Investigating the Demand Side of Semantic Search through Query Log Analysis

Edgar Meij
ISLA, University of Amsterdam
Sciencepark 107
1098 XG Amsterdam
edgar.meij@uva.nl

Peter Mika
Yahoo! Research
177 Avinguda Diagonal
08018 Barcelona, Spain
pmika@yahoo-inc.com

Hugo Zaragoza
Yahoo! Research
177 Avinguda Diagonal
08018 Barcelona, Spain
hugoz@yahoo-inc.com

ABSTRACT

1. INTRODUCTION

Semantic search is by its broadest definition a collection of approaches that aim at matching the Web's content with the information need of Web users at a semantic level. Most of the work in this area has focused on the *supply-side* of semantic search, in particular elevating Web content to the semantic level by relying on methods of information extraction [2] or working with explicit metadata embedded inside or linked to Web resources. With respect to explicit metadata, several studies have been done on the adoption of semantic web formats in the wild, mostly based on statistics from the crawls of semantic web search engines [3, 4, 5, 6, 7]. Much less effort has focused on the *demand-side* of semantic search, i.e. interpreting queries at the semantic level and studying information needs at this level. Conversely, little is known as to how much the supply of metadata actually matches the demand for information on the Web.

In this paper, we address the problem of studying the information need of Web searchers at an ontological level, i.e., in terms of the particular attributes of objects they are interested in. We describe a set of methods for extracting the context words to certain classes of objects from a Web search query log. We do so based on the idea that common context words reflects aspects of objects users are interested in. We implement these methods in an interactive tool called the Semantic Search Assist. The original purpose of this tool was to generate type-based query suggestions when there is not enough statistical evidence for entity-based query suggestions. However, from an ontology engineering perspective, this tool answers the question of what attributes a class of objects would have if the ontology for it was engineered purely based on the information needs of end users. As such it allows us to reflect on the gap between the properties defined in Semantic Web ontologies and the attributes of objects that people are searching for on the Web. We evaluate our tool by measuring it's predictive power on the query log itself. We leave the study of the gap between particular information needs and Semantic Web data for future work.

2. MINING TYPE-BASED QUERY CONTEXTS

We begin by observing that in Web search query logs and in particular for queries that contain a named entity, the *class* of the entity that the user is looking for often determines the query context, i.e. a prefix or suffix written before or after the name of an entity, respectively. Put differently, entities of the same class often occur in the context of similar words, representing specific information users are interested in with respect to that particular class of entities.

2.1 Notation

In this section, we introduce our notations and detail our proposed type-based context extraction methods. We assume queries can be decomposed in an entity part e and a context part f and that entities can be assigned a type T. Table 1 shows some examples. In case the query contains a pre- and suffix, we treat it as two separate queries. Given a set of such queries, we determine the matrix $N = (n_{ef})_{e,f}$, where n_{ef} is the number of times we see f with e. By grouping all entities of a certain type we can, for example, compute $n_{Tf} := \sum_{e \in T} n_{ef}$ which is the number of times we see completion f with an entity of type T. Using N, we can readily estimate probabilities such as $P(f)$, $P(f|e)$, $P(f|T)$, and $P(e|f,T)$.

2.2 Extraction Methods

Imagine that a user is typing a query and we recognise what she has typed so far as an entity with a corresponding type. The most naive approach would be to suggest the most frequent completions for the current entity (**M0**): $score_{M0}(f,e) = P(f|e)$. Given an infinite amount of data this should suffice. However, it will probably fail for rare entities since we will have none or very few completions for them. For this reason we turn to to the entity type and smooth the entity distribution with the type distribution.

M1 looks at the most likely completion for the current type: $score_{M1}(f,T) = P(f|T)$. Another desirable property a completion should have, is being rare over all types. **M2** rewards such completions: $score_{M2}(f,T) = \frac{P(f|T)}{P(f)}$. Another intuition is that completions which are frequent as well as evenly distributed among the entities in the type should be rewarded (**M3**): $score_{M3}(f,T) = G(f|T) = \left(\prod_{e \in T} n_{ef}\right)^{1/|T|}$. The final method (**M4**) only considers the distribution of completions within the type: $score_{M4}(f,T) = H(\theta_{f|T})$, where H is the entropy of the multinomial $\theta_{f|T} = (P(e|f,T))_{e \in T}$.

Query	Entity	Completion	Type
aspirin side effects	ASPIRIN	+*side effects*	Anti-inflammatory drugs
how to take ibuprofen	IBUPROFEN	–*how to take*	Anti-inflammatory drugs
britney spears video	BRITNEY SPEARS	+*video*	American film actors
britney spears shaves her head	BRITNEY SPEARS	+*shaves her head*	American film actors

Table 1: Example queries, extracted entities, completions, and types.

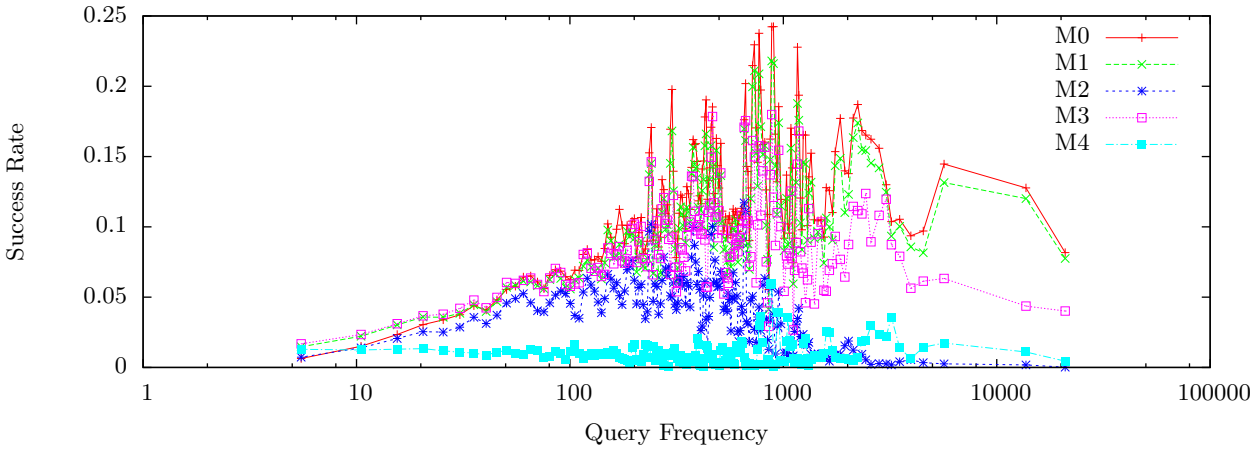

Figure 1: Success rate for queries. The x-axis shows the queries which are averaged and binned by frequency of occurrence. On the left we see rare queries, on the right popular ones.

To detect entities in queries, we require an ontology with a set of classes and a set of instances for each class. Consequently, we match the largest substring common to the query and the label of an instance.[1] In our experiments, we use DBpedia [2] considering both templates and categories as classes. Wikipedia entries can belong to many categories (e.g. 34 for Madonna) and reference a number of templates. We choose only one and try to select the *best* entity type. This is a challenging research question in itself; trivial methods such as choosing the most frequent or rarest type did not work well. Instead, we apply M1 on the training data and evaluate the performance of all possible types of each entity. We then choose the type that led to the best performance on the training set. For entities not present in the training set we select the type with the most entities.

3. EVALUATION OF TYPE-BASED CONTEXT PREDICTION

We evaluate the success of our context extraction by measuring its predictive power. In particular, we compare the highest scoring completions of the various methods with the actual observed remainder of the queries in the test set. We use 6 consecutive days of query logs which we split equally into a training and a test set. We analyze each query and if it contains an entity we keep it. This results in 1,681,753 queries for training and 1,644,033 for testing. For each query, we compute the top $K = 10$ completions predicted by each method. The correct completion for that query is the one typed by the user. We are interested in two evaluation measures: (i) Success Rate @ K (SR), i.e. whether the completion is correctly predicted and (ii) Mean Reciprocal Rank @ K (MRR), i.e. the mean of the inverse of the ranks at which the completion was found, up to K.

[1]We remove any disambiguation part in the entry title. This has the adverse effect of introducing noise, e.g. collapsing Madonna (art) and Madonna (entertainer). Disambiguating such queries is beyond the scope of the current work but could, e.g., be achieved by leveraging a user's history [1].

	M0	M1	M2	M3	M4
MRR	0.081	0.068	0.014	0.046	0.006
SR	0.118	0.104	0.041	0.088	0.010

Table 2: Aggregated results over all queries.

Table 2 shows the results over all test queries. As is clear from the low absolute scores, the task of suggesting the correct completion is a difficult one. The highest obtained MRR lies around 0.18 for M0 with queries that occur around 1000 times. M0 outperforms the type-based methods on almost all queries and measures. However, as indicated by Figure 1, the type-based methods, in particular M1 and M3, perform slightly better than M0 for less frequent queries (occurring 40 times or less and making up 12.7% of the total query volume). For other queries, M0 outperforms all other methods although the difference with M1 is usually small. The reason for the lower scores at the most frequently occurring queries is that these mostly consist of entities such as "in", "to", and "uk" (which are actual Wikipedia entries).

In the future, we plan to complement this evaluation with a user study as we feel that some of the models might achieve a high prediction accuracy by over-fitting popular entities. There are also many query contexts that are particular to the specific entity (e.g. *britney spears shaves her head*) but a user is likely to accept other reasonable suggestions based on

infobox_settlement	infobox_musical_artist	drugbox	infobox_football_club	information_appliance
hotels	lyrics	buy	forum	palm
map	buy	what is	news	review
map of	**pictures of**	tablets	website	software
weather	what is	what is	homepage	reviews
weather in	video	side effects of	tickets	accessories
flights to	download	hydrochloride	official website	manual
weather	hotel	online	**badge**	cases
hotel	dvd	overdose	fixtures	buy
property in	mp3	capsules	free	forum
cheap flights to	best	addiction	**logo**	charger

Table 3: Top ten prefixes and postfixes using our model M5 and Wikipedia templates as classes.

the type (e.g. *britney spears videos*) when offered a choice.

4. QUALITATIVE ANALYSIS OF THE SEMANTIC GAP

Using our tool we can now investigate the semantic gap between the demand-side and the supply-side of semantic search, i.e., the difference between what Web users search for regarding certain classes of entities versus the information that is available in terms of structured data on the Web. Our analysis is qualitative for now and we limit our investigation to Wikipedia, considering Wikipedia templates as lightweight ontology in the style of DBpedia [2]. Nevertheless, given any class in a Semantic Web ontology and a list of instances of that class, the same investigation could be repeated.

Table 3 shows the most common contexts (prefixes or suffixes) for five different Wikipedia templates, computed using method M4. We have chosen this particular model over our other models because it seems to give better type-specific results: even though our M1 has higher predictive power, at the same time it is over-fitting popular entities in the class. We have chosen these five templates because they vary in size from 43225 entities for *infobox_settlement* to 88 entities for *information_appliance*.

We show in **bold** the prefixes or suffixes that match an infobox property, i.e., where the user's query is likely to be satisfied by infobox data (assuming that the particular property is defined for the particular entity the user is searching for, i.e. that the infobox has been completed for this property). It is immediately obvious that there are very few of these. In fact, it seems the majority of these popular information needs cannot even be possibly satisfied by factual data. We leave it for further investigation to study whether it is the case that factual questions –which may be individually uncommon– would still make up a substantial portion of query volume.

It is interesting to note that the there are also information needs where the answer could be relatively concise and expressed in a single sentence or paragraph. This is often reflected in the structure of articles, i.e. the division of information into sections. For example, articles on drugs often have sections titled 'Overdose' and 'Side Effects'. Even if the answer to a query such as *aspirin overdose* can not be answered by a single fact, the information the user is looking for may come from a single section or even a single paragraph within the Wikipedia article. This warrants further investigation of exploiting article structure when searching Wikipedia.

In summary, it is clear that if infobox data would be geared toward answering popular information needs as surfaced by our tool, the infoboxes would need to contain different information at different levels of granularity. This suggests that for answering these 'head' queries one may need to merge the methods of data retrieval with methods of structured retrieval and unstructured retrieval. Put differently, for using the output of our tool as an input for ontology engineering, the list of context words extracted will need to be filtered to those representing attributes of objects, i.e. properties that can be filled with simple values.

5. CONCLUSION AND FUTURE WORK

We have presented a number of methods and their implementation in an online tool for mining type-based query context information, i.e. query prefixes and postfixes that are common to a class of entities, while uncommon to other entities outside of their class. Postulating that these context words represent aspects of entities that search engine users are interested in, we proceeded to investigate on the case of Wikipedia the extent to which this schema of information needs matches the schema of available structured data. We find that at least for the most common context words the overlap is very low as the most common queries are not specific enough to be answered by factual data.

Considering the information needs of end users is critical to the success of Semantic Search and these results suggest that factual information alone may only address a relatively small portion of information needs. One might of course argue that this is a chicken and egg problem: many of these general queries are likely hiding specific needs but unless search engines will be able to satisfy them, users are unlikely to provide more precise definitions of what they are looking for. A promising direction of research is the investigation of how search engines might assist users in formulating more precise queries. However, for the moment a combination of approaches based on document search, structured search (section, paragraph or sentence level search) and factual search promises the best results.

Finally, we suggest that our tool could be used in the future to analyze, extend or create new ontologies based on the information needs extracted from query logs. In this case it

is left to the ontology developer to consider which context words signify relevant attributes of objects to be included in an ontology.

References

J. Bai and J.-Y. Nie. Adapting information retrieval to query contexts. *IPM*, 44(6):1901–1922, 2008.

C. Bizer. Dbpedia: Querying wikipedia like a database. In *WWW '07*, 2007.

M. d'Aquin, C. Baldassarre, L. Gridinoc, S. Angeletou, M. Sabou, and E. Motta. Characterizing knowledge on the semantic web with watson. In R. Garcia-Castro, D. Vrandecic, A. Gómez-Pérez, Y. Sure, and Z. Huang, editors, *EON*, 2007.

L. Ding and T. Finin. Characterizing the semantic web on the web. In *International Semantic Web Conference 2006*, 2006.

L. Ding, L. Zhou, T. Finin, and A. Joshi. How the semantic web is being used: An analysis of foaf documents. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences, 2005. HICSS '05.*, 2005.

M. Hausenblas, W. Halb, Y. Raimond, and T. Heath. What is the size of the semantic web? In *Proceedings of the International Conference on Semantic Systems (I-Semantics2008)*, Graz, Austria, 2008.

G. Tummarello, R. Delbru, and E. Oren. Sindice.com: Weaving the open linked data. *The Semantic Web*, pages 552–565, 2008.