Do We Mean the Same? Disambiguation of Extracted Keyword Queries for Database Search

Elena Demidova, Irina Oelze and Peter Fankhauser
L3S Research Center
Hannover, Germany
{demidova, oelze, fankhauser}@L3S.de

ABSTRACT

Users often try to accumulate information on a topic of interest from multiple information sources. In this case a user's informational need might be expressed in terms of an available relevant document, e.g. a web-page or an e-mail attachment, rather than a query. Database search engines are mostly adapted to the queries manually created by the users. In case a user's informational need is expressed in terms of a document, we need algorithms that map keyword queries automatically extracted from this document to the database content.

In this paper we analyze the impact of selected document and database statistics on the effectiveness of keyword disambiguation for manually created as well as automatically extracted keyword queries. Our evaluation is performed using a set of user queries from the AOL query log and a set of queries automatically extracted from Wikipedia articles both executed against the Internet Movie Database (IMDB). Our experimental results show that (1) knowledge of the document context is crucial in order to extract meaningful keyword queries; (2) statistics which enable effective disambiguation of user queries are not sufficient to achieve the same quality for the automatically extracted requests.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: query formulation.

General Terms

Algorithms, Performance, Design.

Keywords

Extracted keyword queries, keyword disambiguation

1. INTRODUCTION

Usability of keyword queries made them especially popular for database requests under schema- and query language uncertainty. However, lack of expressiveness in the keyword queries poses further challenges to the database search engine. Given a set of keywords, the database has only an uncertain guess about the informational need represented by the query and needs to perform a query disambiguation process to identify the suitable keyword interpretations.

Keyword search in databases has been intensively investigated over the last decade [1, 7, 8, 9, 12, 13]. The approaches range from finding a minimal Steiner tree connecting all keywords in a database graph to recent query ranking algorithms which seek structured queries with the most likely keyword interpretations [10, 18, 19, 21]. However, all these approaches focus on queries manually created by users. In case a user's informational need is expressed in terms of a relevant document, e.g. a web-page or an e-mail attachment, keyword disambiguation procedures require further adaptation. Whereas user queries typically consist of a small number of carefully selected representative keywords, keyword queries extracted from a document in an automatic way can include longer keyword sequences containing non-representative noisy keywords.

We can illustrate an informational need represented as a document along the following scenario: A technician Alice works in the customer support department of an internet sales enterprise. Daily she receives a number of e-mails with product and problem descriptions at different levels of detail. To answer a request, she reads the message and identifies representative keywords to retrieve relevant information from the enterprise database. This time-consuming procedure can require further interaction with the customer before the relevant information can be identified. Alternatively, she can use the whole e-mail content as an input into the novel enterprise search system which will automatically generate a keyword query from this document and retrieve the desired product information as well as trouble-shooting suggestions from the database.

In this paper we analyze the influence of selected document and database statistics on the effectiveness of keyword disambiguation for both manually created as well as extracted queries. We evaluate the impact of several keyword disambiguation factors on a set of user queries from the AOL query log and a set of queries extracted from the movie related Wikipedia articles both executed against the Internet Movie Database (IMDB). Our experimental results show that: (1) knowledge of the document context is crucial in order to extract meaningful keyword queries; (2) statistics which enable effective disambiguation of the user queries are not sufficient to achieve

KEYS'09, June 28, 2009, Providence, Rhode Island, USA.

the same quality for the automatically extracted queries. Therefore, further research is required to raise the quality of results for extracted queries to the level of the user queries.

The paper is organized as follows: Section 2 describes the architecture of our system; Section 3 presents the keyword extraction and query disambiguation procedures; Section 4 discusses the evaluation results; Section 5 contains related work; and finally, Section 6 provides a conclusion.

2. ARCHITECTURE

Our system architecture presented in Figure 1 includes the following components: (1) the ***Query Extraction Module*** is responsible for the generation of the keyword query from an input document; (2) the ***Query Disambiguation Module*** transforms a keyword query to a ranked list of structured queries and (3) the ***Query Execution Module*** executes structured queries resulting from disambiguation until it obtains top-k results.

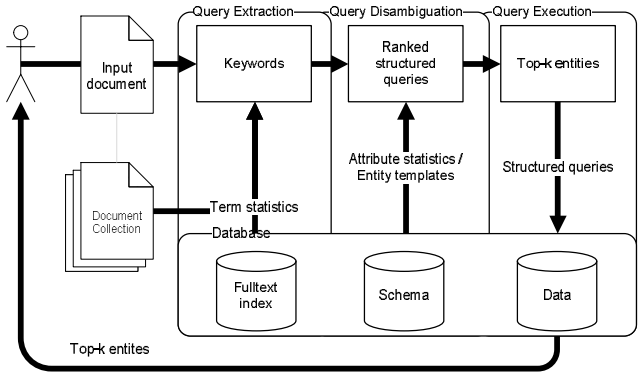

Figure 1. System Architecture

In order to disambiguate keyword queries we index the database content in an offline pre-processing step. Thereby we materialize the primary-to-foreign key joins and index the resulting entities as structured documents using an attribute-specific inverted index. This step enables us to collect precise statistics on the entities stored within the database. We describe these statistics in Section 3.2. In case the database schema is big or the database contains several logical types of entities, like e.g. movies and actors in the movie database, templates can be used to restrict the possible join paths either automatically by the path's length or manually using domain knowledge [2, 8, 9].

3. QUERY DISAMBIGUATION

In our scenario, document and database context represent two sources of statistical information, both of which can be meaningfully employed for the extraction and disambiguation of a keyword query. In this chapter we introduce selected document- and database statistics and discuss their application for keyword extraction and disambiguation.

3.1 Extraction Model

In this work we employ a simple keyword extraction model which enables us to compare the impact of the selected document and database statistics on the representativeness and usefulness of the extracted keywords in the context of database search.

In information retrieval, the importance of a keyword k with respect to an unstructured document D is typically assessed by its TF-IDF score [14]:

$$\text{TF-IDF}(k, D) = TF(k, D) * \log\left(1 + \frac{N}{DF(k, C)}\right) \quad (1)$$

where TF(k, D) is the frequency of the keyword k in the document D, N is the total number of documents in the collection and DF(k, C) is the number of documents in the collection which contain k. The TF-IDF measure requires knowledge of the original document collection. As in our scenario this context is not always available, we also look at the alternative statistics in the database.

The importance of a keyword k in the context of an attribute A can be accessed by its attribute specific selectivity (the TF-IDF score of the keyword within the context of the attribute values), as it is done in structured Information Retrieval [14]:

$$\text{ATF–IDF}(k, A) = \text{ATF}(k, A) * \log\left(1 + \frac{N}{DF(k, A)}\right) \quad (2)$$

where ATF(k, A) is the average frequency of keyword k in the values of attribute A, N is the total number of entities in the database which contain a non-zero value of A and DF(k, A) is the number of entities containing keyword k in attribute A.

In our analysis we extract the set of the most relevant keywords from a document according to the following measures of keyword relevance: (1) TF-IDF score of the keyword in the document within the original document collection; (2) an average ATF-IDF score of the keyword among all database attributes; and (3) the product of (1) and (2).

3.2 Query Model

Given a keyword query, the database system has an uncertain understanding of the informational need represented by this query. However, we can estimate the likelihoods of different structural interpretations for a keyword query using existing database statistics and query logs.

The goal of the query disambiguation is to choose the most likely structural interpretation of a keyword query. To this end we introduce a scoring function which assesses the relevance of a structural interpretation with respect to the keyword query.

3.2.1 Query Ranking

In our query model we make several assumptions with respect to the keyword query: (1) we focus on navigational queries, meaning that every keyword query targets one specific database entity; (2) we assume that each keyword is independent from the other keywords; (3) all keywords are equally weighted; and (4) each keyword targets one specific attribute.

The purpose of query disambiguation then is to select the most likely attribute for each keyword. More formally, given a keyword query $K=\{k_1, \ldots, k_n\}$, we want to determine a conjunctive Boolean query of the form:

$$Q(K) = A_1 : k_1 \text{ AND } \ldots \text{ AND } A_n : k_n \quad (3)$$

where A:k means that keyword k occurs in a value of attribute A within an entity retrieved by the query Q. Note that one possible interpretation for a keyword is a "zero" attribute, meaning that the keyword does not occur in the structured query. This reflects the fact that an extracted query can contain irrelevant keywords, which are left out in the resulting conjunctive query in order to avoid empty results.

The score of the complete query Q which includes interpretations for all keywords of K is an aggregation of the scores of partial queries A:k it contains:

$$Score(Q) = \sum_{0 \le i \le |K|} Score(A_i : k_i) \quad (4)$$

where Q is the complete query, A:k is a partial query assigning keyword k to a value of attribute A in Q, and |K| is the number of terms in the keyword query.

Due to the fact that the number of possible structural interpretations of a keyword query grows exponentially with the number of keywords in K, materialization of the whole space of the possible structural interpretations of K is infeasible. Instead, the query disambiguation module first identifies the most likely target attributes for each keyword using attribute ranking described in Section 3.2.2. Then for each keyword in the keyword query it creates partial queries of the form Q(k)=A:k assigning a keyword to a specific attribute. Finally, the module combines partial queries to build a ranked list of complete Boolean queries each containing interpretations for all keywords.

We developed a pruning algorithm, which enables us to compute a list of complete queries with the highest scores without materializing the whole query space. Owing to the space limitation we skip the details of the algorithm in this paper.

3.2.2 Attribute Ranking

In order to estimate attribute relevance with respect to a keyword we consider keyword specific database statistics as well as the general importance of an attribute in the database.

On the one hand, we can measure relevance of an attribute with respect to a keyword using selectivity of the keyword within the values of this attribute. This can be measured as the ATF-IDF score (see Formula 2). Given a keyword, ATF-IDF score gives preference to the attributes in which multiple occurrences of a keyword are contained in a small number of tuples. On the other hand, some attributes can be considered as a more important keyword target because of domain knowledge or query log information. E.g. although in the movie database the plot text and movie title attributes share vocabulary, user queries request titles more frequently than plots. In case neither domain knowledge nor a query log is available, we consider the heuristics described in the following as an approximation of the general importance of an attribute.

In fact, not all attributes in the database are used in describing entities equally frequently. In case the query log is not available, we can assume that usage of an attribute in describing entities can be employed as an estimate of the user querying behavior. In other words, the more entities have a non-zero value of an attribute, the higher is the general importance of this attribute. We call this factor collection attribute frequency (CAF):

$$CAF(A) = \frac{DF(A)}{N} \quad (5)$$

where DF(A) is the number of entities containing a non-zero value of attribute A, and N is the total number of entities in the collection.

Another keyword independent attribute relevance factor is the document attribute frequency (DAF) which is the average number of non-zero values of the attribute in an entity. This measure gives preference to single-valued attributes which appear only once in an entity description, e.g. an actor name, as opposed to set-valued attributes such as all actors of a movie. Average DAF is calculated as:

$$avgDAF(A) = \frac{1}{\log\left(1 + \frac{\sum_{entities} DAF(A)}{DF(A)}\right)} \quad (6)$$

where DAF(A) is the number of non-zero values of the attribute A in an entity and DF(A) is the total number of entities containing a non-zero value of the attribute A.

Finally, we can combine the scores introduced above to obtain combinations of keyword dependent and independent attribute relevance scores such as ATF-IDF*CAF, ATF-IDF*avgDAF as well as ARank, which is a combination of both keyword independent attribute scores introduced above:

$$ARank(A) = CAF(A) * avgDAF(A) \quad (7)$$

We consider the impact of these factors on the effectiveness of keyword disambiguation in the evaluation section.

4. EVALUATION

To assess the proposed keyword extraction and ranking factors we performed a set of experiments.

4.1 Dataset and Queries

Our experiments were performed on the IMDB dataset which has more than 10 million records and contains several tables such as movies, actors and directors. We materialized movie entities contained in the database by joining the tables following the primary-to-foreign key relations and indexed the resulting structured documents using the Lucene inverted index [6].

As the IMDB dataset does not have an associated query log, we extracted 50 user queries from the query log of the AOL Web search engine having their target URLs in the IMDB domain. The queries we selected for our experiments contain at least two attributes, such as movie-actor and movie-director, and range from 2 to 6 keywords.

Finally, we randomly selected 50 Wikipedia articles belonging to a film-related category. These articles were used later on in our keyword extraction experiments. For every query in both query sets we manually assessed the corresponding database entity. A Wikipedia article is assumed to correspond to the main database entity, e.g. movie, although other database entities, e.g. actors, can be mentioned in the article.

4.2 Effectiveness

In our effectiveness experiments we extracted a keyword query from an input document, translated this query into a ranked list of Boolean queries and executed these queries against

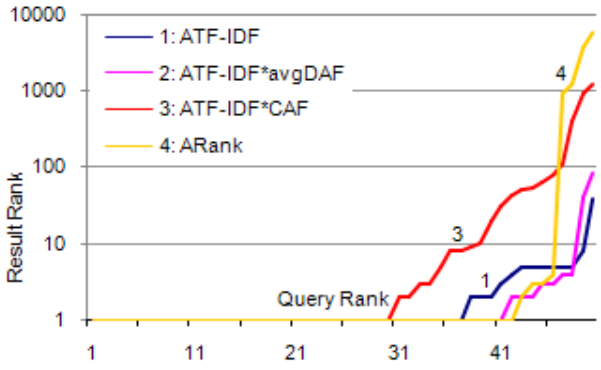

Figure 2a. User Queries

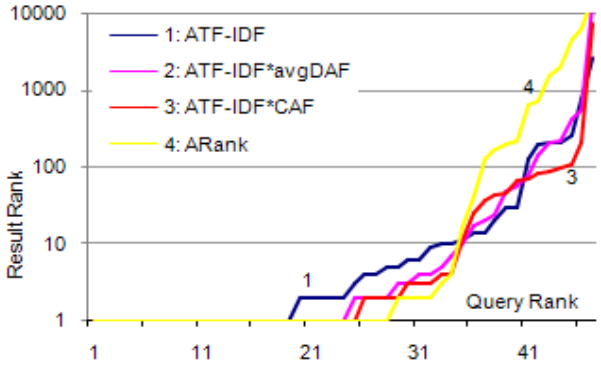

Figure 2b. Document-Based Extraction

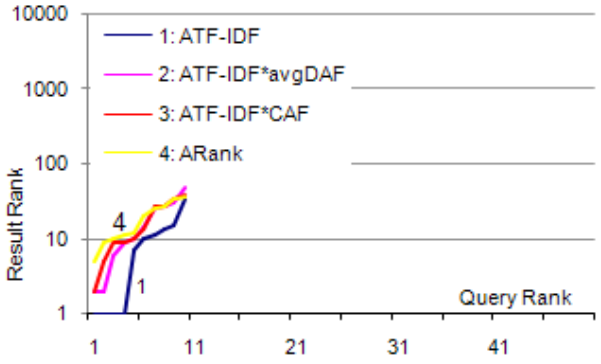

Figure 2c. Database-Based Extraction

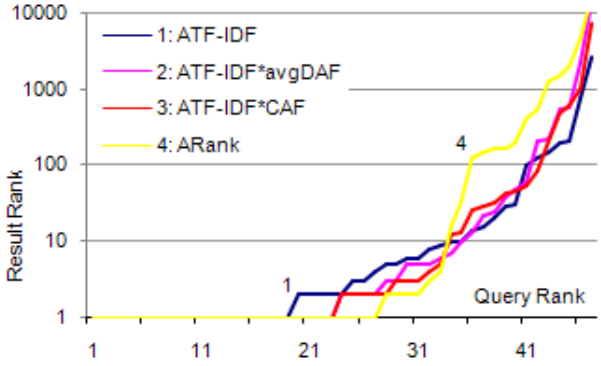

Figure 2d. Combined DB/Document-Based Extraction

the database until the intended entity was retrieved. The results of each executed query were appended to a result list. We measured effectiveness of the particular disambiguation procedure as the rank of the intended entity in the result list.

In order to extract a keyword query from a document we first assigned each term in this document a score. In our experiments we varied the score calculation for a term as follows: (1) TF-IDF weight of the term in the Wikipedia collection as described in Formula 1; (2) an average ATF-IDF weight of the term across the database attributes (see Formula 2); and (3) the product of (1) and (2). Finally, we selected a number of terms with the highest scores as a keyword query representing the document. We extracted five keywords from each Wikipedia article. We translated the user queries as well as the extracted queries into a sorted list of Boolean queries varying attribute ranking factors such as attribute specific ATF-IDF score, ATF-IDF*CAF, ATF-IDF*avgDAF and ARank. As attribute values in the IMDB database are rather short, we assumed that the ATF of a keyword in an attribute is equal to one. The Boolean queries were executed in order of their score, appending their (possibly empty) results, until the intended entity was obtained.

Figure 2 presents our experimental results. Each data point on the X-axis of Figure 2 represents a keyword query. The queries are ordered by the rank of the intended entity (log-scale Y-axis). A result rank of 1 is optimal, meaning that the intended entity was the first non-empty result. Figure 2a shows the results for the user queries, and Figures 2b to 2d show the results for the extracted queries. In Figure 2b, keyword queries are extracted using only statistics in the document collection, and then sorted as in Figure 2a using database statistics. In Figure 2c, they are extracted using only database statistics, and in Figure 2d a combination of database and document statistics is used. Each line on Figures 2a-2d corresponds to the attribute ranking procedure applied for keyword disambiguation.

As Figure 2a shows, ARank is the most effective factor which achieved the optimal result rank for the majority of the user queries (42 out of 50). User queries presented on Figure 2a were disambiguated more efficiently than the queries obtained by any of the applied extraction methods. A comparison between the keyword extraction procedures we applied (Figures 2b, 2c and 2d) shows that the best performing extraction method uses document-based statistics (Figure 2b). In case only database statistics were used in the keyword extraction (Figure 2c), the system was able to identify corresponding entities in the database only for 20% of the queries, whereas 94% of the queries extracted using document statistics were successfully answered (Figure 2b). This illustrates that knowledge of the document context in the collection is essential to extract representative keywords. Owing to this fact, we did not observe any significant improvement by combining database and document statistics (Figure 2d) compared to the document-based extraction (Figure 2b).

In the following we compare the influence of the proposed attribute ranking factors on disambiguation of user queries presented in Figure 2a and queries extracted by the best-performing extraction method (Figure 2b). As we are interested in effective identification of one specific database entity, we call "success at top-x" the situation in which the corresponding

database entity (see Section 4.1) was returned inside of the top-x entities in the result list. Table 1 presents success rate at top-1 and top-10 for the user queries and Table 2 for the queries extracted using document statistics.

Whereas ATF-IDF plays the most important role in increasing success at top-10 for both user and extracted queries, ARank is the most effective factor with respect to success at top-1. As calculation of ARank does not require any knowledge of the keyword specific statistics, this result can be obtained in an efficient manner such that it can be profitable for an application to rank the first result with respect to the ARank factor and return it to the user, while further results are calculated (e.g. based on ATF-IDF). Comparing combinations of the ATF-IDF factor with keyword independent ranking factors like ATF-IDF*CAF and ATF-IDF*avgDAF, we did not observe any significant improvement to using ATF-IDF only. The CAF factor is more important for extracted queries than for user queries.

Table 1. User Queries: Success Rate at Top-1/Top-10

	ATF-IDF	ATF-IDF*avgDAF	ATF-IDF*CAF	ARank
Top-1	0.74	0.82	0.60	**0.84**
Top-10	**0.96**	0.94	0.80	0.92

Table 2. Extracted Queries: Success Rate at Top-1/Top-10

	ATF-IDF	ATF-IDF*avgDAF	ATF-IDF*CAF	ARank
Top-1	0.38	0.48	0.50	**0.56**
Top-10	**0.70**	0.70	0.70	0.70

Although this result gives an important indication for usage of the disambiguation factors, disambiguation of user queries outperforms extracted queries by 27% on average. From this we conclude that statistics which enable effective disambiguation of user queries are not sufficient to achieve the same quality for the automatically extracted requests, such that further research is required to improve this situation.

4.3 Efficiency

Although ATF-IDF-based query ranking enables effective disambiguation of user and extracted queries, it gives preference to the highly selective structured queries. As the probability that many keywords in a query match one and the same database entity decreases, the top queries ranked based on the selectivity criteria typically deliver either an empty result, or a small number of results. This means that the system executes a number of queries which do not deliver any results until it finally finds the query which delivers the right entity. Although this fact has no effect on the result quality, it increases the total execution time, as execution of every empty query invokes a database interaction. To this end efficiency means that we wish to issue as few queries as possible with empty results before we find the query which delivers the right entity. In our experiments we reduced the total number of queries to be executed by testing existence of results of partial queries and their combinations before including them in the complete queries. The basic intuition behind this test is simple. If a partial query returns zero results, then any query with additional conditions will also return an empty result.

Figure 3a and 3b present the rank of the structured query which delivered the right entity for the user queries and queries extracted using document-based statistics in our test collection. Each datapoint of the X-axis represents a keyword query (sorted by the Y-value). The Y-axis represents the rank of the structured query, which delivered the corresponding entity. As Figure 3a shows, ARank is the most efficient factor which achieved the optimal query rank close to one for the majority of the user queries. As we can see, in this perspective the factors for the extracted and user queries correlate. On average, user queries can be answered more efficiently than the extracted queries. ATF-IDF based ranking requires the highest number of queries to be executed. ARank factor, which is not based on selectivity, requires far less database interactions. This result highlights potential advantages of keyword-independent ranking factors like ARank in efficient query answering.

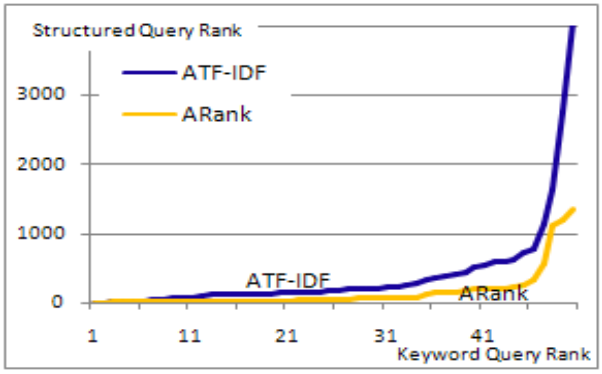

Figure 3a. Query Rank, User Queries

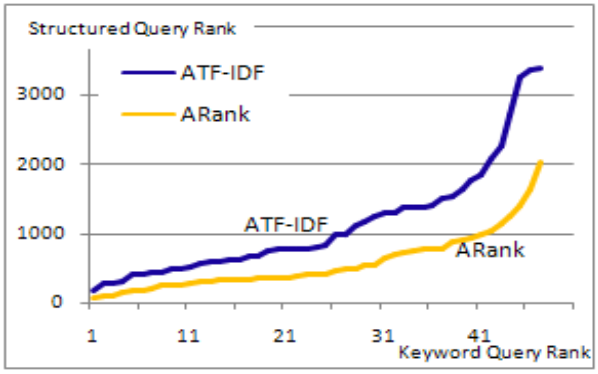

Figure 3b. Query Rank, Document-Based Extraction

5. RELATED WORK

Keyword extraction is a sub-field of Information Extraction (IE) which deals with the automatic extraction of information from unstructured sources. IE has been studied within various communities such as computational linguistics, machine learning, databases and information retrieval [17]. Existing keyword extraction methods can be divided into statistical, NLP-based, machine learning and mixed approaches. Statistical approaches focus on the non-linguistic features of the text such as term frequency, inverse document frequency, n-grams, and position of a keyword, e.g. [3, 15]. The benefits of purely

statistical methods are their ease of use and good quality of results. In this paper we build upon the existing statistical approaches and analyze effectiveness of the different sources for these statistics, such as document collection as well as the database. Dictionary-based named-entity recognition (NER) e.g. [4, 16] focuses on identifying sequences of terms within the documents as named-entities such as person name, company name, location, etc. by exploiting explicit lists (dictionaries) of single and multi-word terms and patterns. Unlike NER, we aim to identify corresponding database entities even when they are not explicitly mentioned in the document. However, we could exploit NER to preprocess the input document to increase the efficiency of the keyword extraction and disambiguation.

In recent years keyword search over structured and semi-structured data has been extensively investigated [1, 7, 8, 9]. Work evolved from retrieval and ranking of the sets of joined tuples connecting keywords, which ignored the rich semantics existing in their structures, to the state-of-the-art query ranking approaches, which exploit structured information available in the database [10, 18, 19, 21]. At the same time, interpretation of the keywords developed from considering attribute values only, to include schema terms (e.g. [12]) and selected operators [18]. SPARK [21] proposed a probabilistic model for query ranking over semantic queries. [20] investigated query disambiguation for OLAP (star) queries. All these approaches focus on user queries, which typically consist of a moderate number of representative keywords. In case of an extracted query a database search engine can face longer keyword sequences containing non-representative noisy keywords such that the existing keyword search approaches require further adaptation.

Matching information across structured and unstructured data is a sub-field of semantic integration, which deals with identifying common concepts across heterogeneous information sources. Traditionally, this work focused on integration of data across heterogeneous structured databases [5]. More recently, semantic integration within and across unstructured documents was considered [11]. Integration across structured and unstructured data was first considered in [2], which identified embeddings of the pre-defined database entities in a document. Like [2], we considered a specific case of data integration across structured and unstructured data. However, we looked at one specific entity corresponding to a Wikipedia article and compared usefulness of different information sources and statistics in identifying this entity in the database.

6. CONCLUSION

In this paper we analyzed influence of selected database and document statistics on keyword extraction and disambiguation of extracted keyword queries in database search. We conducted a number of experiments on real-world data. Our experimental results show that: (1) Knowledge of the document context is crucial in order to extract meaningful keyword queries; (2) statistics which enable effective disambiguation of user queries are not sufficient to achieve the same quality for automatically extracted requests. Therefore, further research is required to raise the quality of results for the extracted queries to the level of the user queries. Furthermore, our experiments indicate the importance of the keyword independent disambiguation factors which were not extensively studied in the previous work.

7. ACKNOWLEDGMENTS

This work has been partially supported by the FP7 EU Project OKKAM (contract no. ICT-215032).

8. REFERENCES

[1] Agrawal, S., Chaudhuri, S., and Das, G., DBXplorer: A System for Keyword-Based Search over Relational Databases. ICDE 2002.

[2] Chakaravarthy, V. T., Gupta, H., Roy, P., and Mohania, M. Efficiently linking text documents with relevant structured information. VLDB 06.

[3] Cohen. J. D., Language and domain-independent automatic indexing terms for abstracting. ASIS 1995.

[4] Cohen, W., and Sarawagi, S. Exploiting dictionaries in named entity extraction: Combining semi-markov extraction processes and data integration methods. SIGKDD 2004.

[5] Doan, A., and Halevy, A. Semantic Integration Research in the Database Community: A Brief Survey. AI Magazine 2005.

[6] Gospodnetic, O. and Hatcher, E., Lucene in Action, Manning 2005.

[7] He, H., Wang, H., Yang, J., and Yu, P.S., BLINKS: Ranked Keyword Searches on Graphs. SIGMOD 2007.

[8] Hristidis, V., Gravano, L., and Papakonstantinou, Y., Efficient IR-Style Keyword Search over Relational Databases, VLDB 2003.

[9] Hristidis, V., and Papakonstantinou, Y., DISCOVER: Keyword Search in Relational Databases, VLDB 2002.

[10] Kandogan, E., Krishnamurthy, R., Raghavan, S., Vaithyanathan, S. and Zhu, H. Avatar semantic search: a database approach to information retrieval. SIGMOD, 2006.

[11] Li, X., Morie, P., and Roth, D. Semantic Integration in Text: From Ambiguous Names to Identifiable Entities. AI Magazine 2005.

[12] Liu, F., Yu, C., Meng, W., and Chowdhury, A., Effective Keyword Search in Relational Databases, SIGMOD 2006.

[13] Luo, Y., Lin, X., Wang, W., and Zhou, X., SPARK: Top-k Keyword Query in Relational Databases. SIGMOD 2007.

[14] Manning, C. D., Raghavan, P. and Schütze, H. Introduction to Information Retrieval, Cambridge University Press. 2008.

[15] Matsuo, Y., and Ishizuka, M. Keyword extraction from a single document using word co-occurrence statistical information. International Journal on Artificial Intelligence Tools, 2004

[16] Mansuri, I., and Sarawagi, S. Integrating unstructured data into relational databases. ICDE 2006.

[17] Sarawagi, S. Information Extraction. Foundations and Trends in Databases 1(3): 261-377 (2008)

[18] Tata, S. and Lohman, G. M. SQAK: doing more with keywords. SIGMOD 2008.

[19] Tran, T., P. Cimiano, Rudolph, S., and Studer, R.: Ontology-Based Interpretation of Keywords for Semantic Search. ISWC 2007.

[20] Wu, P., Sismanis, Y., and Reinwald, B. Towards Keyword-Driven Analytical Processing. SIGMOD 2007.

[21] Zhou, Q., Wang, C., Xiong, M., Wang. H. and Yu, Y. SPARK: Adapting Keyword Query to Semantic Search. ISWC 2007.