Relevance Feedback between Web Search and the Semantic Web

Harry Halpin and **Victor Lavrenko**
School of Informatics
University of Edinburgh
10 Crichton St.
Edinburgh, United Kingdom
h.halpin@ed.ac.uk and vlavrenk@inf.ed.ac.uk

Abstract

We investigate the possibility of using structured data to improve search over unstructured documents. In particular, we use relevance feedback to create a 'virtuous cycle' between structured data from the Semantic Web and web-pages from the hypertext Web. Previous approaches have generally considered searching over the Semantic Web and hypertext Web to be entirely disparate, indexing and searching over different domains. 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, our evaluation is 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 show our relevance model-based system is better than the performance of real-world search engines for both hypertext and Semantic Web search, and we also investigate Semantic Web inference and pseudo-relevance feedback.

1 Introduction

There has recently been gathering interest in combining structured and unstructured search. This seems at least partially inspired by the Semantic Web initiative, which in the form of the graph-based RDF (Resource Description Framework) format has standardized a language for structured data on the Web from a diverse range of sources. Recently, a large amount of RDF has been deployed on the Web as "Linked Data," and so now enough RDF data exists to be reasonably crawled and indexed, but the rankings of Semantic Web search engines are much less well-studied than hypertext Web rankings, and so are thought likely to be sub-optimal [Oren *et al.*, 2008]. The hypothesis put forward by Baeza-Yates is that the search for structured data can be used to improve traditional ad-hoc information retrieval for hypertext Web search engines [Baeza-Yates, 2008], and that techniques from information retrieval can be used to improve the search for structured data.

We are the first to suggest that relevance feedback may be the primary method for creating a 'virtuous cycle' between structured and unstructured search. 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. This requires a Cranfield-style evaluation on a set of hypertext pages and RDF graphs returned by the same query from two different search engines. Even 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?

2 Related Literature

Relevance feedback is the use of explicit relevance judgments from users of a search engine in order to expand the query. By 'expand the query,' we mean that the usually short query is expanded into a larger query by adding terms 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.' 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.' The hypothesis of relevance feedback is that the relevant documents will disambiguate the query and in general give a better description of the information need of the query than the query itself.

Relevance feedback often (but not always) improves information retrieval performance, but almost always the feedback is used to improve rankings over a single source of data and has never been tested on the Semantic Web [Lavrenko, 2008]. Previous approaches have assumed that the Semantic Web and the hypertext Web search are entirely disparate, indexing and searching them differently [Cheng *et al.*, 2008]. Other approaches have tried to use the hypertext Web as a 'backup' when querying RDF fails [Pound *et al.*, 2010]. On a more related note, previous work has focused on deployed hypertext search engines over structured data (albeit without relevance feedback) [Agrawal *et al.*, 2009]. We are assuming that more complex queries (SPARQL structured queries, natural language queries) for the Semantic Web will require keyword search to work reasonably well, so naturally keyword search over RDF is the first part of Semantic Web search to address.

3 System Design

However, could one use structured data in RDF about the Eiffel Tower to expand a query for 'eiffel' in a hypertext search engine? Likewise, could one search over structured data for 'eiffel' and use relevant web-pages from the hypertext web to expand the query? In our system, the query is run first against the hypertext Web search engine and we collect relevance judgments from the results or use pseudo-feedback (assuming the top x results are relevant). We then use this feedback to expand the query to the Semantic Web with certain words, chosen by use of the algorithms described in Section 3.2, from the relevant hypertext web-pages. Finally, the expanded query is used to re-rank the results retrieved by a Semantic Web search engine. We can compare both structured Semantic Web and unstructured hypertext data by considering both to be an unstructured 'bags of words,' albeit weighted by either frequency in unstructured text, weights given to RDF structure, or both. Semantic Web data is 'flattened' into a 'document' of terms derived from both the text and URIs in the RDF graph. The URIs in RDF graphs can be reduced to 'words' by tokenizing on the rightmost hierarchical component (right of fragment identifier if one exists) of the URI. So, the URI `http://www.example.org/hasArchitect` would be reduced to two tokens, 'has' and 'architect.' Of 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 or link to Semantic Web data by a program to be a judgment of relevance.

3.1 Experimental Design

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.[1] This query log contained 6,623,635 unique queries corrected for capitalization. Given that the Semantic Web will not contain data about transactional or navigational queries, we limit the experiment to informational queries. A straightforward gazetteer-based and rule-based named entity recognizer was employed to discover the names of people and places, based off a list of names maintained by the Social Security Administration and a place name database provided by the Alexandria Digital Library Project. 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 a named entity recognizer, for a total of 200 queries to be used in evaluation. Constraints were placed on crawled URIs, such that at least 10 documents from both the Semantic Web and hypertext Web had to be crawled for each query (using FALCON-S and Yahoo! Search respectively), leading to a total of 4,000 results. 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.'

In order to aid the judges, a Web-based interface that presented rendered versions of both the web-page and the Semantic Web result was created to present the queries and results to the judges. Each of the Semantic Web results were rendered using a two-column table for the properties and subjects of each resulting URI. Each result was judged by three judges. Before 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 judgments of relevance, so each result must be either relevant or irrelevant to the query. After the ratings were completed, Fliess's κ statistic was taken in order to test the reliability of inter-judge agreement on relevancy judgments. For both relevance judgments 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 null hypothesis is to be rejected and 'moderate' agreement. For the creation of our evaluation data set, majority voting was used amongst the three judges.

3.2 Language Models

For the relevance feedback methods used in this paper, we employ the well-known *language model* approach as it a formally principled and probabilistic approach to determining the ranking and weighting function. The experiment was run earlier using vector-space models and the results are in general fairly similar although performing slightly worse. Language modeling frameworks in information retrieval represent each document as a language model given by an underlying multinomial probability distribution so that for each word $w \in V$ there is a value that gives how likely an observation of word w is given D, i.e. $P(w|u_D(w))$. The document model distribution $u_D(w)$ is then estimated using the parameter ϵ_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.

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

The parameter ϵ_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 each word v (of the language V) in collection C. Our baseline is called *query-likelihood*, and ranks documents D by the probability that the query Q could be observed during repeated random sampling from the distribution $u_D(\cdot)$. However, the classical

[1]Made available due to an Microsoft 'Beyond Search' award.

language-modeling approach to IR does not provide a natural mechanism to perform relevance feedback. One 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 $R = r_1 \ldots r_k$ 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:

$$u_{R,avg}(w) = \frac{1}{k}\sum_{i=1}^{k} u_{r_i}(w) = \frac{1}{k}\sum_{i=1}^{k}\frac{n(w,r_i)}{|r_i|} \qquad (2)$$

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 as given by equation 3. 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 2 and 3 is that the former treats every document equally, regardless of its length, whereas the latter favors longer documents, i.e. they are not individually penalized by dividing their contributing frequencies $n(w, r_i)$ by their length $|r_i|$.

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

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 [Lavrenko, 2008] is cross entropy. 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 4. Note that either the *averaged* relevance model $u_{R,avg}$ or the *concatenated* relevance model $u_{R,con}$ can be used in Equation 4. We refer to the former as rm and to the latter as tf in the following experiments.

$$-H(u_R||u_D) = \sum_{w \in V} u_R(w) \log u_D(w) \qquad (4)$$

4 Feedback Evaluation

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. For evaluation we used mean average precision (MAP) with the standard Wilcoxian sign-test ('average precision' in figures). These scores will be supplemented with R-precision (RP) to avoid bias.

4.1 Hypertext to Semantic Web Feedback

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. For these experiments, the feature values (f) were uniform. 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 modeling, the relevance models were not smoothed, but a number of different smoothing parameters for ϵ were investigated for the cross entropy comparison function, ranging ϵ between 0.01, 0.10, 0.20, 0.50, 0.80, 0.90, and 0.99. All ranking was done over a pool of *all* Semantic Web documents crawled and evaluated for relevance earlier (the top 10 documents returned by FALCON-S for each of the 200 queries, and so 2,000 documents total). The results are given in Figure 2. The highest performing language model was tf with a cross-entropy ϵ of .2 and a m of 10,000, which produced a MAP of 0.8611 and RP of 0.8142, which was significantly higher than the language model baseline of 0.5043 MAP and a RP of 0.4322 ($p < .05$) using again a m of 10,000 for document models with a ϵ of .99.

Although these results confirm well-known optimal settings for interpolation parameters, there is a new result for this domain, namely that tf always outperformed rm, as rm's best performance had a MAP of 0.7223 and a RP of 0.6518 using an ϵ of .1 and a m of 10,000. FALCON-S, from which we derived our original Semantic Web data in the experiment, had the mean average precision of its original ranking of the top 10 results given by FALCON-S to calculated as 0.6985. To summarize, we compared both the best baseline, rm, as well as the best system with feedback in Figure 1. As shown, our system with feedback had significantly ($p < .05$) better MAP (0.8611) than FALCON-S (0.6985), as well better ($p < .05$) than the 'best' language model baseline without feedback (0.5043) as well as relevance feedback from the Semantic Web to itself (0.7111, $p < .05$).

Hypertext can boost Semantic Web search insofar as it provides more up-to-date and accurate terms in hypertext pages about the information need than may easily be available in the Semantic Web documents alone. One observation is in order; note that tf always outperformed rm. The primary difference is that rm normalizes 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 normalize: tf constructs a relevance model by combining all the relevance documents and then creating the relevance model from the *raw pool* of all relevant document models. So for this task, normalization 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 [Oren *et al.*, 2008], Semantic Web data also follows a non-normal distri-

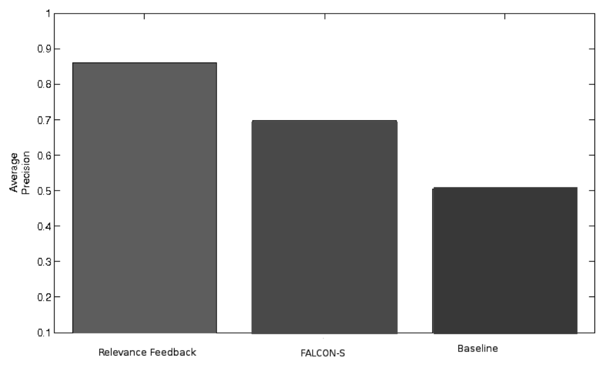

Figure 1: Summary of Best Mean Average Precision Scores: Relevance Feedback From Hypertext to Semantic Web

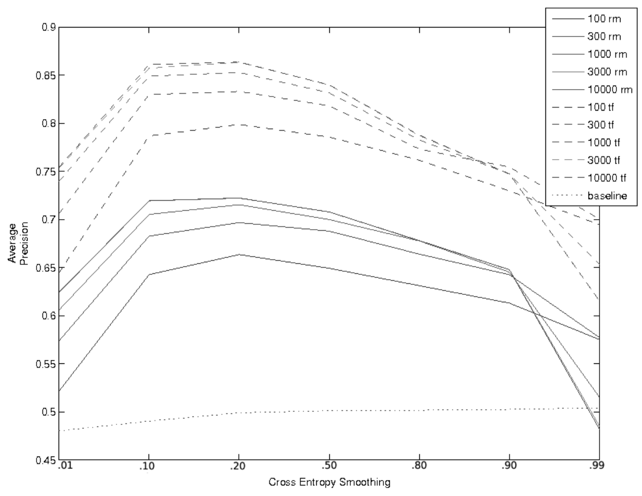

Figure 2: Mean Average Precision Scores for Language Model Parameters: Relevance Feedback From Hypertext to Semantic Web

bution, so there are wildly varying document lengths for both the relevance model and the document models. So document length is a good predictor of relevance on the Semantic Web, as long as the document contains (often as a consequence of its length) high-quality text with informative keywords. Due to these factors, it is unwise to normalize the models, as that will almost certainly dampen the effect of document length and maybe even of crucial keywords in the text.

4.2 Semantic Web to Hypertext Feedback

The results for using Semantic Web documents as relevance feedback for hypertext Web search are surprisingly promising. The results for language modeling were similar to the results in Section 4.1 and are given in Figure 3, although a few differences are worth commenting upon. The best performing language model was tf with m of 10,000 and a ϵ=.2, which produced an average precision of 0.6549 with a RP of 0.6116 ($p < .05$). In contrast, the best-performing rm, with

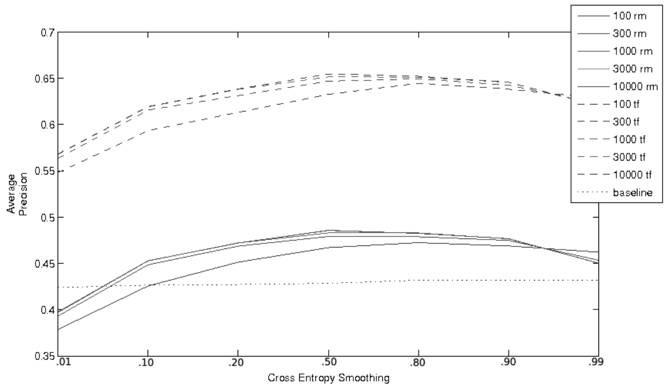

Figure 3: Mean Average Precision Scores for Language Model Parameters: Relevance Feedback From Semantic Web to Hypertext

a m of 3,000 and ϵ=.5, only had a MAP of 0.4858 and a RP of 0.4486 ($p < .05$). The tf relevance models consistently performed better than rm relevance models ($p < .05$). The baseline for language modeling was also fairly poor with a MAP of 0.4284 and a RP of 0.3990 ($p < .05$). This was the 'best' baseline using again a m of 10,000 for document models and a smoothing ϵ of .99. The hypertext results for our experiment were given by Yahoo! Web search, and we calculated the MAP 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. To summarize, in Figure 4, given that our feedback based system 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 as well as relevance feedback from hypertext results to the search engine itself (0.5095,$p < .05$).

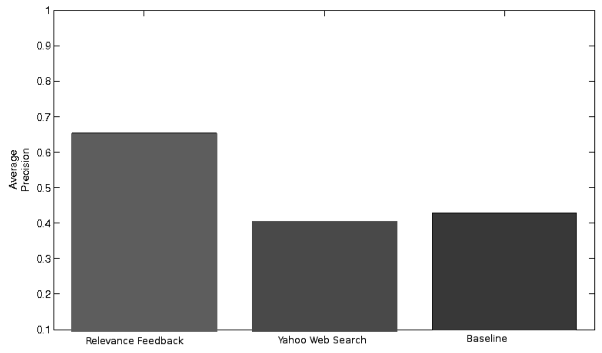

Figure 4: Summary of Best Mean Average Precision Scores: Relevance Feedback From Semantic Web to Hypertext

The general 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. Why does tf relevance modeling allow data from the Semantic

Web to boost hypertext Results? Rather shockingly, as the Semantic Web data often has manually high-quality curated data from sources like DBpedia as its primary components, 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. 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, the hypertext search engine is being 'seeded' with a high-quality accurate description of the information need expressed by the query.

5 Pseudo-relevance feedback

Using *pseudo-relevance feedback*, we assume that the top x documents returned are relevant. Using the same optimal parameters as discovered in Section 4.1, tf with a $m = 10,000$ and $\epsilon = .2$ was again deployed, but this time using pseudo-relevance feedback. Can pseudo-relevance 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 (with the same previously optimized parameters), the best pseudo-relevance feedback result had a MAP of 0.6240 and a RP of 0.5617. This was considerably better than the baseline of just using pseudo-relevance feedback from the Semantic Web to Semantic Web search, which only had a MAP of 0.5251 with a RP of 0.4482 ($p < .05$), and also clearly above the 'best' baseline of 0.5043 MAP and a RP 0.4322 ($p < .05$). However, as shown by Figure 5, the results are still not nearly as good as using actual relevant hypertext pages, which had a MAP of 0.8611 with a RP of 0.8142 ($p < .05$). This is likely because, not surprisingly, the hypertext Web results contain many irrelevant queries that serve as noise, preventing the feedback from boosting the results. Can pseudo-relevance feedback from the Semantic Web improve hypertext search? The answer is yes, but barely. The best result for mean average precision is 0.4321 ($p < .05$), which is better than the baseline of just using pseudo-relevance feedback from hypertext Web results to themselves, which has a mean average precision of 0.3945 ($p < .05$) and the baseline without feedback at all of 0.4284 ($p < .05$). However, the pseudo-relevance feedback results still significantly have a worse performance by a large margin than using relevant Semantic Web data, which had a mean 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 topics even more than the hypertext Web. Therefore, adding pseudo-relevance feedback increases the amount of noise in the language model but still helps performance.

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. In our experiment, we followed both Semantic Web sub-class and type links for one-level. The resulting retrieved and inferred Semantic Web data

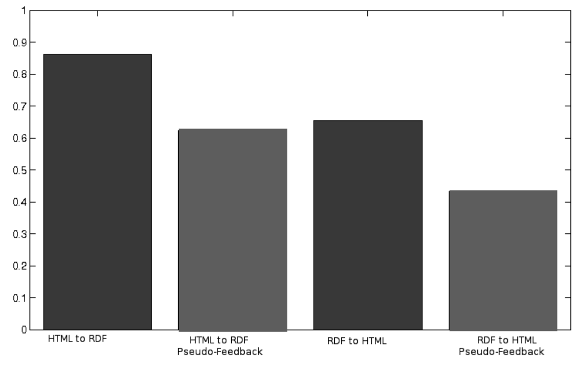

Figure 5: Comparing Pseudo-relevance Feedback (red) to Manual Feedback (blue) on the Semantic Web (RDF) and Hypertext Web (HTML)

were concatenated together. In this way, Semantic Web inference was used as *document expansion*, where the original RDF graph was expanded by adding inferred RDF content [Mayfield and Finin, 2003]. Relevance feedback was tried with these documents expanded by inference, again with the same best-performing parameters (tf with $m = 10,000$ and $\epsilon = .2$). In the first case, inference is used to expand the Semantic Web data, and then the hypertext results are used as relevance feedback. However, as shown in Figure 6, deploying inference only causes a drop in performance. In particular, using hypertext Web results as relevance feedback to Semantic Web search, the system drops from a performance of 0.8611 MAP (0.8142 RP) to a performance of 0.4991 MAP with a RP of 0.4266 ($p < .05$).

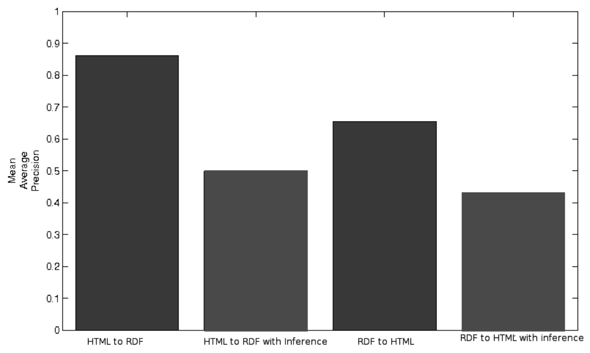

Figure 6: Comparing the Relevance Feedback on the Semantic Web (RDF) and Hypertext Web (HTML) both without (blue) and with (green) Semantic Web inference

Why does inference not help performance in this case? We were assuming the knowledge gained through inference would somehow lead to the discovery of terms also found in relevant documents. However, in some cases of inference with the Semantic Web, adding new terms does not help satisfy the user's information need. For example, simply consider the structured Semantic Web data about the query for the singer 'Shakira.' While the first Semantic Web article about Shakira gives a number of useful facts about her, such as a list of her albums and the fact that her real name is Shakira Isabel Mebarak Ripoll, determining that Shakira is a person via the use of inference is of vastly less utility. For example, the Cyc ontology declares that Shakira 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." This is probably not of interest to whoever is searching about Shakira. In this regard, using inference to expand Semantic Web data can add terms that are irrelevant to the user's information need, so possibly adding distracting noise to the language model. However, using such inference could help in the case of ambiguous queries, such as distinguishing if a query about 'John Deere' is a person or a company.

7 Conclusions

These results show relevance feedback between the Semantic Web and hypertext increases performance for both kinds of search, as our relevance feedback-based system outperforms baselines without feedback as well as state of the art commercial hypertext search engines and research Semantic Web search engines. A thorough discussion of the precise reasons particular algorithms and parameters were chosen has already been given in Section 4.2 and Section 4.1. The gain of our relevance feedback system, a respectable 16% in MAP over 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 Linked Data, we make substantial gains over state of the art systems for hypertext Web search, a 25% gain in MAP over Yahoo! search.

There are a number of further areas of research. One expected criticism of this work is likely the choice of FALCON-S and Yahoo! Web search as a baseline and source of crawled documents, and that we should try this methodology over other Semantic Web search engines and hypertext Web search engines. Also, comparing the Semantic Web to Wikipedia-based relevance feedback would also be of interest, as would collecting a larger standard evaluation corpus for Semantic Web information retrieval. Our experiment was only a *proof of concept* done only over a relatively small (yet statistically significant) set of queries and data, although the "real-world" nature of the sample should allow us to deduce the results will generalize. For actual deployment, detailed performance statistics should be taken and relevance feedback would need to be approximated through click-through logs, which is non-trivial. More analysis should be done on how Semantic Web data can be used in a more structured form; for example, the entity name as given by *rds:label* is usually of higher importance than other features of a RDF graph, so we can now optimize our results by weighing the various terms in the relevance model by weights derived from the structure. Lastly, complex queries (SPARQL and natural language) could be built on top of the current keyword interface.

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. These results demonstrate that our approach of using feedback from hypertext Web search helps users discover relevant Semantic Web data, and vice versa. The gain is significant over baseline systems without feedback and the algorithms used by FALCON-S and Yahoo! Web search. Furthermore, retrieving relevant Semantic Web data can even be improved by pseudo-relevance feedback from hypertext search while further work should be done to explore whether inference can help, as a straightforward approach does not improve results. Yet overall our results point to a high potential for relevance feedback between the hypertext Web and the Semantic Web being deployed on a larger scale to improve search, and this can be extended to feedback between structured and unstructured data in general. The question is: Why does relevance feedback work between seemingly disparate data-sources work on the Web? It is precisely because *information about the same things* is encoded in both hypertext web-pages and the Semantic Web regardless of their particular representation on the Web.

References

[Agrawal *et al.*, 2009] Sanjay Agrawal, Kaushik Chakrabarti, Surajit Chaudhuri, Venkatesh Ganti, Arnd Christian Konig, and Dong Xin. Exploiting web search engines to search structured databases. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 501–510, New York, NY, USA, 2009. ACM.

[Baeza-Yates, 2008] 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.

[Cheng *et al.*, 2008] Gong Cheng, Weiyi Ge, and Yuzhong Qu. FALCONS: Searching and browsing entities on the Semantic Web. In *Proceedings of the the World Wide Web Conference*, 2008.

[Lavrenko, 2008] Victor Lavrenko. *A Generative Theory of Relevance*. Springer-Verlag, Berlin, Germany, 2008.

[Mayfield and Finin, 2003] J. Mayfield and T. Finin. Information retrieval on the semantic web: Integrating inference and retrieval. In *Proceedings of the SIGIR Workshop on the Semantic Web*, Toronto, Canada, 2003.

[Oren *et al.*, 2008] 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.

[Pound *et al.*, 2010] Jeffrey Pound, Ihab F. Ilyas, and Grant Weddell. Expressive and flexible access to web-extracted data: a keyword-based structured query language. In *Proceedings of the 2010 international conference on Management of data*, SIGMOD '10, pages 423–434, New York, NY, USA, 2010. ACM.