inverted lists. In *Proceedings of ACM SIGMOD 2004*.

[29] V. Raman and J. Hellerstein. Potter's wheel: An interactive data cleaning system. In *The VLDB Journal*, pages 381–390, 2001.

[30] S. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of SIGIR'94*, pages 232–241, 1994.

[31] G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 44(4):288–297, 1990.

[32] G. Salton, C. S. Yang, and C. T. Yu. A theory of term importance in automatic text analysis. *Journal of the American Society for Information Science*, 26(1):33–44, Jan-Feb 1975.

[33] S. Sarawagi and A. Bhamidipaty. Interactive deduplication using active learning. In *Proc. of 8th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 2002.

[34] U. Shah, T. Finin, and A. Joshi. Information retrieval on the semantic web. In *Proc. of The Conf. on Information and Knowledge Management (CIKM)*, 2002.

[35] K. Sparck Jones and P. Willett, editors. *Readings in Information Retrieval*. Morgan Kaufmann Publishers, 1997.

[36] S. Tejada, C. Knoblock, and S. Minton. Learning domain-independent string transformation weights for high accuracy object identification. In *Proc. of the 8th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining*, 2002.

[37] E. Voorhees and D. Harman, editors. *Proceedings of Text REtrieval Conference (TREC1-9)*. NIST Special Publications, 2001. http://trec.nist.gov/pubs.html.

[38] J. Xu and W. Croft. Query expansion using local and global document analysis. In *Proceedings of the SIGIR'96*, pages 4–11, 1996.

[39] W. Yih and D. Roth. Probabilistic reasoning for entity and relation recognition. In *Proc. of International Conf. on Computational Linguistics (COLING)*, 2002.