Semantic Entity Retrieval using Web Queries over Structured RDF Data

Jeff Dalton and Sam Huston
CS645 Project Final Report
May 11, 2010

Abstract
We investigate the problem of performing entity retrieval over a large collection of RDF data. As part of this project we took part in the Semantic Search 2010 Workshop. We show that retrieval performance depends on the effective use of the structure present in the data. This requires the mapping of textual queries to structured queries. The scale and diversity of the collection make this a particularly challenging task. First, we present an overview of the systems we investigated to store and query the dataset. We then investigate the effectiveness of both structured and unstructured retrieval models. We find that a semi-structured retrieval model utilizing hand selected attributes outperformed unstructured models, placing second overall at the Semantic Search 2010 Workshop. Finally, we show initial work further exploring the mapping of terms and queries into more structured representations structured queries. The initial work shows that the problem is very challenging and a key problem is that data modeling needs to be performed in the context of entity retrieval to ensure the correct level of granularity for types and attributes.

Introduction
This project documents our efforts to take part in the Semantic Search 2010 Workshop (http://km.aifb.kit.edu/ws/semsearch10/).

In particular, the workshop challenge is designed to find out:
- How to allow users to ask complex questions and interact with structured semantic web data using keyword queries?
- How can structured semantic information be applied to IR problems?

The goal of our project is to improve entity retrieval of structured documents from queries specified to a web search engine using keywords. For example a keyword query [rated G movies released in 2009], may be mapped into a structured query as: [entityType=Movie att:Rating=G att:releaseYear=2009]. We can then represent this translated query in a structured query language and then use a DBMS or IR System to retrieve the matching entities.

With this as our stated aim, there are several key challenges to translate keywords to structured queries. First the ambiguity of natural language leads to many plausible interpretations of any textual query. This ambiguity is unavoidable, however, we can model the ambiguity by using a probabilistic mapping for each query to a set of structured interpretations. Second there is a complex implicit transformation from some query terms to a value for an attribute. We do not directly address this challenge in this project due to time constraints. We do however consider a sub-problem. We assume that all query terms are informational keywords, this means that predicate names are missing from the query. Third there are no guarantees that any particular entity will have data for any particular attribute. This can lead to problems involving insufficient statistics for particular attributes or types. This third challenge implies an appropriate model for the data; a large heterogeneous collection of entities with diverse schemas.

Semantic Retrieval Systems

We explored loading the dataset into several different systems for retrieval, the HBase column store, Neo4J graph store, and Indri retrieval engine.

We were unable to use HBase because it is not compatible with our cluster management software. HBase requires persistent services running on a dedicated cluster, which does not interface with the GridEngine software used on the Sydney IR cluster or the Swarm cluster used in our experiments. Secondarily HBase does not currently provide a SPARQL interface. Instead it provides a proprietary Key-Value API. This API could be used for non-ranked boolean query processing. However our needs extend far beyond this use.

We then explored the Neo4J graph store. Neo4J is an open-source graph database system that stores the data as nodes and relationships as edges. The RDF triples fit naturally into this model. Neo4J supports the SPARQL query language to search the data. Given that our queries are text keywords, an important feature is full-text search over the nodes. Neo4J supports this using the Lucene search engine to index the node content. We were not able to get the RDF indexing components to work out-of-the-box.

The Indri retrieval engine is comparable to the Lucene system used by Neo4J to retrieve nodes. Thus, retrieving these document nodes stored in Indri is equivalent to the text search capability of Neo4J on our dataset. Since we have significant prior experience with Indri, we chose to use this system to index and test the performance of our queries. Additionally the Indri query language provides a simple interface to incorporate belief weights for attributes or types.

Related Work

Kim et al. (2009) presented a probabilistic retrieval model for semi-structured data. The model allows the words of a textual query to be weighted over several matching attributes. However, a key limitation is that it assumes a collection with a single or very few clearly defined entity types. Our proposed problem extends this work to heterogenous collections where there are many heterogenous (thousands or more) entity types.

In Information Retrieval, the field of Entity Retrieval has been well-studied. In 2009, the TREC Entity Track focused on retrieving entities from the ClueWeb web crawl. Since 2007, INEX Entity-Ranking Track has focused semi-structured retrieval of entities from Wikipedia (Demartini, 2007). Both of these tracks focus on entities with large quantities of text in the form of Wikipedia entries or web homepages. In contrast, the collection we focus on is primarily structured data in the form of RDF with some associated text.

In the database community, there is work on keyword search over structured databases (Agrawal, et al. 2002; Agrawal, et al., 2009) and translating text queries into SQL (Paparizos 2009). For an overview of recent work in this area, the SIGMOD 2009 tutorial by Chen, et. al. (2009) provides a good introduction. The problem of keyword search over heterogeneous web data was described by Madhavan et. al, in work integrating data from Information Extraction (Madhavan et al., 2006, Madhavan et al., 2007). The previous work does not utilize real-world web queries and does not focus specifically on search over semantic data.

Recently, Pound et al. (2010) defined the Adhoc Object Retrieval (AOR) task. This task extends previous IR document evaluation to semantic web objects. Their evaluation queries are composed of a subset of the queries that they annotated from the Yahoo! search log, restricted to entity queries. To evaluate retrieval effectiveness, they use a simple baseline TF-IDF text retrieval system over a collection of RDF data. They focus on the evaluation framework and did not explore the retrieval algorithms. We explore the more advanced language modeling based retrieval methods used by Indri. Furthermore, they do not attempt to infer desired object types or map query terms to fields, which we perform initial tests.

Retrieval Models

There are four types of query models that we test; *Full-Text*, *Select-Attributes*, *Probable-Attributes* and *Probable-Types*.

Full-Text

The first set of models we use are a baseline for the following models. Two standard IR models are used over the above described documents. Query-Likelihood, and Sequential Dependancies. Neither of these models makes use of any attribute designations. They both treat each document as an bag of words. Within our dataset the bag of words is constructed as the the set of all <object> values within the RDF tuples for a given <subject>.

The query likelihood model ranks documents according to the probability of relevance. This model makes the underlying assumption of term independence. It uses the following formulation:

$$P(D|Q) = \prod_{q \in Q} P(q|D)$$

The probability of a query term given a document is estimated using a Dirichlet smoothing with a default parameter $\mu = 2500$:

$$P(q|D) = \frac{tf_q + \mu P(q|C)}{|D| + \mu}$$

The sequential dependancy model uses a similar model, except it relaxes the independence assumption allowing for neighboring dependancies. It uses the following formulation:

$$P(D|Q) = \sum_{c \in T} \lambda_T f_T(c) + \sum_{c \in O} \lambda_O f_O(c) + \sum_{c \in U} \lambda_U f_U(c)$$

Where T is the set of query terms in Q, O is the set of ordered neighboring query word pairs, and U is the set of unordered neighboring query word pairs. The f functions produce Jelenik-Mercer smoothed log probabilities. Each of the λ values are weighting parameters, by as suggested by Metzler et al. (2005) they are set to:

λ_T	0.8
λ_O	0.1
λ_U	0.1

For example given the query: [banana paper making]. We have T = {banana, paper, making}, O={'banana paper', 'paper making'}, and U={(banana,paper), (making, paper)}.

Select-Attributes

The second set of models use a very limited set of attributes. This is based upon TRECWeb Track results that use specific metadata fields to improve retrieval results. Commonly the fields used include; title, subtitle, meta-description and anchor text. In order to test this approach we performed some simple ontology folding, and collected 4 attribute-sets; <name>, <title>, <dbpedia-title>, and <text>. The first three are fairly self-explanatory. The <text> field encapsulates all of the textual <object> values of the entity. This is distinct from the <object> values that contain urls or a link to another entity.

In particular we performed this ontology folding based upon the final term in the attribute name. For example: <http://someones.ontology.website.org/resources#name> would be interpreted as: <name>. These four condensed attributes were indexed as fields within the document. The idea here is to weight documents with correct titles more heavily than other similar documents. This model uses a similar formulation to the sequential dependancy model above, however, instead of query word pairs we incorporate the 4 attribute-sets:

$$P(D|Q) = \sum_{c \in A} \lambda_A f_A(c) + \sum_{c \in N} \lambda_N f_N(c) + \sum_{c \in T} \lambda_T f_T(c) +$$

$$\sum_{c \in X} \lambda_X f_X(c) + \sum_{c \in B} \lambda_B f_B(c)$$

Where A indicates the whole document; this includes all of the selected attributes, N corresponds to the name attribute, T corresponds to the title attribute, X corresponds to the text attribute, and B corresponds to the dbpedia-title attribute. Since were not provided with training data, setting these weights can not be accomplished using a hill-climbing or grid sweep method. Indeed the only option was to use intuitive values. We chose to use:

λ_A	0.61
λ_N	0.06
λ_T	0.12
λ_X	0.06
λ_B	0.15

Our intuition is clearly apparent from these parameters; dbpedia-titles are the most informative of the extracted attributes. The other parameter values that the extracted attributes are more valuable that the remaining parts of the document, thus they use a small parameter value.

Weighted-Attributes
This approach is identical to the PRM-S model described in our related work. As a baseline, we assume a homogeneous collection; all entities are 'documents'. This approach tests the PRM-S assumptions over a large scale collection with a very large number of attributes. We do not perform any ontology folding within these models. The approach is to weight each word in a query according to the likelihood that it will be in each attribute.

This model is formulated as follows:

$$P(D|Q) = \prod_{q \in Q} \sum_{a_d \in D} P_M(A|q) P(q|a_d)$$

Note that this formula is very similar to the query likelihood model. The key difference here is that it introduces a mapping probability (P_M) over each attribute. A represents the accumulation of all instances of a particular attribute within the collection. a_d represents a particular instance of the attribute in the document d.

Select-Types
The third set of models will use attribute <type> values that often represent the object class(es) of a subject. A member of this set of attributes exists in most entities. Given that each document has one or more types, we will assign probabilities for each <type> value, given a query. We will use the top-k most probable object types to limit the retrieval document set. Then we query this limited set using each of the

baseline models. This approach assumes a heterogeneous collection. In this manner we are testing one possible method of assigning entity types to a given query. This model is formulated as follows:

$$P(d|Q,T) = \sum_{t \in D} P(t|Q) \prod_{q \in Q} P(q|d)$$

Where T is the set of document types assigned to the query Q.

Dataset and indexing process

The set of queries were provided by the Semantic Search 2010 Workshop. They consist of 92 *entity* queries. The first 42 were extracted from the Yahoo Query Log Tiny Sample version 1.0. The second 50 were extracted from the Microsoft Live Search Log. These queries represent 40.6% of this type of query in a web search log. Only queries submitted by at least 10 users were sampled.

We do not have any training queries or relevance judgements. This was an implicit part of the Semantic Search Workshop. This means that none of the above model parameters may be estimated. We used default values where possible, and guessed where it was not possible. Given additional time we would have liked to train our models over previous ad-hoc entity retrieval tasks, then use these parameters for this project.

The dataset is the Billion Triple Challenge dataset available at http://vmlion25.deri.ie/. It was crawled during February/March 2009. It is based on datasets provided by Falcon-S, Sindice, Swoogle, SWSE, and Watson using the MultiCrawler/SWSE framework. To ensure wide coverage, it also included a (bounded) breadth-first crawl of depth 50 starting from http://www.w3.org/People/Berners-Lee/card. It is worth noting here that this dataset is extremely diverse.

It contains many different ontologies, from many different database schemas. This diversity contributes significantly to many of the challenges we faced in this project.

The dataset is provided in RDF format. The Resource Description Framework (RDF) is a series of triples; <Subject> <Predicate> <Object>. Intuitively, the subject is related to the object through the predicate. Objects can either be textual nodes or entities, as represented by the URI of a subject. Each Predicate is referred to be as an Attribute within this paper.

Size (Compressed)	17 GB
Triples	1.14 Billion
Unique Attributes	85,594

Table 1: Billion Triple Challenge Dataset

The RDF data was converted into a specific XML variant used for parsing and indexing with Indri. To perform this conversion, the RDF triples were first sorted by subject. The triples for each subject were grouped together to form a document. The RDF predicates were mapped to XML element names. The RDF objects were treated as element values. The result was 126GB of uncompressed XML documents. These were then indexed using the Indri retrieval engine. No stop words were removed and stemming was not applied during indexing.

Indri was able to index the full-text of the documents when the predicates (fields) were ignored. The full-text corpus statistics from Indri are highlighted in Table 1. The indexing process took approximately one day on a single computation node and resulted in an index that is 155 GB. This is the index used for our baseline query likelihood runs.

Documents (subjects)	191,397,652
Unique Terms	6,822,6071
Total Terms	12,423,901,192

Table 2: Indri statistics on the Billion Triple dataset

Another index was built that indexed the documents, including a limited set of predicates. These predicates will be described in our description of the *Select Attribute* model.

Indexing these fields allows the matches to be treated separately, allowing field weighting and restriction. This index separated the text values from the URL references and treats them separately. This created a larger index that is 187 GB. This index is used for the *Select-Attribute* and *Probable-Type* retrieval models.

Indri was not able to index the dataset using all of the predicate types as Indri extents (fields). The dataset contains 85,594 unique predicates. Indri is not designed to indexed structured documents with this number of fields. Indri uses a B-Tree representation for terms in its dictionary and stores statistics for each term-field pair. Unfortunately, this is not a sparse representation and large numbers of fields cause a single B-tree entry to exceed a page size, causing Indri to crash. These data structures mean that it is unable to index a collection with more than a few hundred fields, at most. Re-engineering the system to handle with additional fields is beyond the scope of this project.

Instead of indexing all fields, we decided to compute the probabilities of the attributes for the queries off-line. Upon inspection of this data, we saw that this attribute mapping approach does not return useful predicates as probable. These findings are further discussed in the discussion section.

SemSearch evaluation process

The Semantic Search Workshop recieved 14 runs from six teams. For each of the 92 queries, the top 10 results were evaluated. This resulted in a total of 5,786 query-URI pairs. The workshop used Mechanical Turk to evaluate query-result pairs for relevance. The turkers were paid $.20 for each set of 10 results. More detail on the evaluation process and inter annotator agreement are available in the conference proceedings.

Each result was evaluated on a scale from 1 to 3. A judgment of 1 is not relevant, 2 is somewhat relevant, and 3 is an exact match.

The measures used for evaluation are: precision at 10 retrieved documents (P@10), mean average precision (MAP), and normalized discounted cumulative gain (NDCG) are commonly used evaluation metrics for web retrieval.

SemSearch Results

Participant	Run	P@10	MAP	NDCG
Yahoo BCN	sub30-RES.3	0.4924	0.1919	0.3137
UMass	sub31-run3	0.4826	0.1769	0.3073
Yahoo BCN	sub30-RES.2	0.4185	0.1524	0.2697
UMass	sub31-run2	0.4239	0.1507	0.2695
Yahoo BCN	sub30-RES.1	0.4163	0.1529	0.2689
U of Delaware	sub28-Okapi	0.4228	0.1412	0.2591
U of Delaware	sub28-AX	0.4359	0.1458	0.2549
UMass	sub31-run1	0.3717	0.1228	0.2272
DERI Galway	sub27-dpr	0.3891	0.1088	0.2172
DERI Galway	sub27-dlc	0.3891	0.1088	0.2171
U of Delaware	sub28-Dir	0.3652	0.1109	0.2140
DERI Galway	sub27-gpr	0.3793	0.1040	0.2106
L3S	sub29	0.2848	0.0854	0.1861
KIT	sub32	0.2641	0.0564	0.1181

Table 3: SemSearch evaluation results

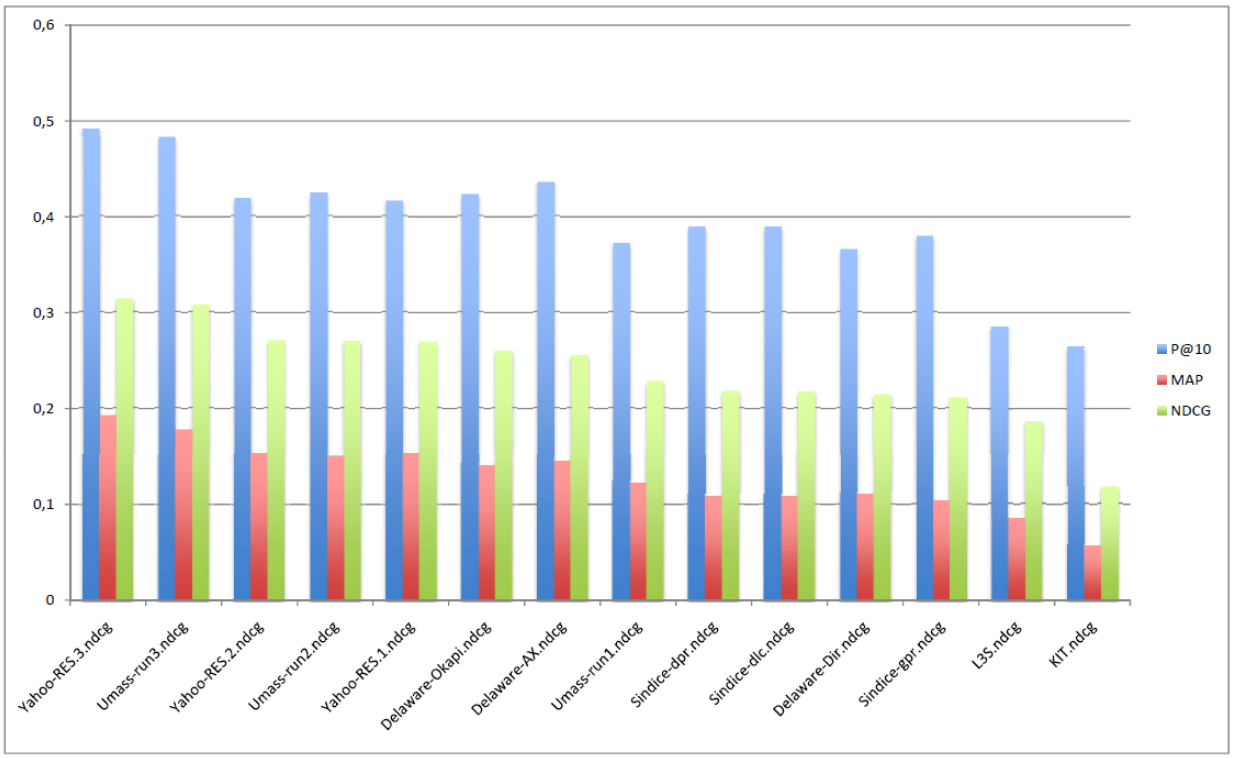

Figure 1: SemSearch entity retrieval evaluation results

The results in Table 3 show that the UMass select types retrieval comes in second across all retrieval metrics and performs comparably to the best run from the conference organizers. For p@10 the sequential dependency full-text model comes in third, and places fourth for both NDCG and MAP. This indicates that it is returning more relevant results in the top 10 than the BCN RES.2 submission.

In the above experiment, the use of selected fields provides an improvement over the text only retrieval runs. The Select-Types run returns 93.8% DBPedia Results. The top performing BCN RES.3 run returns only 40.6% DBPedia results. Interestingly, they achieve similar effectiveness. This indicates that there

likely room for significant improvement by combining the two approaches. We did not perform a run treating DBPedia titles separate from the other field features. Treating the DBPedia field separately appears to boosting those documents over non-DBPedia documents. Consequently, it is unclear whether the use of fields results in improvement or whether the benefit can be attributed to higher rank for DBPedia documents. A similar effectiveness gain might be achieved only boosting DBPedia results. We were unable to perform more detailed analysis of the differences between the runs because the individual query results are not yet available on the workshop website.

PRM-S for mapping keywords to attributes

As described in our models section above, Kim et al. outlines a simple approach for adding structure to keyword queries. PRMS calculates the likelihood that a query word belongs to a candidate document field by aggregating the field text across all documents in the collection. A key assumption in the model is that the collection is homogeneous. In our case, this is false, and we demonstrate the drawbacks of utilizing the method cross a heterogenous collection.

We were not able to submit this run to the workshop because of time constraints and due to problems indexing fields with Indri. However, after the workshop we were able to compute the necessary PRM-S values and manually inspect the keyword to attribute mappings. The results for [david suchet] presented in Table X appear to be representative of results on many of the queries. For brevity, when discussing the attribute names we use only the last portion of the URL.

Using PRMS to compute attribute probabilities for the individual query words illustrates several problems. The first problem is attribute sparsity. For the [david] query term, the most likely attribute, *first*, is problematic. The probability value of 1 indicates that the attribute is homogeneous, with all values in the field being david. Upon inspection, we find that this and many attributes like it contain little text and often occur in few documents. This sparsity problem results in noisy mappings. This could be helped by smoothing from the background collection model, but it would not fix the overall problem. The main issue is that the collection is heterogenous and the model breaks down.

The second problem with PRM-S in this case is that the probability is estimated for each query term independently. For the [suchet] keyword below, the *commander* attribute achieves a high likelihood score because of the French general Louis Gabriel Suchet. Relaxing the independence assumption and utilising sequential dependency relationships and term proximity as done for our text model may improve this problem. However, relaxing these assumptions means that the statistics necessary are no longer directly available from an inverted index and depend on individual occurrences in each document. This makes it prohibitively expensive to compute. Because of the structured nature of the documents, it may also not be as helpful as in unstructured contexts.

Word	Prob	Attribute name
david	1	httpwwwexamplecomnsfirst
david	1	httpdbpediaorgpropertydoh
david	0.356436	httpexampleorgattendancename
david	0.333333	httpdbpediaorgpropertytooltip
david	0.333333	httpdbpediaorgpropertyseconddirector
david	0.333333	httpdbpediaorgpropertyfightchoreography
david	0.333333	httpdbpediaorgpropertydirectorpercentcoltsrugbypercent
david	0.333333	httpdbpediaorgpropertydavid
david	0.333333	httpdbpediaorgpropertycommercialdirector
david	0.333333	httpdbpediaorgpropertyactingorganistanddirectorofmusic

Table 4: Likelihood probabilities for David from David Suchet query

Word	Prob	Attribute name
suchet	2.15E-04	httpdbpediaorgontologyrelatives
suchet	8.56E-05	httpdbpediaorgontologypresenter
suchet	3.79E-05	httpdbpediaorgpropertylifetimeproperty
suchet	2.97E-05	httpdbpediaorgpropertyrelatives
suchet	2.60E-05	httpdbpediaorgpropertycommander
suchet	7.03E-06	httpdbpediaorgpropertybirthname
suchet	5.19E-06	httpdbpediaorgpropertyawards
suchet	2.55E-06	httpwwwpurlorgstuffrevtext
suchet	2.13E-06	httpdbpediaorgpropertystarring
suchet	1.31E-06	httpdbpediaorgpropertybirthdate

Table 5: Likelihood probabilities for Suchet from David Suchet query

Type Matching

Within this corpus many entities have type attributes, also referred to as entity classes. We compute the probability that a type psuedo-document corresponds to a given query. This information allows us to use the *probable-type* model. We show the results for the queries [44 magnum hunting] and [charles darwin].

We can clearly see that there are some promising results here; within the magnum query we have types revolvers, pistols and semi-automatic pistols. The [darwin] query is also promising; Charles Darwin was a life scientist, who lived in Downe until his death. Arguably 3 of the top 4 could contain an entity describing [charles darwin].

We can also see that there is some significant noise present here. 5 of the top 10 types from the first result clearly do not contain an entity describing a [44 magnum hunting]. Similarly we find types relating to the city of Darwin in Australia in the second query. We expect that some of these anomalies are derived from the maximum likelihood estimations that were used to create these probabilities. There are a variety of smoothing techniques that could reduce this problem.

There is also the problem of type resolution. The types referenced here are also entities, that means that they themselves have types. This creates a hierarchy of types. Within a full system it would be useful to propagate probabilities for types up and down this hierarchy. For example within the first query [44 magnum hunting] most of the types returned a classes of firearms. Utilizing this hierarchy would enable us to put a high probability on the more generic [firearm] type. So we can see that this technique could significantly improve type classification, and further improve entity retrieval.

Sadly due to time constraints, and a lack of relevance data we were unable to evaluate the retrieval performance of these queries, nor investigate the type hierarchy. We do hold out hope that these mappings would improve retrieval results over the *select-attribute* model. However without evaluation data we can not be certain.

Object Type	QL Probability
cartridge102971691	1.73E-09
revolvers	7.45E-10
revolver	6.00E-10
rifle	5.31E-10
handgun	6.75E-11
12.7_mm_firearms	4.56E-11
pistol	1.01E-11
shotgun	9.33E-12
semi-automaticpistols	6.38E-12
weapon	4.75E-12

Table 6: Type likelihood results for [.44 magnum hunting]

Object Type	QL Probability
life scientist	3.43E-04
peoplefromdowne	1.83E-04
radiostationsindarwin	9.95E-05
user.coco.science.conpts_theories	7.31E-05
english_eugenicists	4.54E-05
australian_immunologists	2.21E-05
geology_books	1.74E-05
ecuadorian_scientists	1.42E-05
honoraryfellowsofclarehall,cambridge	1.34E-05
biological_literature	8.41E-06

Table 7: Type results for [charles darwin]

Discussion

Entity queries

The power of semantic retrieval lies in modeling the relationships between objects and in the structured representation. However, the entity queries in the Semantic Search workshop mostly consist of a single noun phrase with one or two entities. The main relationship type that appears to be important is location within a city or state. Furthemore, the entity queries usually refer to the name of the entity and not to more structured attributes. Therefore, many of the queries, although common, may be better executed using unstructured retrieval techniques.

Leveraging Ontology Structure

In our experiments, we utilize the leaf nodes for both type and attributes. However, these exist within a larger tree structure defined by an ontology. A key problem for future work is to discover the correct level of granularity to compute the likelihoods for both attribute and types. This was not feasible in the time frame of this project for the given collection because the ontologies were not distributed with the collection. Additionally, the DbPedia ontologies were automatically generated and may need refinement to be useful for retrieval. Lastly, we made no attempts to perform schema matching to collapse attributes and types that are analogous across ontologies. Integrating these schemas would allow us to more accurately estimate type and attribute statistics. Integrating schemas is a well studies are in the DB and Semantic Web communities and outside the scope of this initial work.

Semantic Link Structure

The SemSearch results demonstrate that documents with rich text representations provide better retrieval than short, sparse documents. An important feature of semantic web data is that it is densely connected. Besides containing links to internal resources, documents contain links to external unstructured documents, such as Wikipedia articles and company websites. We investigated links to unstructured documents outside the collection by crawling links to Wikipedia. However we found difficulties in the scale of the crawl, and in how to incorporate the crawled data without introducing noise. This data could possibly be used to perform document expansion and enrich the representation of a semantic entity.

We did not exploit the graph structure defined by RDF links. Web link structure has been successfully exploited by Google and other web search engine to improve retrieval. It is likely that much of that work could be applied to the semantic web. However it should be noted that this data has a very large variety of different link types. Every RDF triple whose <object> is a reference to another object can be considered a link of the type <attribute>.

It is important to note that many of the semantic web tools are limited by their ability to handle large volumes of linked data. In particular, to answer complex queries efficiently they rely on storing the semantic graph in memory. The key problem in producing a truly scalable system lies in the need to traverse the graph structure. This requires a complex indexing that balances the need for disk locality of neighboring nodes, and the need to quickly compute the node statistics traditionally extracted from an inverted index.

Conclusions

As per our initial goals, we competed in the Semantic Search 2010 Workshop and placed second overall out of six teams. The *Select-Types* UMass run is comparable to the best performing submission from the conference organizers. Overall, our results demonstrate that proven retrieval techniques for unstructured text documents are also effective on structured documents. We have shown that using some simple frequent document attributes can improve retrieval performance. We believe one reason for this success is that many of the relevant entities from the query logs are covered by DbPedia, which contain significant textual data from abstracts.

We also performed initial explorations using PRM-S to map keywords to attributes and to predict object classes for queries. The PRM-S attribute predictions do not appear to be a promising technique for this very heterogeneous corpus. We believe that there are at least several main underlying causes for this problem. The first, is very sparse data within rare attributes. The second problem is the bag-of-words independence assumption which is excerbated for entity queries which consist mainly of coherent noun phrases.

Our explorations show that first mapping keywords to object classes appears to hold more promise. We find that a key challenge in this mapping is modeling the data at the correct level of specificity. We believe that there is potential in selecting the appropriate level of class abstraction from ontologies that are neither too general nor too broad. Furthermore, we believe that PRM-S and type prediction could be combined for further improvement. First the entity could be classified to a type, then PRM-S performed over the resulting sub-collection. This blocking approach would reduce the total number of potential keyword to attribute mappings.

Finally, we've shown the limitations of IR Systems in indexing large scale heterogeneous structured data collections, and to exploit link structure within the data. We've also encountered some of the problems of mapping keyword queries to structured query languages in a probabilistic manner. Through these problems we have shown a need to develop ontology-based hierarchical language models for this task. We believe that these types of models should vastly improve our ability to use types and attributes within semi-structured queries.

References

S. Agrawal, S. Chaudhuri, and G. Das. Dbxplorer: A system for keyword-based search over relational databases. *ICDE'02*.

S. Agrawal, K. Chakrabarti, S. Chaudhuri, V. Ganti, A. C. Konig, and D. Xin. Exploiting web search engines to search structured databases. In *WWW* Conference, pages 501–510, 2009.

Chen, Y., Wang, W., Liu, Z., and Lin, X. 2009. Keyword search on structured and semi-structured data. In *Proceedings of the 35th SIGMOD international Conference on Management of Data*. ACM, New York, NY, 1005-1010.

E. Chu, A. Baid, T. Chen, A. Doan, and J. Naughton. A relational approach to incrementally extracting and querying structure in unstructured data. In *VLDB*, 2007.

G. Demartini, A. P. de Vries, T. Iofciu, and J. Zhu. Overview of the INEX 2008 entity ranking track. In *This volume*, 2009.

Kim, J., Xue, X. and Croft, W. B. , A Probabilistic Retrieval Model for Semistructured Data, *ECIR* 2009, PP. 228-239.

Paparizos, S., Ntoulas, A., Shafer, J., and Agrawal, R. 2009. Answering web queries using structured data sources. *SIGMOD* '09. ACM, New York, NY, 1127-1130.

Pound, J., Mika, P., Zaragoza, H. Ad-hoc Object Ranking in the Web of Data. In *WWW Conference*, 2010.

Madhavan, J. , Jeffery S., Cohen, S., Dong, X., Ko, D., Yu, C., and Halevy, A. Web-scale data integration: you can only afford to Pay As You Go. In *CIDR*, 2007.

Madhavan, J. , Halevy, A, Cohen, S., Dong, X., Jeffery S., Ko, D., and Yu, C. Structured Data Meets the Web: A few Observations. *IEEE Data Eng. Bull*. 29, 4 (2006). Page 19-26.

Metzler, D. , Croft, W. B. , A Markov Random Field Model for Term Dependancies. 28th ACM SIGIR, 2005.