

Relevance Feedback Between Hypertext and Semantic Web Search

Harry Halpin
Institute for Communicating and Collaborative
Systems
University of Edinburgh
10 Crichton St.
Edinburgh, United Kingdom
H.Halpin@ed.ac.uk

Victor Lavrenko
Institute for Communicating and Collaborative
Systems
University of Edinburgh
10 Crichton St.
Edinburgh, United Kingdom
vlavrenk@inf.ed.ac.uk

ABSTRACT

We investigate the possibility of using structured data to improve unstructured search. In particular, we use relevance feedback to create a ‘virtuous cycle’ between structured data gathered from the Semantic Web of Linked Data and unstructured gathered from the hypertext Web. Previous approaches have generally considered the searching over the Semantic Web and hypertext Web to be entirely disparate, indexing and searching over different domains. While relevance feedback has traditionally improved information retrieval performance, relevance feedback is normally used to improve rankings over a single data-set. Our novel approach is to use relevance feedback from hypertext Web results to improve Semantic Web search, and results from the Semantic Web to improve the retrieval of hypertext Web data. In both cases, an evaluation based on certain kinds of informational queries (abstract concepts, people, and places) selected from a real-life query log and checked by human judges. We evaluate our work over a wide range of algorithms and options, and show it improves baseline performance on these queries for deployed systems as well, such as the Semantic Web Search engine FALCON-S and Yahoo! Web search. We further show that the use of Semantic Web inference seems to hurt performance, while the pseudo-relevance feedback increases performance in both cases, although not as much as actual relevance feedback.

1. INTRODUCTION

There has recently been a return of interest in combining structured and unstructured search. This seems at least partially inspired by the Linked Data initiative, which has released a massive amount of public structured data on the Web from a diverse range of sources in common Semantic Web formats like RDF [10], which has led to the rise of specialized Semantic Web search engines. The hypothesis put forward by Baeza-Yates is that the search for structured data - called ‘Semantic Search’ - can be used to improve traditional ad-hoc information retrieval for hypertext Web search engines [1], and that techniques from information retrieval can be used to improve the search for structured data. While we realize the amount and sources of structured data on the Web are huge, to restrict and test the hypothesis of Baeza-Yates, from hereon we will assume from that ‘structured search’ (also called ‘semantic search’ or more precisely, ‘Semantic Web search’) refers to indexing and retrieving of Linked data by search engines like Sindice and FALCON-S [4], and hypertext search refers to the indexing and retrieval of hypertext

Copyright is held by the author/owner(s).

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

documents on the World Wide Web by search engines like Google and Yahoo! Search. We also assume a traditional, keyword-based ad-hoc information retrieval paradigm for both kinds of search as both kinds of search engines support this paradigm.

We are the first to suggest that relevance feedback may be the primary method for creating a ‘virtuous cycle’ between structured and unstructured search. Relevance feedback usually improves information retrieval performance, but almost always the feedback is used to improve rankings over a single source of data. Our novel approach is to use relevance feedback from hypertext Web search, of which there is a massive amount of data available to commercial search engines, to improve the retrieval of structured Semantic Web data, since Semantic Web data has little relevance feedback data itself. Then more interestingly, we attempt to run the relevance feedback in reverse: Assuming we have structured Semantic Web data, can it be used to improve hypertext search? Previous approaches to searching structured data on the Semantic Web like FALCON-S have assumed that the Semantic Web and the hypertext Web search to be entirely disparate, indexing and searching them differently [4], although structured search engines like Sindice that index microformat are blurring this distinction [15].

2. SYSTEM DESIGN

In order to deal with these problems, we will employ *relevance feedback*, the use of explicit relevance judgements from users of a query in order to expand the query. By ‘expand the query,’ we mean that the usually rather short query is expanded into a much larger query by adding words from known relevant documents. For example, a query on the hypertext Web for the Eiffel Tower given as “eiffel” might be expanded into “paris france eiffel tour.” The same principle applies to the Semantic Web. If the relevant pages instead were about an Eiffel Tower replica in Texas, the same results query could be expanded into “paris texas eiffel replica.” In our experiment, the selection of a document by a user and their staying on the document for some period of time is a sign of relevance. The hypothesis of relevance feedback, as pioneered by Rocchio in the SMART retrieval system, is that the relevant documents will disambiguate and in general give a better description of the information need of the query than the query itself [17]. Relevance feedback has been shown in certain cases to improve retrieval performance significantly, both in early studies and in later work like relevance modelling that creates relevance directly from the indexed documents rather than explicitly waiting for the user to make a relevance judgement [11].

Semantic Web search engines exist that specifically index and return ranked Linked Data in RDF (Resource Description Frame-

work, a W3C standard for structured Web data [10]) in response to keyword queries, but their rankings are much less well-studied than hypertext Web rankings, and so are thought likely to be sub-optimal. However, in general Semantic Web search suffers from a lack of a thorough and neutral Cranfield-style evaluation, and we carefully employ the standard information retrieval evaluation in our experiment, which is a first in searching on the Semantic Web.

Our novel contribution to ranking structured Semantic Web data is to use selected hypertext web-pages as relevance feedback for improving the ranking of Semantic Web data. In our system, the query is run first against the hypertext Web search engine and we collect relevance judgements from the results. We then use these judgements to expand the query with certain words from the relevant documents. The expanded query is used to re-rank the results retrieved by a Semantic Web search engine. We can compare both structured semantic and unstructured hypertext data by considering both to be an unstructured ‘bags of words.’ Semantic Web data can be flattened, and URIs can be reduced to ‘words’ by the following steps:

- Reduce to rightmost hierarchical component.
- If the rightmost component contains a fragment identifier (#), consider all characters right of the fragment the rightmost hierarchical component.
- Tokenize rightmost component on space, capitalization, and underscore.

So, the URI `http://www.example.org/hasArchitect` would be reduced to two tokens, ‘has’ and ‘architect.’ Of immense interest for hypertext search, we can then run the process backwards, using relevant Semantic Web data as relevance feedback to improve hypertext Web search. This is not unfeasible, as one could consider the consumption of Semantic Web data by a program to be a judgement of relevance.

3. IS THERE ANYTHING WORTH FINDING ON THE SEMANTIC WEB?

The main problem confronting of any study of the Semantic Web is one of *sampling*. As almost any large-data database can easily be exported to RDF, statistics demonstrating the actual deployment of the Semantic Web can be biased by the automated release of large, if useless, data-sets, the equivalent of ‘Semantic Web’ spam. Also, large specialized databases like Bio2RDF can easily dwarf the rest of the Semantic Web in size. A more appropriate strategy would be to try to answer the question: What information is available on the Semantic Web that users are actually interested in? The first large-scale analysis of the Semantic Web was done via an inspection of the index of Swoogle by Ding and Finin [5]. The primary limitation of that study was that the large majority of the Semantic Web resources sampled did not contain rich information that many people would find interesting. For example, the vast majority of data on the Semantic Web in 2006 was Livejournal exporting every user’s profile as FOAF and RSS 1.0 data that used Semantic Web techniques to structure the syntax of news feeds. Yet with information-rich and interlinked databases like Wikipedia being exported to the Semantic Web, today the Semantic Web may contain information needed by actual users.

3.1 Inspecting the Semantic Web

In order to select real queries from users for our experiment, we used the query log of a popular hypertext search engine, the

Web search query log of approximately 15 million distinct queries from Microsoft Live Search. This query log contained 6,623,635 unique queries corrected for capitalisation. The main issue in using a query log is to get rid of navigational and transactional queries. A straightforward gazetteer-based and rule-based named entity recogniser was employed to discover the names of people and places [14], based off a list of names maintained by the Social Security Administration and a place name database provided by the Alexandria Digital Library Project. From the query log a total of 509,659 queries were identified as either people or places by the named-entity recogniser, and we call these queries *entity queries*. Employing WordNet to represent abstract concepts, we chose queries recognised by WordNet that have *both* a hyponym and hypernym in WordNet. This resulted in a more restricted 16,698 queries that are supposed to be about abstract concepts, which we call *concept queries*.

The results of running the crawled queries against a Semantic Web search engine, FALCON-S’s Object Search [4], were surprisingly fruitful. For entity queries, there was an average of 1,339 URIs (S.D. 8,000) returned for each query. On the other hand, for concept queries, there were an average of 26,294 URIs (S.D. 14,1580) returned per query, with no queries returning zero documents. As shown in Figure 1, when plotted in logarithmic space, both entity queries and concept queries show a distribution that is heavily skewed towards a very large number of high-frequency results, with a steep drop-off to almost zero results instead of the characteristic long tail of a power law. For the vast majority of queries, far from having no information, the Semantic Web of Linked Data appears to have *too much data*. For a more fine-grained analysis of domain names and sources of data, see our previous work [7].

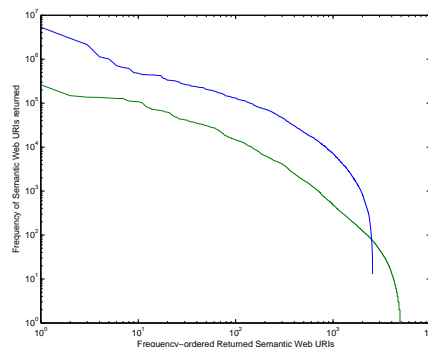


Figure 1: The rank-ordered frequency distribution of the number of URIs returned from entity and concept queries, with the entity queries given by green and the concept queries by blue.

Another question is whether or not there is any correlation between the amount of URIs returned from the Semantic Web and the popularity of the query. As shown by Figure 2, there is *no* correlation between the amount of URIs returned from the Semantic Web and the popularity of the query. For entity queries, the correlation coefficient was 0.0077, while for concept queries, the correlation coefficient was still insignificant, at 0.0125. The popularity of query is not related to how much information the Semantic Web possesses on the information need expressed by the query: Popular queries may have little data, while infrequent queries may have a lot. This is likely due to the rapidly changing and event-dependent nature of hypertext Web queries versus the Semantic Web’s preference for more permanent and less temporally-dependent data.

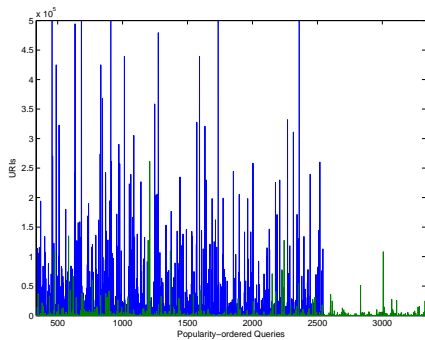


Figure 2: The rank-ordered popularity of the queries is on the x -axis, with the y axis displaying the number of Semantic Web URIs returned, with the entity queries given by green and the concept queries by blue.

1	ashville north carolina
2	harry potter
3	orlando florida
4	ellis college
5	university of phoenix
6	keith urban
7	carolina
8	el salvador
9	san antonio
10	earl may

Table 1: 10 Selected Entity Queries

3.2 Selecting Queries for Evaluation

In order to select a subset of informational queries for evaluation, we randomly selected 100 queries identified as abstract concepts by WordNet and then 100 queries identified as either people or places by the named entity recogniser, for a total of 200 queries to be used in evaluation. Constraints were placed on crawled URIs, such that at least 10 Semantic Web documents (the structured data retrieved by the Semantic Web URI) were crawled for each query, leading to a total of 1,000 structured Semantic Web documents about entities and 1,000 Semantic Web documents about concepts, for a total of 2,000 Semantic Web documents for relevance judgements. Then, the same experimental query log was used to crawl the hypertext Web using Yahoo! search, resulting in the same number of web-pages about concepts and entities for relevance judgements. A random selection of ten queries from the entity queries is given in Table 1 and another random selection of ten concept queries is given in Table 2. As one can tell, the queries about entities and concepts are spread across quite diverse domains, ranging from entities about both locations (El Salvador) and people (both fictional such as Harry Potter and non-fictional such as Earl May) to concepts about a whole range of abstraction, from sociology to ale.

3.3 Relevance Judgements

For each of the 200 experimental queries selected in Section 3.2, 10 hypertext web-pages and 10 Semantic Web documents need to be judged for relevance, leading to a total of 4,000 human judgements for relevance for our entire experiment. Human judges each judged 25 queries presented in a randomized order, and were given a total of 3 hours to judge the entire sample for relevancy. No researchers were part of the rating. The judges were each presented

131	sociology
133	clutch
134	telephone
135	ale
136	pillar
137	sequoia
138	aster
139	bedroom
140	tent
141	cinch

Table 2: 10 Selected Concept Queries

first with ten hypertext web-pages and then with ten Semantic documents, leading to a total of 20 judgements per query per judge. Each result therefore was judged by three judges, with a total of 24 judges were used in the entire experiment. Before starting judging, the judges were given instructions and trained on 10 sample results (5 web-pages and 5 Semantic Web documents). The human judges were forced to make binary judgements of relevance, so each result must be either relevant or irrelevant to the query. In their instructions, relevance was defined as *whether or not a result is about the same thing as the query, which can be determined by whether or not accurate information about the information need is expressed by the result*. A number of kinds of Web results that would ordinarily be considered relevant are therefore excluded. In particular, there is a restriction that the relevant information must be present in the result itself. This excludes possibly relevant information that is accessible via outbound links, even a single link. All manner of results that are collections of links are excluded from relevancy, including both ‘link farms’ purposely designed to be highly ranked by page-rank based search engines, as well as legitimate directories of high-quality links to relevant information. These hubs are excluded precisely because the information, even if it is only a link transversal away, is still not directly present in the retrieved result. By this same principle, results that merely redirect to another resource via some method besides the standardized HTTP methods are excluded, since a redirection can be considered a kind of link. They would be considered relevant only if additional information was included in the result besides the redirection itself.

In order to aid the judges, a Web-based interface was created to present the queries and results to the judges. Although an interface that presented the queries and the search interface in a manner similar to search engines was created, human judges preferred an interface that presented them the results for judgements one-at-a-time, forcing them to view a rendering of the web-page associated with each URI originally offered by the search engine. For each hypertext web-page, the web-page was rendered using the Firefox Web Browser and PageSaver Pro 2.0. For each Semantic Web document, the result was rendered (i.e. the triples, any associated text in the subject, and any associated Semantic Web document) by using the open-source Disco Hyperdata Browser with Firefox.¹ In both cases, the resulting rendering of the Web representation was saved at 469×631 pixel resolution. The reason that the web-page was rendered instead of a link given directly to the URI is because of the unstable state of the Web, especially the hypertext Web. Even caching the HTML would have risked losing much of the graphic element of the hypertext Web. By creating ‘snapshot’ renderings, each judge at any given time was guaranteed to be presented with

¹The Disco Hyperdata Browser, a browser that renders Semantic Web data to HTML, is available at <http://www4.wiwiw.fu-berlin.de/bizer/ng4j/disco/>.

the result in the same visual form. One side-effect of this is that web-pages that heavily depend on non-standardised technologies or plug-ins would not render and were thus presented as blank screen shots to the user. The user-interface broke the evaluation into two steps:

- *Judging relevant results from a hypertext Web search:* The judge was given the search terms created by an actual human user for a query and an example relevant web-page whose full snapshot could be viewed by clicking on it. A full rendering of the retrieved web-page was presented to the user with its title and summary (as produced by Yahoo! Search) easily viewed by the judge as in Figure 3. The judge clicked on the check-box if the result is considered relevant. Otherwise, the web-page was by default recorded as not relevant. The web-page results were presented to the judge one at a time, ten times for each query.
- *Judging relevant results from a Semantic Web search:* Next, the judge assessed all the Semantic Web results for relevancy. These results were retrieved from the Semantic Web using the same interface displayed to the judge in Step 1 as shown in Figure 4, and a title was displayed by retrieving any literal values from `rdfs:label` properties and a summary by retrieving any literal values from `rdfs:comment` values. Using the same interface as in Step 1, the judge had to determine whether or not the Semantic Web results were relevant.

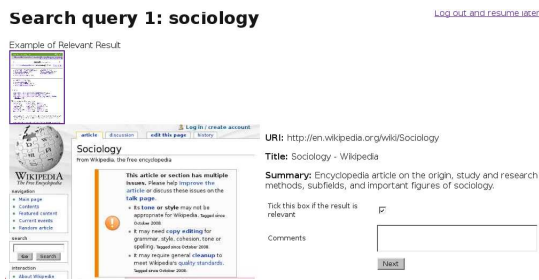


Figure 3: The interface used to judge web-page results for relevancy.

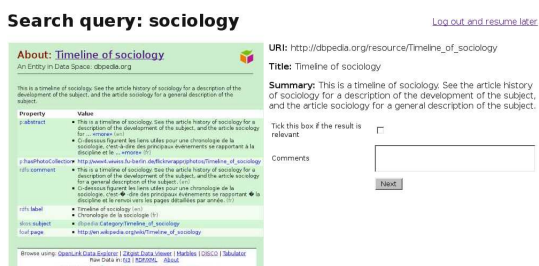


Figure 4: The interface used to judge Semantic Web results for relevancy

After the ratings were completed, Fleiss's κ statistic was taken in order to test the reliability of inter-judge agreement on relevancy judgements [6]. Simple percentage agreement is not sufficient, as it does not take into account the likelihood of purely coincidental agreement by the judges. Fleiss's κ both corrects for chance

Results:	Hypertext	Semantic Web
Resolved:	197 (98%)	132 (66%)
Unresolved:	3 (2%)	68 (34%)
Top Relevant:	121 (61%)	76 (58%)
Top Non-Relevant:	76 (39%)	56 (42%)

Table 3: Results of Hypertext and Semantic Web Search Relevance Judgements

agreement and can be used for more than two judges [6]. The null hypothesis is that the judges cannot distinguish relevant from irrelevant results, and so are judging results randomly. Overall, for both relevance judgements over Semantic Web results and web-page results, $\kappa = 0.5724$ ($p < .05$, 95% Confidence interval [0.5678, 0.5771]), indicating the rejection of the null hypothesis and 'moderate' agreement. For web-page results only, $\kappa = 0.5216$ ($p < .05$, 95% Confidence interval [.5150, 0.5282]), also indicating the rejection of the null hypothesis and 'moderate' agreement. Lastly, for only Semantic Web results, $\kappa = 0.5925$ ($p < .05$, 95% Confidence interval [0.5859, 0.5991]), also indicating the rejection of the null hypothesis is to be rejected and 'moderate' agreement. So, in all cases there is 'moderate' agreement, which is sufficient given the general difficulty of producing perfectly reliable relevancy judgements. Interestingly enough, the difference in κ between the web-page results and Semantic Web results show that the judges were actually *slightly* more reliable in their relevancy judgements of information from the Semantic Web rather than the hypertext Web. This is likely due to the more widely varying nature of the hypertext results, as compared to the more consistent informational nature of Semantic Web results.

Were judges more reliable with entities or concepts? Recalculating the κ for all results based on entity queries, $\kappa = 0.5989$ ($p < .05$, 95% Confidence interval [0.5923, 0.6055]), while for all results based on concept queries was $\kappa = 0.5447$ ($p < .05$, 95% Confidence interval [0.5381, 0.5512]). So it appears that judges are slightly more reliable discovering information about entities rather than concepts, backing the claim made by Hayes and Halpin that there is more agreement in general about 'less' abstract things like people and places rather than abstract concepts [9]. However, agreement is still very similar and 'moderate' for both information about entities and concepts.

For the queries, much of the data is summarised in Table 3. **Resolved** queries are queries that return at least one relevant result in the top 10 results, while **unresolved** queries are queries that return no relevant queries in the top 10 results. 'Hypertext' means that the result was taken only over the hypertext Web results and 'Semantic Web' indicates the same for the Semantic Web results. The percentages for resolved and unresolved for 'hypertext' and 'Semantic Web' were taken over the hypertext and Semantic Web relevancy corpora in order to allow direct comparison. The percentages for 'Top Relevant' and 'Non-Relevant Top' were computed as percentages over all relevant queries, and so excludes unresolved queries.

For both hypertext and Semantic search, there were 71 (18%) unresolved queries that did not have any results. For the hypertext Web search, only 3 (2%) queries were unresolved, while 68 (34%) of the queries were unresolved for the Semantic Web. This simply means that the hypertext search engines almost always returned at least one relevant results in the top 10, but that for the Semantic Web almost a third of all queries did not return any relevant result in the top 10. This only means there is much information that does not yet have a relevant form on the Semantic Web, unless it is hidden by the perhaps sub-optimal ranking by FALCON-S.

Another question is how many queries had a relevant result as their top result? In general, 197 queries (50%) had top-ranked relevant results over both Semantic Web and hypertext search. While the hypertext Web search had 121 (61%) top-ranked relevant results, the Semantic Web only had 76 (58%) top-ranked results. What is more compelling for relevance feedback is the number of relevant results that were *not* the top-ranked result. Again for both kinds of searches, there were 132 (33.0%) queries where a relevant result was *not* in the top position of the returned results. For the hypertext Web, there were 76 (39%) queries with a top non-relevant result. Yet for the Semantic Web there were 56 (42%) queries that had a top non-relevant result. So queries on the Semantic Web are more likely to turn up no relevant results in the top 10. When a relevant query is returned in the top 10 results it is quite likely that a non-relevant result will be in the top position for both the hypertext Web and the Semantic Web.

3.4 Information Retrieval using Language Models

For the relevance feedback methods used in this paper, we employ the *language model* approach as it a formally principled and probabilistic approach to determining the ranking and weighting function. For those interested in *vector space models*, the results using vector-space models are reported in previous work [8], and the results are in general fairly similar. In early language modelling efforts [16], the probability that the language model of a document would generate the query was the comparison function of the document. A more sophisticated approach to language model considers that the query was a sample from an underlying *relevance model* of unknown relevant documents, but that the model could be estimated by computing the co-occurrence of the query terms with every term in the vocabulary. In this way, the query itself was just considered a limited sample, so the it is automatically expanded before the search has even begun by re-sampling the underlying relevance model. A number of different options for the parameters of each weighting function, and the appropriate ranking function, will be considered.

3.4.1 Representation

Language modelling frameworks in information retrieval represent each document as a language model given by an underlying multinomial probability distribution of word occurrences. Thus, for each word $w \in V$ there is a value that gives how likely an observation of word w is given D , i.e. $P(w|u_D(v))$ [16]. The document model distribution $u_D(v)$ is then estimated using the parameter λ_D , which allows a linear interpolation that takes into account the background probability of observing w in the entire collection C . This is given in Equation 1.

$$u_D(w) = \lambda_D \frac{n(w, D)}{|D|} + (1 - \lambda_D) \frac{n(w, C)}{\sum_{v \in V} n(v, C)} \quad (1)$$

The parameter λ_D just takes into account the relative likelihood of the word as observed in the given document D compared to the word given the entire collection of documents C . $|D|$ is the total number of words in document D , while $n(w, D)$ is the frequency of word d in document D . Further, $n(w, C)$ is the frequency of occurrence of the word w in the entire collection C divided by the occurrence of all words v in collection C .

3.4.2 Language Modeling Baseline

When no relevance judgements are available, the language modeling approach ranks documents D by the probability that the query

Q could be observed during repeated random sampling from the distribution $u_D(\cdot)$. The typical sampling process assumes that words are drawn independently, with replacement, leading to the following retrieval score being assigned to document D :

$$P(Q|D) = \prod_{q \in Q} u_D(q) \quad (2)$$

The ranking function in Equation 2 is called *query-likelihood* ranking and is used as a baseline for our language-modeling experiments.

3.4.3 Language Models and Relevance Feedback

The classical language-modeling approach to IR does not provide a natural mechanism to perform relevance feedback. However, a popular extension of the approach involves estimating a relevance-based model u_R in addition to the document-based model u_D , and comparing the resulting language models using information-theoretic measures. Estimation of u_D has been described above, so this section will describe two ways of estimating the relevance model u_R , and a way of measuring distance between u_Q and u_D for the purposes of document ranking.

Let $R = r_1 \dots r_k$ be the set of k relevant documents, identified during the feedback process. One way of constructing a language model of R is to average the document models of each document in the set:

$$u_{R,avg}(w) = \frac{1}{k} \sum_{i=1}^k u_{r_i}(w) = \frac{1}{k} \sum_{i=1}^k \frac{n(w, r_i)}{|r_i|} \quad (3)$$

Here $n(w, r_i)$ is the number of times the word w occurs in the i 'th relevant document, and $|r_i|$ is the length of that document. Another way to estimate the same distribution would be to *concatenate* all relevant documents into one long string of text, and count word frequencies in that string:

$$u_{R,con}(w) = \frac{\sum_{i=1}^k n(w, r_i)}{\sum_{i=1}^k |r_i|} \quad (4)$$

Here the numerator $\sum_{i=1}^k n(w, r_i)$ represents the total number of times the word w occurs in the concatenated string, and the denominator is the length of the concatenated string. The difference between Equations 3 and 4 is that the former treats every document equally, regardless of its length, whereas the latter favors longer documents (they are not individually penalized by dividing their contributing frequencies $n(w, r_i)$ by their length $|r_i|$).

3.4.4 Comparison Function: Cross Entropy

We now want to re-compute the retrieval score of document D based on the estimated language model of the relevant class u_R . What is needed is a principled way of comparing a relevance model u_R against a document language model u_D . One way of comparing probability that has shown the best performance in empirical information retrieval research [12] is cross entropy. Intuitively, cross entropy is an information-theoretic measure that measures the average number of bits needed to identify the probability of distribution p being generated if p was encoded using given probability distribution q rather than q itself. For the discrete case this is defined as:

$$H(p, q) = - \sum_x p(x) \log(q(x)) \quad (5)$$

If one considers that the $u_R = p$ and that document model distribution $u_D = q$, then the two models can be compared directly

using cross-entropy, as shown in Equation 6. This use of cross entropy also fulfills the Probability Ranking Principle and so is directly comparable to vector-space ranking via cosine [12].

$$-H(u_R|u_D) = \sum_{w \in V} u_R(w) \log u_D(w) \quad (6)$$

Note that either the *averaged* relevance model $u_{R,avg}$ or the *concatenated* relevance model $u_{R,con}$ can be used in Equation 6. We refer to the former as *rm* and to the latter as *tf* in the following experiments.

4. FEEDBACK EVALUATION

We evaluated the language models described in Section 3.4 on queries selected in Section 3.2 with relevance judgements on these queries selected in Section 3.3. First we inspect relevance feedback from hypertext web-pages improving Semantic Web document rankings, and then the reverse.

4.1 Hypertext to Semantic Web Feedback

4.1.1 Results

A number of parameters for our system were evaluated to determine which parameters provide the best results. For each of the parameter combinations, we compared the use of relevance feedback to a baseline system which did not use relevance feedback, yet used the same parameters with the exception of any relevance feedback-related parameters. The baseline system without feedback can also be considered an unsupervised algorithm, while a relevance feedback system can be thought of as a supervised algorithm. For example, the relevant hypertext web-pages R can be considered to be training data, while the Semantic Web data D we wish to re-rank can be considered to be test data. The hypertext web-pages and Semantic Web data are disjoint sets ($D \cap R = \emptyset$). For evaluation we used mean average precision (MAP) with the standard Wilcoxon sign-test, which we will often just call ‘average precision.’

Both averaged relevance models *rm* and concatenated relevance models *tf* were investigated, with the primary parameter being m , the number of non-zero probability words used in the relevance model. The parameter m was varied between 100,300,1000,3000, and 10000. Remember that the query model *is* the relevance model for the language model-based frameworks. As is best practice in relevance modelling, the relevance models were not smoothed, but a number of different smoothing parameters for ϵ were investigated for the cross entropy comparison function, ranging from ϵ between 0.01,0.10,0.20, 0.50,0.80,0.90, and 0.99. The results are given in Figure 5.

The highest performing language model was *tf* with a cross-entropy ϵ of .2 and a m of 10,000, which produced an average precision of 0.8611, which was significantly higher than the language model baseline of 0.5043 ($p < .05$) using again a m of 10,000 for document models with a cross entropy ϵ of .99. Rather interestingly, *tf* always outperformed *rm*, and *rm*’s best performance had a MAP of 0.7223 using an ϵ of .1 and a m of 10,000.

4.1.2 Discussion

One observation is in order; note that for language models *tf* always outperformed *rm*. The primary difference is that *rm* normalise by documents length: *rm* constructs a relevance model on a per-relevant document basis before creating the average relevance model. In contrast, *tf* does not normalise: *tf* constructs a relevance model by combining all the relevance documents and then

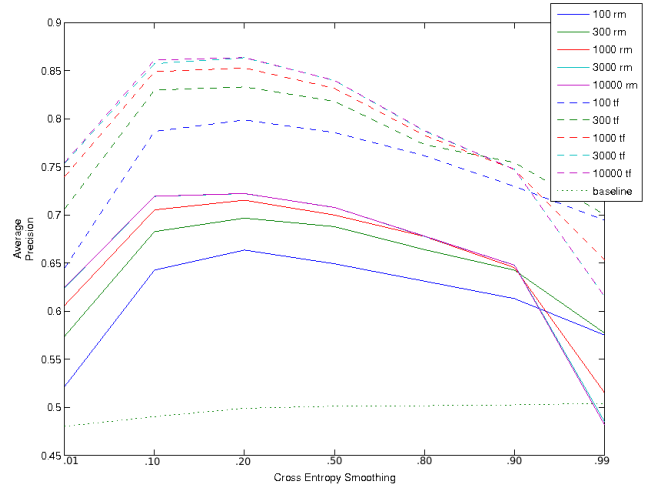


Figure 5: Average Precision Scores for Language Model Parameters: Relevance Feedback From Hypertext to Semantic Web

creating the relevance model from the *raw pool* of all relevant document models. Thus it appears the answer is that any kind of normalisation by the length of the document hurts performance. The reason for this is likely because the text automatically extracted from hypertext documents is ‘messy,’ being of low quality and bursty, with highly varying document lengths. As observed earlier [15], the amount of triples in Semantic Web documents follow a power-law, so there are wildly varying document lengths of both the relevance model and the document models. It appears that document length is a good predictor of relevance on the Semantic Web. Due to these factors, it is unwise to normalise the models, as that will almost certainly dampen the effect of document length and maybe even crucial keywords (such as ‘alien’ and ‘fiction’ in disambiguating a ‘sf’ query).

4.2 Semantic Web to Hypertext Feedback

In this section, we assume that the user or agent program has somehow accessed or otherwise examined the Semantic Web document from URIs, and these Semantic Web documents then form a relevant document model that is compared to hypertext documents in order to produce rankings. In this way, the feedback cycle has been reversed, so we can test if relevant structured Semantic Web documents can improve unstructured hypertext search.

4.2.1 Results

The results for using Semantic Web documents as relevance feedback for hypertext Web search are surprisingly promising. The results for language modelling were similar to the results in Section 5 and are given in Figure 6, although a few differences are worth comment. The best performing language model was *tf* with m of 10,000 and a $\epsilon=.2$, which produced an average precision of 0.6549 ($p < .05$). In contrast, the best-performing *rm*, with a m of 3,000 and $\epsilon=.5$, only had an average precision of 0.4858 ($p < .05$). The *tf* relevance models consistently performed better than *rm* relevance models ($p < .05$). The baseline for language modelling was also fairly poor with an average performance of 0.4284 ($p < .05$). This was the ‘best’ baseline using again a m of 10,000 for document models and cross entropy smoothing ϵ of .99. The general

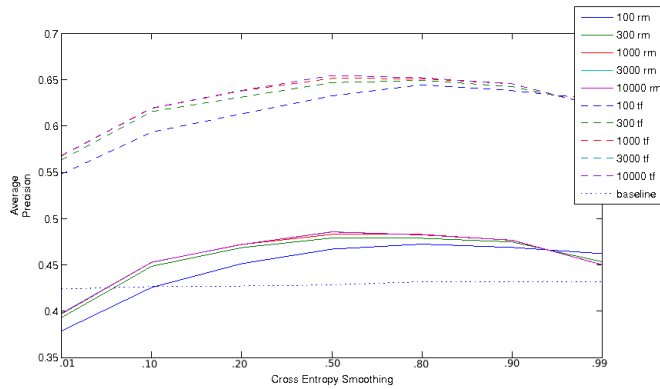


Figure 6: Average Precision Scores for Language Model Parameters: Relevance Feedback From Hypertext to Semantic Web

trends from the previous experiment then held, except the smoothing factor was more moderate and the difference between tf and rm was even more pronounced.

4.2.2 Discussion

Why is tf relevance modelling allow data from the Semantic Web to boost hypertext Results? Rather shockingly, as the Semantic Web data is mostly manually high-quality curated data from sources like DBpedia, so the actual natural language fragments on the Semantic Web (found for example in Wikipedia abstracts) are much better samples of natural language than the natural language samples found in hypertext. Furthermore, the distribution of ‘natural’ language terms extracted from RDF terms (such as ‘sub class of’ from `rdfs:subClassOf`), while often irregular, will either be repeated very heavily or fall into the sparse long tail. Lastly, it appears once again that document length is a good predictor of relevance on the hypertext Web. So, it is on some level not surprising that even hypertext Web search results can be improved by Semantic Web data, because used in combination with the right relevance feedback parameters, in essence the hypertext search engine is being ‘seeded’ with a high-quality accurate description of the information need expressed by the query to be used for query expansion.

5. PSEUDO-FEEDBACK

A well-known technique within relevance feedback is *pseudo-feedback*, namely simply assuming that the top x documents returned are relevant. Then, one can use this as a corpus of relevance documents to expand the queries in the same manner using language models as described in Section 3.4. Using the same optimal parameters as discovered in Section , tf with a $m = 10,000$ and $\epsilon = .2$ was again deployed, but this time using pseudo-feedback. Can pseudo-feedback from hypertext Web search help improve the rankings of Semantic Web data? The answer is clearly positive. Employing all ten results as pseudo-relevance feedback and the same previously optimized parameters, the best pseudo-relevance feedback result had an average precision of 0.6240. This was considerably better than the baseline of just using relevance pseudo-feedback from the Semantic Web to itself, which only had an average precision of 0.5251 ($p < .05$), and also clearly above the ‘best’ baseline of 0.5043 ($p < .05$). However, as shown by Figure 7, the results are still not nearly as good as using actual relevant hypertext pages, which had an average precision of 0.8611 ($p < .05$). This is

likely because, not surprisingly, the hypertext Web results contain many irrelevant queries that serve as noise, preventing the relevant feedback from boosting the results.

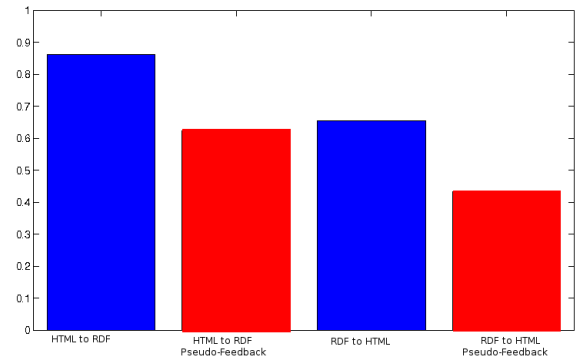


Figure 7: Comparing Relevance Feedback (red) to Relevance Feedback (blue) on the Semantic Web (RDF) and Hypertext Web (HTML)

Can pseudo-feedback from the structured Semantic Web improve hypertext search? The answer is yes, but barely. The best result for average precision is 0.4321 ($p < .05$), which is better than the baseline of just using pseudo-feedback from hypertext Web results to themselves, which has an average precision of 0.3945 ($p < .05$) and the baseline without feedback at all of 0.4284 ($p < .05$). However, the pseudo-feedback results are both still significantly worse performance by a large margin than using relevant Semantic Web data, which had an average precision of 0.6549 ($p < .05$). These results can be explained because, given the usual ambiguous and short one or two word queries, the Semantic Web tends to return structured data spread out of over multiple subjects even moreso than the hypertext Web. Therefore, adding pseudo-relevance feedback increases the amount of noise in the language model as opposed to using actual relevance feedback, hurting performance while still keeping it above baseline.

6. INFERENCE

One of the characteristics of structured data in general is that the structure should allow one - at least ‘in theory’ - to discover more relevant data. The Semantic Web formalizes this in terms of type and sub-class hierarchies in RDF using RDF Schema [3]. While inference routines are quite complicated in theory as regards the various Semantic Web specifications, as shown in detail by our survey of Linked Data done over query logs [7], in practice the vast majority of actual inference that can be used is on the Semantic Web is of two types, *rdfs:subClassOf*, which indicates a simple sub-class inheritance hierarchy, and *rdf:type*, which indicates a simple type. For our experiment, we followed all explicit *rdfs:subClassOf* statements up one level in the sub-class hierarchy and explicit *rdf:type* links. The resulting retrieved Semantic Web data was all concatenated together, and then concatenated yet again with their source structured document from the Semantic Web. In this way, Semantic Web inference was used as *document expansion*.

Inference was first tried using normal relevant feedback, again with the same best-performing parameters (tf with $m = 10,000$ and $\epsilon = .2$). In the first case, the inference is used to expand the Semantic Web data, and then the hypertext results are used as relevance feedback. However, as shown in Figure 8, deploy-

ing inference only causes a drop in performance. In particular, using hypertext Web results as relevance feedback to the Semantic Web, the system drops from a performance of 0.8611 to a performance of 0.4991 ($p < .05$). With pseudo-feedback over the top 10 documents, the performance drops even more, from 0.6240 to 0.4557 ($p < .05$). The use of inference actually makes the results worse than the baseline performance of language models of 0.5043 ($p < .05$) without either relevance feedback or inference.

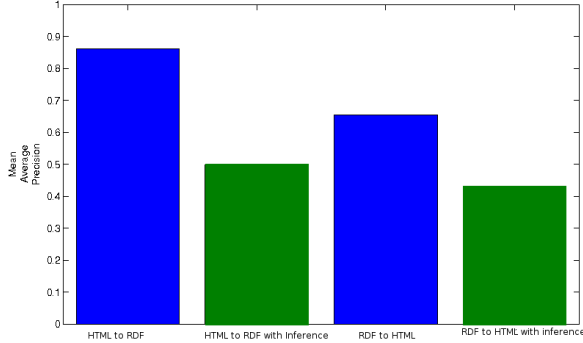


Figure 8: Comparing the Relevance Feedback on the Semantic Web (RDF) and Hypertext Web (HTML) both without (blue) and with (green) Semantic Web inference

The results of using inference to boost hypertext Web results using Semantic Web is equally grim. Using the same parameters as above, the feedback from the expanded Semantic Web data to the hypertext Web results in an average precision of 0.4273, which is insignificantly different from the baseline of not using relevance feedback at all of 0.4284 ($p < .05$) and considerably worse than not using inference at all, which has a MAP of 0.6549 ($p < .05$). When pseudo-feedback is used, the results fall to the rather low score of 0.3861, which is clearly below the baseline of 0.4284 ($p < .05$). So, at least the obvious way of use of native type and sub-class based Semantic Web inference seems to only lead to a decline in performance.

Why does inference hurt rather than help performance? One would naively assume that adding more knowledge in the form of structured data would help the results. However, this assumes the knowledge gained through inference would somehow lead to the discovery of terms also found in relevant documents. However, in the case of much inference with the Semantic Web, this is not the case. For example, simply consider the structured Semantic Web data about the query for the football player ‘Dick Anderson.’ While the first Semantic Web article about Dick Anderson gives a number of useful facts about him, such as the fact that he is a football player, determining that Dick Anderson is a person via the use of inference is of vastly less utility. For example, the Cyc ontology [13] declares that Dick Anderson is a person, namely that “Something is an instance of Person if it is an individual Intelligent Agent with perceptual sensibility, capable of complex social relationships, and possessing a certain moral sophistication and an intrinsic moral value.” In this regard, inference only serves as noise, adding irrelevant terms (or rarely relevant terms) to the language models. While further experiments in interpolation would be useful, our explorations in this direction showed no performance increase, only less performance decrease.

7. DEPLOYED SYSTEMS

One area we have not explored is how relevance-feedback systems performs against systems that are actually deployed, as our previous work has always been evaluated against systems and parameters we created specifically for experimental evaluation. For example, our performance in Section 5 and Section 4.2.1 was only compared to baselines that were versions of our weighting function without a relevance feedback component. While that particular baseline is principled, the obvious needed comparison is against actual deployed commercial or academic systems where the precise parameters deployed may not be publicly available and so not easily simulated experimentally.

7.1 Results

The obvious baseline to choose to test against is the Semantic Web search engine, FALCON-S, from which we derived our original Semantic Web data in the experiment. We used the original ranking of the top 10 results given by FALCON-S to calculate its average precision, 0.6985. We then compared both the best baseline, rm , as well as the best system with feedback in Figure 9. As shown, our system with feedback had significantly ($p < .05$) better average precision (0.8611) than FALCON-S (0.6985), as well better ($p < .05$) than the ‘best’ language model baseline without feedback (0.5043) as reported earlier as given in Section 5.

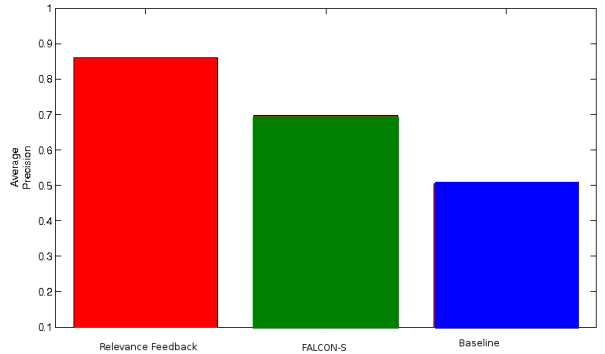


Figure 9: Summary of Best Average Precision Scores: Relevance Feedback From Hypertext to Semantic Web

Average precision does not have an intuitive interpretation, besides the simple fact that a system with better average precision will in general deliver more accurate results closer to the top. In particular, one scenario we are interested in is having *only* the most relevant RDF data accessible from a single URI returned as the top result, so that this result is easily consumed by some program. For example, given the search ‘amnesia nightclub,’ a program should be able to consume RDF returned from the Semantic Web to produce with high reliability a single map and opening times for a particular nightclub in Ibiza in the limited screen space of the browser, instead of trying to display structured data for every nightclub called ‘amnesia’ in the entire world. In Table 4, we show that for a significant minority of URIs (42%), FALCON-S returned a non-relevant Semantic Web URI as the top result. Our feedback system achieves an average precision gain of 16% over FALCON-S. While a 16% gain in average precision may not seem huge, in reality the effect is quite dramatic, in particular as regards boosting relevant URIs to the top rank. So in Table 4, we present results of how our best parameters tf with $m = 10,000$ lead to the most relevant Semantic data in the

Results:	Feedback	FALCON-S
Top Relevant:	118 (89%)	76 (58%)
Non-Relevant Top:	14 (11%)	56 (42%)
Non-Relevant Top Entity:	9 (64%)	23 (41%)
Non-Relevant Concept:	5 (36%)	33 (59%)

Table 4: Table Comparing Hypertext-based Relevance Feedback and FALCON-S

top result. In particular, notice that now 89% of resolved queries now have relevant data at the top position, as opposed to 58% without feedback. This would result in a noticeable gain in performance for users, which we would argue allows Semantic Web data to be retrieved with high-enough accuracy for actual deployment.

While performance is boosted for both entities and concepts, the main improvement comes from concept queries. Indeed, as concept queries are often one word and ambiguous, not to mention the case where the name of a concept has been taken over by some company, music band, or product, it should not be surprising that results for concept queries are considerably boosted by relevance feedback. Results for entity queries are also boosted. A quick inspection of the results reveals that the entity queries were the most troublesome, and that these entity queries gave both FALCON-S and our feedback system problems. These problematic queries were mainly very difficult queries that have a number of Semantic Web URIs that all share similar natural language content. An example would be a query for ‘sonny and cher,’ which results in a number of distinct Semantic Web URIs: one for *Cher*, another one for *Sonny and Cher* the band, and another for ‘The Sonny Side of Cher,’ an album by Cher. For concepts, one difficult concept was the query `rock`. Although the system was able to disambiguate the musical sense from the geological sense, there was a large cluster of Semantic Web URIs for rock music, ranging from *Hard Rock* to *Rock Music* to *Alternative Rock*. These types of queries seem to present the most difficulties for Semantic Web search engines.

Although less impressive than the results for using hypertext web-pages for relevance feedback for the Semantic Web, the feedback cycle from the Semantic Web to hypertext does improve significantly the results of even commercial hypertext web-engines, at least for our set of queries about concepts and entities. The hypertext results for our experiment were given by Yahoo! Web search, and we calculated a mean average precision for Yahoo! Web search to be 0.4039. This is slightly less than our baseline language model ranking, which had an average precision of 0.4284. As shown in Figure 10, given that our feedback based had an average precision of 0.6549, our relevance feedback system performs significantly ($p < .05$) better than Yahoo! Web search and ($p < .05$) the baseline *rm* system.

7.2 Discussion

These results are not in need of a large discussion, as they clearly show our relevance feedback method works significantly better than various baselines, both internal baselines and state of the art commercial hypertext search engines and Semantic Web search engines. The parametrisation of the precise information retrieval components used in our system is not entirely arbitrary, as argued above in Section 4.1.2 and Section 4.2.2. The gain of our relevance feedback system, a respectable 16% in average precision over the engine FALCON-S, intuitively makes the system’s ability to place a relevant structured Semantic Web data in the top rank acceptable for most users.

More surprisingly, by incorporating human relevance from the

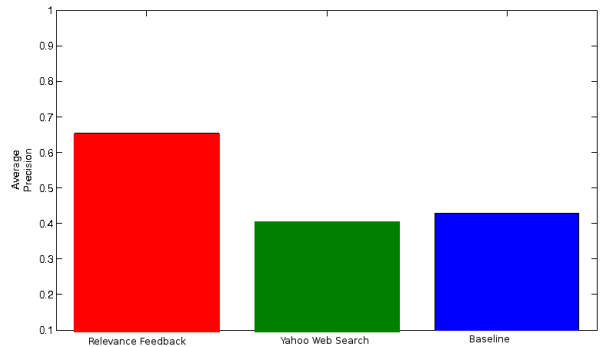


Figure 10: Summary of Best Average Precision Scores: Relevance Feedback From Semantic Web to Hypertext

Semantic Web, we make substantial gains over state of the art systems for hypertext Web search, a 25% gain in average precision over Yahoo! search. One important factor is the constant assault of hypertext search engines by spammers and others. Given the prevalence of a search engine optimisation and spamming industry, it is not surprising that the average precision of even a commercial hypertext engine is not the best, and that it performs less well than Semantic Web search engines. Semantic Web search engines have a much smaller and cleaner world of data to deal with than the unruly hypertext Web, and Web search must be very fast and efficient. Even with relevance from the Semantic Web, an average precision of 40% is impressive, although far from the 65% precision using relevance feedback from the Semantic Web. Even with the help of the Semantic Web, hypertext search is unlikely to achieve near perfect results anytime soon.

Interestingly enough, it seemed that pseudo-feedback only helps marginally in improving hypertext Web search using Semantic Web data, although it helps a fair amount in improving Semantic Web search using hypertext results. This should not be a surprise, as pseudo-feedback in general performs worse than relevance feedback. However, the amount of a performance lost given by pseudo-feedback show that both for the Semantic Web and hypertext Web search for concepts and entities is difficult, as many results that are about highly different things and subject matters may be returned.

8. FUTURE WORK

There are a number of areas where our project needs to be more thoroughly integrated with other approaches and improved. The expected criticism of this work is likely the choice of FALCON-S and Yahoo! Web search as a baseline, and that we should try this methodology over other Semantic Web search engines and hypertext Web search engines. At this point, we can only leave this for future work. Yet in our opinion the most exciting work is to be done as regards scaling our approach to work with live large-scale Web search engines.

8.1 Scaling to the Web

While language models, particularly generative models like relevance models [12], should have theoretically higher performance than vector-space models, the reason why large-scale search engines do not in general implement language models for information retrieval is that the computational complexity of calculating distributions over billions of documents does not scale. However, there

is reason to believe that relevance models could be scaled to work with Web search if they built their language sample from suitably large ‘clean’ sample of natural language and also compressed the models by various means. Lastly, our system and experiment was only a *proof of concept* system, and it was tested only over a relatively small (although statistically significant) query sample. Far better would be to deploy this system with a global-scale hypertext search engine that then fed its relevance feedback statistics, easily garnered from query logs, to a Semantic Web search engine. Lastly, as more and more structured data is used on the Web, if this data is stored in either a format like that of the Semantic Web or formats like microformats that can easily be converted to the Semantic Web, then this data can be used to boost the performance of hypertext search engines.

8.2 Creation of New Semantic Web data

One of the looming deficits of our system is that for a substantial amount of our queries there are *no* relevant Semantic Web URIs with accessible RDF data. This amount is estimated to be 34% of all queries. However, these queries with no Semantic Web URIs in general *do* have relevant information on the hypertext Web, if not the Semantic Web. The automatic generation of Semantic Web triples from natural language text as explored by Brewster et al. [2] could be used in combination with our system to create automatically generated Semantic Web data, in response to user queries.

9. CONCLUSION

Relevance feedback from unstructured data can improve structured search. Furthermore, relevance feedback from structured search can improve unstructured search. While relevance feedback is known to in general improve results, our use of wildly disparate sources of data such as the structured Semantic Web and the unstructured hypertext Web to serve as relevance feedback for each other is novel. We do this by treating both data sources as ‘bags of words’ in order to make them compatible and find from structured data high quality terms for use in language models.

These results demonstrate that our approach of using feedback from hypertext Web search helps users discover relevant Semantic Web data. The gain is significant over both baseline systems without feedback and the state of the art page-rank based mechanism used by FALCON-S and Yahoo! Web search. Furthermore, the finding of relevant structured Semantic Web data can even be improved by pseudo-feedback from hypertext search. These results, due to the significant and randomized number of queries used and the fact that relevance judgements involved three judges, point to a high reliability for these results, so we have reason to believe the results will scale.

More exciting to the majority of users of the Web is the fact that apparently relevance feedback from the Semantic Web can improve hypertext Web. However, pseudo-feedback only marginally improves the quality of results of hypertext Web search engines. This result means that there needs to be considerable effort in determining when structured data from the Semantic Web is actually relevant. While it is easy enough for query logs to determine relevant hypertext Web data, it is more difficult to determine when structured data is relevant. However, there are clear signs. For example, structured data that is consumed by applications like maps and calendar programs can be ascertained to be actually used. This could be detected in formats like microformats by the use of a plugin or Web service. Interestingly enough, using inference only hurt performance, due to the rather obscure terms from higher-level ontologies serving functionally as ‘noise’ in the feedback. While this kind of data information may be useful for automated reasoning,

it does not really provide hints for the kinds of information that a hypertext Web searcher may be looking for.

The operative question is: Why does relevance feedback work? Although there appears to be a huge gulf between structured data and unstructured data, it is precisely because the same *kind of information* is encoded in the unstructured hypertext and the structured Semantic Web that these two disparate sets of data can be used as relevance feedback for each other. Indeed, the key to true high-level performance for search engines is the use of high quality data of any kind for query expansion, whether it is stored in a structured format or unstructured formats.

10. REFERENCES

- [1] Ricardo Baeza-Yates. From capturing semantics to semantic search: A virtuous cycle. In *Proceedings of the 5th European Semantic Web Conference*, pages 1–2, Tenerife, Spain, 2008.
- [2] Christopher Brewster, Jose Iria, Ziqi Zhang, Fabio Ciravegna, Louise Guthrie, and Yorick Wilks. Dynamic iterative ontology learning. In *Proceedings of the Recent Advances in Natural Language Processing Conference (RANLP)*, Borovets, Bulgaria, 2007.
- [3] Dan Brickley and R. V. Guha. RDF Vocabulary Description Language 1.0: RDF Schema. Recommendation, W3C, 2004. <http://www.w3.org/TR/rdf-schema/> (Last accessed on Nov. 15th 2008).
- [4] Gong Cheng, Weiyi Ge, and Yuzhong Qu. FALCONS: Searching and browsing entities on the Semantic Web. In *Proceedings of the World Wide Web Conference*, 2008.
- [5] Li Ding and Tim Finin. Characterizing the Semantic Web on the Web. In *Proceedings of the International Semantic Web Conference (ISWC)*, pages 242–257, 2006.
- [6] J.L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76:378–382, 1971.
- [7] Harry Halpin. A query-driven characterization of linked data. In *Proceedings of the Linked Data Workshop at the World Wide Web Conference*. Madrid, Spain, 2009.
- [8] Harry Halpin and Victor Lavrenko. Relevance feedback between hypertext search and semantic search. In *Proceedings of the Semantic Search Workshop at the World Wide Web Conference*. Madrid, Spain, 2009.
- [9] Patrick Hayes and Harry Halpin. In defense of ambiguity. *International Journal of Semantic Web and Information Systems*, 4(3), 2008.
- [10] Graham Klyne and Jeremy Carroll. Resource description framework (rdf): Concepts and abstract syntax. Recommendation, W3C, 2004. <http://www.w3.org/TR/rdf-concepts/>.
- [11] V. Lavrenko, J. Allan, E. DeGuzman, D. LaFlamme, V. Pollard, and S. Thomas. Relevance models for topic detection and tracking. In *Proceedings of Human Language Technologies Conference, HLT 2002*, pages 104–110, 2002.
- [12] Victor Lavrenko. *A Generative Theory of Relevance*. Springer-Verlag, Berlin, Germany, 2008.
- [13] Douglas Lenat. Cyc: Towards programs with common sense. *Communications of the ACM*, 8(33):30–49, 1990.
- [14] A. Mikheev, C. Grover, and M. Moens. Description of the LTG system used for MUC. In *Seventh Message Understanding Conference: Proceedings of a Conference*, 1998.
- [15] Eyal Oren, Renaud Delbru, Michele Catasta, Richard Cyganiak, Holger Stenzhorn, and Giovanni Tummarello. Sindice.com: a document-oriented lookup index for open linked data. *International Journal of Metadata, Semantics, and Ontologies 2008*, 3(1):37–52, 2008.
- [16] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of the Twenty-First Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 275–281, Melbourne, Australia, 1998. ACM Press.
- [17] J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART Retrieval System: Experiments in Automatic Document Processing*, pages 313–32. Prentice-Hall, Inc., Uppder Saddle River, New Jersey, USA, 1971.