A Study of Entity Search in Semantic Search Workshop

Xitong Liu
Department of Electrical and Computer Engineering
University of Delaware
xliu@ece.udel.edu

Hui Fang
Department of Electrical and Computer Engineering
University of Delaware
hfang@ece.udel.edu

ABSTRACT

The paper describes the system we developed for the Entity Search Track of Semantic Search 2010 Workshop. The problem of entity search is to retrieve relevant results from a semantic data set about entities. Our general goal is to study how we may apply existing Information Retrieval (IR) techniques to solve the problem. Specifically, we focus on how to utilize the IR systems to store the semantic data about entities and how to leverage the IR models to retrieve the entities. We evaluate the proposed system with the sample queries and report the results.

1. INTRODUCTION

The InfoLab from University of Delaware participated in the Entity Search Track of the Semantic Search 2010 Workshop. The task of entity search is to retrieve relevant entities from a semantic data set for a given query, where the data set contains semi-structured information about different entities and the query is a keyword query.

Entity search is an important problem and has recently started attracting attention from research communities. Many problems centering around entity search, such as expert finding problem [3, 8, 1] and related entity finding problem [2] have been proposed and studied, but the setups of these problems are different. In particular, *expert finding* is to retrieve entities (i.e., experts) from a unstructured data set (i.e., a document collection) given a keyword query, *related entity finding* is to retrieve entities from a unstructured data set given a structured query, and the *entity search* problem that we study in this paper is to retrieve entities from a semi-structured data set given a keyword query.

The problem of entity search track can be formulated as a keyword search problem over a semi-structured data set. Since most studies on keyword retrieval utilize IR techniques, a straightforward solution is to leverage existing IR techniques and adapt them to solve the entity search problem, which is the main focus of our study. The main challenges include (1) how to convert the semantic data set to a format that existing IR systems can process, and (2) how to model the relevance based on the query and the data set.

To address the first challenge, we explore two options and discuss the trade-off for each option. Based on the discussions, we then decide to convert the semantic data set into a document collection, where each document corresponds to the profile of an entity. To address the second challenge, we need to understand the semantic meaning of an entity query. Unfortunately, since this is the first year of the entity search track, the semantic meaning of a query is not well defined and understood. Thus, we decide to directly apply existing retrieval models to rank the results this year, and leave the futher study of the second challenge as our future work.

The paper is organized as follows. We describe the developed system in Section 2, report the experiment results for sample queries in Section 3, and conclude in Section 4.

2. SYSTEM DESCRIPTION

We now describe an overview of the system that we developed for the entity search track. The system can be decomposed into two components: *entity profile building* and *entity retrieval*. Specifically, we first construct an entity profile collection based on the given semantic data set, and then discuss how to retrieve relevant entities based on their profiles for a given query.

2.1 Entity Profile Building

In order to retrieve entities that are relevant to a query, we propose to build a *profile* for every entity so that each entity profile contains all the information about the entity based on the data set.

The data set used for the track is Billion Triple Challenge 2009 dataset [1], which contains the semantic information about entities. An entity is described with a set of attributes and their values. For example, a movie entity may be described with the following attributes, i.e., entity name, director, release date, actors, country and language. Each line of the data set describes the information of an attribute for a particular entity in the following N-Quads format [2]:

```
<subject> <predicate> <object> <context>.
```

N-Quads format is a line-based plain text presentation of RDF graph. The subject field includes the entity name or URI for the entity, the predicate field indicates the attribute type, and the object field contains the attribute value. The fourth element, i.e. the context field, is the URI of the graph that provides the context information of other fields. Thus, if two lines have the same subject field values, it means that these two lines contain the information about the same entity.

For example, the following N-Quads statements contain the information of two entities, i.e., a movie "Forrest Gump"

[1]http://vmlion25.deri.ie/
[2]http://sw.deri.org/2008/07/n-quads/

Table 1: Statistics of top 10 attribute types

Attribute Type	Occurrences
wikilink	156448093
title	37923223
name	23372231
weblog	23266489
subject	22144813
has-author	19567160
links_to	19157151
label	16835111
interest	16715068
description	15211377

(first three statements) and a book "Pride and Prejudice" (last three statements).

```
<Forrest Gump> <director> <Robert Zemeckis> <LINK1> .
<Forrest Gump> <actor> <Tom Hanks> <LINK1> .
<Forrest Gump> <release_date> <July 6, 1994> <LINK1> .
<Pride and Prejudice> <author> <Jane Austen> <LINK2> .
<Pride and Prejudice> <country> <United Kingdom> <LINK2> .
<Pride and Prejudice> <genre> <novel of manners> <LINK2> .
```

where LINK1 and LINK2 are the URIs of the graphs that contain each statement. We can see that a movie is described with three attributes, i.e., director, actor and release date, and a book is described with three attributes, i.e., author, country and genre. The first field describes the entity name, the second describes the attribute type, the third describes the attribute value, and the fourth describes the context information of the statement.

Since an entity may contain multiple attributes and each attribute is described in one line (i.e., one N-quads statement), the information of an entity could be distributed in multiple lines. Clearly, given an entity, we may construct its profile by merging the information of all the attributes of the entity, i.e., by merging all lines whose subject field values are the name or the URI of the entity. Thus, an entity profile contains the information of its attributes and corresponding attribute values.

Given an entity profile, we explore two options to represent the information of the profile. The first option is to store the profiles as XML documents [6]. A commonly used existing strategy is to store the information of each attribute separately, retrieve the information based on each attribute, and then combine the retrieval results based on all the attributes of an entity. The second option is to store the profiles as a unstructured document, where we simply merge all the information of an entity (i.e., attribute types and attribute names) together and represent them as "bag-of-terms".

The advantage of the first option is to allow us to utilize the semantic meaning of the data. Unfortunately, our preliminary results suggest that this option may not be feasible for the given data set due to a wide variety of entities and entity attributes. We find that there are 117,383 different attribute types in the data set, and the most frequent attribute types are shown in Table 1. Thus, we choose the second option to represent the profile in our system.

To construct an entity profile, we propose to merge all the useful information of an entity together into a document. Specifically, we extract the predicate fields and object fields associated with the same subject fields and merge them together. We do not utilize the context fields because they only provide the origins of the RDF graphs and do not contribute additional information of an entity.

For the object fields (i.e., attribute values), we include all the information of the fields into the entity profiles. For the predicate fields, we find that they have the following two formats:

1. `<http://dbpedia.org/property/abstract>`, in which the attribute type is located in the last part of URI, i.e. `abstract`.

2. `<http://www.aktors.org/ontology/portal#year-of>`, in which the type is located in the string after `#`, i.e. `year-of`.

Since we are more interested in the attribute type, the entity profile only includes the attribute type information rather than the whole predicate fields. The extraction of the attribute types is done based on the regular expression matching for the above two formats.

Moreover, since there are blank nodes in the data set, we need to discuss how to process them in our profile building process. We use the data set preprocessed by the organizers, in which the blank node is encoded into
`http://example.org/URLEncode(BNID)`,
where BNID is a unique ID in the dataset. In the data set, a blank node is either a subject field or an object field whose value is encoded with BNID. Note that a blank node serves as a bridge that connects two N-quads statements. If the subject field value of one statement contains the same BNID as the object field value of the other statement, the former statement provides additional information for the entity related to the later statement. Thus, when building the profile for the entity, we need to include the information from both statements. This process will be done iteratively until all the statements with the blank nodes as their subject fields are merged into the entity profiles.

Although representing an entity profile as a unstructured document is simple and straightforward, it merges all the attribute types and values together and ignores the semantic information of an entity. However, intuitively, we may need to utilize these semantic information to improve search accuracy. For example, if a user wants to retrieve all the books written by Jane Austin, the system with the current profile representation may return the movie "Becoming Jane" to the user due to the loss of the semantic meaning of the data. We plan to explore other ways of representing profile that would better suit for semantic search in our future work.

2.2 Entity Retrieval

After building the entity profiles, each profile can be regarded as a document including all the information of an entity. As a result, the problem of entity search over the original semantic data set is the same as document retrieval over the newly constructed entity profile collection. Motivated by this observation, we can then leverage the existing IR techniques. Specifically, we first build index over the entity profile collection, and then run existing retrieval models to retrieve relevant entities based on the corresponding profiles.

We apply three existing retrieval functions, i.e., Okapi, Dirichlet Prior and axiomatic retrieval function. We now define the notations used in the functions. $S(Q,D)$ denotes the scoring function. $c(t,D)$ is the count of term t in document D and $c(t,Q)$ is the count in query Q. N is the number of documents in the collection. $|D|$ is the document length, and $avdl$ is the average document length of the collection. $df(t)$ is the number of documents containing term t. $p(t|C)$ is the probability of a term t given by the collection.

- Okapi [7] (a retrieval function derived from the classical probabilistic retrieval model):

$$S(Q,D) = \sum_{t \in Q \cap D} ln \frac{N - df(t) + 0.5}{df(t) + 0.5} \times \frac{2.2 \times c(t,Q)}{1.2 + c(t,Q)} \times \frac{1001 \times c(t,D)}{1000 \times ((1-b) + b\frac{|D|}{avdl}) + c(t,D)}, \quad (1)$$

  where b is a parameter.

- Dirichlet Prior [9] (a retrieval function derived using language modeling approach):

$$S(Q,D) = \sum_{t \in Q \cap D} c(t,Q) \cdot ln(1 + \frac{c(t,D)}{\mu \cdot p(t|C)}) + |Q| \cdot ln \frac{\mu}{|D| + \mu}, \quad (2)$$

  where μ is a parameter.

- Axiomatic retrieval function [4] (a retrieval function derived using axiomatic approach):

$$S(Q,D) = \sum_{t \in Q \cap D} c(t,Q) \times \frac{c(t,D)}{c(t,D) + s + s \cdot \frac{|D|}{avdl}} \times (\frac{N+1}{df(t)})^{0.35}, \quad (3)$$

  where s is a parameter.

3. EXPERIMENTS

3.1 Data Sets

We now report the statistics of the data sets. There are 1,464,829,200 lines in the Billion Triple Challenge 2009 dataset, and 79,272,352 distinct entities in the entity profile dataset. The document length distribution of entity profile dataset is shown in Figure 3.1. Obviously most of the documents are short and nearly half of them are shorter than 50 terms. The average document length is 115 terms and the maximum document length (i.e., MaxDocLen) is 3,244,241 terms.

3.2 Preliminary Results

We use the sample queries provided by the organizer as the training set to measure the performance of different models under different parameter settings. Before we conduct the experiments, we construct the judgements for the sample queries using the pooling strategy [5]. Specifically, we treat a retrieval model with a parameter setting as a run, and merge the top 5 documents from all the runs into a pool. For each query, all the documents in the pool are judged as either relevant or non-relevant.

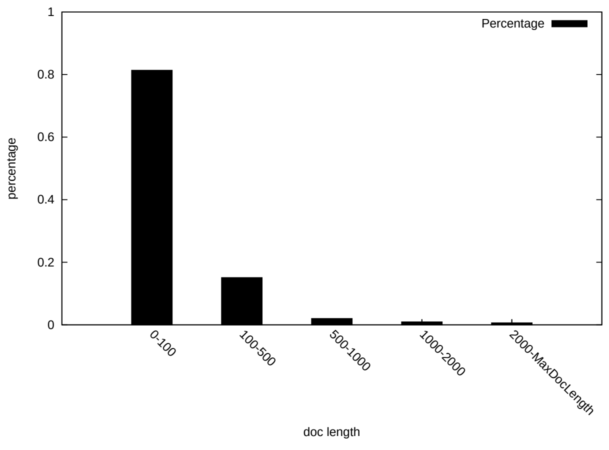

Figure 1: Profile Document Length Distribution.

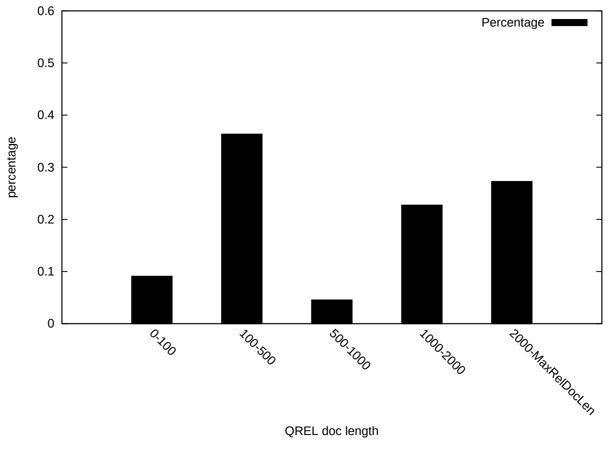

Figure 2: QREL Document Length Distribution.

Table 2: Performance comparison of different functions with different parameters

Retrieval Model	MAP
Okapi, b=0.05	0.4915
Okapi, b=0.1	0.4912
Okapi, b=0.2	**0.5130**
Okapi, b=0.5	0.4487
Okapi, b=0.7	0.3372
Dirichlet Prior, μ=500	0.4599
Dirichlet Prior, μ=1000	**0.4714**
Dirichlet Prior, μ=2000	0.4591
Dirichlet Prior, μ=5000	0.3967
Dirichlet Prior, μ=10000	0.3465
Axiomatic, s=0.05	**0.4746**
Axiomatic, s=0.1	0.4180
Axiomatic, s=0.2	0.4085
Axiomatic, s=0.5	0.4520
Axiomatic, s=0.7	0.4316

Figure 3.2 shows the document length distribution over the relevant documents based on the judgements we construct for the sample queries. The average relevant document length is 2,933 terms and the maximum relevant document length (i.e., MaxRelDocLen) is 15,594 terms.

It is interesting to see that the document length distribution for relevant documents is not consistent with the distribution of the dataset. Since most of the documents in the dataset are short documents, the relevant document are usually much longer. An intuitive explanation is that longer documents have higher term frequency of the terms in the entity search queries. This observation suggests that we do not need to penalize longer documents as harshly as for document retrieval. As a result, we make the hypothesis that the parameters should be set to smaller values than the ones used for the document retrieval task. In fact, the retrieval performance confirms our hypothesis as shown in Table 2. The default parameter values used for document retrieval are 0.75 for Okapi, 2000 for Dirichlet Prior and 0.5 for axiomatic retrieval function. However, the optimal values for the entity search track are 0.2 for Okapi, 1000 for Dirichlet Prior, and 0.05 for the axiomatic retrieval function.

3.3 Submitted Runs

Based on the results shown in Table 2, we use the following strategies for the three submitted runs.

- **U**delOkapi: Use the Okapi as the retrieval function and the value of parameter b is set to 0.2.

- **U**delDir: Use the Dirichlet Prior as the retrieval function and the value of parameter μ is set to 1000.

- **U**delAX: Use the axiomatic retrieval function and the value of parameter s is set to 0.05.

4. CONCLUSION

We describe the system that we developed for the entity search track. The main focus is to study whether we could adapt existing IR techniques to solve the problem. We propose to construct a unstructured entity profile collection based on the semantic data set, and then apply existing IR retrieval models to retrieve relevant entities based on their profiles.

In our future work, we plan to revisit the two challenges and explore how to adapt the current strategies to improve the search accuracy. In particular, with the availability of the official judgement files, we will be able to understand the semantic meaning of the entity queries and utilize such information to study how to better build the entity profiles and how to better model the relevance for the entity queries.

5. REFERENCES

[1] P. Bailey, N. Craswell, A. P. de Vries, and I. Soborof. Overview of the trec 2007 enterprise track. In *Proceedings of Text Retrieval Conference*, 2007.

[2] K. Balog, A. P. de Vries, P. Serdyukov, P. Thomas, and T. Westerveld. Overview of the trec 2009 entity track. In *Proceedings of Text Retrieval Conference*, 2009.

[3] N. Craswell, A. P. de Vries, and I. Soboroff. Overview of the trec 2005 enterprise track. In *Proceedings of Text Retrieval Conference*, 2005.

[4] H. Fang and C. Zhai. An exploration of axiomatic approaches to information retrieval. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 480–487, 2005.

[5] D. Harman. Overview of the fourth text retrieval conference (trec-4). In *Proceedings of Text Retrieval Conference*, 1995.

[6] M. Lalmas. *XML Retrieval*. Morgan & Claypool, 2009.

[7] S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *Proceedings of Text Retrieval Conference*, pages 109–126, 1996.

[8] I. Soborof, A. P. de Vries, and N. Craswell. Overview of the trec 2006 enterprise track. In *Proceedings of Text Retrieval Conference*, 2006.

[9] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions of Information Systems*, 22(2):179–214, 2004.