User Modelling for Adaptive Question Answering and Information Retrieval

Silvia Quarteroni and **Suresh Manandhar**
Department of Computer Science
University of York
York, YO10 5DD
United Kingdom
{silvia,suresh}@cs.york.ac.uk

Abstract

Most question answering (QA) and information retrieval (IR) systems are insensitive to different users' needs and preferences, and also to the existence of multiple, complex or controversial answers. We propose the notion of adaptivity in QA and IR by introducing a hybrid QA-IR system based on a user model. Our current prototype filters and re-ranks the query results returned by a search engine according to their reading level. This is particularly useful in school environments, where it is most needed to adjust the presentation of complex information to the pupils' level of understanding. ***Keywords:*** *adaptive systems, question answering, information retrieval, user modelling, readability.*

Introduction

Question answering (QA) systems are information retrieval systems accepting queries in natural language and returning the results in the form of sentences (or paragraphs, or phrases). This is different from standard information retrieval (IR) where results are presented in the form of a ranked list of query-relevant documents. Such a finer answer presentation is possible thanks to the application of computational linguistics techniques in order to filter irrelevant documents, and of a consistent amount of question preprocessing and result post-processing.

However, in most QA systems the output remains independent of the questioner's characteristics, goals and needs. In other words, there is a lack of *user modelling*: a 10-year-old and a University History student may need different answers to the question: "*When did the Middle Ages begin?*".

Secondly, most QA systems focus on *factual* questions, i.e. questions concerning people, dates, numerical quantities etc., which can generally be answered by a short sentence or phrase (Kwok, Etzioni, & Weld 2001). The mainstream approach to QA, represented by the TREC-QA evaluation campaign[1], has long encouraged the assumption that a "good" system is one that returns the "correct" answer in the shortest possible formulation. We argue that this approach may be unsuitable for some queries, in particular for those with multiple, complex or controversial answers (e.g. "*What were the causes of World War II?*"): in such situations a short paragraph or text snippet is more appropriate than exact answer spotting. Some users may prefer longer answers than a simple phrase, in order to quickly grasp the answer context: for instance, the answer to "*What is a metaphor?*" may be better understood with the inclusion of examples.

Although recent editions of TREC (Voorhees 2004) denoted more interest towards some types of non-factoid *questions* such as list and definitional ones, the issue which we believed is not sufficiently addressed is the detection and treatment of non-factoid *answers*. We thus believe that TREC-QA evaluation is not the optimal approach to assess the performance of systems providing such types of answers. Moreover, TREC does not evaluate the adaptivity of the QA systems to the users' search needs, by assuming a fixed questioner profile instead (Voorhees 2004).

Finally, QA systems rarely interact with the user: the typical session starts with a query and ends with its result. A possible solution to this problem would be a dialogue-based system with a history component: however this is not the subject of this paper, which addresses in particular the first and third of the deficiencies mentioned above.

To solve the first issue, we propose an *adaptive* system which adjusts its output with respect to a user model. The system can be seen as an enhanced IR system which adapts both the content and presentation of the final results, improving their *quality*. We explain how both QA and IR systems can benefit from the contribution of user models, and especially how these can be used to filter the information presented as an answer. We show an application where one UM dimension – reading level – is used as a filter for documents retrieved for school assignments. We address the third problem by designing an evaluation framework for the system inspired by search engine evaluation, and in particular by user-centered metrics. A QA system able to distinguish between simple/factoid answers and more complex answers – presenting them in a TREC-style manner in the first case and more appropriately in the second – will be the solution to the second problem and the final objective of our research.

Global Architecture

The high-level architecture as represented in Fig. 1 shows the basic components of the system, the QA module and the user model.

[1] http://trec.nist.gov

QA module

The structure of the QA module, described in detail in the following section, is organized according to the three-tier partition underlying most state-of-the-art systems: 1) question processing, 2) document retrieval, 3) answer generation. The module makes use of a web search engine for document retrieval and consults the user model to obtain the necessary criteria to filter and re-rank the search engine results and eventually to present them appropriately to the user. The QA module would fit nicely in the framework of a dialogue-based system, in which the dialogue interface would be both a source of query terms and answer formulator; however this paper focuses on the module as a standalone system.

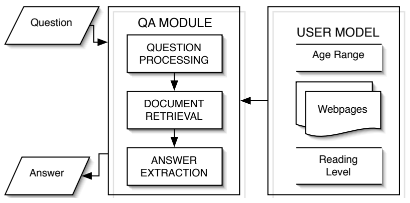

Figure 1: High level system architecture

User model

Depending on the application of interest, the user model (UM) can be designed to suit the information needs of the user in different ways. As our current application is a learning-oriented, web-based system, we have defined the UM as consisting of the user's:
1) age range, $a \in \{7..11, 11..16, adult\}$;
2) reading level, $r \in \{poor, medium, good\}$;
3) set of documents/webpages of interest/bookmarks, w.
Analogies can be found with the SeAn (Ardissono, Console, & Torre 2001), and SiteIF (Magnini & Strapparava 2001) news recommender systems, where information such as age and browsing history, resp. are part of the UM.
This information is explicitly collected from the user; eventually, a dialogue framework with a history component will provide additional data to update the UM. Although in this paper we focus on adapting the presentation of the search results using the reading level parameter only, we are now developing the strategy to account for the user's interests. These will be extracted both statically from the user's document set w (e.g. using keyphrase extraction – cf. **Retrieval**) and dynamically from the dialogue history. They could be weighed and combined using either implicit relevance feedback (Teevan, Dumais, & Horvitz 2005) or multi-attribute utility metrics such as Rank Order Centroids (Edwards & Barron 1994).

Related work

To our knowledge, our system is among the first to address the need for a different approach to non-factual questions. The typical web-based QA system, e.g. MULDER (Kwok, Etzioni, & Weld 2001), is structured in three phases: question processing, answer extraction which obtains relevant snippets from the retrieved documents, and answer generation which produces a ranked list of answers. Although we maintain this partition as a general framework for our system, a significant aspect of novelty in our architecture is that the QA component is supported by the UM. Additionally, we have changed the relative importance of the different tiers: we drastically reduce the amount of NLP techniques applied during question processing and answer generation, while giving more importance to the post-retrieval phase and to the role of the UM. In this, our approach differs from that of most QA systems that typically perform intensive lexical, syntactic, logic and semantic processing (Harabagiu, Pasca, & Maiorano 2000): this allows us to evaluate the usefulness of user modelling.

Applications

Potential applications of the adaptive system include:

- *Learning environments*: our current application, YourQA (http://www.cs.york.ac.uk/aig/aqua/), is a system helping school children to find information on the Web for their assignments. Our goal here is to adjust the search engine results to the pupils' age and reading level. This is particularly useful in areas such as science and history, where the complexity of the information available online could be a barrier to the pupil's understanding.
- *Access to knowledge and culture*, in particular when a comparative approach is needed: for example the question *"What subjects did Michelangelo paint?"* can have several inter-related answers. A similar approach can be found in ALFRESCO (Stock & Team 1993), an interactive, natural language system presenting frescoes. Although it does not perform user modelling, ALFRESCO can give punctual information (e.g. location) or more complex descriptions about a fresco, and is able to propose related artworks to the user.
- *Leisure*: web personalization tools show that adaptivity is a key quality in a web application (especially in the field of *infotainment*). For instance, the SeAN news recommender (Ardissono, Console, & Torre 2001) is based on a UM which presents news according to the user's preferred topics and level of detail. Our system aims at introducing user modelling in the wider context of IR; also, our UM incorporates not only the interests but also the reading level of the user.

QA Module

In this section we discuss the individual subcomponents of the QA module and describe how the information flow is organized among them (see diagram in Fig. 2).

Question processing

Query Expansion The first step performed by our system is query expansion, which is common practice in most QA systems (Harabagiu, Pasca, & Maiorano 2000) that expands the original query with alternative formulations with the purpose of increasing the number of documents retrieved by the search engine (i.e. recall). These are created by replacing query words with synonyms, which we obtain using the WordNet 2.0 lexical database[2].

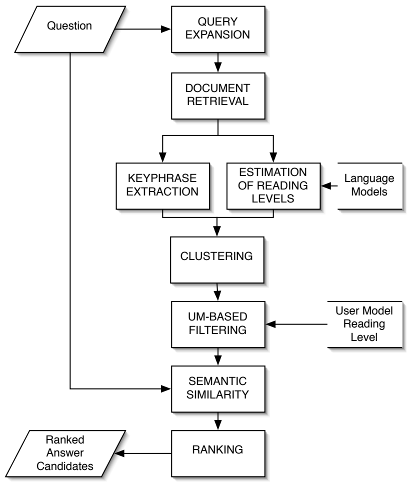

Figure 2: Diagram of the QA module

Retrieval

Document retrieval We use Google[3] to retrieve the top 20 documents returned for each of the queries issued from the query expansion phase. These documents are the basis for the subsequent steps, which will progressively narrow the part of the text where relevant information is located.

Keyphrase extraction Once the documents are retrieved, we perform keyphrase extraction to determine their most relevant topics using Kea (Witten *et al.* 1999), an extractor based on Naïve Bayes classification. Kea first splits each document into phrases and then takes short subsequences of these initial phrases as candidate keyphrases. Two attributes are used to discriminate between keyphrases and non-keyphrases: TF $\times$ IDF score within the set of retrieved documents (in short, T), and the offset D in the document of the phrase's first appearance. T and D are assumed independent following Naïve Bayes; the probability that a phrase is a keyphrase is therefore:
$P(key|T,D) = \frac{P(T|key) \cdot P(D|key) \cdot P(key)}{P(T,D)}$, where $P(key)$ is the a priori probability that the phrase is a keyphrase, $P(T|key)$ the probability that it has $TF \times IDF$ score T, $P(D|key)$ the probability that it has offset D and $P(T|D)$ a normalization factor. Kea outputs a list of phrases ranked according to the probability computed as above, among which we select the top three as keyphrases for each document.

Estimation of reading levels In order to better match the search result presentation to the reading ability of the users, we estimate the reading difficulty of the retrieved documents. To perform the estimation, we have chosen to use the Smoothed Unigram Model (Collins-Thompson & Callan 2004), a variation of a Multinomial Bayes classifier which is particularly successful for Web documents if compared to approaches such as Flesch-Kincaid (Kincaid *et al.* 1975).

It proceeds in two phases: in the training phase, given a range of reading levels, we collect a set of representative documents for each level. We then build a unigram language model lm_s for each set s; the model consists of a list of word stems appearing in the training documents and their individual probabilities. Our unigram language models account for the following reading proficiency levels in English: 1) *poor*, suitable for ages 7–11; 2) *medium*, suitable for ages 11–16; 3) *good*, suitable for adults. Our training data consists of about 180 Web pages (from e.g. "BBC schools"[4], "Cassini Huygens for schools" [5]), explicitly annotated by the publishers according to the three reading levels above.

The reading level assigned to an unknown document D will be that of the language model lm_i most likely to have generated D. In the test phase, such likelihood is estimated as:

$$L(lm_i|D) = \sum_{w \in D} C(w,D) \cdot log(P(w|lm_i))$$

where w is a word in D, $C(w,d)$ represents the number of occurrences of w in D and $P(w|lm_i)$ is the probability that w occurs in lm_i (approximated by its frequency).

Clustering In order to obtain an indicator of inter-document relatedness, we apply document clustering (Steinbach, Karypid, & Kumar 2000) to group them using both their estimated reading difficulty and their topic (i.e. their keyphrases). In particular we chose to implement hierarchical clustering (Fung, Wangy, & Ester 2003) as it produces a tree structure which is particularly intuitive to analyse: each leaf corresponds to one document, and sibling leaves denote documents that are strongly related both in topic and in reading difficulty. We implemented the Cobweb hierarchical algorithm using the WEKA suite of tools (Witten & Frank 2000) . Fig. 3 illustrates an example cluster tree for the the query: *"Who painted the Sistine Chapel?"*. Leaf labels represent document keyphrases, and ovals represent cluster nodes (oval labels are the common keyphrases of underlying leaves). Three different reading levels are identified: good (leaf 3, leaves under nodes 8 and 12), medium (leaf 15, leaves under node 4), and poor (leaf 2).

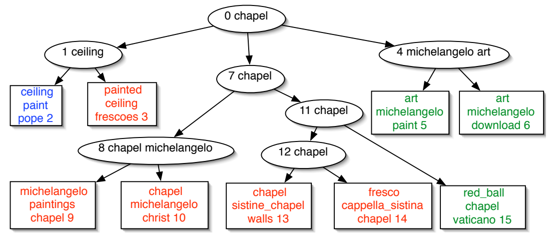
0 chapel
1 ceiling
7 chapel
4 michelangelo art
ceiling paint pope 2
painted ceiling frescoes 3
11 chapel
art michelangelo paint 5
art michelangelo download 6
8 chapel michelangelo
12 chapel
michelangelo paintings chapel 9
chapel michelangelo christ 10
chapel sistine_chapel walls 13
fresco cappella_sistina chapel 14
red_ball chapel vaticano 15

Figure 3: Tree for *"Who painted the Sistine Chapel?"*

[2] `http://wordnet.princeton.edu`
[3] `http://www.google.com`
[4] `http://bbc.co.uk/schools`
[5] `http://www.pparc.ac.uk/Ed/ch/Home.htm`

Answer extraction

The purpose of answer extraction is to adjust the query results to the needs of the user and to present the most interesting excerpts of the retrieved documents according to both the user's query topics and reading level. This process follows the diagram in Fig. 2: we use the UM to filter the clustered documents, then compute the similarity between the question and the filtered document snippets in order to return the best ones in a ranked list.

UM-based filtering The documents in the cluster tree are filtered according to the user's reading level: only those compatible with the user's reading ability are retained as candidates for further analysis. However, if the number of retained documents does not exceed a given threshold, we accept in our candidate set part of the documents having the next lowest readability – in case the user's reading ability is good or medium – or a medium readability – in case the user has a low reading level.

Semantic similarity Within each of the documents retained, we seek the sentence which is semantically most relevant to the query by applying the following semantic distance metric. Given a sentence p and the query q, we represent them as two sets of words $\mathcal{P} = \{pw_1, \dots, pw_m\}$ and $\mathcal{Q} = \{qw_1, \dots, qw_n\}$; their semantic distance is then:

$$dist_q(p) = \sum_{1 \leq i \leq m} min_j [d(pw_i, qw_j)]$$

where $d(pw_i, qw_j)$ represents the word-level distance between qw_j and pw_i. The intuition is that for each question word, we find the word in the candidate answer sentence which minimizes the word-level semantic distance and we compute the sum of such minima (DeBoni & Manandhar 2003). This coarse, bag-of-words approach is sufficient to narrow the location of interesting information in the document.

To compute $d(w_1, w_2)$, the word-level distance between a word w_1 and a candidate related word w_2, we adopt the metric in (Jiang & Conrath 1997). Given a lexical database $\mathcal{W}$ (WordNet 2.0 in our case), it combines $P(w_1)$ and $P(w_2)$, the frequencies of w_1 resp. w_2 in $\mathcal{W}$, with the frequency of their *most specific common superordinate*[6] ($mscs_{1,2}$):

$$d(w_1, w_2) = IC_1 + IC_2 - 2 \cdot IC(mscs_{1,2}),$$

where $IC_i = log^{-1} P(w_i)$ is the information content of w_i.

Ranking In each candidate document D, for a given query q, we locate the sentence $p^* = argmin_{p \in D}[dist_q(p)]$; then, $dist_q(p^*)$ becomes the document score. Additionally, we take advantage of the document tree created during the clustering phase, as this structure constitutes an indicative thematic partition of the query results. To do this, document *clusters* are also ranked: the score of a cluster is the maximal score obtained by its underlying documents. This allows to present results grouped by cluster in decreasing order of document score.

[6]for instance, $mscs(cat, panther) = feline$

Answer presentation We present our answers in an HTML page, where results are listed following the ranking described above. Each result consists of the title and clickable URL of the originating document, and the passage where the sentence which best answers the query is located and highlighted. Question keywords and potentially useful information such as named entities are in colour.

Results

We report the results of running our system on a range of queries, which include factoid/simple, complex and controversial questions[7].

Simple answer

As an example of a simple query, we present the results for: *"Who painted the Sistine Chapel?"*, the system returned the following answer snippets.

- for UM_{good}: ***"Sistine Chapel (sis-teen). A chapel adjoining Saint Peter's Basilica, noted for the frescoes of biblical subject painted by Michelangelo on its walls and ceilings."***
- for UM_{med}: ***"In all Michelangelo painted more than 300 different figures on the Sistine Chapel ceiling."***
- for UM_{poor}: ***"My name is Jacopo L'Indaco and I was an assistant to Michelangelo when he painted the Sistine Chapel."***

To obtain the above answers the system was run 3 times with the values defined earlier for the UM reading level parameter. As we can see, in all cases the correct information is present although not always explicitly, as in the second case. This is understandable since our current similarity metric does not perform any deep processing. The difference in language complexity is clearly discernible in the different answers.

Complex answer

We illustrate the results of our system with the query *"Definition of metaphor"*, which relates to a difficult concept. Our top results, highlighted in boldface within their containing passages, are:

- for UM_{good}: *"**Definition of Metaphor. Metaphor is a process by which we conceive "one thing in terms of another, and its primary function is understanding" (Lakoff and Johnson, Metaphors We Live By, pg 36).** That is, we often do (and must) talk about things that do not physically exist, that we cannot reach out and touch or take apart with our hands to see what is inside and how it works: love, time, anger, life. [...] We make them tangible by, in a process of thought, applying models of physical things to them: love is falling; argument is war. "*
- for UM_{med}: *"**Definition: A metaphor is a figure of speech that compares unlike things by saying that one thing is the other.** Often metaphors are simple comparisons, but they can be extended so that different aspects of the things compared are treated separately. "*

[7]Notice that this partition is not to be interpreted as a methodological division, as we currently approach complex and controversial answers the same way.

- for UM_{poor}, we did not find any document exactly matching the reading level, we thus returned the results for UM_{med} (see above).

As before, there is a clear difference in the language complexity of the two answers.

Controversial answer

We illustrate this case using the query *"Who wrote the Iliad?"*, which is a subject of debate: the existence of a single author, as well as the historicity of the poet Homer are under discussion. These are among our top results:

- for UM_{good}: *"Most Classicists would agree that, whether there was ever such a composer as "Homer" or not, the Homeric poems are the product of an oral tradition, a generations-old technique that was the collective inheritance of many singer-poets, aoidoi. [...]* ***Could the Iliad and Odyssey have been oral-formulaic poems, composed on the spot by the poet using a collection of memorized traditional verses and phases?"***
- for UM_{med}: *"No reliable ancient evidence for Homer – earliest traditions involve conjecture (e.g. conflicting claims to be his place of origin) and legend (e.g. Homer as son of river-god).* ***General ancient assumption that same poet wrote Iliad and Odyssey (and possibly other poems) questioned by many modern scholars: differences explained biographically in ancient world (e g wrote Od. in old age); but similarities could be due to imitation."***
- for UM_{poor}: ***"Homer wrote The Iliad and The Odyssey (at least, supposedly a blind bard named "Homer" did)."***

In this case we can see how the problem of attribution of the *Iliad* is made clearly visible: in the three results, document passages provide a context which helps to explain such controversy at different levels of difficulty.

Evaluation

Methodology

Our system is not a QA system in the strict sense, as it does not single out one correct answer phrase. The key goal of our system is the improved satisfaction of the user towards more adaptive results, that suit the user's reading level. A user-centred evaluation methodology that assesses how the system meets individual information needs is more appropriate for our system, since it is intended to handle complex and controversial answers in addition to factoid answers.

We draw our evaluation guidelines from (Su 2003), which proposes a comprehensive search engine evaluation model. We define the following metrics (Tab. 1):

1. *Relevance*: we compute strict precision (P_1) – the ratio between the number of results rated as relevant and all the returned results, and loose precision (P_2) – the ratio between the number of results rated as relevant or partially relevant and all the returned results.
2. *User satisfaction*: a 7-point Likert scale [8] is used to assess satisfaction with loose precision of results (S_1) and with the success of the query (S_2).
3. *Reading level accuracy (A_r)*. This metric was not present in (Su 2003) and has been introduced to assess the reading level estimation. Given the set $\mathcal{R}$ of results returned by the system for a reading level r, it is the ratio between the number of documents $\in \mathcal{R}$ rated by the users as suitable for r and $|\mathcal{R}|$. We compute A_{good}, A_{med} and A_{poor}.
4. *Overall utility (U)*: the search session as a whole is assessed via a 7-point Likert scale.

We have discarded some of the metrics proposed by (Su 2003) linked to more technical aspects of search engines and response time, which has not been considered an issue at the present stage. Also, we exclude metrics relating to the user interface which is not relevant for this study.

Metric	*field*	*description*
Relevance	P_1	strict precision
	P_2	loose precision
Satisfaction	S_1	with loose precision
	S_2	with query success
Accuracy	A_{good}	good reading level
	A_{med}	medium reading level
	A_{poor}	poor reading level
Utility	U	overall session

Table 1: Summary of evaluation metrics

Evaluation results

We performed our evaluation by running 24 queries – samples are reported in Tab. 3– on both Google and our system[9]. We submitted the results (i.e. Google result snippets and YourQA result passages) to 20 evaluators. Each evaluator provided feedback on the relevance of each fragment proposed, on the success and result readability of the single queries, and on the overall utility of the system; values were hence computed for the metrics in Tab. 1.

Relevance The precision results (Tab. 2) for the whole search session were computed by averaging the values obtained for the six queries. Although quite close, they show a 10-15% difference in favour of YourQA for both strict precision (P_1) and loose precision (P_2). This proves that the coarse semantic processing applied and the context visualisation contribute to the creation of more relevant passages.

	P_1	P_2	S_1	S_2	U
Google	0.39	0.63	4.70	4.61	4.59
YourQA	0.51	0.79	5.39	5.39	5.57

Table 2: Evaluation results

[8]This measure – ranging from 1= "extremely unsatisfactory" to 7="extremely satisfactory" – is particularly suitable to assess the degree to which the system meets the user's search needs. It was reported in (Su 1991) as the best single measure for information retrieval among 20 tested.

[9]To make the two systems more comparable, we turned off query expansion and only submitted the original question sentence

User satisfaction After each of the six queries, we asked evaluators the following questions: *"How would you rate the ratio of relevant/partly relevant documents returned?"* (assessing S_1) and *"How would you rate the success of this search?"* (assessing S_2). Tab. 2 denotes a higher level of satisfaction tributed to YourQA in both cases.

Reading level accuracy Adaptivity to the users' reading level is the distinguishing feature of YourQA: we were thus particularly interested in its performance in this respect. Tab. 3 shows that altogether, evaluators found our results appropriate for the reading levels to which they were assigned. The accuracy tended to decrease (from 94% to 72%) with the level: this was predictable as it is more constraining to conform to a lower reading level than to a higher one. However it also suggests that our estimation of document difficulty was perhaps too "optimisitic": we are now working with better quality training data which provides more accurate language models.

Query	A_g	A_m	A_p
When did the Middle Ages begin?	0.91	0.82	0.68
Who painted the Sistine Chapel?	0.85	0.72	0.79
When did the Romans invade Britain?	0.87	0.74	0.82
Who was a famous cubist?	0.90	0.75	0.85
Who was the first American in space?	0.94	0.80	0.72
Definition of metaphor	0.95	0.81	0.38
average	**0.94**	**0.85**	**0.72**

Table 3: Sample queries and reading level accuracy

Overall utility At the end of the whole session, users answered the question: *"Overall, how was this search session?"* relating to their search experience with Google and YourQA. The values obtained for U in Tab. 2 show a clear preference (a difference of 1 on the 7-point scale) of the users for YourQA, which is very positive considering that it represents their general judgement on the system.

Future work We plan to run a larger evaluation by including more metrics, such as user *vs* system ranking of results and the contribution of cluster by cluster presentation. Additionally, we will analyse in depth the result of the evaluation metrics with respect to the individual reading levels and the different types of questions proposed. We also intend to involve more evaluators, selecting representatives of all three reading levels (in the present case all were adults).

Conclusion

A user-tailored QA system is proposed where a user model contributes to: 1) elaborating answers corresponding to the user's needs; 2) presenting them efficiently. In this paper we have focused on the user adaptivity issue and discussed the architecture of the core QA module, which we are currently testing. Preliminary results show a positive feedback from human assessors, which we hope to confirm after a more extensive evaluation. Our next goal will be implementing a dialogue interface to improve the system's interactivity.

References

Ardissono, L.; Console, L.; and Torre, I. 2001. An adaptive system for the personalized access to news. *AI Commun.* 14(3):129–147.

Collins-Thompson, K., and Callan, J. P. 2004. A language modeling approach to predicting reading difficulty. In *Proceedings of HLT/NAACL*.

DeBoni, M., and Manandhar, S. 2003. An analysis of clarification dialogue for question answering.

Edwards, W., and Barron, F. 1994. Smarts and smarter: Improved simple methods for multiattribute utility measurements. 60:306–325.

Fung, B.; Wangy, K.; and Ester, M. 2003. Hierarchical document clustering using frequent itemsets. In *Proceedings of the SIAM International Conference on Data Mining*.

Harabagiu, S.; Pasca, M.; and Maiorano, S. 2000. Experiments with open-domain textual question answering. In Kaufmann, M., ed., *Proceedings of COLING-2000*.

Jiang, J. J., and Conrath, D. W. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the International Conference Research on Computational Linguistics (ROCLING X)*.

Kincaid, J.; Fishburne, R.; Rodgers, R.; and Chissom, B. 1975. Derivation of new readability formulas for navy enlisted personnel. Technical Report Branch Report 8-75, Chief of Naval Training.

Kwok, C. C. T.; Etzioni, O.; and Weld, D. S. 2001. Scaling qa to the web. In *World Wide Web*, 150–161.

Magnini, B., and Strapparava, C. 2001. Improving user modelling with content-based techniques. In *UM: Proceedings of the 8th Int. Conference*.

Steinbach, M.; Karypid, G.; and Kumar, V. 2000. A comparison of document clustering techniques.

Stock, O., and Team, A. P. 1993. ALFRESCO: Enjoying the combination of NLP and hypermedia for information exploration.

Su, L. T. 1991. *An investigation to find appropriate measures for evaluating interactive information retrieval*. Ph.D. Dissertation, New Brunswick, NJ, USA.

Su, L. T. 2003. A comprehensive and systematic model of user evaluation of web search engines: Ii. an evaluation by undergraduates. *J. Am. Soc. Inf. Sci. Technol.* 54(13):1193–1223.

Teevan, J.; Dumais, S. T.; and Horvitz, E. 2005. Personalizing search via automated analysis of interests and activities. In *Proceedings of SIGIR '05*, 449–456. New York, NY, USA: ACM Press.

Voorhees, E. M. 2004. Overview of the TREC 2004 question answering track. In *Text REtrieval Conference*.

Witten, H., and Frank, E. 2000. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementation*. Morgan Kaufmann.

Witten, I. H.; Paynter, G. W.; Frank, E.; Gutwin, C.; and Nevill-Manning, C. G. 1999. KEA: Practical automatic keyphrase extraction. In *ACM DL*, 254–255.