X-ENS: Semantic Enrichment of Web Search Results at Real-Time

Pavlos Fafalios and Yannis Tzitzikas

Institute of Computer Science, FORTH-ICS, GREECE, and
Computer Science Department, University of Crete, GREECE
{fafalios,tzitzik}@csd.uoc.gr

Abstract. While more and more semantic data are published on the Web, an important question is how typical web users can access and exploit this body of knowledge. Although, existing interaction paradigms in semantic search hide the complexity behind an easy-to-use interface, they have not managed to cover common search needs. In this paper, we present X-ENS (eXplore ENtities in Search), a web search application that enhances the classical, keyword-based, web searching with semantic information, as a means to combine the pros of both Semantic Web standards and common Web Searching. X-ENS identifies *entities of interest* in the snippets of the top search results which can be further exploited in a faceted search-like interaction scheme, and thereby can help the user to limit the - often very large - search space to those hits that contain a particular piece of information. Moreover, X-ENS permits the exploration of the identified entities by exploiting semantic repositories.

1 Introduction

While more and more semantic data are published on the Web, the question of how typical web users can access this body of knowledge becomes of crucial importance. There is a growing amount of research on interaction paradigms that allow end users to profit from the expressive power of Semantic Web standards. These paradigms range from keyword search (e.g. [3]), faceted browsing and exploration (e.g. [1]) to natural language question answering (e.g. [4]). Although most hide the complexity behind an easy-to-use interface, regular users do not take advantage of using them. Their usefulness in comparison to common web search engines (especially for regular users) has not been proved, mainly because they aim at providing the user with (linked) data and relationships between these data, instead of providing links to web pages.

To address this issue we focus on *enriching* the classical web search engines with *Linked Open Data* (LOD), as a means to combine the pros of both semantic web standards and common web searching. X-ENS is a *web application* that identifies *at query-time* entities (i.e. LOD of a particular class, e.g. Persons, Locations, Writers, etc.) in the snippets of the top results returned by a search system and allows the user to explore *at real-time* their properties. The entities are exploited in a faceted search-like interaction scheme, allowing the user to narrow the search space to those hits that contain a particular piece of information.

Note also that *Google Knowledge Graph*[1] evidences the increasing interest on exploiting Semantic Data in Web Searching. However, *Google Knowledge Graph* does not locate entities in the search results but just presents a semantic description of what user is *maybe* searching.

The *innovation* brought by `X-ENS` consists in *exploiting at real-time semantic repositories* in web searching for a) *configuring the entities* that are interesting for the application, and b) *further exploring the properties of the identified entities*. In general, `X-ENS` bridges the gap between the responses of "non-semantic" search systems and semantic information.

A deployment of `X-ENS` is available at `http://139.91.183.72/x-ens/`. Figure 1 depicts an indicative screen dump. For a particular query, the user gets a list of results and a grouped list of discovered entities. User can easily limit the search space to those results that contain one or more entities (Fig. 1A). Moreover, he can explore at real-time the properties of an entity (Fig. 1B) or semantically analyze a particular hit (Fig. 1C). Note that this functionality would be impossible in plain web searching or in plain semantic search. The entities are used as the "glue" for automatically connecting the results with data and knowledge.

Fig. 1: An indicative screen dump of `X-ENS`.

2 System Description

2.1 The Process

We focus on a *dynamic* approach where no pre-processing of the resources has been done. Specifically, the user a) submits a *keyword query* through `X-ENS`'s web

[1] http://www.google.com/insidesearch/features/search/knowledge.html

interface, b) `X-ENS` fetches the *top results* of the same query from the underlying search system, c) it applies *entity mining* at the snippets of the results, d) *ranks* the discovered entities, e) *groups* them according to the categories they belong to, f) *ranks* the categories, and finally g) *presents* the categorized entities in a *faceted search-like* interface. At that point, the user has many (session-based) options for interacting with the search system that we analyze at Section 2.5.

2.2 Discovering Entities in the Results

We currently use GateAnnie[2] for entity mining. In our setting it takes as input a set of document snippets, specifically those of the top-L hits of the query answer, and it returns as output a set of entity lists (one list for each category). We have automated the procedure of adding a new category of entities in GateAnnie. Thereby, we can easily configure the entity names that are interesting for the application at hand (e.g. LOD of a particular type). As we will see later (Section 2.7), for defining the *entities of interest* we can exploit any semantic repository that is accessible via a SPARQL endpoint, or we can load our own lists.

2.3 Ranking of Entities

The user submits a keyword query q, and let A be the set of the top-L hits (e.g. $L = 200$) returned by the underlying search system. For an $a \in A$, let $rank(a)$ be its position in the answer (the first hit has rank equal to 1, the second 2, and so on). We apply entity mining in A, get a set of entities E, and rank E according to the formula: $Score(e) = \frac{\sum_{a \in docs(e)} ((|A|+1) - rank(a))}{\frac{|A|(|A|+1)}{2}}$, where $docs(e)$ denote the elements of A in which an entity e has been identified.

We can see that the entities occurring in the top hits are promoted, i.e. we exploit the ranking of the documents. The rational behind this ranking formula is that the top hits in the ranked list probably contain more useful entities that the last hits since they are considered "better" results.

2.4 Ranking of Categories

For a category c, if $inst(c)$ denotes the entities that fall in c, we rank the categories according to the formula: $Score(c) = \sum_{e \in inst(c)} Score(e)$. We can see that the categories which contain the more highly scored entities are promoted. We also apply entity mining in the *query terms* and promote in the top positions the categories of the identified entities. For example, suppose that a user submits the query `modern Greek writers`. We would like to promote categories like Person, Location, Poet, Writer, etc.

2.5 Interaction Model

Faceted search-like exploration of the results. The results of entity mining (LOD grouped in categories) are visualized and exploited according to the *faceted exploration* interaction paradigm [5]: when the user clicks on an entity, the hits are restricted to those that contain that entity (Figure 1A). Specifically, the

[2] http://gate.ac.uk/ie/annie.html

user is able to gradually select entities from one or more categories and refine the answer set accordingly (the mechanism is session-based). If such selections belong to the same category, they have disjunctive (OR) semantics and if they belong to separate categories they have conjunctive (AND) semantics. Furthermore, the user can see only the top-10 entities in each category and by simply clicking the *"show all"* hyperlink she/he can inspect all of them.

On-click semantic exploration of the Linked Data. There are already vast amounts of structured information published according to the principles of LOD. The availability of such datasets enables not only to configure easily the entity names that are interesting for the application at hand (see Section 2.7), but also the enrichment of the identified entities with more information about them. In this way the user not only can get *useful information* about one entity without having to submit a new query, but he can also start *browsing* the entities that are linked to that entity.

Another important point is that exploiting LOD is more dynamic, affordable and feasible, than an approach that requires each search system to keep stored and maintain its own knowledge base of entities and facts. Returning to our setting, a question is which LOD dataset(s) to use. One approach is to identify and specify one or more appropriate datasets for each category of entities. For example, for entities in the category Location, the *GeoNames*[3] dataset is ideal since it offers access to millions of placenames. Furthermore, *DBpedia*[4] is appropriate for multiple categories such as Organizations, Persons, Locations, etc. Other sources that could be used include *FreeBase*[5] (for persons, places and things) and *YAGO* [6] which includes *Wikipedia*, *WordNet* and *GeoNames*. In addition *FactForge* [2] includes 8 LOD datasets (including *DBpedia*, *Freebase*, *Geonames*, *Wordnet*). *DBpedia* and *FactForge* offer access through SPARQL endpoints[6].

Running one (SPARQL) query for each entity would be a very expensive task, especially if the system has discovered a lot of entities. For this reason, we offer this service *on demand*. Specifically when the user clicks on the small icon at the right of an entity, the system at that time collects more information about that entity which are visualized in a popup window as shown in Figure 1B. As we will see later, the user is able to define the SPARQL endpoint and a SPARQL template query for each category of entities (see Section 2.7).

On-demand semantic analysis of individual hits. User is able to perform entity mining at the contents of a hit *on-demand*, by simply clicking the *"find its entities"* hyperlink (Figure 1C). In that case, the contents of the particular document are downloaded and entity mining is performed. The discovered entities are grouped according to the category they belong to. Then the user is able to continue his interaction with the system, by investigating a particular entity (inspecting where in the document the entity was found) or semantically explore its properties.

[3] http://www.geonames.org/
[4] http://dbpedia.org/
[5] http://www.freebase.com/
[6] DBpedia: http://dbpedia.org/sparql, FactForge: http://www.factforge.net/sparql

2.6 X-ENS during Plain Web Browsing

X-ENS also offers entity discovery and exploration while user is browsing on the web. Specifically, the user is able to inspect the entities of a particular web page by simply clicking a *bookmarklet*[7] and then to further retrieve more information about an entity by querying (instantly) the LOD cloud. Namely, the user can (at real-time) exploit the proposed functionality while browsing.

2.7 Configurability

We give particular emphasis on the configurability of X-ENS. The administrator of the system can specify various parameters of X-ENS through a configuration page.

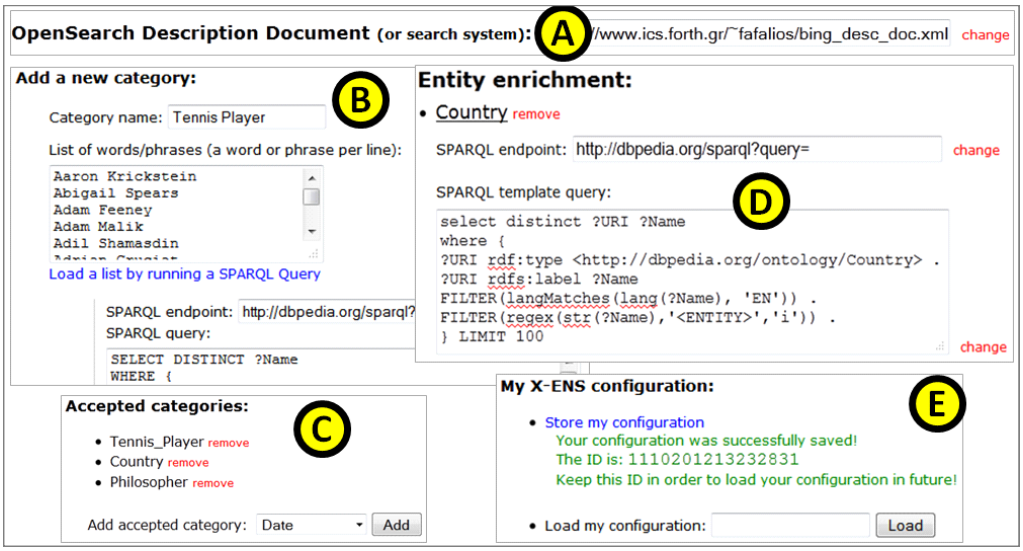

Fig. 2: Configuring X-ENS.

Specifying the underlying search system. We can define the *underlying search system* (e.g. Bing) by giving an *OpenSearch*[8] description document (Fig. 2A). *OpenSearch* is a collection of simple formats for the sharing of search results and the *OpenSearch description document* format can be used to describe a search engine. Nevertheless, we could plug any search system by creating the appropriate results parser.

Adding a new category of entities. We are able to *add a new category* of entities by giving a *category title* and a *list of words/phrases* (Fig. 2B). The list can be loaded by running a SPARQL query over a knowledge base that offers a SPARQL endpoint. For example, we can run a SPARQL query over DBpedia's SPARQL

[7] http://en.wikipedia.org/wiki/Bookmarklet
[8] http://www.opensearch.org/

endpoint that returns a list of all objects of `rdf:type dbp-ont:TennisPlayer` and thereby offer the ability to explore Tennis Players in the search results.

Specifying the desired categories. We can define the *desired categories of entities* (from the list of available) for which `X-ENS` will identify entities (Fig. 2C). For example, we may want to detect only Countries and Writers.

Specifying the underlying knowledge bases. We are able to define how to *semantically explore* an identified entity by giving a *SPARQL template query* and a *SPARQL endpoint* for each category of entities that we want to offer entity exploration (Fig. 2D). The SPARQL template query must contain the character sequence <ENTITY> (including the < and >). When a user ask for more information about an entity, we read the template query of the category in which the selected entity belongs, and we replace each occurrence of <ENTITY> with the entity's label name.

Storing the configuration. Finally, we can store the current configuration and load it in the future (Fig. 2E). In particular, by clicking the *"Store my configuration"* hyperlink, `X-ENS` returns a unique id that can be used for loading the current configuration in the future.

3 Conclusion and Lessons Learned

We described `X-ENS`, a web search application that enhances at *real-time* the classical web searching with semantic information, as a means to combine the pros of both Semantic Web standards and common Web Searching. `X-ENS` is applicable to any search system that offers textual results, it does not require any pre-processing and it does not use any caching scheme. It exploits at *real-time* semantic repositories (the LOD cloud) for both configuring the entities of interest and further exploring their properties. The entities are used as the "glue" for automatically connecting the documents with data and knowledge. This approach does not require deciding or designing an integrated schema/view, nor mappings between concepts as in knowledge bases, or mappings in the form of queries as in the case of databases.

The proposed functionality a) gives the user an overview of the answer space, b) allows the user to restrict his focus on the part of the answer where a particular entity has been identified, c) is convenient for the user needs/tasks that require collecting entities, and d) allows the user to get useful information about one entity without having to submit new queries.

Lessons learned (and issues for further research). Interacting with the LOD cloud implies dealing with a heterogeneous, distributed and very large set of highly interconnected data. The average response times of `X-ENS` services highly depend on the underlying search system, the knowledge bases that we exploit (SPARQL endpoints) and the SPARQL queries that we run. Furthermore, the quality of the results, i.e. the quality of the presented entities, highly depends on the quality of the snippets; "rich" snippets that better reflect the relation between the query and the document can result in more and better entities. Moreover, *entity disambiguation* is a problem that also affects the quality

of the presented entities and an important issue that worths further research. Ambiguity in an entity name can arise from variations in how an entity may be referenced, e.g. *IBM* and *International Business Machines*, or from the existence of several entities with the same name, e.g. *Argentine (the country)* and *Argentine (the fish)*. Finally, a better understanding of the query context and of the search space will probably enable `X-ENS` to generate better results.

References

1. S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. Dbpedia: A nucleus for a web of open data. *The Semantic Web*, pages 722–735, 2007.
2. B. Bishop, A. Kiryakov, D. Ognyanov, I. Peikov, Z. Tashev, and R. Velkov. Factforge: A fast track to the web of data. *Semantic Web*, 2(2):157–166, 2011.
3. A. Hogan, A. Harth, J. Umbrich, S. Kinsella, A. Polleres, and S. Decker. Searching and browsing linked data with SWSE: the semantic web search engine. *Web Semantics: Science, Services and Agents on the World Wide Web*, 2011.
4. V. Lopez, M. Pasin, and E. Motta. Aqualog: An ontology-portable question answering system for the semantic web. *The Semantic Web: Research and Applications*, pages 135–166, 2005.
5. G. Sacco and Y. Tzitzikas. *Dynamic taxonomies and faceted search*, volume 25. Springer, 2009.
6. F. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *Procs of the 16th World Wide Web conf.*, pages 697–706, 2007.

Appendix: Meeting the Challenge Requirements

Minimal requirements:

- The application has to be an end-user application. ⇒ `X-ENS` *is an end-user application and targets anyone that wants to search the web.*

- The information sources used should be under diverse ownership or control. ⇒ `X-ENS` *is applicable to any search system that offer textual snippets and can exploit any semantic repository.*

- The information sources used should be heterogeneous. ⇒ `X-ENS` *analyzes the unstructured contents of the search results and is able to discover and explore entities of any type.*

- The information sources used should contain substantial quantities of real world data. ⇒ `X-ENS` *is independent of the size of the underlying sources. It acts as a meta-search service over any search system or semantic repository.*

- The meaning of data has to play a central role. ⇒ *Entity names are used as the "glue" for automatically connecting documents with data and knowledge.*

- Meaning must be represented using Semantic Web technologies. ⇒ *The meaning of data lies in the semantic knowledge bases that are exploited. The meaning of an entity (and its properties) can be explored by running SPARQL queries (on the back-end).*

- Data must be manipulated/processed in interesting ways to derive useful information. ⇒ X-ENS *applies entity mining in the snippets of the top results in order to offer real-time exploration of the discovered entities.*
- This semantic information processing has to play a central role in achieving things that alternative technologies cannot do as well, or at all. ⇒ X-ENS*'s features are available thanks to the semantic information processing (i.e. entity mining) that is applied in the snippets of the top results.*

Additional Desirable Features:

- The application provides an attractive and functional Web interface. ⇒ X-ENS *is accessible through a simple, user-friendly and attractive interface.*
- The application should be scalable. Ideally, the application should use all data that is currently published on the Semantic Web. ⇒ X-ENS *can be used with any web search system and any knowledge base that lies in the LOD cloud.*
- Novelty, in applying semantic technology to a domain or task that have not been considered before. ⇒ X-ENS *is the first web meta-search application that exploits at real-time semantic repositories for both specifying the entities that are interesting for the application and further exploring their properties.*
- Functionality is different from or goes beyond pure information retrieval. ⇒ X-ENS *applies semantic analysis on the results of an information retrieval task and offers further entity exploration.*
- The application has clear commercial potential and/or large existing user base. ⇒ X-ENS *is currently a prototype but we intent to continue its development and investigate the possibility to apply* X-ENS *over vertical and professional search systems.*
- Contextual information is used for ratings or rankings. ⇒ X-ENS *gives higher rank to the categories of the entities that were discovered in the query (after having applied entity mining in the query terms).*
- Multimedia documents are used in some way. ⇒ X-ENS *currently shows images and maps in the popup window during entity exploration.*
- There is a use of dynamic data, perhaps in combination with static information. ⇒ X-ENS *applies entity mining and entity exploration at real-time without any pre-processing or any caching scheme.*
- The results should be as accurate as possible (e.g. use a ranking of results according to context). ⇒ X-ENS *exploits the ranking of the results in order to promote the entities (and the categories) discovered in the top results.*
- There is support for multiple languages and accessibility on a range of devices. ⇒ X-ENS *can be applied over any search system of any language, and the SPARQL queries can retrieve information in any language (since the LOD cloud contains information in many languages). Moreover,* X-ENS *is a Web application, so it is accessible by any device that has internet access (e.g. smart phones, tablets, etc.).*