Web Information Extraction and User Modeling: Towards Closing the Gap

Eugene Agichtein
Mathematics and Computer Science Department
Emory University
eugene@mathcs.emory.edu

Abstract

Web search engines have become the primary method of accessing information on the web. Billions of queries are submitted to major web search engines, reflecting a wide range of information needs. While significant progress has been made on improving the relevance of the results, web search process often remains a frustrating experience. At the same time, web information extraction has seen tremendous progress, such that knowledge bases of millions of facts extracted from the web are now a reality. Yet it is not clear how effectively these knowledge bases support common user information needs. We posit that a key for web information extraction to significantly impact the web search experience is to connect the extraction process with user modeling, particularly with automatic methods for inferring user information needs and anticipated interaction patterns. In this paper we overview some recent efforts for user modeling and inferring user preferences in the context of closing the gap between web information extraction and user modeling.

1 Introduction

The ultimate goal of web search is to provide answers for user information needs – where the answers may be documents, lists of items for sale, lists of structured objects, or even multi-document summaries. According to recent studies, between 39% and 60% of the queries submitted to web search engines are informational in nature [9]. The information needs behind the queries have been specifically classified into Directed, Undirected, Advice, Locate, and List categories. Of these, the Directed, Undirected, and Locate categories account for more than 56% of all queries [35]. Many of these information needs may be answered more effectively by providing structured information –by allowing precise queries, and integrating and aggregating relevant information from multiple documents.

Significant progress has been made on scalable and accurate web information extraction, allowing state-of-the-art systems to automatically extract knowledge bases of millions of facts from hundreds of millions of web pages (e.g., [17, 31]), thus bridging the gap between structured and unstructured data. These efforts have focused on accuracy and scalability of information extraction. Yet, a different gap –between user modeling to infer user information needs and the information extraction process– remains largely open. Recently, user behavior has been shown to be one of the most valuable indicators for accurate web search (e.g., [1]), and is becoming increasingly useful as more powerful and unobtrusive monitoring and analysis techniques are developed. The

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

question we explore in this paper is how user behavior –via user modeling– might be useful to guide and tune web information extraction. We describe the building blocks towards bridging the gap between web information extraction (Section 2) and user modeling (Section 3), and outline related research questions that we currently pursuing (Section 4).

2 Web Information Extraction: User Control vs. Effort

We briefly survey web information extraction from the user perspective –focusing on the experience of a casual web searcher rather than a developer or sophisticated analyst. Information extraction refers to the process of representing information in text in a structured form, and representative tasks include identifying entities, facts, events and relations extracted from the documents in either the surface web [28] or in the "deep web" [32]. We briefly review different types of extraction systems, categorized by the way the user input is solicited, and how it is incorporated into the extraction process. This is not a comprehensive review, and we only mention a few representative systems among the many influential information extraction systems reported.

Language Engineering: This approach to information extraction has stood the test of time as the resulting systems performed consistently well in DARPA-sponsored complex information extraction scenarios such as terrorist attacks, natural disasters, and corporate succession events extracted mostly from newspaper text. These systems offer rich development environments for constructing, manipulating, and testing extraction rules and lexicons for a variety of information extraction steps. With sufficient domain knowledge, resources, and expert tuning, this approach can result in highly accurate systems. A prominent example is the *PET* system [38], developed by the Proteus project at the New York University. *PET* allows a system user (typically a domain expert or developer) to create, generalize, and test extraction patterns based on the manual examination of example text documents. Another example of a language-engineering environment is *GATE* [15], a system developed at the University of Sheffield. In both systems, the actual users of the final extraction system either need to acquire the expertise with the engineering environment (an unlikely prospect), or more commonly must interact with the developers to build a system that supports the users' information needs. Typically this is an iterative process that may require weeks or months depending on the complexity of the task.

Supervised Machine Learning: A promising approach is to automatically train an information extraction system and generate rules or extraction patterns for new tasks. The training is done over a large *manually tagged* corpus, where a system can apply machine learning techniques to generate extraction patterns. Examples of such systems include CRYSTAL [20], BWI [25], and Rapier [11], and successful extraction tasks include address segmentation, entity tagging, resume field extraction, gene and protein interactions and many others. A drawback of this approach is the need for a large tagged corpus, which involves a significant amount of manual labor to create. To reduce the amount of required annotation, the AutoSlug system [33] generated extraction patterns automatically by using a training corpus of documents labeled as either relevant or irrelevant for the topic. This requires less manual labor than annotating the documents, but nevertheless the effort involved is substantial. In all cases above, a large annotated corpus was created - a daunting prospect for most casual users. In this setting, the user "interacts" with the system by defining the target entity types and relations or templates, and providing many tagged training instances. The user needs are then "frozen" in the beginning of the training process. Unfortunately, as user needs evolve, or the text corpus changes, the labeling process would have to be re-done, potentially from scratch.

Partially Supervised Machine Learning: A powerful approach to exploiting large amounts of *un-annotated* text was proposed in [14, 39], where starting with a few *seed extraction patterns*, a system can acquire and refine additional patterns. In this setting, the user may either provide patterns directly, select and possibly extend some subset of patterns in the system, or interact with the environment (e.g., *Proteus*) to build patterns. A significant improvement to the approach was introduced by the KnowItAll system [17] and subsequent variants, which starts with *general* patterns (shared by all extraction tasks) and proceeds to automatically refine and generate specific rules for particular classes of entities or relationships. A different approach was recently developed in the ODIE project [23, 36] for automatically clustering and classifying co-occurring pairs of entities given a description of the topic of interest. In this setting, the user input is limited to a description (e.g., a few keywords) for a topic, and the system attempts to discover important relations and entities. More recently, the URES system [19] was introduced that operates similarly, but focuses on the rare tuples by generating more flexible extraction patterns (e.g., with gaps). These approaches are –by design– almost completely unsupervised, or data-driven, and hence the user has minimal participation in the extraction process per se. Rather, a user may provide restrictions at query time by specifying the type(s) of entities or relations to use to process the query.

A complementary approach for extracting relations was introduced by DIPRE [8] and significantly extended in the *Snowball* system [5, 6]. *Snowball* starts with a few user-provided examples, or *seed tuples*, for the target relation. To identify new tuples, the named entities of interest must be embedded in similar contexts as the seed tuples. *Snowball* was designed to be flexible about variations inherent in natural language text and can in some cases discover a vast majority of the tuples in a collection while starting with just a few example tuples. This variant of partially-supervised extraction allows a user to focus on particular subset of tuples by manipulating the seed tuples. Nevertheless, both pattern-seeded and tuple-seeded approaches assume a "batch" mode of interaction (i.e., that the same relations will be used in the future) and is not amenable to "one-of" question answering, or on-demand extraction; Also, such systems may have to be re-tuned (albeit at much lower costs) for new tasks or with changes in the user information needs.

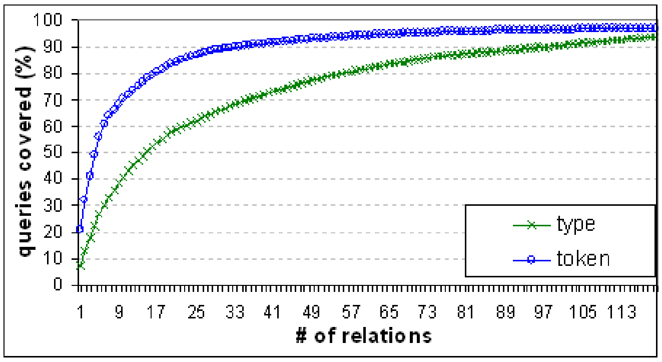
Web Question Answering: The task of returning short answers to natural language questions – where the user information need is inferred at query time is commonly referred to as question answering (QA). Many question answering systems are represented in the yearly TREC Question Answering competition (e.g., [37]). For example, Moldovan et al., and Aliod et al. [29, 3] present systems that re-rank and postprocess the results of regular information retrieval systems with the goal of returning the best passages. Cardie et al. [12] describe a system that combines statistical and linguistic knowledge for question answering and employs sophisticated linguistic filters to postprocess the retrieved documents and extract the most promising passages to answer a question. In all cases, the user (in principle) can specify any information need over supported answer types at query time. In practice, the questions usually are simple "factoid" question – that could be extracted from a short string in a single document, and thus a QA system typically cannot support aggregate queries, range queries and joins that would be possible if the information was pre-computed by an information extraction system. In order to pre-extract the necessary information to answer such queries, good understanding of the user information needs and the anticipated interactions and access patterns is required, as we discuss next.

3 User Modeling: What Do Users Want?

Understanding user information needs and query intents are crucial tasks for accurate information retrieval and web search. As information needs vary widely, user modeling research evolved largely along two paths: careful manual analysis of the queries and the information needs behind them, as well as *automatically* inferring information need through models of observable actions of varying complexity (e.g., query refinements, result interactions, and browsing patterns). Again, this section is not meant to be a comprehensive overview of user modeling (please refer to excellent books on the subject, e.g., [27]). Rather, we focus on techniques we developed recently that may be helpful for improving web information extraction performance.

3.1 Manual Analysis of User Information Needs

Among many domains that have benefited from information extraction, some of the most extensive user and query modeling and analysis has been done for factoid question answering [37], and an important variant of the task, medical question answering (e.g., [26]). Other domains include shopping, image and library search (e.g., faceted search and browsing [16]) for which distinct user models have been developed based on extensive manual analysis. In the TREC domain, fine-grained question types have been developed, such as the 140 question-type Webclopedia question taxonomy [21], with associated rules for mapping questions to answer types. As another related approach, [22] used a large number of dictionaries, or lists, some of which were constructed dynamically by querying sites such as Amazon.com based on careful analysis of previous, and anticipated, TREC QA queries.



Relation	*Type %*	*Instance %*
<person> discovers <concept>	7.7	2.9
<person> has position <concept>	5.6	4.6
<location> has location <location>	5.2	1.5
<person> known for <concept>	4.7	1.7
<event> has date <date>	4.1	0.9
<object> has discovery date <date>	3.3	1.0
<person> creates <object>	3.3	1.5
<animate> eats <object>	2.9	1.8
<event> has location <location>	2.4	1.6
<object> has alias <title>	2.3	0.7
Total coverage	41.5	18.2

Figure 1: Coverage of queries by type and token (a) and the list of the most frequently queried relation types (b) [4].

.

In the web search domain, questions do not necessarily follow the TREC model as the TREC questions are grammatical, and well-formed, whereas the web questions are often short, and vague. To better understand the user needs expressed in web search, we performed an extensive examination of more than 2,000 question-like queries sampled from the millions of queries submitted to the web-searchable Microsoft Encarta. Specifically, we focused on questions that could be answered by some binary (and possibly tertiary) relation. Interestingly, a fairly small set of relations cover a large fraction of user queries. Figure 1 (a) reports the number of relations (horizontal axes) that results in the reported coverage as a percentage of factoid questions in the sample (vertical axes). For illustration, Figure 1 (b) lists the relations identified by our annotators as most frequently needed to answer the questions in query log. From the sample, fewer than 25 relations are needed to cover more than 50% of the queries, with a very skewed frequency distribution of both relation types and relation instances. Reference [4] is a promising step towards answering factoid questions using knowledge bases extracted from trusted resources or the web. The skewed distribution of relationships observed in the annotated queries indicates that a limited number of fact tables can cover the bulk of user factoid questions. This approach could be extended to handling complex questions through decomposition into simpler factoid questions. Our results suggest focusing computational and annotation resources on extracting fact tables for frequently queried relationships, and on mapping user questions to appropriate relations. While for particularly important resources or questions types we could manually analyze query logs for re-tuning extraction, a more promising approach would be to *automatically* identify user interests, preferences, and access strategies from observable behavior data as we describe next.

3.2 Automatic Analysis of User Behavior to Infer Information Needs and Preferences

The best indicator of user intent and interest and relevance of results is explicit human feedback. Unfortunately, such feedback is expensive, and often not realistic to obtain. Hence, reducing the dependence on explicit human judgments by using implicit relevance feedback has been an active topic of research. Several research groups have evaluated the relationship between implicit measures and user interest. For example, reference [30] presented a framework for characterizing observable user behaviors using two dimensions-the underlying purpose of the observed behavior and the scope of the item being acted upon. Reference [10] studied how several implicit measures related to the interests of the user using a custom browser to gather data about implicit interest indicators and to probe for explicit judgments of Web pages visited. Reference [10] also found that indicators such as the time spent on a page, and the amount of scrolling on a page have a strong positive relationship with explicit interest.

In the context of web search, Fox et al. [18] explored the relationship between implicit and explicit measures. Reference [18] built an instrumented browser to collect data and developed Bayesian models to relate implicit measures and explicit relevance judgments for both individual queries and search sessions. They found that clickthrough was the most important individual variable but that predictive accuracy could be improved by using additional variables. Joachims et al. [24] presented an empirical evaluation of interpreting clickthrough evidence. By performing eye tracking studies and correlating predictions of their strategies with explicit ratings, the authors showed that it is possible to accurately interpret clickthrough events in a controlled, laboratory setting. More recently, Radlinski and Joachims [34] exploited session-level implicit feedback to improve ranking further.

Recently, we presented a real-world study of modeling the behavior of web search users to predict search result preferences [2]. We introduced more robust probabilistic techniques for interpreting clickthrough evidence by aggregating across users and queries, resulting in more accurate than previously published results on interpreting implicit feedback. Specifically, we focused on the *deviations* of user behavior from the "expected" behavior –estimated from the aggregated behavior patterns computed across all users and queries. Figure 2 illustrates this idea for one particular indicator, result clickthrough frequency. Figure 2(a) reports the overall clickthrough frequency, while Figure 2(b) separates the queries by the position of known top relevant result (PTR), and subtracts the "expected" clickthrough fraction at each corresponding position –thereby computing the deviation from the "expected" clickthrough at that position. For example, on average clickthrough at position 2 may be 0.55, but for queries with first *relevant* result at 2, clickthrough at position 2 is 0.65 for positive deviation of 0.1. By modeling these deviations, we can significantly increase the accuracy of clickthrough interpretation [2].

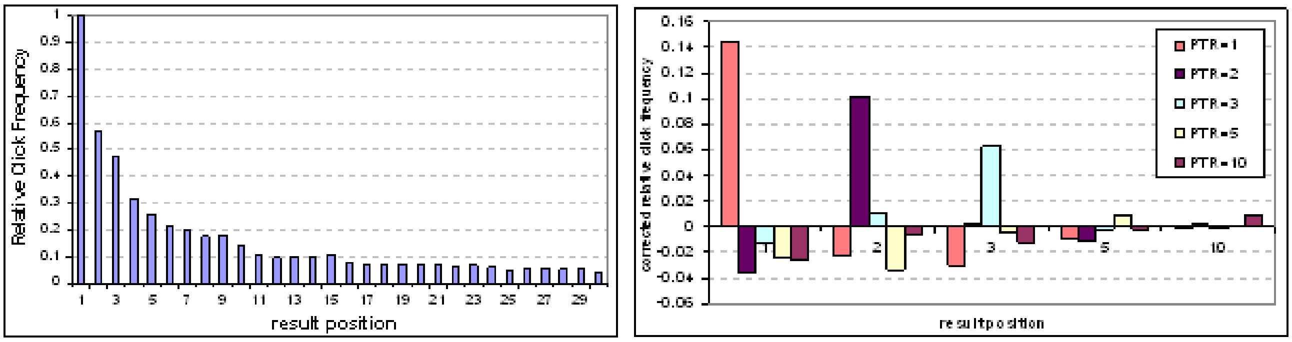


Figure 2: Overall clickthrough rate (a) and (b) Deviations from expected clickthrough rates for varying positions of top relevant result, per [2].

.

Furthermore, reference [2] introduced a general model for interpreting post-search user behavior that incorporates clickthrough, browsing, and presentation features. A sample of features, derived from server and client-side instrumentation, are shown in Figure 3(a). By considering the complete search experience after the initial query and click, we demonstrated prediction accuracy dramatically exceeding that of interpreting only the limited clickthrough information. Automatically learning to interpret user behavior results in substantially better performance than the human-designed ad-hoc clickthrough interpretation strategies, as reported in the pairwise agreement plot in Figure 3(b). The horizontal axes refers to the fraction of pairwise preferences (explicitly stated) that a system was able to predict correctly, and the vertical axes measures the fraction of preferences for which a prediction was attempted, that were predicted correctly. The full model, reported as *UserBehavior*, significantly outperforms the clickthrough Deviation model, which in turn outperforms the previous state-of-the-art clickthrough model.

Presentation Features	
ResultPosition	Position of the URL in Current ranking
QueryTitleOverlap	Fraction of query terms appearing in result Title
QueryURLOverlap	Fraction of query terms appearing in the URL
QuerySummaryOverlap	Fraction of query terms in result summary
Clickthrough and Browsing features	
TimeToClick	Seconds between query and click on the URL
TimeToFirstClick	Seconds between query and click on any result
ClickFrequency	Fraction of clicks for this query on this URL
ClickDeviation	Deviation from expected click frequency
TimeOnPage	Result page dwell time
CumulativeTimeOnPage	Cumulative time for all subsequent pages visited

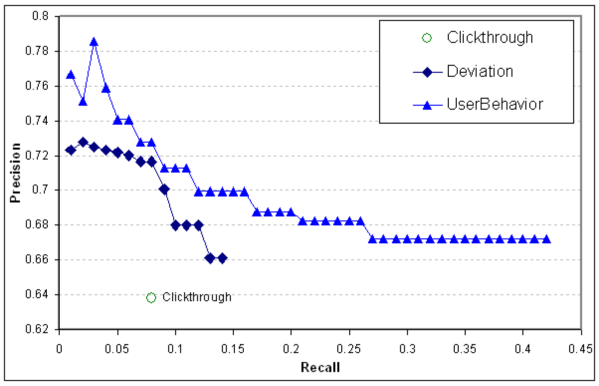


Figure 3: Sample user behavior features (a) and the accuracy of predicting user pairwise result preferences (b), per [2].

While our techniques perform well on average, our assumptions about clickthrough distributions (and learning the user behavior models) may not hold equally well for all queries. For example, queries with divergent access patterns (e.g., for ambiguous queries with multiple meanings) may result in behavior inconsistent with the model learned for all queries. Specifically, for informational queries, the user models and behavior patterns are expected to be divergent from navigational queries. In particular, we showed that we can distinguish navigational queries (and their target websites) from the rest of the queries by automatically classifying user behavior patterns [7]. The classification framework we developed is amenable to easy maintenance and updating, in particular to be tuned for evolving user behavior patterns and query distributions. A promising direction is how to extend and apply these models to identify implicitly structured queries (i.e., the queries that would benefit from information extraction) as such queries appear to also have distinct interaction patterns. This is one step towards integrating information extraction and automatic user modeling, as we discuss next.

4 Towards Integrating Web Information Extraction and User Modeling

So far we have considered some of the recent information extraction and user modeling research. If we knew the expected workload of types of relations to expect, we could more effectively optimize the resource allocation for the information extraction process accordingly. Unfortunately, a large gap remains between detailed manual analysis directly usable for extraction tuning (e.g., [4]) and the more course-grained automatic user modeling (e.g., [2]). The missing piece is the ability to automatically analyze the query and interactions stream, and to automatically identify the relations and entities that would have been helpful if pre-extracted for processing these queries. An interesting approach to improving question answering accuracy by automatically indexing the more frequent answer types was introduced recently by Chakrabarti et al. [13]. The distribution of entity types to be indexed is computed according to workload distribution of questions and answers from the TREC QA benchmark [37]. For the types of questions well represented in the TREC benchmark, this strategy is feasible. Unfortunately, web search queries exhibit different properties from the TREC questions, and hence tuning on the TREC set exclusively may not be optimal for the web search setting.

The gap between automatic user modeling and information extraction suggests a promising area of research that requires addressing multiple challenges. First, we need to identify queries amenable to answers from information extraction output. To do this, we can train a classifier (e.g., as in reference [7]) to identify queries and behavior patterns indicative of a user searching for a specific answer that could be pre-computed by information extraction. After such queries are identified, the next step is to infer the actual information need (and not just the documents likely to be relevant) – in effect to identify the types of answers appropriate for the query.

We are currently improving techniques for mining the user behavior information to improve the "resolution" of automatically identifying user information needs. Finally, the ranking of candidate answers generated from the extracted information could be improved by incorporating intelligent user modeling techniques and past user behavior.

To summarize, there is a large potential in "closing the loop" between web information extraction and automatic user modeling. However, so far the user need analysis for information extraction has been primarily manual. At the same time, automatic user modeling to infer information needs and preferences have so far focused on general web search, due to availability of both user data and explicit ratings. We described some of our recent results in user modeling –manual and automatic– that could potentially serve as first steps towards automatically tuning web information extraction systems, thus closing the gap between the information extraction process and user information needs.

ACKNOWLEDGEMENTS: The experimental results were reported in prior work ([4, 2, 7]) done at Microsoft Research in collaboration with Eric Brill, Silviu Cucerzan, Susan Dumais, and Zijian Zheng.

References

[1] Eugene Agichtein, Eric Brill, and Susan Dumais. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2006.

[2] Eugene Agichtein, Eric Brill, Susan Dumais, and Robert Ragno. Learning user interaction models for predicting web search result preferences. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2006.

[3] Diego Aliod, Jawad Berri, and Michael Hess. A real world implementation of answer extraction. In *Proceedings of the 9th International Workshop on Database and Expert Systems, Workshop: Natural Language and Information Systems*, 1998.

[4] Eugene Agichtein, Silviu Cucerzan, and Eric Brill. Analysis of factoid questions for effective relation extraction. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval Poster Session*, 2005.

[5] Eugene Agichtein and Luis Gravano. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the 5th ACM International Conference on Digital Libraries*, June 2000.

[6] Eugene Agichtein. *Extracting Relations From Large Text Collections*. PhD thesis, Columbia University, 2005.

[7] Eugene Agichtein and Zijian Zheng. Identifying "best bet" web search results by mining past user behavior. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

[8] Sergey Brin. Extracting patterns and relations from the World-Wide Web. In *Proceedings of the 1998 International Workshop on the Web and Databases (WebDB'98)*, March 1998.

[9] Andrei Broder. A taxonomy of web search. *SIGIR Forum*, 2002.

[10] Mark Claypool, David Brown, Phong Le, and Makoto Waseda. Inferring user interest. *IEEE Internet Computing*, 2001.

[11] Mary E. Califf and Raymond J. Mooney. Relational learning of pattern-match rules for information extraction. In *Sixteenth National Conference on Artificial Intelligence*, 1999.

[12] Claire Cardie, Vincent Ng, David Pierce, and Chris Buckley. Examining the role of statistical and linguistic knowledge sources in a general-knowledge question-answering system. In *Proceedings of ANLP*, 2000.

[13] Soumen Chakrabarti, Kriti Puniyani, and Sujatha Das. Optimizing scoring functions and indexes for proximity search in type-annotated corpora. In *Proceedings of the 15th International Conference on World Wide Web (WWW)*, 2006.

[14] Michael Collins and Yoram Singer. Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 1999.

[15] Hamish Cunningham. *Software Architecture for Language Engineering*. PhD thesis, University of Sheffield, 2000.

[16] Wisam Dakka, Panagiotis G. Ipeirotis, and Kenneth R. Wood. Automatic construction of multifaceted browsing interfaces. In *Proceedings of the International Conference on Knowledge Management (CIKM)*, 2005.

[17] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A. Popescu, T. Shaked, S. Soderland, D. Weld, and A. Yates. Web-scale information extraction in KnowItAll. In *Proceedings of the World Wide Web Conference*, 2004.

[18] Steve Fox, Kuldeep Karnawat, Mark Mydland, Susan Dumais, and Thomas White. Evaluating implicit measures to improve web search. *ACM Transactions on Information Systems*, 2005.

[19] Ronen Feldman and Benjamin Rosenfeld. Boosting unsupervised relation extraction by using ner. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2006.

[20] David Fisher, Stephen Soderland, Joseph McCarthy, Fangfang Feng, and Wendy Lehnert. Description of the UMass systems as used for MUC-6. In *Proceedings of the 6th Message Understanding Conference*, 1995.

[21] H. Hovy, U. Hermjakob, and D. Ravichandran. A question/answer typology with surface text patterns. In *Proceedings of HLT*, 2001.

[22] Wesley Hildebrandt, Boris Katz, and Jimmy Lin. Answering definition questions with multiple knowledge sources. In *Proceedings of HLT/NAACL 2004*, 2004.

[23] Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. Discovering relations among named entities from large corpora. In *Proceedings of the Annual Meeting of Association of Computational Linguistics (ACL)*, 2004.

[24] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2005.

[25] Nickolas Kushmerick, Daniel S. Weld, and Robert B. Doorenbos. Wrapper induction for information extraction. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 1997.

[26] Jimmy Lin and Dina Demner-Fushman. The role of knowledge in conceptual retrieval: A study in the domain of clinical medicine. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2006.

[27] Gary Marchionini. *Information Seeking in Electronic Environments*. Cambridge University Press, 1997.

[28] Andrew McCallum. Information extraction: Distilling structured data from unstructured text. *ACM Queue*, 3, 2005.

[29] Dan Moldovan, Sanda Harabagiu, Marius Pasca, Rada Mihalcea, Richard Goodrum, Roxana Girju, and Vasile Rus. Lasso: A tool for surfing the answer net. In *Proceedings of TREC-8*, 1999.

[30] D. Oard and J. Kim. Modeling information content using observable behavior. In *Proceedings of the 64 Annual Meeting of the American Society for Information Science and Technology*, 2001.

[31] Marius Pasca, Dekang Lin, Jeffrey Bigham, Andrei Lifchits, and Alpa Jain. Organizing and searching the world wide web of facts - step one: The one-million fact extraction challenge. In *Proceedings of the 21st National Conference on Artificial Intelligence*, 2006.

[32] Sriram Raghavan and Héctor García-Molina. Crawling the hidden web. In *Proceedings of the Conference on Very Large Databases*, 2001.

[33] Ellen Riloff. Automatically generating extraction patterns from untagged text. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 1996.

[34] Filip Radlinski and Thorsten Joachims. Query chains: learning to rank from implicit feedback. In *Proceeding of the 11th ACM SIGKDD international Conference on Knowledge Discovery in Data Mining*, 2005.

[35] Daniel E. Rose and Danny Levinson. Understanding user goals in web search. In *Proceedings of the International World Wide Web Conference (WWW)*, 2004.

[36] Satoshi Sekine. On-demand information extraction. In *Proceedings of the COLING/ACL 2006 Poster Session*, 2006.

[37] Ellen M. Voorhees. Overview of the TREC 2003 question answering track. In *Proceedings of the Text REtrieval Conference*, 2004.

[38] Roman Yangarber and Ralph Grishman. NYU: Description of the Proteus/PET system as used for MUC-7. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, 1998.

[39] Roman Yangarber, Ralph Grishman, Pasi Tapanainen, and Silja Huttunen. Unsupervised discovery of scenario-level patterns for information extraction. In *Proceedings of Conference on Applied Natural Language Processing (ANLP-NAACL)*, 2000.