AQUAM: Automatic Query Formulation Architecture for Mobile Applications

Fotis Menemenis
Multimedia Knowledge Laboratory
Informatics and Telematics Institute
GR-57001 Thermi, Thessaloniki, Greece
fotis@iti.gr

Symeon Papadopoulos
Multimedia Knowledge Laboratory
Informatics and Telematics Institute
GR-57001 Thermi, Thessaloniki, Greece
papadop@iti.gr

Ben Bratu
Centre for Applications Research
Motorola Labs
St. Aubin, 91193 Gif sur Yvette, France
ben.bratu@motorola.com

Simon Waddington
Centre for Applications Research
Motorola Labs
Basingstoke RG22 4PD, UK
simon.waddington @motorola.com

Yiannis Kompatsiaris
Multimedia Knowledge Laboratory
Informatics and Telematics Institute
GR-57001 Thermi, Thessaloniki, Greece
ikom@iti.gr

ABSTRACT

When a user performs a web search, the first query entered will frequently not return the required information. Thus, one needs to review the initial set of links and then to modify the query or construct a new one. This incremental process is particularly frustrating and difficult to manage for a mobile user due to the device limitations (e.g. keyboard, display). We present a query formulation architecture that employs the notion of context in order to automatically construct queries, where context refers to the article currently being viewed by the user. The proposed system uses semantic metadata extracted from the web page being consumed to automatically generate candidate queries. Novel methods are proposed to create and validate candidate queries. Further two variants of query expansion and a post-expansion validation technique are described. Finally, insights into the effectiveness of our system are provided based on evaluation tests of its individual components.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Design, experimentation

Keywords

Mobile search, query validation, query expansion

1. INTRODUCTION

Web search is currently one of the most commonly performed tasks for the average computer user. Typically, the user enters query terms and the search engine retrieves the most relevant information for that query. With the introduction of higher bandwidth wireless data services such as HSDPA, and improved mobile web browsers, users are increasingly searching the web from their mobile devices [12], i.e. they carry out mobile searches. There are two main problems when performing a mobile search: (a) it takes effort to type the query due to the device input limitations (i.e. keyboard), (b) reviewing the search results is hard due to constraints in the device output (i.e. screen). In case the results are not satisfactory, the query typing and result reviewing tasks need to be repeated which leads to user frustration and is thus a key restricting factor for the use of search in mobile environments.

Consider a scenario, in which a business traveler arrives in the city where her meeting takes place (e.g. Stockholm) and she has some spare time in the evening. While reading the news on her mobile phone, she reads an interesting article about the merits of Mediterranean diet. Assuming that it is about dinner time, she thinks that it would be great to have a healthy Mediterranean dinner in a Greek restaurant; however, the chance that she will embark on an online search session in order to locate such a restaurant is rather slim due to the frustration she would feel by typing one or more search queries and reviewing the results.

In this paper, we present the AQUAM system, a context-aware mobile search architecture as a means to alleviate the aforementioned sources of user frustration. The proposed system makes use of semantic metadata extracted from the text-based item currently being consumed by the user in order to create an initial set of queries. Subsequently, the system analyzes the search results returned by these queries in order to rank them and only present the most relevant queries to the user. After carrying out an evaluation of the proposed system, we found that it produces queries which are judged by the users as relevant with respect to their context.

The rest of the paper is structured as follows. Section 2 provides an overview of research related to our work. The general architecture and the individual components of the proposed system are discussed in Section 3. Section 4 presents the user study which was carried out in order to evaluate the utility of the proposed system. Finally, Section 5 presents the conclusions of our research.

2. RELATED WORK

A number of information retrieval (IR) systems incorporating the notion of contextual search have been presented in the related literature.

Lawrence [8] provides some insight into the problem of understanding the context of a web search and outlines the importance of personalized content delivery in response to a query submitted to a search engine. Other IR tools, such as Inquirus 2 presented in [5], require explicit input from the user in the form of queries in order to recommend relevant web content, to modify (alter or expand) the input query or to classify web documents. The system introduced by [3] utilizes persistent query logs containing user queries along with the web documents each user clicked on in order to expand the input query. A similar approach is followed by [1] with graphs containing the history of the users' clicks on advertisements to be used instead of query logs. Semantic networks are used in [7] for analyzing search context. A form of query refinement is also attempted in [2] where lexical affinities or, alternatively, co-occurring terms are identified within the results of a query and the information gain of the query terms is calculated in order to retrieve query terms that result in more relevant web results. There, the authors employed a system that incorporates two forms of persistent knowledge, i.e. processing of search engine query logs to detect co-occurrence patterns that can be used for query expansion and usage of the WordNet lexical database for synonym detection.

An alternative category of such systems [4] also requires manual intervention by the user. The user tags queries within a given text, and feature extraction and clustering are used to expand the marked queries. A similar idea is presented in [9]. First, the web results for the input query are retrieved. Next, the user manually characterizes the web results as relevant or irrelevant and finally based on this annotation stage the amelioration of the input query is attempted. Apart for the standard statistical (term frequency) and lexico-syntactic analysis (Part of Speech tagging) the feature selection mechanism of our system employs lexical rules to automatically extract queries from the input text. Such combinations carry a valuable amount of information for the given text and comprise the building blocks for relevant query formulation.

The approach of [6] encompasses tools that automatically infer the contextual information in order to improve the user experience with relevant content. To this end, queries are extracted from the consumed content and different algorithms are employed to provide the best possible news results pertaining to each query. While this system involves a post-processing step for improving the final news results, no step for evaluating the produced queries is performed. In AQUAM, we address this issue by means of a validation mechanism that examines the validity and ambiguity of the automatically produced queries. Thus, irrelevant or vague queries are filtered out, thus improving the quality of the search engine results.

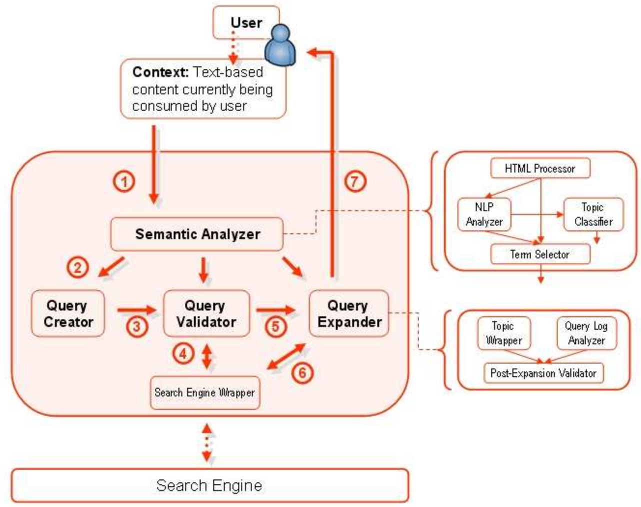

Figure 1: System overview.

3. SYSTEM OVERVIEW

Figure 1 provides an overview of AQUAM. Through this architecture, we attempt to address three issues present in context-based recommenders:

- Feature selection methods may sometimes result into false positive terms for formulating the query so a validation mechanism is required in order to increase the accuracy of the automatically generated queries.
- The user may want to search for information related to the content but using terms that are not explicitly present in the text.
- In environments where webpage-like articles are consumed an efficient method to detect the relevant textual content within an HTML page is required.

AQUAM comprises the following modules, detailed in the next sections:

- The Semantic Analyzer extracts from the text-based input content the semantic metadata, i.e. the significant keywords and the topic to which the content item is classified.
- The Query Creator automatically constructs a set of queries based on the semantic metadata.

- The Query Validator handles the validation of the automatically constructed queries.
- The Query Expander automatically generates one or more search queries using relevant terms that are not present in the initial content item and performs a final post-expansion query validation step.

3.1 Semantic Analysis

The Semantic Analyzer processes the current context (i.e. the text that the user is reading) in order to extract semantic metadata that is required for the automatic query creation process. The module comprises the HTML processor, the NLP analyzer, the Term selector and the Topic classifier.

HTML processor: This unit is applied on webpage-like textual content in order to extract the useful text contained in it. To this end, the proposed technique involves processing of the elements that the webpage comprises. Assume an element e of a webpage w that contains t as its textual part (if any). Further, assume that $l(e)$ stands for the length of e (in characters) and $l(t)$ for the length of t. We define the text density of e as the value $td(e)$ for element e as follows:

$$td(e) = \frac{l(t)}{l(e)}$$

The hypothesis made is, that the higher text density an element has, the more useful text the element contains. This hypothesis is based on the observation that in article-like web pages it is customary to have a plain-text component (article) and decorative elements (side-bars, ads, etc). In order to achieve maximum accuracy in text extraction, we empirically determined certain optimal thresholds for the parameters of the algorithm, namely a lower $ll(t)$ and an upper $ul(t)$ limit for $l(t)$ and a lower $ltd(e)$ and an upper $utd(e)$ limit for the text density value. Then, the HTML processor retains t as valid text, if one of the following two conditions is satisfied:

$$(td(e) > utd(e)) \cup \{(td(e) > ltd(e)) \cap (l(t) > ul(t))\} \quad (1)$$

$$(e = p) \cup \{(td(e) > ltd(e)) \cap (l(t) > ll(t))\} \quad (2)$$

where p stands for the HTML element of type *paragraph* (<p>).

NLP analyzer: This unit encompasses Natural Language Processing (NLP) tools for the semantic analysis of the textual content. The processing steps involve: a) Text preprocessing (stop-word and punctuation removal, plural-to-singular inflection), b) Part-of-Speech (PoS) tagging, c) Single and multi-word Named Entity (NE) recognition (Persons, Organizations and Locations).

Term selector: This unit constructs the semantic vector representation of the content. The selection of the text features is based on three criteria:

- The frequency of the term in the text.
- The location of the term in the text. Terms that appear in the title of the text or in the first part of the text are usually more informative about the text context.
- Number of words per NE. Multi-word NEs take priority over single-word ones.

Topic classifier: This unit classifies the term vector extracted from the content to the most similar topic of interest from a predefined compilation of topics. A simple cosine similarity-based classification algorithm is used for this process.

3.2 Query Creation

The system incorporates a straightforward query creation mechanism based on the notion of entity-typed patterns. These patterns consist of sequences of NE types, i.e. *Persons* (P), *Organizations* (O) and *Locations* (L). For instance, pattern (PO) would result in queries consisting of tokens denoting a Person and an Organization. All possible pattern-based queries are created for a given input piece of text. More specifically, the query creation is based on the following rules pertaining to the NE types (here, *Nouns* (N) are considered as a distinct NE type) of the text:

- *PERSON+PERSON* (PP)
- *PERSON+LOCATION* (PL)
- *PERSON+ORGANIZATION* (PO)
- *PERSON+NOUN* (PN)
- *ORGANIZATION+NOUN* (ON)
- *ORGANIZATION+LOCATION* (OL)
- *NOUN+NOUN* (NN)

At first, all possible queries are created, based on the combination of their types as described in the above rules. Since the number of queries created in this way is extremely large, a query ranking scheme is devised in order to select the M most representative of them. To rank the queries, the sum of their terms significance is used as a metric. The criteria used for the selection of the top N produced queries are the following:

- **Sum of query terms significance.** A score value is assigned to each query term according to its significance (times of appearance) in the text. The score of each query is calculated by adding the scores of its constituent terms.
- **Term diversity.** Terms that have already been included in queries penalize the containing query.
- **Rule diversity.** Rules that have been used should be penalized.

3.3 Query Validation

The Query Validation module has the purpose of estimating the validity of the queries that are produced from the query creator in order to:

- Filter out noisy queries that have falsely been created (e.g. due to a misclassification of terms during the text feature selection).
- Order the remaining valid queries and select a small number of queries that the end-user will receive as recommendation.

Four different techniques are employed and evaluated as a means to implement the query validation function, namely: (a) similarity-based, (b) topic-based, (c) rank-based and (d) clustering-based validation.

Similarity-based query validation: A simple principle for estimating the validity of a given query is to observe the similarity between the text from which the query comes and the results that are returned from a search engine once the query is submitted. If the similarity exceeds a certain threshold on average, then the decision is made that the query under test is valid. As a measure of similarity, several choices are available, among which we used: (1) simple L1-distance with use of TF-IDF features and (2) Jaccard distance [11].

Topic-based query validation: The criterion for considering a query valid according to this approach is the percentage of returned web pages that belong to the same topic as the webpage from which the query under test originated.

Rank-based query validation: This variant of query validation is based on the principle of ranking the queries under test according to the similarity of the returned results with the input content. The top N queries are selected as valid and pass successfully to the query expansion module. More specifically, the process can be decomposed in the following steps:

1. All queries produced by the Query Creator are submitted to the search engine.
2. For each submitted query the top K results are retrieved from the search engine and the average similarity of their texts with the input text is calculated. The similarity is calculated in the same way as in the case of similarity-based query validation.
3. The queries are ranked according to the average similarity calculated for each of them. The top L ranked queries are stored in a separate list called the Ranked Query Set (RQS).
4. Another ranking process follows, in which the terms of all queries are ranked according to their significance (calculated through the containing queries significance). The Ranked Term Sets (RTS) is containing the ranked list of all terms that are possible candidates for the query construction process.
5. During the selection of queries an examination of the ranked results as stated above, takes place. A query is selected as valid if it contains the higher ranked term and none of its terms is contained in the queries that have already been added to the Valid Query Set (VQS).
6. When all possible combinations are examined, the VQS is supplemented with the queries of RQS that follow in the ranking and have not already been selected.

Clustering-based query validation: This variant of query validation is based on clustering the search results returned for each query and deriving a validity score for the query by examining the resulting cluster structure. For web results clustering, standard algorithms such as k-means and its variants (e.g. X-means, G-means) were tested. The first results were not satisfactory due to the high dependence of the clustering results on the number of clusters (determined manually). Therefore, we are currently experimenting with with Expectation-Maximization clustering, which is unsupervised. The second step is to compute a validation score for the query. We are exploring a validation mechanism where the premises of the input space are:

- Cluster number - if most of the results are classified into only one cluster then the proposed query can be considered valid because the results are semantically close to each other.
- Topic attached to each cluster - if most of the clusters are classified into the same topic then the proposed query can be considered valid.
- Position of the results returned by the search engine, such that the more top-ranking results belong to the same cluster, the higher the validity score the query receives.

Evaluation results for this module are still preliminary and thus will not be presented in this paper.

3.4 Query Expansion

Once a set of valid queries is collected, a query expansion step is carried out in order to present the user with queries whose terms were not explicitly present in the text, but would potentially be of interest. The Query Expansion module consists of units that handle topic creation and classification, query log analysis, search and retrieval as well as expansion term selection and post-expansion query validation. These units are described below.

Topic wrapper: This unit involves the construction of prototype vectors containing keywords for predefined topics of interest (e.g. football, music). The steps followed in the creation of the prototype vectors are:

- Submit several topic queries (e.g. 'football', 'music') to the search engine to get results that are associated with these topics. The queries may also be based on lists containing same-kind terms (e.g. football teams, music bands).

Table 1: Specification of result availability metric.

level	name	# hits
7	HUGE	... $> 10^8$
6	VERY-BIG	$10^8 >$ ... $> 10^7$
5	BIG	$10^7 >$ ... $> 10^6$
4	MEDIUM-HIGH	$10^6 >$ ... $> 10^5$
3	MEDIUM-LOW	$10^5 >$ ... $> 10^4$
2	LOW	$10^4 >$ ... $> 10^3$
1	SCARCE	... $< 10^3$

- Extract keywords (nouns) and their corresponding frequencies for each topic.
- Use a list of 'common' keywords to refine the entries in the prototype vectors. We regard as common, those keywords, which have a low IDF (Inverse Document Frequency) value in a TF-IDF scheme.
- The produced prototype vectors are finally stored in an embedded local database for later use.

Query-log analyzer: This unit analyzes user queries available in large query log files. After pre-processing of the log file during which queries with webpage links are removed, log file analysis follows. The idea is to construct a repository of frequently co-occurring terms. Each query term is initially filtered (punctuation, stop words) and stemmed using an implementation of Porter's stemming algorithm [10]. The co-occurring terms are stemmed as well. Then, the times of appearance of co-occurring terms are calculated for each term. The results (terms, stems and frequencies) are locally stored and indexed. During the on-line query expansion step, the system retrieves from the local database the co-occurring terms for a given query term and appends the valid expansion terms to the input query.

Post-expansion query validator: This unit checks the validity of the expansion terms and filters out expansion terms of low quality. The rule used for the validation procedure is the change in the number of results (hits) between the original and the expanded query. The number of hits for a query is available through the API of most search engines. Seven levels of result availability were defined based on the number of hits and presented in Table 1.

If the expanded query results in a change (compared to the original query) in the number of web results of more than one level, then the expansion is considered invalid and therefore it is not presented to the end user.

4. EVALUATION

This section details the evaluation results for each of the modules comprising the system. The test data set consists of 40 URLs pointing to article-like web pages related to four categories: music, movies, football and politics. Evaluation tests based on this data set will be presented that will provide insights into the performance of each of the modules presented in Section 3. As will appear from the tests, there were many cases where the evaluation was possible only by means of personal judgement. In order to mitigate the subjectivity of evaluation in such cases, we employed three independent evaluators/users and considered their majority opinion as the ground truth for the evaluation.

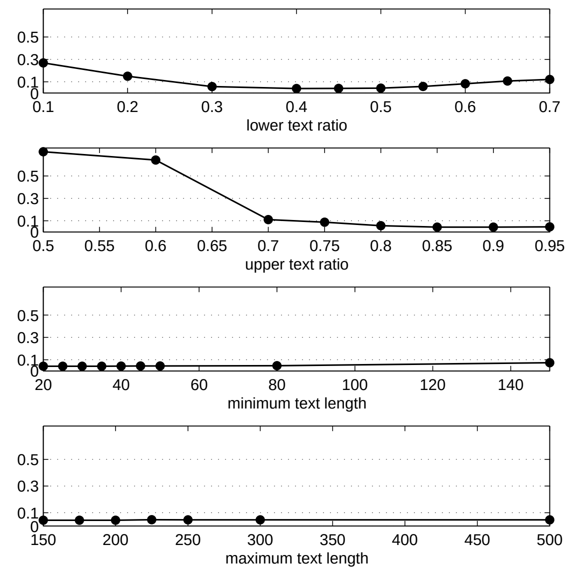

Figure 2: Error dependency on parameters of the HTML processing module.

4.1 Semantic Analysis

HTML processor: Assuming that a human reader views a web page on their browser and selects the text S_{manual} as the interesting text from the web page, then the performance of the module can be assessed by comparing the text S_{manual} with the automatically extracted (from the same web page) text S_{auto}. The texts are considered as arrays of words. By comparing the two texts, one can identify the set S_{common} of shared words between the two texts. Then, the error of the module can be quantified as follows:

$$e = \frac{1}{2}\left(\frac{\#S_{manual} - \#S_{common}}{\#S_{manual}} + \frac{\#S_{auto} - \#S_{common}}{\#S_{auto}}\right)$$

where $\#S$ is the number of words of the word array S. The performance of the module depends on the values of the parameters $ltd(e)$, $utd(e)$, $ll(t)$, and $ul(t)$. This dependency is illustrated in Figure 2. According to this, the module performance remains stable for a wide range of the parameters. The expected error rate for parameters values set within this range is approximately 4%.

NLP analyzer / Topic Classifier / Term Selector: Since the accuracy of NLP techniques and topic classification methods are widely researched topics, no evaluation test results are presented here. It was confirmed by inspection of the results that the accuracy of these modules was sufficiently high so as not to impede the performance of the subsequent modules. Thus, the outcome of testing these components was to gain understanding on its pruning effec-

Table 2: Average statistics for the semantic analysis process. The following abbreviations were used: (a) Tok = Tokenization, (b) SWR = Stop Word Removal, (c) NLP = Natural Language Processing, and (d) TS = Term Selection

	Tok	SWR	NLP	TS
Terms	229.8	160.9	78.6	11.8
Person	-	-	9.2	6.8
Organization	-	-	2.5	2.2
Location	-	-	3.8	0.9
Noun	-	-	63.1	1.9

Table 3: Query creation rule success rates.

Rule ID	$\frac{Q_{valid}}{Q_{valid}+Q_{invalid}}$	Success Rate
PP	181 / 247	73.3%
PL	66 / 100	66.0%
PO	93 / 172	54.4%
PN	206 / 401	51.4%
OL	16 / 43	37.2%
ON	35 / 143	24.5%
NN	5 / 38	13.2%

tiveness. Table 2 presents the number of tokens and entities remaining after each step of the semantic analysis process is applied. It appears that the term selection step is of crucial importance in order to limit the number of possible queries that can be formulated as combinations of the content terms.

4.2 Query Creation

The evaluation tests of the Query Creation module establish the overall effectiveness of the entity-type pattern rules presented in Section 3.2 and the dependency of their effectiveness on the input topic. Table 3 summarizes the mean success of each pattern rule over the whole test data set. In principle, it appears that the rules involving entities of type Person result in more relevant queries in comparison to the rest of the rules.

Table 4 illustrates the dependency of the query creation rule effectiveness on the topic of the input text. For instance, the music topic calls for a different query creation rule strategy since the most effective rules for this topic are the ones comprising Person and Organization entities (note that music bands are recognized by NLP as Organizations).

Table 4: Query creation rule success rates.

Rule ID	Success Rate			
	football	movies	music	politics
PP	**100.0%**	**83.1%**	34.0%	**63.8%**
PL	**100.0%**	76.5%	17.4%	54.2%
PO	**100.0%**	47.7%	**40.9%**	51.6%
PN	91.9%	45.7%	39.3%	40.3%
OL	77.8%	9.0%	37.5%	33.3%
ON	55.6%	7.0%	34.4%	20.8%
NN	25.0%	0.0%	20.0%	10.0%

Table 5: Definition of classification performance metrics.

Query-set		Ground truth	
		YES	**NO**
Automatic decision	**YES**	TP	FP
	NO	FN	TN

Table 6: Comparison of query validation methods.

Method	π	ρ	f_α ($\alpha = 0.5$)
Sim (TF-IDF)	48.56%	**98.06%**	58.39%
Sim (Jaccard)	50.00%	98.00%	59.76%
Topic-based	**52.17%**	85.16%	**59.90%**
Rank-based	48.37%	96.13%	57.97%

4.3 Query Validation

The performance of the proposed query validation mechanisms has been assessed by means of standard IR success metrics, namely precision π, recall ρ, and F-measure f_α, as computed by the following equations:

$$\pi = \frac{TP}{TP + FP}, \qquad \rho = \frac{TP}{TP + FN}, \qquad f_\alpha = \frac{1+\alpha}{\alpha \times \pi + \rho}$$

where TP, FP and FN stand for True Positive, False Positive and False Negative respectively and are specified in Table 5.

The ground truth for the particular data set was based on the opinion of three independent annotators using a majority vote. Table 6 presents the maximum attainable performance measures for the different implementations of the query validation module as described in Section 3.3. Such performance was attained by empirically tuning the parameters of the methods. From the table, it appears that the topic-based and Jaccard similarity-based are the two most effective query validation methods.

Table 7 provides insight into the filtering rate of one of the query validation modules (namely the Jaccard similarity-based validation). The filtering rate is presented for two modes of operation, namely Strict (S) and Relaxed (R), based on two different parameter values for the module. On average, by employing the S filtering mode, the user will be provided with a set of recommended queries that is less than half the size of the original query set.

Table 7: Filtering rate of the query validation module. The number of 'passing' queries to the total input queries is presented together with the precision-recall ($\pi - \rho$) level.

		$\frac{\#pass}{\#total}$	pass (%)	(π-ρ)
football	S	85 / 234	36.3	95.3 - 38.0
	R	217 / 234	92.7	91.7 / 93.4
movies	S	135 / 319	42.3	53.3 - 46.5
	R	304 / 319	95.3	49.7 / 97.4
music	S	97 / 227	42.7	26.8 - 33.3
	R	200 / 227	88.1	32.5 / 83.3
politics	S	158 / 360	43.9	42.4 - 43.2
	R	349 / 360	96.9	42.9 / 96.8

Table 8: Comparison of the two proposed query expansion methods. Accuracy is computed as the ratio $\frac{TP+TN}{TP+TN+FP+FN}$, where TP, TN, FP and FN are defined in Table 5.

	Topic-based	**Query Log-based**
Accuracy	91.80% (280 / 305)	57.78% (104 / 180)
($\boldsymbol{\pi}$-$\boldsymbol{\rho}$)	85.88% / 97.22%	57.14% / 99.04%

Table 9: Sample query expansions produced by the two proposed query expansion methods. In the particular cases presented here, the query-log based expansion method appears superior to the topic-based one.

Input Query	**Topic-based**	**Query Log-based**
sir alex david beckham	+ results + reports + supporters	+ boxing + book + soccer + owen
ben haim game	+ fan + team	+ gallery
michael myers rob zombie	+ board + star	+ bar + horror + living + music
theron war	+ matrix	+ pics + charlize + photos
madonna malawi	+ leader	+ child
president bush campaign	+ article + archival + minister	+ approval + attack

4.4 Query Expansion

The performance of the Query expansion module was based on manual evaluation of the expansion results produced by the two alternative expansion schemes presented in Section 3.4, namely topic-based and query log-based expansion. Table 8 illustrates the performance differences between the two methods. The *accuracy* of the methods was estimated by means of the ratio of $TP+TN$ over the total number of evaluated queries. The large difference in performance between topic-based and query log-based expansion can be largely attributed to the fact that outdated query logs were used by the query log-based expander (the most recent query log set was the one released by AOL in 2006). Table 9 lists several sample expansion results that are indicative of the performance of the two expansion modules. In the particular examples, the query log-based expansion appears to yield more interesting expansion terms.

Finally, the performance of the post-expansion query validation mechanism was evaluated. It appears from Table 10 that the presence of this module in the system is crucial in improving the relevance of the final queries presented to the user. The operation of the post-expansion query validation resulted in a 20% increase in the accuracy of the proposed queries for both expansion methods. Table 11 illustrates by means of examples how the different query expansions are handled by the post-expansion technique.

Table 10: Comparison of accuracy between using and not using the proposed validation mechanism.

	Topic-based	**Query Log-based**
Validation off	71.11%	27.13%
Validation on	91.80%	57.14%

Table 11: Sample results based on the proposed post expansion validation mechanism. The Validation Result can be either Invalid (I) or Valid (V) and the *Reason* depicts the change in the number of hits from the original to the expansion query (see Table 1 to translate the numbers of the *Reason* column to number of hits).

Query	**Proposed Expansion**	**Validation Result**	**Reason**
salma hayek series	discography	I	$5 \rightarrow 2$
	music	V	$5 \rightarrow 4$
bill murray nbc	award	V	$5 \rightarrow 5$
	bates	I	$5 \rightarrow 3$
young jose mourinho	analysis	V	$4 \rightarrow 3$
	alumni	I	$4 \rightarrow 1$

5. CONCLUSIONS

This paper tackled several interesting research and technical issues involved in the development of AQUAM, an automatic query recommendation system intended for use in context-relevant search scenarios, as a means to alleviate users from the burden of typing search queries.

First, we made use HTML and NLP analysis techniques in order to extract from the input web content a set of semantic metadata, namely the most prominent terms, their types (POS and NE), as well as the topic of the text. Further, we devised an entity-template based technique to formulate queries. We observed that different query formulation rules are suitable depending on the topic of the input content.

In addition, four variants of query validation were presented and two query expansion schemes along with a post-expansion validation technique were developed. Evaluation tests were carried out to confirm the accuracy and robustness of the individual AQUAM modules. More specifically, the topic-based query validation proved to be the most effective in filtering irrelevant queries, and the topic-based query expansion appeared more accurate in augmenting the input queries with expansion terms. We also found that the query log-based expansion mechanism is promising under the condition that up-to-date query logs are available. Finally, the devised post-expansion validation technique contributed to an improvement of 20% in the accuracy of the proposed queries.

Future work involves the refinement and evaluation of the proposed clustering-based query validation technique. Furthermore, we plan to investigate ways of automatically setting the parameters of the query validation methods avoiding the need for performance tuning tests. Finally, we plan to study potential methods for performing query validation without resorting to third-party search engines.

6. ACKNOWLEDGMENTS

Part of the costs incurred by this work were covered by the BOEMIE project, partially funded by the European Commission, under contract number FP6-027538. We would also like to thank the reviewers for their insightful comments that contributed to the paper improvement.

7. REFERENCES

[1] I. Antonellis, H. Garcia-Molina, and C.-C. Chang. Simrank++: query rewriting through link analysis of the clickgraph (poster). In *WWW*, pages 1177–1178, 2008.

[2] D. Carmel, E. Farchi, Y. Petruschka, and A. Soffer. Automatic query refinement using lexical affinities with maximal information gain. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 283–290, New York, NY, USA, 2002. ACM.

[3] H. Cui, J.-R. Wen, J.-Y. Nie, and W.-Y. Ma. Query expansion by mining user logs. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):829–839, 2003.

[4] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin. Placing search in context: the concept revisited. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 406–414, New York, NY, USA, 2001. ACM.

[5] E. Glover, G. Flake, S. Lawrence, W. P. Birmingham, A. Kruger, C. L. Giles, and D. Pennock. Improving category specific web search by learning query modifications. In *Symposium on Applications and the Internet, SAINT*, pages 23–31, San Diego, CA, January 8–12 2001. IEEE Computer Society, Los Alamitos, CA.

[6] M. Henzinger, B.-W. Chang, B. Milch, and S. Brin. Query-free news search. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 1–10, New York, NY, USA, 2003. ACM.

[7] R. Kraft, C. C. Chang, F. Maghoul, and R. Kumar. Searching with context. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 477–486, New York, NY, USA, 2006. ACM.

[8] S. Lawrence. Context in web search. *IEEE Data Engineering Bulletin*, 23(3):25–32, 2000.

[9] S. Patro, V. M. Malhotra, and D. Johnson. An algorithm to use feedback on viewed documents to improve web query. In *WEBIST (Selected Papers)*, pages 177–189, 2006.

[10] M. F. Porter. An algorithm for suffix stripping. pages 313–316, 1997.

[11] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[12] J. Yi, F. Maghoul, and J. Pedersen. Deciphering mobile search patterns: a study of yahoo! mobile search queries. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 257–266, New York, NY, USA, 2008. ACM.