Web Searching with Entity Mining at Query Time

Pavlos Fafalios[1], Ioannis Kitsos[1], Yannis Marketakis[1],
Claudio Baldassarre[2], Michail Salampasis[3], and Yannis Tzitzikas[1]

[1]Institute of Computer Science, FORTH-ICS, and
Computer Science Department, University of Crete, GREECE
[2]Food and Agriculture Organization of the United Nations
[3] Vienna Univ. of Technology, Institute of Software Technology and Interactive Systems
Emails: {fafalios, kitsos, marketak, tzitzik}@ics.forth.gr,
Claudio.Baldassarre@fao.org, salampasis@ifs.tuwien.ac.at

Abstract. In this paper we present a method to enrich the classical web searching with *entity mining* that is performed at *query time*. The results of entity mining (entities grouped in categories) can complement the query answers with useful for the user information which can be further exploited in a *faceted* search-like interaction scheme. We show that the application of entity mining over the *snippets* of the top-hits of the answers, can be performed at real-time. However mining over the snippets returns less entities than mining over the full contents of the hits, and for this reason we report comparative results for these two scenarios. In addition, we show how *Linked Data* can be exploited for specifying the entities of interest and for providing further information about the identified entities, implementing a kind of entity-based integration of documents and (semantic) data. Finally, we discuss the applicability of this approach on professional search, specifically for the domains of fisheries/aquaculture and patents.

1 Introduction

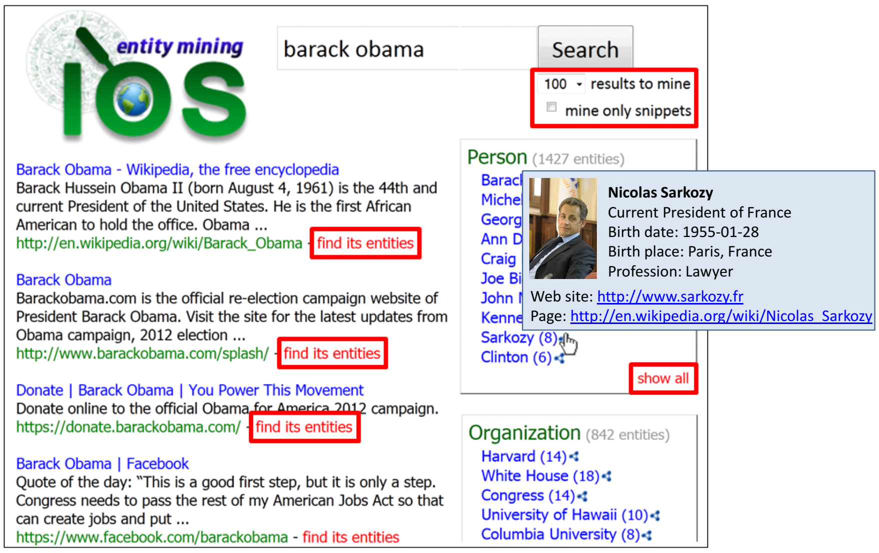
Entity search engines aim at providing the user with entities and relationships between these entities, instead of providing the user with links to web pages. Although this is an interesting line of research and there are already various entity search engines and approaches [6, 20, 8], according to our opinion, these approaches/tools are still in their infancy in the sense that they are not really useful for the usual information needs of the users. For this reason, instead of radically changing the way users search for information, we propose *enriching* the classical interaction scheme of web search engines with entity mining, as a means to combine the pros of both families of systems. For instance, showing to the user the recognized categories and entities (e.g. at a left bar of the user interface), can be useful for the user in various contexts and for various information seeking tasks. For example, consider the case of a user wanting to find persons who work on a research topic, places related to one particular painter, telephones of restaurants on a particular area, etc. If such entities are made available to the user then he can simply collectively use these entities instead of having to open several pages and searching in each one of them. Furthermore, these entities allow the user to restrict his focus on the part of the answer where a particular entity has been identified. In addition, when the user views an entity that he already knows that is relevant to his information need, this allows him to realize that his query is appropriate for that need. Recognizing categories and entities is not only useful to public web search, but it could be particularly useful in professional search that is, search in the workplace, especially in industrial research and development [14]. For example in professional patent search, in many situations, one must look beyond keywords to find and analyse patents based on a more sophisticated understanding of the patents content and meaning [13]. Technologies such as entity identification and analysis could become a significant aid to such searches and can be seen, together with other text analysis technologies, as becoming the cutting edge of information retrieval science [2].

From an information integration point of view we could say that *entity names* are used as the “glue” for automatically connecting 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 key point is that entities can be identified in documents, data, database cells, metadata attributes and knowledge bases.

To enrich web searching with *Named Entity Mining* (for short *NEM*), we have to tackle (at least) two main challenges: (a) *Real-time Response*: Real time interaction is very important in web searching (and generally in any setting of an interactive search system), however *NEM* is in general computationally expensive in the sense that the required processing time (for extracting entities) is proportional to the size (contents) of the documents. (b) *Selection/Ranking of Entities:* We have to specify criteria that determine the selection and ranking of the (often numerous) discovered entities.

In this work we focus on *NEM* that is performed at *query time* and no preprocessing or indexing has been done. Figure 1 shows an indicative screendump of a prototype system that we have designed and developped[1]. This prototype retrieves the top-K hits (the user is able to set the value of K) of a WSE (Web Search Engine), in our case Google, and mines entities at that time either from the *snippets*, or from the *full contents* (depending on what the user wants) of the top hits of the query answer. The discovered entities are grouped according to their categories and are visualized and exploited according to the *faceted exploration* interaction paradigm [18]: when the user clicks on an entity, the hits are restricted to those that contain that entity, and so on. Furthermore, the system, after user's request, can apply mining over a desired hit and discover all entities of that hit. All these are performed at query time, without any preprocessing.

[1] Accessible through http://www.ics.forth.gr/isl/ios



Barack Obama - Wikipedia, the free encyclopedia
Barack Hussein Obama II (born August 4, 1961) is the 44th and current President of the United States. He is the first African American to hold the office. Obama ...
http://en.wikipedia.org/wiki/Barack_Obama - find its entities

Barack Obama
Barackobama.com is the official re-election campaign website of President Barack Obama. Visit the site for the latest updates from Obama campaign, 2012 election ...
http://www.barackobama.com/splash/ - find its entities

Donate | Barack Obama | You Power This Movement
Donate online to the official Obama for America 2012 campaign.
https://donate.barackobama.com/ - find its entities

Barack Obama | Facebook
Quote of the day: "This is a good first step, but it is only a step. Congress needs to pass the rest of my American Jobs Act so that can create jobs and put ...
https://www.facebook.com/barackobama - find its entities

Person (1427 entities)
Barac
Miche
Georg
Ann D
Craig
Joe Bi
John M
Kenne
Sarkozy (8)
Clinton (6)
show all

Nicolas Sarkozy
Current President of France
Birth date: 1955-01-28
Birth place: Paris, France
Profession: Lawyer
Web site: http://www.sarkozy.fr
Page: http://en.wikipedia.org/wiki/Nicolas_Sarkozy

Organization (842 entities)
Harvard (14)
White House (18)
Congress (14)
University of Hawaii (10)
Columbia University (8)

Fig. 1: A prototype offering Web Searching enriched with entity mining

In Figure 1 we can see what the user is getting after having submitted the query "Barack Obama" and selected to mine the full contents of the top-100 results. We observe entities like *Joe Biden*, *John McCain*, *Michelle Obama* and *Clinton* under the category "Person", and *White House*, *Congress*, *Columbia University* and *Harvard* under the category "Organization". Only the top-10 entities of each category are shown, but the user can see all of them by clicking on "show all". By clicking on an entity, e.g on *Joe Biden*, the search space narrows to those containing that entity, while with "find its entities" on a hit, the user can ask and get all entities of that hit. Furthermore, for each entity the user can ask the system to fetch more information from the *Linked Open Data* cloud (e.g. as we see for Nicolas Sarkozy in Figure 1).

In a nutshell, in this paper: (a) we detail a novel combination of *NEM* technologies for *enriching* the classical web (meta) searching process with entity mining performed at query time, where the mined entities are exploited for offering faceted exploration, (b) we compare the results of *NEM over document snippets* versus *NEM* over the *full document contents* according to various perspectives (mined entities, computational cost), (c) we elaborate on the *ranking of entities* and we report the results of a comparative evaluation with users, and (d) we show how to exploit the LOD (Linked Open Data) cloud for enriching the identified entities with links to their corresponding semantic descriptions.
The rest of this paper is organized as follows. Section 2 discusses the context, related works, and possible approaches for enriching web and professional search with *NEM*. Section 3 describes the approach that we investigate in this work (architecture, entity ranking, LOD-based enrichment), and a prototype over the fisheries/aquaculture domain. Section 4 reports experimental results, and finally, Section 5 concludes and identifies directions that are worth further research.

2 Background and Related Work

The idea of enriching the classical *query-and-response* process of current web search engines, with *static* and *dynamic* metadata for supporting *exploratory search* was proposed in [17] and it is described in more detail (enriched with the results of a user-based evaluation) in [16]. In that work the notion of *dynamic metadata* refers to the outcome of *results clustering* algorithms which take as input the *snippets* of hits, where snippets are *query word dependent* (and thus they cannot a-priori be extracted, stored and indexed). Note that the results of *NEM* if applied over the textual snippets also falls into the case of *dynamic metadata*.

We can model our setting as follows. Let D be the set of all documents, and C the set of all supported categories, e.g. $C = \{Locations, Persons, Organizations, Events\}$. Let M be the set of all minable entities where each entity is described by a string. We can now model the functionality of a *NEM* tool by two functions mc and mec. Specifically, let $mc : D \to 2^C$ be the function that takes as input a document (say $d1$) and returns the categories of the entities that have been recognized in $d1$, e.g. $mc(d1) = \{Locations, Persons\}$. Now let $mec : D \times C \to 2^M$ be the function that takes as input a document d and a category c and returns entities belonging to that category which have been recognized in that document. For example, $mec(d1, Location) = \{Crete, Athens\}$.

There are several approaches that could be used in order to enrich the classical web searching with *NEM*. Some of them are described below.

$\mathcal{RS}$**: Real-time *NEM*** over the **Snippets** of the **top hits** of the answer. Here entity mining is performed only over the *snippets* of the top hits of the returned answer.

$\mathcal{RC}$**: Real-time *NEM*** over the **Contents** of the **top hits** of the answer. Here the full contents of the top hits of the returned answer are downloaded and then entity mining is performed. Clearly, this process can take much more time than $\mathcal{RS}$.

$\mathcal{OC}$**: Off-line *NEM*** over the **entire Corpus**.
Here we mine all entities of the corpus *offline* (assuming that the corpus is available), and we build an appropriate index (or database) for using it at run time. For each incoming query, the entities of the top-K (e.g. $K = 100$) hits of the answer are fetched from the index, and are given to the user. An important observation is that the size of the entity index in the worst case could be in the scale of the corpus. Also note that this approach cannot be applied in an uncooperative search environment where full access to the resources is not given.

$\mathcal{OFQ}$**: Offline *NEM*** over the **top hits** of the answers of the **Frequent Queries**. Here, also *offline*, for each frequent query of the log file (e.g. for those which are used for query suggestion), we compute its answer, we fetch the top-K hits, we apply *NEM* and save its results as they should be shown (i.e. what the left bar should show) using the approach and indexes described at [10, 9]. The benefit of this approach in comparison to $\mathcal{OC}$ is that we do not have to apply *NEM* at the entire collection but only at the top hits of the most frequent queries. This significantly reduces the required computational effort and storage space. The downside of this approach is that if a user submits a query which does not belong to the frequent queries, and thus it has not been processed, then the system cannot provide and suggest entities. In that case the system could offer the choice to the user to apply *NEM* at query time, i.e. approach $\mathcal{RS}$ or $\mathcal{RC}$ as described earlier. Finally, we should note that this approach is applicable also at a meta search level since it does not require access to the entire corpus (only to the query answers), but periodically the index has to be refreshed (mainly incrementally).

There is a plethora of related works and systems that offer a kind of *entity search*, below we briefly describe few of them.

The *Entity Search Engine* [6, 5, 4] supports only two categories (phone and email) and users have to type formatted queries (using # to denote entities). *NEM* is applied over the entire corpus and the extracted entities are stored in the form of ordered lists based on document ID (much like storing inverted indices), in order to provide their results instantly.

EntityCube[2] is an entity search engine by Mircosoft which extracts entities and relationships from semistructured as well as natural-language Web sources. The goal is to automatically construct and maintain a knowledge base of facts about named entities, their semantic classes, and their mutual relations as well as temporal contexts.

MediaFaces [20, 21] provides faceted exploration of media collections and offers a *machine learned* ranking of entity facets based on user click feedback and features extracted from three different ranking sources. For a given entity of interest, they have collected (from knowledge bases like Wikipedia and GeoPlanet) a large pool of related candidate facets (actually related entities).

The approach described at [8] aims at identifying related entities by analyzing a user query (that describes one entity in a TREC-like manner) and then generating and sending (to various search engines) an enriched query, and finally analyzing the (full contents) of the returned results.

With respect to the approaches $\mathcal{RS}$, $\mathcal{RC}$, $\mathcal{OC}$ and $\mathcal{OFQ}$ described earlier, most systems follow approach $\mathcal{OC}$. [8] follows the $\mathcal{RC}$ approach, while the only system that offers $\mathcal{OFQ}$ is [9]. To the best of our knowledge the current paper is the first that investigates the $\mathcal{RS}$ approach.

3 *NEM* at Query Time

We focus on a *dynamic* approach where no pre-processing of the resources has been done. Analogously to works like [22] which compares the outcomes of *clustering over snippets* with the outcomes of *clustering over contents*, in this work we investigate the same question but for the case of *entity mining*. Furthermore we investigate linking the identified entities with *Linked Data*.

[2] http://entitycube.research.microsoft.com/

Architecture Our prototype system IOS, supports the approaches $\mathcal{RS}$, $\mathcal{RC}$ and $\mathcal{OFQ}$. The default choice is $\mathcal{RS}$ and $\mathcal{OFQ}$, while $\mathcal{RC}$ is offered on demand (see the options at the upper right corner of Figure 1). Approach $\mathcal{OFQ}$ is offered for the frequent queries only and the reason for providing it is because: (i) it can offer instant behavior for a significant percentage of incoming queries, (ii) it is less space consuming than the $\mathcal{OC}$ approach that mines everything, and (iii) it is beneficial for the server since it reduces the number of incoming queries and the same precomputed information is exploited in several requests. We do not further analyze this case since it has been described in detail at [10] and [9]. Figure 2 shows the architecture. We currently use *GateAnnie*[3][3, 7] for *NEM*. In our case it takes as input a set of documents (or document snippets), specifically those of the top-K hits of the query answer. It returns as output a set of lists (one list for each category). In general, *GateAnnie* relies on finite state algorithms and the JAPE (regular expressions over annotations) language. It consists of various components, in our case the following are used: *Unicode Tokeniser* (for splitting text into simple tokens such as numbers, punctuation and words), *Gazetteer* (predefined lists of entity names), and *Sentence Splitter* (for segmenting text into sentences). The prototype also supports *faceted search-like* restriction of the answer, i.e. the user is able to *gradually* select entities from one or more categories and refine the answer set accordingly (the mechanism is *session*-based). So far all such selections have disjunctive (OR) semantics.

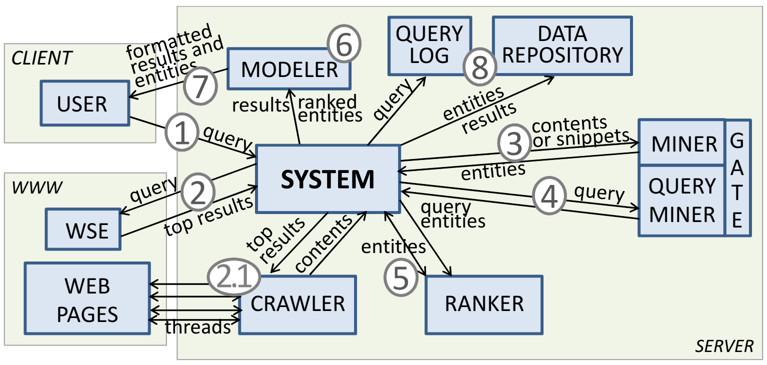


Fig. 2: The architecture (and main flow of control) of the prototype system

Entity Ranking Entity selection and ranking is important since usually the UI has limited space therefore only a few can be shown at the beginning. We propose tackling this problem by (a) ranking all identified entities for deciding the top-10 entities to be shown for each category, and (b) offer to user the ability to show more entities (all) on demand. Below we focus on ranking methods that do not rely on any log analysis, so they are aligned with the dynamic nature of our approach.

Consider a query q and let A be the set of returned hits (or the top-K hits of the answer). Now consider a category c. The entities that fall in this category

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

are $E = \cup_{a \in A} mec(a, c)$. For an $e \in E$, let $docs(e) = \{a \in A \mid e \in mec(a, c)\}$, i.e. $docs(e)$ is the set of documents in which entity e is identified.

We need a method to rank the elements of E. One approach is to count the elements of A in which the entity appears, i.e. its *frequency*. Furthermore, we can take into account the rank of the documents that contain that entity in order to promote those entities that are identified in more highly ranked documents (otherwise an entity occurring in the first two hits will receive the same score as one occurring in the last two). 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 can capture this requirement by a formula of the form:

$$Score_{rank}(e) = \sum_{a \in docs(e)} ((|A| + 1) - rank(a)) \tag{1}$$

We can see that an occurrence of e in the first hit, counts $|A|$, while an occurrence in the last document of the answer counts for 1.

Another approach is to take into account the *words* of the entity name and the query string. If for an entity $e \in E$, we denote by $w(e)$ the words of its name, and by $w(q)$ the words of the query, then we can define $Score_{name}(q, e) = \frac{|w(q) \cap w(e)|}{|w(e)|}$. To tolerate small differences (due to typos or lexical variations), we can define an alternative scoring function that is based on the *Edit Distance*:

$$Score_{nameDist}(q, e) = \frac{|\{\ a \in w(q) \mid \exists b \in w(e), EDist(a, b) \leq 2\}|}{|w(q)|} \tag{2}$$

which returns the percentage of the words of q which can be "matched" to one word of the entity e either exactly or up to an Edit distance equal to 2.

The above scores can be combined to reach a final score that considers both perspectives. We can adopt the *harmonic mean* for promoting those entities which have high scores in both perspectives. However notice that if an entity has not any query word (or a word that is close to a query word), that entity would take zero at $Score_{name}$ and that would zero also the harmonic mean. One approach to tackle this problem is to compute the plain (instead of the harmonic) mean, or in place of $Score_{nameDist}(q, e)$ to have the sum $Score_{nameDist}(q, e) + b$ for a very small positive constant b (e.g. $b = 0.01$). To conclude we could use:

$$Score(q, e) = \frac{2\ Score_{rank}(q, e)\ Score_{nameDist}(q, e)}{Score_{rank}(q, e) + Score_{nameDist}(q, e)} \tag{3}$$

3.1 On Exploiting Linked Open Data

There are already vast amounts of structured information published according to the principles of *Linked Open Data (LOD)*. The availability of such datasets enables not only to configure easily the entity names that are interesting for the application at hand, 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, the first 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 category "Location", the *GeoNames*[4] dataset is ideal since it offers access to millions of placenames. Furthermore, *DBpedia*[5] is appropriate for multiple categories such as "Organizations", "People" and "Locations". Other sources that could be used include: *FreeBase*[6] (for persons, places and things), *YAGO[19]* (for Wikipedia, WordNet and GeoNames), *Wikicompany*[7] (for organizations). In addition *FactForge*[1] includes 8 LOD datasets (including DBpedia, Freebase, Geonames, UMBEL, Wordnet). DBpedia and FactForge offer access through SPARQL endpoints[8].

Running one (SPARQL) query for each entity would be a very expensive task, especially if the system has discovered a lot of entities. Some possible ways to tackle this problem are: (a) offer this service on demand, (b) for the frequent queries pay this cost at pre-processing time and exploit the results as described in [10, 9], (c) periodically retrieve and store locally all entities of each category, so at real time only a matching process is required (however here we have increased space and maintenance requirements). Note however that approach (c) is essentially the approach of our prototype (even though no Linked Data are used), since the Gazetteers of *GateAnnie* that we use include names of persons (11,974), organizations (8,544), and locations (29,984); in total about 50,502 names are used in our setting. Furthermore, lists of prefixes and postfixes are contained that aid the identification of entities (e.g. from a phrase "Web *inventor* Tim Berners-Lee", it recognizes "Tim Berners-Lee" as a *person* due to the prefix "inventor"). So the essential difference could be the following: instead of having a *NEM* component that contains predefined named lists/rules, it is beneficial (certainly from an architectural point of view) to offer the ability to the system to download the required lists (from the constantly evolving LOD) that are appropriate for the application at hand. For example, we can run a SPARQL query that returns a list with all objects of `rdf:type dbp-ont:Artist` and thereby offer the ability to explore artists in the search results.

Currently our prototype adopts the (a) approach for the general web search scenario, and the (c) approach for vertical search scenarios. Specifically when the user clicks on the small icon at the right of an entity, the system at that time checks if that entity lies in the LOD cloud (by performing a SPARQL query) and if yes it collects more information about that entity which are visualized in a popup window as shown in Figure 1. For instance, the following query, evaluated over *FactForge* SPARQL Endpoint, returns the basic information about a

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

foaf:Person with *foaf:name 'Barack Obama'*:

```
SELECT DISTINCT
?person ?name ?comment ?birthDate ?birthPlace ?homepage ?thumbnail WHERE {
?person rdf:type foaf:Person; foaf:name ?name
FILTER(regex(str(?name),'Barack Obama','i'))
?person dbp-ont:birthDate ?birthDate .
OPTIONAL{?person rdfs:comment ?comment
          FILTER(langMatches(lang(?comment),"EN"))}
OPTIONAL{?person dbp-ont:thumbnail ?thumbnail}
OPTIONAL{?person foaf:homepage ?homepage}
OPTIONAL{?person dbp-ont:birthPlace ?place .
?place rdfs:label ?birthPlace FILTER(langMatches(lang(?birthPlace), "EN"))}
```

Case Study: Fisheries and Aquaculture publications Apart from the case of general purpose Web searching, we have started investigating this approach in vertical search scenarios. One of this is the domain of *FAO* (Food and Agriculture Organization) publications about *fisheries and aquaculture*. The underlying keyword search system is the *FIGIS search component*[9] which can receive queries through an HTTP API. The search result apart from formatted HTML can be returned in XML format which uses Dublin Core schema to encapsulate bibliographic information. Each returned hit has various textual elements, including publication title and abstract. The first is around 9 words, the second cannot go beyond 3,000 characters. As concern NEM, we identified the following relevant categories: *Countries, Water Areas, Regional Fisheries Bodies*, and *Marine Species*. For each one there is a list of entities: 240 countries, 28 water areas, 47 regional fisheries bodies and 8,277 marine species, in total 8,592 names. Each such entity is also described and mutually networked in the *Fisheries Linked Open Data (FLOD)* RDF dataset. FLOD extended network of entities is exposed via a public SPARQL endpoint[10] and web services.

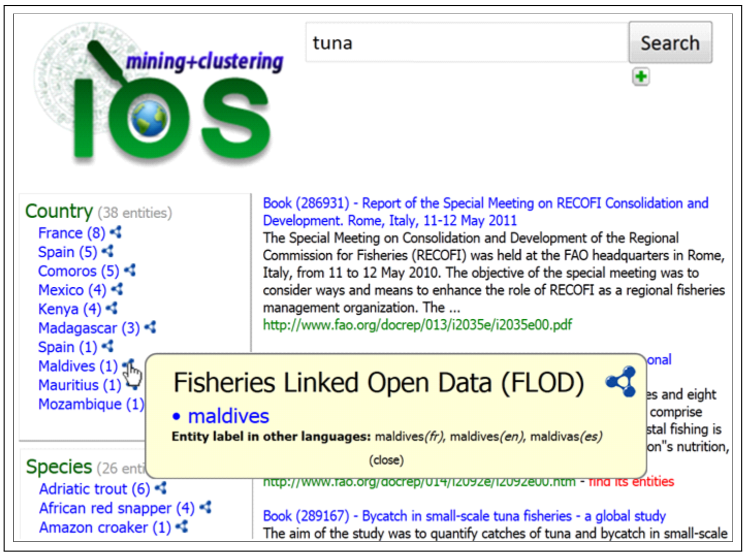
The objective is to investigate how to enrich keyword search with entity mining where the identified entities are linked to entities in FLOD endpoint, and from which semantic description can be created and served. A screendump of this prototype is shown in Figure 3. The link in the popup window redirects the users to the FLOD graph browser (a customized version of the Pubby web application[11] interfacing with FLOD endpoint).

Patent Search Entity mining at query time can also be beneficial for patent search. *Patent search* is a kind of professional search, and most patent searches (e.g. patentability and validity) are crucially important for businesses' patent management success. Missing relevant documents is unacceptable therefore the retrieval of all relevant documents is usually necessary. Clearly, this is a kind of *recall-oriented* search, and thus the support of an interactive and gradual

[9] http://www.fao.org/fishery/search/en

[10] http://www.fao.org/figis/flod/endpoint/sparql

[11] http://www4.wiwiss.fu-berlin.de/pubby/



Country (38 entities)
France (8)
Spain (5)
Comoros (5)
Mexico (4)
Kenya (4)
Madagascar (3)
Spain (1)
Maldives (1)
Mauritius (1)
Mozambique (1)

Species (26 enti
Adriatic trout (6)
African red snapper (4)
Amazon croaker (1)

Book (286931) - Report of the Special Meeting on RECOFI Consolidation and Development. Rome, Italy, 11-12 May 2011
The Special Meeting on Consolidation and Development of the Regional Commission for Fisheries (RECOFI) was held at the FAO headquarters in Rome, Italy, from 11 to 12 May 2010. The objective of the special meeting was to consider ways and means to enhance the role of RECOFI as a regional fisheries management organization. The ...
http://www.fao.org/docrep/013/i2035e/i2035e00.pdf

Fisheries Linked Open Data (FLOD)
• maldives
Entity label in other languages: maldives*(fr)*, maldives*(en)*, maldivas*(es)*
(close)

onal
es and eight
comprise
stal fishing is
on"s nutrition,
http://www.fao.org/docrep/014/i2092e/i2092e00.htm - find its entities

Book (289167) - Bycatch in small-scale tuna fisheries - a global study
The aim of the study was to quantify catches of tuna and bycatch in small-scale

Fig. 3: A prototype over FAO publications with links to FLOD

(session-based) multi-faceted search process is required. In that context the provision of facets that correspond to various kinds of entities can help the user to get an overview and to quickly restrict the search space. The usefulness of entity mining in patent search is also signified by the emergence of systems like *quantalyze*[12] in which quantities such as temperatures are spotted in the documents (patent documents in this case), their respective semantic context is identified and the quantity itself is normalized to a standard unit. However, more kinds of entities would be useful to be supported, e.g. companies, countries, persons (of related publications), product types, laws, other patents, etc. Since most of them are *named* entities the exploitation of LOD is indispensable. In this setting *NEM* could be applied not only over the textual snippets returned by a simple search, but also over the *abstracts* or *descriptions* (full contents) of the patents.

4 Experimental Results

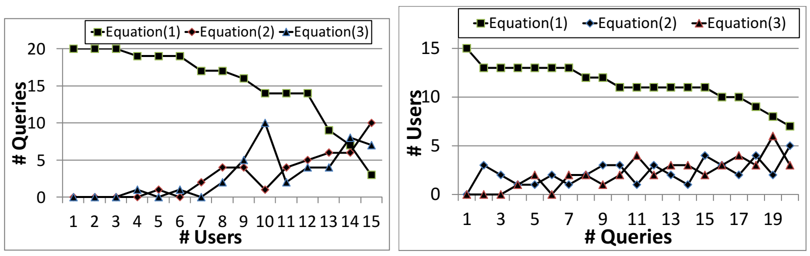
At section 4.1 we report the results of a comparative evaluation of the three *entity ranking methods* by users, while at section 4.2 we compare *snippet-mining* versus *contents-mining* from various perspectives. Finally, in section 4.3 we report the results of LOD-related experiments.

4.1 Comparative Evaluation of Entity Ranking Methods by Users

We comparatively evaluated with users the *three* methods for entity ranking, i.e. equations (1), (2) and (3) of Section 3. Fifteen users participated in the evaluation with ages ranging from 20 to 28, 73.3% males and 26.6% females. We selected a set of 20 queries and for each one we printed one page consisting of three columns, one for each ranking method. Each column was showing

[12] https://www.quantalyze.com/en/

the top-10 entities for the categories *Person, Location, Organization. NEM* over full contents was used. Each participant was asked to mark the most preferred ranking. If the user could not identify the most preferred, the user could mark two or even all three as equally preferred. We aggregated the results based on plurality ranking (by considering only the most preferred options). The results showed that the most preferred ranking is that derived by equation (1), which was the most preferred in 228 of the 15x20=300 questions. The equations (2) and (3) received almost the same preference; they were selected as most preferred options in 43 and 44 of the 15x20=300 questions.
In more detail, Figure 4a shows that for 13 of the 15 participants, equation (1) was the most preferred, while Figure 4b shows that equation (1) was the most preferred for all 20 queries.
From these we can conclude that the string similarity between the query and the entity name did not improve entity ranking in our setting.



(a) (b)

Fig. 4: Left: Aggregated preferences for each user. Right: Aggregated preferences for each query.

4.2 Contents Mining versus Snippet Mining

Since a snippet is part of a document, the entities discoverable in a snippet are *subset* of those discoverable at the full document contents. From this perspective we could say that the results of snippet mining are "sound" with respect to the results of documents mining.

To check how different they are we performed various measurements. For a set of 1,000 queries we compared the results of *snippet-mining* and *contents-mining* over the top-50 hits of the query answers. In our experiments we considered the categories *Person, Location* and *Organization*. The results are shown in Figure 5a, 5b and 5c respectively. The y-axis is in log scale and the queries are ordered in descending order with respect to the number of mined entities over their full contents.

Figure 5a shows that the average number of identified *persons* over full contents is about 527, while the average number of identified persons over snippets is about 18, meaning that content mining yields around 29 times more persons. We should also note that 50% of the queries return less than 500 entities, 43% of queries retrieve from 500 to 1000 entities and only 7% return more than 1000 entities.

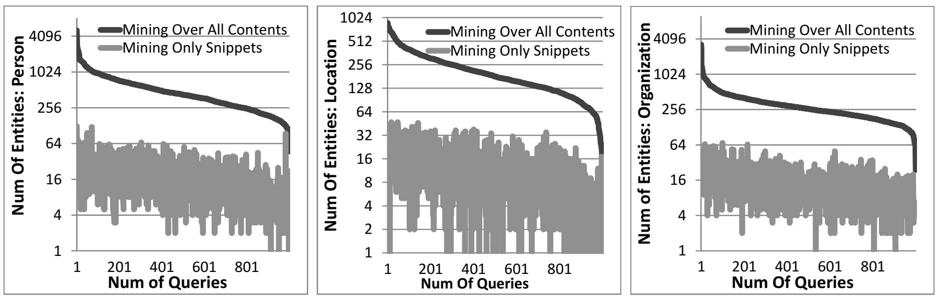


Fig. 5: Comparing the number of mined entities (for the categories *Person, Location, Organization*) over the contents and over the snippets for 1,000 queries (for each query its top-50 hits are mined).

In Fig. 5b we observe the same pattern for *locations*: contents-mining in average returns about 219 entities per query while snippet-mining about 12 entities. Finally, according to Fig. 5c, contents mining identifies on average 309 organizations while snippet-mining 14 organizations (22 times less).

To sum up, we could say that contents mining yields around 20 times more entities than snippet mining.

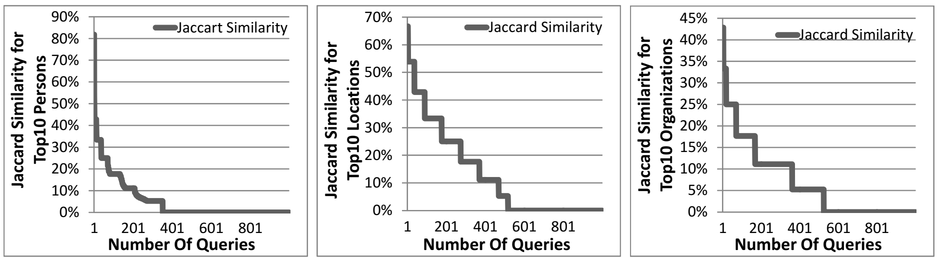


Fig. 6: Jaccard Similarity between top-10 mined entities for the categories *Person, Location, Organization* over snippets and over full contents for 1,000 queries (for each query answer the top-50 hits are considered).

In addition, we compared only the top-10 mined entities as produced by equation (1). We compared these lists as sets using the Jaccard similarity. The results are shown in the three diagrams of Figure 6. We observe that the entities in the category "Person" have Jaccard similarity 0% for the 65% of queries and more than 20% for less than the 10% of queries. For the categories "Location" and "Organization", about half of the queries have 0% similarity. Furthermore, for entities in "Organization" there are no queries with Jaccard similarity more than 50%, while for entities in "Location" there is not any query with more than 70% similarity. From the above we can conclude that the top-10 entities yielded from snippets mining are quite different from those yielded from contents mining. This is quite expectable since (as we described earlier) contents mining yields around 20 times more entities than snippets mining.

Computational and Memory Costs

Below we report execution times. All experiments were carried out using a laptop with processor Intel Core 2 Duo T8300 @ 2.40Ghz CPU, 4GB RAM and running Ubuntu 10.04 (32 bit), and Google was used as the underlying engine. The implementation of the system is in Java 1.6 (J2EE platform), using Apache Tomcat 7 (2GB of RAM).

Snippet-mining. *Time:* The whole process over the top-50 snippets (each having the size of 193 bytes in average) for one query takes 1.5 seconds in average. The process comprises the following steps. At first we retrieve the results pages from the underlying WSE which costs about 547 ms (36,2% of the total time). Second, we apply *NEM* over the snippets of the retrieved query answers which takes about 901 ms (60,4% of the total time). Third, we apply *NEM* over the query string which costs about 5,6 ms (0,37% of the total time). Finally, we create a string representation of the first page of results with cost about 36 ms (2,4% of the total time) and also a string representation of all entities in about 9 ms (0,6% of total time). The time for ranking entities and categories is negligible (less than 1 ms). In more detail, and only for the *NEM* task, some indicative times follow: 0.2 secs for 10 snippets of total size 0.1 MB, 1.2 secs for 100 snippets of total size 1.94 MB.

The average main memory requirements for one query (for the whole process) is about 37MB.

Contents-mining *Time:* The whole process over the top-50 full documents (of total size about 6.8 MB) for one query takes 78 seconds in average. The retrieval of results from the underlying WSE costs less than one second (1% of the total time). The downloading of the contents of each hit costs about 28 seconds (36% of the total time). The application of *NEM* over the contents of the downloaded documents takes about 45 seconds (57% of the total time). The creation of the string representation of the first page of results costs about 33 ms (0.04% of the total time), while the construction of the string representation of all entities takes about 4,5 seconds (6% of total time)[13]. The sorting of the categories and entities costs only a few ms. Some indicative times for *NEM* only: 5.2 seconds for 10 documents of total size 1.5 MB, 107 seconds for 100 documents of total size 16.3 MB.
The average main memory requirements for one query (the whole process) is about 300 MB.

Synopsis. The comparison between snippet and contents mining can be summarized as:

[13] Notice that this step takes 6%, while in snippets it takes only 0.6% of the total time. This is due to the much higher number of entities.

	$\mathcal{RS}$	$\mathcal{RC}$
entities per hit:	1.2	10.1
overall time:	1.5 secs	78 secs
main memory footprint for one query:	37 MB	300 MB

4.3 Linked Data-related Experimental Results

In general the time required for getting the *semantic description of an entity* by querying a SPARQL endpoint, depends on the particular query and the particular endpoint. In our case, and using the FLOD Endpoint, the average time for an entity of the category *Location* is 230 ms, while the average time for the rest categories is 450 ms.

Below we report the times required for getting the *entire list of entities* of the FAO categories mentioned earlier, by submitting one query for each category: Countries (240) in about 450 ms, Species (8,277) in about 12,000 ms, Water areas (28) in about 300 ms, and Regional Fisheries Bodies (47) in about 340 ms.

5 Concluding Remarks

In this paper we have discussed methods to enhance web searching with entity mining. Such enhancement is useful because it gives the user an overview of the answer space, it allows the user to restrict his focus on the part of the answer where a particular entity has been mined, it is convenient for user needs (or user tasks) that require collecting entities, and it can assist the user to assess whether the submitted query is the right one.

We described four main approaches for supporting this functionality and we focused on two *dynamic methods*, i.e. methods that are performed at query time and do not require any pre-processing. Since such methods have not been studied in the literature (nor supported by existing systems), we compared the application of *NEM* over textual snippets versus *NEM* over the full contents (after having downloaded them at real time) of the top hits (according to various criteria). In brief, the experimental results showed that real time *NEM* over the top snippets is feasible (requires less than 2 secs for the top-50 hits) and yields about 1.2 entities per snippet. On the other hand the approach "download and mine over the full contents" is more time consuming (requires 80 secs for the top-50), but mines much more entities (in average 10.1 per hit).

As regards *entity ranking* we comparatively evaluated three methods; one based on the frequency of the entity and the rank of the hits in which it occurs, one based on similarity with the query string, and one that combines both. The user study showed that the string similarity between the query and the entity name did not improve entity ranking in our setting. Another important point is that the top-10 entities derived from snippet mining and the top-10 entities derived from contents mining for the same queries are quite different; their Jaccard similarity is less than 30% for the majority of the queries. Therefore one issue that is worth further research is to compare the *quality* of the identified entities in snippets versus those identified in contents. Towards the same direction, it is worth investigating approaches for entity deduplication and cleaning that are appropriate for our setting.

In addition, and since there are already vast amounts of structured information published according to the principles of *Linked Open Data (LOD)*, we discussed and showed how they can be exploited for enriching the semantic descriptions of identified entities. 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.

In future we plan to apply and evaluate empirically this approach in the domains of fisheries/aquaculture and patent search. Regarding the latter, we currently develop a patent search system for the whole spectrum of patent users based on the *ezDL* system[14] (framework for interactive search applications and system for performing evaluations [12]). The plan is to integrate multiple patent data sources, patent search tools and UIs, and one of these tools will offer *NEM*.

The long term vision is to be able to mine not only correct entities but probably entire conceptual models that describe and relate the identified entities (plus other external entities) and are appropriate for the context of the user's information need. After reaching that objective the exploratory process could support the interaction paradigm of faceted search over such (crispy or fuzzy) semantic models, e.g. [11] for plain RDF/S, or [15] for the case Fuzzy RDF.

Acknowledgement Work done in the context of the *iMarine* (FP7 Research Infrastructures, 2011-2014) and *MUMIA* (COST action IC1002, 2010-2014).

References

1. 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.
2. D. Bonino, A. Ciaramella, and F. Corno. Review of the state-of-the-art in patent information and forthcoming evolutions in intelligent patent informatics. *World Patent Information*, 32(1), 2010.
3. K. Bontcheva, V. Tablan, D. Maynard, and H. Cunningham. Evolving GATE to meet new challenges in language engineering. *Nat. Lang. Eng.*, 10:349–373, 2004.
4. T. Cheng and K. Chang. Entity search engine: Towards agile best-effort information integration over the web. In *Proc. of CIDR*, pages 108–113. Citeseer, 2007.
5. T. Cheng, X. Yan, and K. Chang. Entityrank: searching entities directly and holistically. In *Procs of the 33rd intern. VLDB conf*, pages 387–398, 2007.
6. T. Cheng, X. Yan, and K. Chang. Supporting entity search: a large-scale prototype search engine. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 1144–1146. ACM, 2007.
7. H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Procs of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*, 2002.
8. B. Ernde, M. Lebel, C. Thiele, A. Hold, F. Naumann, W. Barczyn'ski, and F. Brauer. ECIR - a Lightweight Approach for Entity-centric Information Retrieval. In *Proceedings of the 18th Text REtrieval Conference (TREC 2010)*, 2010.

[14] http://www.ezdl.de/

9. P. Fafalios, I. Kitsos, and Y. Tzitzikas. Scalable, flexible and generic instant overview search. In *WWW'12 (Demo Paper), Lyon, April*, 2012.
10. P. Fafalios and Y. Tzitzikas. Exploiting avaliable memory and disk for scalable instant overview search. In *WISE '11*, October 2011.
11. S. Ferré and A. Hermann. Semantic search: reconciling expressive querying and exploratory search. *The Semantic Web–ISWC 2011*, pages 177–192, 2011.
12. N. Fuhr. An infrastructure for supporting the evaluation of interactive information retrieval. In *Procs of the 2011 workshop on Data infrastructurEs for supporting information retrieval evaluation*, DESIRE '11, NY, USA, 2011.
13. H. Joho, L. Azzopardi, and W. Vanderbauwhede. A survey of patent users: an analysis of tasks, behavior, search functionality and system requirements. In *Procs of the 3rd symposium on Information interaction in context*. ACM, 2010.
14. A. Kohn, F. Bry, and A. Manta. *Professional Search: Requirements, Prototype and Preliminary Experience Report*. 2008.
15. N. Manolis and Y. Tzitzikas. Interactive Exploration of Fuzzy RDF Knowledge Bases. *Procs of the 8th Extended Semantic Web Conference (ECSW'2011)*, 2011.
16. P. Papadakos, N. Armenatzoglou, S. Kopidaki, and Y. Tzitzikas. On exploiting static and dynamically mined metadata for exploratory web searching. *Knowledge and Information Systems*, 30:493–525, 2012. 10.1007/s10115-011-0388-2.
17. P. Papadakos, S. Kopidaki, N. Armenatzoglou, and Y. Tzitzikas. Exploratory web searching with dynamic taxonomies and results clustering. In *ECDL '09: Proceedings of the 13th European Conference on Digital Libraries*, September 2009.
18. G. Sacco and Y. Tzitzikas. *Dynamic taxonomies and faceted search: theory, practice, and experience*, volume 25. Springer-Verlag New York Inc, 2009.
19. 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.
20. R. van Zwol, L. Garcia Pueyo, M. Muralidharan, and B. Sigurbjörnsson. Machine learned ranking of entity facets. In *Procs of the 33rd intern. ACM SIGIR conf.*, pages 879–880. ACM, 2010.
21. R. van Zwol, B. Sigurbjornsson, R. Adapala, L. Garcia Pueyo, A. Katiyar, K. Kurapati, M. Muralidharan, S. Muthu, V. Murdock, P. Ng, et al. Faceted exploration of image search results. In *Procs of the 19th World Wide Web*, 2010.
22. O. Zamir and O. Etzioni. Web document clustering: A feasibility demonstration. In *Procs of SIGIR'98*, pages 46–54, Melbourne, Australia, 1998.