Using BM25F for Semantic Search

José R. Pérez-Agüera
Metadata Research Center.
SLIS. UNC
jaguera@email.unc.edu

Javier Arroyo
Computer Science School.
UCM
javier.arroyo@fdi.ucm.es

Jane Greenberg
Metadata Research Center.
SLIS. UNC
janeg@email.unc.edu

Joaquin Perez Iglesias
Computer Science School.
UNED
joaquin.perez@lsi.uned.es

Victor Fresno
Computer Science School.
UNED
vfresno@lsi.uned.es

ABSTRACT

Information Retrieval (IR) approaches for semantic web search engines have become very populars in the last years. Popularization of different IR libraries, like Lucene, that allows IR implementations almost out-of-the-box have make easier IR integration in Semantic Web search engines. However, one of the most important features of Semantic Web documents is the structure, since this structure allow us to represent semantic in a machine readable format. In this paper we analyze the specific problems of structured IR and how to adapt weighting schemas for semantic document retrieval.

Keywords

Information Retrieval, Semantic Search, BM25F

1. INTRODUCTION

Keyword-based Semantic Web search engine development has become a major research area garnering much attention in the Semantic Web community over the last seven years [10]. Research initiatives target not only simple semantic document and ontologies retrieval [12, 2] but explore the development of gateways supporting the implement Semantic Web applications [6]. Semantic search engines represent the Semantic Web Community's contribution to semantic services and are important for moving forward Semantic Web research.

Current practice with semantic web search queries frequently involve SPARQL language expressions. However, exact-match query semantics, rendered with these approaches, generally retrieve too many or too few results. The approach, therefore falls short and fails to satisfy the user's needs. One potential approach to addressing these shortcomings is to integrate the use of an IR-style ranking model based on keywords. In other words, exact-match similarity searching might be improved by integrating known statistical algorithms, although research exploring this method is extremely limited [8].

Information Retrieval (IR) plays a very important role in the design and implementation of nearly any if not all digital information systems. However, most of the times, the IR techniques used are basic, out-of-the-box and do not really improve the performance of semantic search engines. In other words the perceived needed algorithmic changes are mainly implemented to solve efficiency problems. As a result, little work has been conducted to adapt weighting schema that might improve the relevance of the results.

Current keyword-based semantic search engine work in the Semantic Web domain needs to consider how to take advantage of RDF structure. Document structure has been very useful for traditional Web Search Engines like Google or Yahoo, since the relevance of the documents for a given a query will be different depending where query terms occur. Information about document structure can be used to boost (more highly rank) terms that occur in any specific document field. For example, documents terms occurring in field *title* might be more important than terms occurring in, say, the third paragraph. These heuristics have been applied with success to optimize Web search engines [21].

The IR community has been working on structured documents from 90s [14]. In fact, this early work supports more recent developments like the initiative for Evaluation of XML retrieval (INEX) [1] competition, where different XML collections have been used to evaluate the performance of IR systems when the structure of the document is available.

The amount of RDF data has been growing steadily over the last five years thanks of Web of Data proposed by [1], and a good corpus of RDF semantic encoding is, in fact, available via the Web. At the same time, the general increase in Web of Data aggravates retrieval performance, and, in many ways, highlights a need to explore means for improving traditional web search engines. If researchers are going to advance Semantic web search, it seems not only reasonable, but imperative that they consider how RDF and encoded document structure might be used to determine weighting and assist the Information Retrieval ranking functions. The research presented in this paper considers this problem, specifically to best way to ingrate RDF encoded document structure and information retrieval ranking algorithms to improve the current state of Keyword-based Semantic Web search engine development.

The three research questions guiding this research are:

- Can semantic structure improve quality results in terms of relevance?
- What is the behavior of IR ranking functions when we use semantic information?

[1] http://www.inex.otago.ac.nz/

WWW2010, April 26-30, 2010, Raleigh, North Carolina.
.

- What features have to have a corpus for Semantic Search evaluation?

We will try to answer these questions in the next sections of this paper.

The remainder of this paper is organized as follows: Section 2 briefly reviews the status of RDF and semantic web search, and presents methods to create an RDF inverted index using indexing practices based on links. Section 3 reviews structured IR and ranking functions for structured documents; and present a method of addressing problems in Lucene weighting schema. Section 4 reviews some approaches to Semantic Web Search evaluation. In this section we present also our corpus for Semantic Search evaluation combining dbpedia + INEX collection, and, finally we review the evaluation metrics used in this paper. Section 5 presents the results and discussion, and Section 6 includes the conclusion and next steps.

2. RDF RETRIEVAL

Currently, most web search engines, like Google or Bing, have difficulties accurately retrieving semantic information represented in RDF. For example, if we try to use Google to retrieve entities from dbpedia, the results that we will obtain will be very bad in terms of relevance. One way to address this limitation and take advantage of RDF data, and thus improve retrieval performance, is to combine IR approaches with inference searching based on SPARQL. Examples is this area include Sindice [19], Falcon [2], Watson [5] or SEMPLORE model [20].

One of the main challenges for applying IR approaches to semantic search is how to represent RDF based data using an inverted index. The next section explains how the Semantic Web community has tried to solve this problem.

2.1 Modeling RDF data using an Inverted Index

The Semantic Web community has given a lot of attention to representing the RDF data model using Inverted Indexes that have been traditionally used in information retrieval for term-document indexing. A major challenge is how to adapt the semantic web data model, based on RDF triples, to an inverted index which is a data model used to index unstructured or semi-structured information. Inverted indexes are implemented using a posting list, where each term in the document collection have a pointer to the documents where it occurs. Main term-document features, like term frequency or term positions are also stored in the inverted index.

When semi-structured information is indexed an additional element must be considered to represent the structure of the document. This element is represented as a field of the document, in order to consider a different relevant value for each term depending on the field where it occurs. Field structure can be easily integrated in traditional inverted indexes, but ranking functions have to be modified in order to take into account the structure of the document for relevance computation. The possibility to represent structure for documents is not new, and, in fact, a basic document structure can be inferred from HTML documents. However, the Semantic Web supports a more sophisticated means of presenting and securing document structure, moving from descriptive markup to semantic markup. For this reason, using inverted indexes to index RDF triples, is very convenient, since RDF triples can contain non-atomic information like textual nodes.

The main problem for storing RDF triples in inverted indexes is how to represent subjects, predicates, and objects information in a $n \times m$ matrix. Several approaches to this problem have been presented in literature; and, the most successful models seem to be SIREN [7], based on XML indexing techniques, and SEMPLORE model [20], based on the idea of artificial documents.

2.2 Indexing based on links

The approach, presented in this paper, is similar to SEMPLORE model [20], although we have included some modifications in order to improve keywords based retrieval. The main innovation is the use of link-based indexing. One of the most important source of keywords for indexing in web retrieval is the text that is linked to other web page [13]. This text is not very useful to describe the document where occurs, but it is extremely useful to describe the document linked. So, if we are analyzing a page about the film *The Godfather*, and we find the text *director* linked to other page with information about *Francis Ford Coppola*, the word *director* will be a very good descriptor of the web page talking about *Francis Ford Coppola*, instead of the web page about the movie where the word actually occurs. We have applied the same idea to RDF retrieval, since usually the words that occur in the name of the predicates are useful to describe one of the nodes of the triple. As a result, we can use the aforementioned example, but using dbpedia entries.

In dbpedia we have a RDF entry for *The Godfather* [2] and another different entry for *Francis Ford Coppola* [3].The first one have a property *dbpprop:director* wich have a link to a URI [4], so the word director can be used as keyword to index the entry described by this URI [5].

The other fields that we have defined are similar to the fields used in SEMPLORE model. We have defined a *rdf:type* field, where the different types used to describe the node are included, a *text:field*, where textual information about the node is stored, and a URI field, which is used not only like a unique identifier, but it is also analyzed like a very relevant source of keywords to describe every node, we have call *title* to this field.

Finally we have included another field, called *obj*, to make reference to the relation between subjects and objects. So *Francis Ford Coppola* is also used to index *The Godfather* entry and vice/versa. Table 1 shows the fields defined in our inverted index.

3. STRUCTURED IR

For the last forty years, search engines have been dealing with flat documents, that is, without structure. The main consequence of this approach is the fact that terms within a document are considered to have the same relevance (or value), disregarding their role in the document. This assumption implies a relevance model simplification based on bag of words, and, therefore, useful information is lost. The retrieval on flat documents, also known as ad-hoc retrieval,

[2]http://dbpedia.org/page/The_Godfather
[3]http://dbpedia.org/page/Francis_Ford_Coppola
[4]http://dbpedia.org/page/Francis_Ford_Coppola
[5]http://dbpedia.org/page/Francis_Ford_Coppola

FIELD	CONTENT
text	plain text
title	keywords from the URI
obj	object
inlinks	incoming link defined by a predicate
type	rdf:type

Table 1: Fields used to represent RDF structure in the inverted index.

has been the main task in the different IR conferences like Text REtrieval Conference (TREC) [11].

With the growth of the Web and the increase of the mark-up languages, the structure of the documents has become explicit. As a result, a more complex information source is available that may, in fact, allow us to improve the performance of the information retrieval. The Semantic Web seeks to add meaning to this structure, trying to replace descriptive mark-up by semantic mark-up.

In fact, around the structured documents a new field has born within IR community named structured IR. This new field, tries to adapt the classical ranking functions and models to the new scenario by means of considering the explicit document structure.

From an IR theoretical standpoint, the main issue to solve is the adaptation of the standard ranking functions, like Vector Space Model (VSM) or BM25, to score structured documents. Currently the most important initiative for this task is the INEX initiative, which tries to define a new experimentation framework, similar to TREC, but focused on structured documents.

One of the main assumptions within IR theory is that exists an statistical independence between terms. From this assumption the standard ranking functions are composed by a term weights summation. The same independence assumption between terms has been extrapolated to independence between the different document parts, that is, fields that represent the document structure.

Starting from this assumption, ranking functions are defined by two summations, the first one at term level, and the second one at field level. This independence assumption applied to the fields of the documents make necessary the definition of a set of values which represent the document fields weights in the ranking functions. These values can be seen as a boost factor assigned to each field. They become an additional element of the ranking functions necessary to compute the final score of the documents. Optimization of these values must be done for each different collection, since the importance of a field can change between collections. For instance, the significance of a field like *title* in governmental web pages could be different than in commercial webs. Usually the estimation of these boost factors is carried out applying optimization methods and it is one of the most difficult task in structured IR [18].

For web pages, document structure can be inferred from HTML mark-up. The problem here is that structure is no evident, since HTML mark-up is used to define how documents will be rendered in the browser. The main approach is to define heuristics that help us to represent the information inferred from the HTML mark-up and that consider different relations between fields. However, this heuristics are always based in implicit information where semantic is not present. Semantic Web add meaning to this structure based on RDF elements and ontologies. This semantic information could be used in traditional IR systems in order to improve document and entity retrieval, and actually is one of the mains achievements in Semantic Search.

3.1 Ranking for structured documents

Vector Space Model (VSM) can be represented by many different functions, normally a VSM relevance score function is defined as follows:

$$score(d,q) = \sum_{t \in q} idf(t) \cdot tf(t,d),$$

where t represents the query terms for the query q, $idf(t)$ the Inverse Document Frequency of the term t in the collection and $tf(t,d)$ the document term frequency function of the term t in the document d.

If we consider the use of the document fields, we have to define a field term frequency function for the term t, and therefore:

$$tf(t,d) = \sum_{c \in d} w_c \cdot tf_c(t,d).$$

where c represents each field contained in the document d, w_c is the weight or boost factor for each field in the document and $tf_c(t,d)$ is the field term frequency function of the term t in the field c. This simple modification of the classical $tf - idf$ equation allow us to compute the score of the documents considering the structural information.

To study the viability of our approach we have implemented BM25F, which is state-of-art in structured information retrieval. BM25F is an extension of the BM25 ranking function adapted to score structured documents. Some properties about the impact of the term document frequency on document relevance are introduced with the BM25 model. From [17] it is well known that the probability of relevance of a document increases as the term frequency of a query term increases, and that this increase is non-linear. For these reasons most modern ranking functions use an increasing saturation function to weight document terms that appear in the query. An example is the term saturation function used as part of BM25 [17]:

$$\frac{tf(t,d)}{tf(t,d) + k_1}, \tag{1}$$

where $tf(t,d)$ is the term frequency function of term t in document d, and k_1 is a constant that allows us to control the non-linear growing term frequency function.

We have follow the steps proposed by Robertson in [4] to implement BM25F for our work. We first calculate a normalized term frequency for each field c:

$$tf_c(t,d) = \frac{occurs_c(t,d)}{1 + b_c(\frac{l_{d,c}}{l_c} - 1))},$$

where $occurs_c(t,d)$ are the occurrences of the term t in the field c of the document d, $l_{d,c}$ is the length of the field, and l_c is the average field length for that field. Moreover, b_c is a field-dependant parameter similar to the b parameter

in BM25. These term frequencies can be combined linearly using the boost factor w_c

$$tf(t,d) = \sum_{c \in d} w_c \cdot tf_c(t,d).$$

At this point we can use BM25 saturating, as in equation (1), to obtain final BM25F:

$$BM25F_d = \sum_{t \in q \cap d} \frac{tf(t,d)}{k_1 + tf(t,d)} \cdot idf(t)$$

$$tf(t,d) = \sum_{c \in d} w_c \cdot tf_c(t,d)$$

where c represents each field contained in the document d, w_c is the weight or boost factor for each field in the document and $tf_c(t,d)$ is the field term frequency function of the term t in the field c.

This simple modification of the classical $tf - idf$ equation allow us to compute the score of the documents considering the structural information.

3.2 Dangers to combine scores from different document fields

As we mentioned in section 3.1, most ranking functions, use a non-linear method to saturate the computation of the frequencies. This is due to the fact that the information gained on observing a term the first time is greater than the information gained on subsequently seeing the same term. The non-linear method can be as simple as a logarithmic or a square-root function or more complex parameter-based approaches like BM25 k_1 parameter. S. Robertson [17] has described the dangers to combine scores from different document fields and what are the most tipical errors when ranking functions are modified to consider the structure of the documents.

A good example of these errors is Lucene library. Apache Lucene[6] is a high-performance and full-featured text search engine library written entirely in Java. It is a scalable technology suitable for nearly any application that requires full-text search. Lucene offers high-performance indexing and has become one of the most used search engine libraries in both academia and industry.

The ranking function is the core of any search engine applied to determine how relevant a document is to a given query. Lucene ranking function is built on a combination of the Vector Space Model (VSM) and the Boolean model of Information Retrieval. The main idea behind Lucene approach is the more times a query term appears in a document relative to the number of times the term appears in the whole collection, the more relevant that document will be to the query. Lucene uses also the Boolean model to first narrow down the documents that need to be scored based on the use of boolean logic in the query specification. Lucene is able to deal with document structure and actually implements the necessary functionalities to index structured documents based on fields. To rank these structured documents, Lucene combines the scores from document fields.

The method used by Lucene to compute the score of an structured document is based on the linear combination of the scores for each field of the document[7]:

$$score(q,d) = \sum score(q,c) \tag{2}$$

where

$$score(q,c) = \sum tf_c(t,d) * idf(t) * w_c \tag{3}$$

and

$$tf_c(t,d) = \sqrt{freq(t)} \tag{4}$$

As we can see, Lucene's ranking function uses $\sqrt{freq(t)}$ to implement the non-linear method to saturate the computation of the frequencies, but the linear combination of the $tf_c(t,d)$ values that Lucene implements breaks the saturation effect, since field's boost factors w_c are applied after of non-linear methods are used. The consequence is that a document matching a single query term over several fields could score much higher than a document matching several query terms in one field only, which is not a good way to compute relevance and use to hurt dramatically ranking function performance.

This problem hurt search engine performance where two or more fields are used to represent a document, as we will see in section 5.1.

4. A SEMANTIC SEARCH EVALUATION FRAMEWORK

To the best of our knowledge, there is no standard evaluation frameworks to analyze how semantic information affects retrieval process. The best effort to offer a standard evaluation framework for Semantic Search is the work by [9], where TREC collections are proposed like test beds for semantic retrieval. A very good state-of-art about previous semantic search evaluations efforts can also be found in this paper.

On the contrary, evaluation in IR has been a very active research area in the last 40 years. As a result, there are a several well known competitions and test IR Systems that can be evaluated. The most important IR contest is TREC, which offers collections of documents and queries in order to provide a good standard framework for different kind of evaluations.

There are two typical evaluation approaches in IR. One of them is user-focused evaluation, where real users are asked to judge search engine performance. This evaluation approach is used in Interactive Information Retrieval, where user feedback and user interaction are the main issues to evaluate. The second IR approach is a method is based on automatic evaluation. It requires a data corpus and binary evaluations in order to evaluate search engines performance. The evaluation score is automatically calculated, based on a method coming from the experiments conducted by Cyril W. Cleverdon at Cranfield University in the 60s to evaluate the efficiency of indexing systems [3]. The Cranfield methodology uses three basic elements: Document Collection, a set of Topics, used to define queries and the correspondent relevance judgments, with examples of relevant and non-relevant documents for the queries generated from topics. This methodology have been developed for the last 30 years in the TREC conference providing to IR community with a robust evaluation framework.

[6]http://lucene.apache.org

[7]Formulas have been simplified for the sake of clarity in order to highlight the important terms

Moreover, other IR contests following the Cranfield methodology have appeared in the last 20 years. They focus on different IR areas such as Web Search, multilingual IR or domain-oriented IR. A Semantic Search evaluation framework can be proposed in the same way. We firmly believe that this would lead to interesting and fruitful comparisons amongst the different approaches that would make possible to make great progress in the field.

Unfortunately, none of the already-proposed collections for IR evaluation are useful for Semantic Search evaluation, since they have been designed to evaluate traditional IR systems. However, INEX, the XML retrieval contest, offer a collection of documents taken from the Wikipedia that can be adapted to Semantic Search. This is due to the fact that the semi-structure that characterize XML documents is similar to the structure of RDF documents used in Semantic Search and because some semantic knowledge bases like dbpedia and Yago have been already used in some INEX tracks.

In next section, we propose an evaluation framework based on the INEX contest. It is based on the idea of mapping the Wikipedia documents used in INEX to their corresponding dbpedia RDF entries.

4.1 A first evaluation framework using dbpedia + INEX

INEX evaluation framework fits good enough to the goal of evaluating Semantic Search systems, but some changes must be done to adapt it to the semantic field. The main problem is that Wikipedia, which is the source of the INEX collection, has no semantic information itself. To solve this limitation we have mapped Dbpedia to the Wikipedia version used in the INEX contest. Dbpedia entries contain semantic information drawn from Wikipedia pages. In addition, Dbpedia has been used in the evaluation of semantic web search engines like SEMPLORE, so it can be considered as a good corpus for this kind of tasks.

Currently Dbpedia contains almost three millions of entries and the INEX-Wikipedia collection contains 2,666,190 documents. As a result, our corpus only takes into account the 2,233,718 document-entities that result from the intersection of both collections.

Given the corpus, INEX 2009 topics and assessments are adapted to this intersection. The result of this operation have been 68 topics and a modified assessments file.

The resulting framework makes possible to evaluate the retrieval performance in a collection of RDF documents. At this point, it is needed to establish a metric that allow us to evaluate. For this purpose, we have used TREC-eval software [8], which implements state-of-art retrieval metrics. It will allow us not only to assess the precision and recall, but also the quality of our ranking.

A standard, accepted set of metrics has been defined by IR community for search engine performance evaluation [15]. IR evaluation metrics are oriented in two main directions: the ability to retrieve relevant documents and the ability to sort them properly. We have used different measures to evaluate our system. Each of these measures assess the quality of some aspect of the retrieved documents:

- Mean Average Precision (MAP): the average of the precision value (percent of retrieved documents that are relevant) obtained for the top set documents existing after each relevant document is retrieved. In this way, MAP measures precision at all recall levels and provides a view of both aspects.
- Geometric Mean Average Precision (GMAP): a variant of MAP that uses a geometric mean rather than an arithmetic mean to average individual topic results. It informs about how robust a systems is.
- Precision after X documents (P@X): It measures the precision after X documents (whether relevant or non-relevant) have been retrieved. If X documents were not retrieved for a query, then all missing documents are assumed to be non-relevant.
- R-Precision: it measures precision after R documents have been retrieved, where R is the total number of relevant documents for a query.

[8] http://trec.nist.gov/trec_eval/

5. THE CASE STUDY: LUCENE VS BM25

The evaluation framework proposed in the previous section makes possible a rigorous comparison between different ranking and indexing models. In this case, we have used to analyze the impact of documents structure in the retrieval with two different ranking functions, Lucene and BM25/BM25F [16] This will also allow us to empirically illustrate the shortcomings of the Lucene weighting scheme for structured documents.

In order to do that, we have generated two different indexes based on the corpus described in Section 4.1. The first index have a plain structure, i.e., the RDF-document structure is not taken into account. This is done by merging the content of all the fields described in Section 2.2, that is, *text*, *title*, *inlinks*, *obj* and *type*. This index will be used to establish a baseline to compare non-structured retrieval against the structured one. The second index contains the aforementioned fields, so structure is taken into account. This multi-field index will make possible to evaluate the impact of document structure in the retrieval and also the impact of each individual field.

Each index is used with two different ranking functions, one from Lucene and the other a BM25. More specifically, the plain index has been used to retrieve documents with the BM25 and the standard Lucene ranking functions; while the multi-field index has been used with the BM25F and the field-based Lucene ranking functions. All of them have been previously discussed in Section 3.

The multi-field ranking approaches assign boost factors to each field in order to assign different weights depending on the field where query words occur. The boost factors used in our case study are $text = 1$, $title = 3$, $inlinks = 2$, $obj = 2$, $type = 2$. Given that boost factors, the frequency of words occurring in, say, URI, which is the field *title* for us, is multiplied by 3. The BM25F ranking function needs additional parameters. For them we have considered the following values $K_1 = 1.7$, $b_{title} = 0.4$, $b_{inlinks} = 0.4$, $b_{type} = 0.4$, $b_{obj} = 0.4$ and $b_{text} = 0.3$. The optimization of boost factors and the rest of parameters usually render a better performance in the retrieval task. However, it implies machine learning methods that require two sets of queries, one to train the methods and other to evaluate them. Given that the number of queries in our evaluation framework is not high, we have not carried out the optimization and we have

used values guided by our judgment. However, the study of the effect of other values is a line of future work.

As we have already mentioned, the set of queries in our evaluation framework are taken from the INEX 2009 contest. INEX 2009 provides judgments for 68 topics. Each topic made up three different versions of the same topic, *title*, *description* and *narrative*, which is similar to other evaluation frameworks like TREC.

We have run two different experiments, each one using a different kind of queries, the short and the long version of the 68 queries. In the short queries, only the *title* version of the topic is considered. This kind of queries is very short and is similar to the words that people use to retrieve from Web Search Engines like Google or Bing. The average length of this queries is around 3 words. In the extended queries, we merge the content of the three versions of each topic obtaining a much more verbose query. Extended queries use three versions of the same topic. The main average length of these queries after concatenate *title*, *description* and *narrative* is around 30 words which imply more verbose and expressive queries where the topic is well explained and represented.

5.1 Results and discussion

The performance of different ranking models is evaluated. Tables 2 and 3 show the results obtained by the Lucene and the BM25 ranking functions using both the plain and the multi-field indexes. While Table 2 shows the results for the title-based queries, Table 3 does it for the extended ones.

	MAP	P@5	P@10	GMAP	R-Prec
Lucene	.1411	.3676	.3132	.0789	.1949
LuceneF	.1243	.4088	.3088	.0637	.1657
BM25	.1659	.4235	.3588	.0881	.2145
BM25F	**.1743**	**.4412**	**.3765**	**.1030**	**.2172**

Table 2: MAP, P@5, P@10, GMAP, R-Prec for title based queries. All this measures ranges from 0 to 1

	MAP	P@5	P@10	GMAP	R-Prec
Lucene	.1560	.4147	.3368	.0957	.2100
LuceneF	.1200	.3971	.2971	.0578	.1632
BM25	.1746	**..4735**	**.3868**	.1081	.2257
BM25F	**.1822**	.4647	.3824	**.1170**	**.2262**

Table 3: MAP, P@5, P@10, GMAP, R-Prec for long queries. All this measures ranges from 0 to 1

The main conclusion that can be drawn from the experiments is that BM25-based ranking functions overcomes the ones from Lucene in every single measure. This is not surprising since BM25 is the state-of-the-art approach in IR. From the results it is also evident that Lucene performance dramatically decreases when structured information is used in the index. In our view, this results illustrate the problems of Lucene when dealing with structured documents that were explained in Section 3.1. It can be seen, that BM25F obtains the best performance, although the improvement over BM25 is not significant for most of the considered metrics. Regarding the type of queries, slightly better results are obtained with the extended queries, which means that this extra information helps in certain situations.

Another major conclusion that can be obtained from the tables is that the overall results are not satisfactory, if we compare them with those usually obtained by the participants in the INEX contests, whose metric values are higher. This is due to the fact that our corpus documents, the RDF Dbpedia entries, are not as verbose as those used in INEX, the Wikipedia entries. As it is well-known, IR methods take profit from verbose documents and this fact burdens the performance when retrieving documents from Dbpedia.

In order to deepen into the analysis of the retrieval of structured documents, we have also run experiments to study the impact of each field in the performance. Tables 5 and 4 show the sensibility analysis carried out for the multifield-Lucene and the BM25F ranking methods. In this analysis, we compare the results obtained only taking into account the *text* field, the *text* field along with one of the rest of the fields or all the fields. According to these results, it is shown again that Lucene does not work well with structured documents. Regarding BM25F, we can conclude that most of the fields does not imply a significant improvement in the results. Only *title*, the field with more relevant keywords, slightly boosts the performance.

	te	te+ti	te+in	te+ob	te+ty	all
MAP	.1756	**.1867**	.1760	.1749	.1750	.1822
GMAP	.1084	**.1190**	.1098	.1080	.1080	.1170
P@5	.4529	.4559	.4500	.4500	.4559	**.4746**
P@10	.3882	**.3941**	.3897	.3853	.3853	.3824

Table 4: Sensibility test for BM25F. All this measures ranges from 0 to 1. $te = text$, $ti = title$, $in = inlinks$, $ob = obj$, $ty = type$, $all = allfields$

	te	te+ti	te+in	te+ob	te+ty	all
MAP	**.1657**	.1211	.0691	.0707	.1422	.1200
GMAP	**.0998**	.0567	.0193	.0205	.0749	.0578
P@5	**.4265**	.4088	.1176	.2647	.4000	.3971
P@10	**.3676**	.3191	.1353	.2074	.3294	.2971

Table 5: Sensibility test for Lucene with fields. All this measures ranges from 0 to 1. $te = text$, $ti = title$, $in = inlinks$, $ob = obj$, $ty = type$, $all = allfields$

These results can be explained if we take into account that BM25F is able to deal with the fields in RDF documents, but again the content of these fields is not verbose. BM25F deals with the structure of the document, but not with the semantic associated to the structure. This conclusion points to the fact that it seems possible to improve the results of the BM25 approach, which is the usual benchmark in IR, if we are able to take into account the semantic information in an explicit way and not only implicitly, as BM25F does when it just considers the fields and ignores their meaning.

In order to gain more insight into the results, we have also estimated the densities of the MAP values obtained by the 68 queries of the evaluation framework for the different retrieval approaches. They are shown in Figure 1. The plot confirms the conclusions drawn from the tables regarding

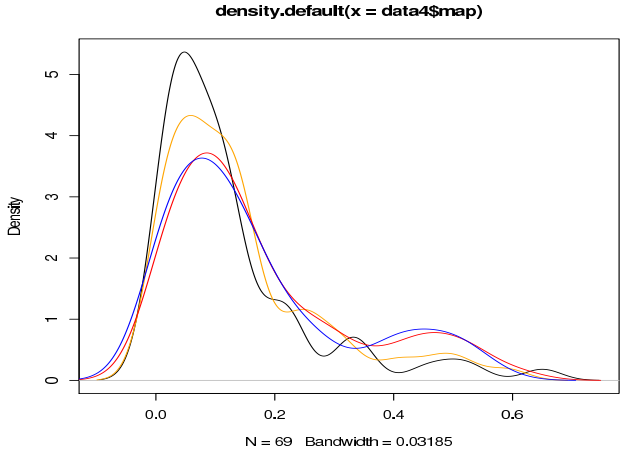

Figure 1: Density of the MAP values for different ranking approaches (BM25=blue, BM25F=red, Lucene=yellow, Lucene multifield=black)

the performance of the different approaches. However, it highlights a much more interesting feature of the result that remained hidden when we analyzed the mean MAP value of the 68 queries: it can be clearly seen that the BM25 and BM25F distributions have two modes, one corresponding to the queries whose results are poor and the other corresponding to the queries whose results are surprisingly good. However, the number of poor-performance queries is much higher. In short, the query results are not homogeneous and they do not follow a unimodal distribution, as it could be expected. This result opens a new line of research, as it would be very interesting to identify the features of the queries that belong to each of the two kinds. Knowing the reasons behind the bimodal distributions would make possible to propose better retrieval approaches that are able to enhance the performance of the queries for which the current approaches fail to offer satisfactory results.

6. CONCLUSIONS AND FUTURE WORK

This work has analyzed structured IR in the field of Semantic Search. In order to do so, a comparison between the popular Lucene and the state-of-art BM25 methods has been carried out. The effect of considering the documents structure in both methods has also been studied. Some interesting results have arisen from this work. It has been shown that BM25 is better than Lucene, specially when taking into account the document structure. In this case, Lucene hurts the retrieval performance, while BM25F does not. However BM25F is not able to take profit from the semantic information contained in the fields with less text.

These conclusions can be helpful to improve the performance of Semantic search engine implementations based on Lucene, such as Sindice, Watson, Falcons or SEMPLORE. They also highlight that there is plenty of room for collaboration between IR and semantic search. We believe that hybrid methods combining BM25F retrieval method and semantic technology should be able to increase the results obtained. In addition, the analysis of the results opened some interesting research issues such as the bimodal distribution of query results that should be addressed in future studies.

Another interesting contribution of this work is that a first evaluation framework for Semantic Search has been proposed. It could be extended and improved, but in its present form is an extremely useful tool to objectively test the performance of other Semantic Search approaches.

References

7. ADDITIONAL AUTHORS

8. REFERENCES

[1] C. Bizer, T. Heath, K. Idehen, and T. Berners-Lee. Linked data on the web (ldow2008). In *WWW*, pages 1265–1266, 2008.

[2] G. Cheng and Y. Qu. Searching linked objects with falcons: Approach, implementation and evaluation. *Int. J. Semantic Web Inf. Syst.*, 5(3):49–70, 2009.

[3] C. Cleverdon. The Cranfield tests on index language devices. pages 47–59, 1997.

[4] N. Craswell, H. Zaragoza, and S. Robertson. Microsoft cambridge at trec-14: Enterprise track. In *TREC*, 2005.

[5] M. d'Aquin, C. Baldassarre, L. Gridinoc, S. Angeletou, M. Sabou, and E. Motta. Characterizing knowledge on the semantic web with watson. In *EON*, pages 1–10, 2007.

[6] M. d'Aquin, M. Sabou, E. Motta, S. Angeletou, L. Gridinoc, V. Lopez, and F. Zablith. What can be done with the semantic web? an overview Watson-based applications. In *SWAP*, 2008.

[7] R. Delbru. SIREn: Entity retrieval system for the web of data. In *Proceedings of the 3rd Symposium on Future Directions in Information Access (FDIA).*, 2009.

[8] S. Elbassuoni, M. Ramanath, R. Schenkel, M. Sydow, and G. Weikum. Language-model-based ranking for queries on RDF-graphs. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 977–986, New York, NY, USA, 2009. ACM.

[9] M. Fernandez, V. Lopez, E. Motta, M. Sabou, V. Uren, D. Vallet, and P. Castells. Using TREC for cross-comparison between classic IR and ontology-based search models at a Web scale. In *Workshop: Semantic search workshop at 18th International World Wide Web Conference*, 2009.

[10] R. V. Guha, R. McCool, and E. Miller. Semantic search. In *WWW*, pages 700–709, 2003.

[11] D. Harman. Overview of the first text retrieval conference (trec-1). In *TREC*, pages 1–20, 1992.

[12] A. Harth, A. Hogan, R. Delbru, J. Umbrich, S. O'Riain, and S. Decker. SWSE: Answers before links! In *Semantic Web Challenge*, 2007.

[13] D. Hawking, E. M. Voorhees, N. Craswell, and P. Bailey. Overview of the trec-8 web track. In *TREC*, 1999.

[14] M. Lalmas. *XML Retrieval*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool Publishers, 2009.

[15] C. D. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[16] J. Pérez-Iglesias, J. R. Pérez-Agüera, V. Fresno, and Y. Z. Feinstein. Integrating the Probabilistic Models BM25/BM25F into Lucene. *CoRR*, abs/0911.5046, 2009.

[17] S. E. Robertson, H. Zaragoza, and M. J. Taylor. Simple BM25 extension to multiple weighted fields. In *CIKM '04*, pages 42–49, 2004.

[18] M. J. Taylor, H. Zaragoza, N. Craswell, S. Robertson, and C. Burges. Optimisation methods for ranking functions with multiple parameters. In *CIKM*, pages 585–593, 2006.

[19] G. Tummarello and R. Delbru. Entity coreference resolution services in sindice.com: Identification on the current web of data. In *IRSW*, 2008.

[20] H. Wang, Q. Liu, T. Penin, L. Fu, L. Zhang, T. Tran, Y. Yu, and Y. Pan. Semplore: A scalable IR approach to search the web of data. *J. Web Sem.*, 7(3):177–188, 2009.

[21] H. Zaragoza, N. Craswell, M. J. Taylor, S. Saria, and S. E. Robertson. Microsoft cambridge at trec 13: Web and hard tracks. In *TREC*, 2004.