Advanced Techniques For Personalized, Interactive Question Answering

SILVIA QUARTERONI

PH. D. THESIS

THE UNIVERSITY OF YORK - Department of Computer Science
York, YO10 5DD - UNITED KINGDOM

September 30th, 2007

Abstract

Using a computer to answer questions has been a human dream since the beginning of the digital era. A first step towards the achievement of such an ambitious goal is to deal with natural language to enable the computer to understand what its user asks.

The discipline that studies the connection between natural language and the representation of its meaning via computational models is computational linguistics. According to such discipline, Question Answering can be defined as the task that, given a question formulated in natural language, aims at finding one or more concise answers in the form of sentences or phrases.

Question Answering can be interpreted as a sub-discipline of information retrieval with the added challenge of applying sophisticated techniques to identify the complex syntactic and semantic relationships present in text. Although it is widely accepted that Question Answering represents a step beyond standard information retrieval, allowing a more sophisticated and satisfactory response to the user's information needs, it still shares a series of unsolved issues with the latter.

First, in most state-of-the-art Question Answering systems, the results are created independently of the questioner's characteristics, goals and needs. This is a serious limitation in several cases: for instance, a primary school child and a History student may need different answers to the question: *When did the Middle Ages begin?*.

Moreover, users often issue queries not as standalone but in the context of a wider information need, for instance when researching a specific topic. Although it has re-

cently been proposed that providing Question Answering systems with dialogue interfaces would encourage and accommodate the submission of multiple related questions and handle the user's requests for clarification, interactive Question Answering is still at its early stages.

Furthermore, an issue which still remains open in current Question Answering is that of efficiently answering complex questions, such as those invoking definitions and descriptions (e.g. *What is a metaphor?*). Indeed, it is difficult to design criteria to assess the correctness of answers to such complex questions.

These are the central research problems addressed by this thesis, and are solved as follows.

An in-depth study on complex Question Answering led to the development of classifiers for complex answers. These exploit a variety of lexical, syntactic and shallow semantic features to perform textual classification using tree-kernel functions for Support Vector Machines.

The issue of personalization is solved by the integration of a User Modelling component within the the Question Answering model. The User Model is able to filter and re-rank results based on the user's reading level and interests.

The issue of interactivity is approached by the development of a dialogue model and a dialogue manager suitable for open-domain interactive Question Answering. The utility of such model is corroborated by the integration of an interactive interface to allow reference resolution and follow-up conversation into the core Question Answering system and by its evaluation.

Finally, the models of personalized and interactive Question Answering are integrated in a comprehensive framework forming a unified model for future Question Answering research.

Contents

List of Figures 6

List of Tables 8

List of Algorithms 10

Acknowledgments 10

Author's Declaration 12

1 Introduction 14

1.1 A Long Researched Field 16
1.1.1 Early Question Answering 16
1.1.2 TREC-QA 18
1.1.3 Recent Advances in Question Answering 19
1.2 Some Open Issues in Question Answering 20
1.2.1 Lack of Personalization 20
1.2.2 Lack of Interactivity 21
1.2.3 Complex Questions 22
1.2.4 System Evaluation 23
1.3 Main Thesis Contributions 24
1.3.1 Standard Question Answering 25
1.3.2 Advanced Question Answering 26
1.3.3 Personalized Question Answering 27
1.3.4 Interactive Question Answering 29
1.3.5 A Unified Model of Personalized, Interactive QA 31
1.3.6 Deployment of the YourQA System 31
1.4 Thesis Overview 32

2 The YourQA Question Answering System 33
2.1 Overview of Standard Question Answering Systems 34
2.1.1 Early Question Answering Systems 34
2.1.2 TREC-QA Systems 37
2.2 High Level Architecture of YourQA's standard QA Module 46
2.3 Question Processing 49
2.3.1 Question Classification 49
2.3.2 The YourQA Question Taxonomy 51
2.3.3 An Experimental Approach to Question Classification 54
2.3.4 Related Work on Question Processing 58
2.4 Document Retrieval 59
2.4.1 Web-based Retrieval 59
2.4.2 Document Processing 61
2.5 Answer Extraction 62
2.5.1 Factoid Answers 63
2.5.2 Non-factoid Answers 65
2.6 Result Format 69
2.6.1 Answer Header 69
2.6.2 Answer Passage 70
2.6.3 Standard User Interface 71

3 Advanced Question Answering 76
3.1 Advanced Models for Sentence Representation 77
3.1.1 Syntactic Structures 78
3.2 Encoding Shallow Semantic Structures 79
3.2.1 Nested Predicate Argument Structures 81
3.3 Syntactic and Semantic Kernels for Text 83
3.3.1 Collins & Duffy's Syntactic Tree Kernel 83
3.3.2 The Shallow Semantic Tree Kernel 85
3.4 Experiments 87
3.4.1 Question Classification 88
3.4.2 Answer Classification 91
3.4.3 Answer Re-ranking 94
3.4.4 Related Work 97
3.5 Conclusion 98

4 Personalized Question Answering **99**
4.1 High Level Architecture of a Personalized QA System 101
4.2 Previous Work on User Modelling 101
4.2.1 Current Applications of User Modelling 104
4.3 User Model Design 106
4.3.1 Building a User Model 106
4.3.2 The User Model in YourQA 109
4.3.3 Reading Level Component 111
4.3.4 Profile Component 114
4.4 User Modelling for Personalized Question Answering 116
4.4.1 Document Retrieval 118
4.4.2 Answer Extraction 120
4.4.3 Related Work 122
4.5 Personalized User Interface of YourQA 123
4.5.1 Session Setup 124
4.5.2 Session Execution 124
4.6 Evaluation 125
4.6.1 Reading Level Evaluation 127
4.6.2 Profile Evaluation 129
4.6.3 First Experiment 130
4.7 Conclusions 144
4.7.1 Future Work 145

5 Interactive Question Answering **147**
5.1 Desiderata for Interactive QA 148
5.1.1 Salient Features of Human Information-Seeking Dialogue 149
5.1.2 Previous Work on Information-Seeking Dialogue 150
5.1.3 Issues in Modelling Information-Seeking Dialogue 152
5.1.4 Summary of Desiderata for Interactive QA 155
5.2 A Dialogue Model for Interactive QA 156
5.2.1 Dialogue Scenario 156
5.2.2 Towards a Dialogue Taxonomy 158
5.2.3 YourQA's Dialogue Moves 160
5.3 A Dialogue Manager for Interactive QA 161
5.3.1 Pattern Based Approaches 163
5.3.2 Plan Based Approaches and the Information State 164

5.3.3 Discussion 166
5.3.4 Chatbot-based Interactive Question Answering 167
5.4 A Wizard-of-Oz Experiment 172
5.4.1 Design 172
5.4.2 Results 175
5.5 Implementation 176
5.5.1 Dialogue Management Algorithms 176
5.5.2 AIML Framework 179
5.5.3 Dialogue Interface 186
5.6 Evaluation 186
5.6.1 Initial Evaluation 189
5.6.2 Final Evaluation 190
5.7 Conclusions 195

6 Future Research in Question Answering **199**
6.1 Future Directions for Question Answering 199
6.1.1 Standard Question Answering 199
6.1.2 Personalized Question Answering 201
6.1.3 Interactive Question Answering 202
6.2 A Unified Model of Personalized, Interactive QA 203
6.2.1 Modelling Issues 204
6.2.2 Previous Approaches 206
6.2.3 A Personalized, Interactive QA Scenario 208
6.2.4 High-Level Architecture 210
6.2.5 Future Challenges 215

7 Conclusions **217**

Appendix **219**

A Publications **220**

References **221**

List of Figures

1.1 High level view of personalized, interactive QA 32

2.1 High level architecture of the standard version of YourQA 47
2.2 Extract of a Google result page 61
2.3 Answer Extraction phase 64
2.4 YourQA: Result format 70
2.5 Standard YourQA interface: entry page 73
2.6 Standard YourQA interface: result page 74
2.7 YourQA: Result list 75

3.1 Parse tree of a question from TREC-QA 2001 78
3.2 Parse tree of s_1 79
3.3 Parse tree of s_2 79
3.4 PASs of two sentences expressing similar semantics. 80
3.5 Compact PASs of two different sentences 81
3.6 Two PASs composing a PASN 82
3.7 Example of a PASN 82
3.8 Two parse trees with their fragments derived by the STK function 84
3.9 A PAS with two of its fragments. 85
3.10 Impact of the BOW and PT features on answer classification 94
3.11 Impact of PAS and PASN combined with BOW and PT on answer classification 95
3.12 Comparison between PAS and PASN when used as standalone for answer classification 96

4.1 Personalized Question Answering Architecture 102
4.2 First answer to the question: "What is *Ginger and Fred?*" 121
4.3 Profile creation 125

4.4 Profile modification and save, query submission 126
4.5 First evaluation: perceived answer usefulness 133
4.6 First evaluation: occurrence of useful answers 134
4.7 First evaluation: perceived answer relatedness 136
4.8 First evaluation: perceived system sensitivity 137
4.9 Final evaluation: questionnaire 138
4.10 Final evaluation: average perceived usefulness 139
4.11 Final evaluation: perceived usefulness by rank 140
4.12 Final evaluation: average perceived relatedness 141
4.13 Final evaluation: perceived relatedness by rank 142
4.14 Final evaluation: perceived time 143
4.15 Final evaluation: perceived sensitivity 144

5.1 Wizard-of-Oz experiment questionnaire 174
5.2 A conversation between YourQA and user U_1 180
5.3 Extract from a conversation between YourQA and user U_2 184
5.4 Extract from a conversation between YourQA and user U_3 185
5.5 High-level architecture of the interactive version of YourQA 186
5.6 YourQA's interactive interface: desktop (applet) version 187
5.7 YourQA's interactive interface: Web service (servlet) version 188
5.8 First evaluation: tasks 189
5.9 Final evaluation: perceived accessibility of information 194
5.10 Final evaluation: perceived pace and speed of interaction 195
5.11 Final evaluation: perceived context understanding and reformulation ease 196
5.12 Final evaluation: overall satisfaction 197
5.13 Extract from a conversation between YourQA and user U_4 198

6.1 Black-box view of the standard version of YourQA 210
6.2 Black-box view of the personalized version of YourQA 211
6.3 Black-box view of the interactive version of YourQA 211
6.4 Black-box view of the personalized, interactive version of YourQA 212
6.5 Screenshot from a personalized, interactive QA session 214

List of Tables

2.1 Example of TREC 2004 question series 43
2.2 Example of TREC relationship question 45
2.3 YourQA's eleven class taxonomy 53
2.4 Accuracy of the SVM question classifier 55
2.5 Performance of the best SVM classifier by question class: UIUC 56
2.6 Performance of the best SVM classifier by question class: YourQA 57
2.7 Summary of question classification results 58
2.8 Sample patterns used during answer extraction 65
2.9 Result passages: expected factoid estimation 72
2.10 Factoid type and corresponding coloring scope 72

3.1 Accuracy of the question classifier with different feature combinations . . 90
3.2 Performance of the best SVM classifier by question class 91
3.3 Baseline classification accuracy of YourQA and Google 96
3.4 MRR of YourQA, Google and the best re-ranker 97

4.1 Example of personalized answer re-ranking 122
4.2 Reading level accuracy evaluation: cross validation 128
4.3 Examples of queries and reading level agreement 129
4.4 Perceived answer usefulness and relatedness to the user profile 135
4.5 Second evaluation: summary of results 138

5.1 User dialogue moves 160
5.2 System dialogue moves 160
5.3 Wizard-of-Oz questionnaire results 176
5.4 First evaluation: questionnaire results 191
5.5 Example of TREC 2007 question series 191
5.6 Second evaluation: questionnaire results 192

List of Algorithms

1 YourQA's answer extraction algorithm 63
2 The Syntactic Tree Kernel 84
3 The Shallow Semantic Tree Kernel 86
4 Answer re-ranking algorithm 95
5 Personalized QA algorithm 117
6 Dialogue scenario 157
7 Dialogue scenario annotated with dialogue moves 162
8 Context management in the Information State (Ginzburg & Sag, 2004) . . 166
9 Example of an ALICE category 167
10 A greeting category used in YourQA 173
11 Multiple question handling algorithm 177
12 Follow-up detection algorithm (De Boni & Manandhar, 2005) 178
13 Follow-up resolution algorithm 179
14 The YourQA `DO YOU KNOW` category 181
15 The YourQA `CLARIFY` category 182
16 The YourQA `QUERY` category, invoking the standard QA component . . . 183
17 The YourQA `FOLLOWUP` category, encoding follow-up proposal 183
18 The YourQA `ANSWER` category, returning the answer to the user 183
19 Personalized, interactive Question Answering 209
20 YourQA greeting category prompting for the user's name. 213
21 YourQA category assigning the user's name to `userID` 213
22 The YourQA `HELLO1` category, greeting the user with his/her name. . . . 214

Acknowledgments

I am grateful to my supervisor, Dr Suresh Manandhar, for proposing to me a very interesting subject of research, for his constant encouragement from the beginning of my PhD and for keeping my motivations high throughout these three years.

I would like to thank my internal examiner, Professor Helen Petrie, for her useful advice on my user experiments and for her interest toward my work.

My warm acknowledgments also to Professor Bonnie Webber, my external examiner, for her interest and suggestions to my research. Her friendly conversations gave me many useful ideas.

I am indebted to Professor Roberto Basili for inviting me to a very fruitful collaboration at the University of Rome "Tor Vergata" in the summer of 2006.

The work in this thesis would be much less interesting were it not for Professor Alessandro Moschitti, with whom I carried on very interesting joint research. His countless scientific advice and teachings were extremely useful to my work.

I will never be able to express all my love and gratitude to Alfio, Fulvia and Marzia, my family, for their constant support and for always believing in me: words cannot express how happy I am to be a part of them.

Thank you also to all my friends, for staying close in spite of the distance: a list of names would be insufficient and reductive, so I will not attempt it.

Finally, this PhD would never have happened without Pierre, who walked side-to-side with me during all these years. For this, I will never be grateful enough.

Silvia Quarteroni

York, September 30th, 2007

Author's Declaration

The Author wishes to certify that the work presented in this thesis is the product of original research.
Part of such research is documented in a number of publications in conferences, workshops and journals with public proceedings. These are listed in Appendix A.

The research reported in Chapter 3 is based on joint work with Alessandro Moschitti from the University of Trento (Italy) and Roberto Basili from the University of Rome "Tor Vergata" (Italy). Such collaboration led to the publication of the following papers:

- A. Moschitti, S. Quarteroni, R. Basili and S. Manandhar, *Exploiting Syntactic and Shallow Semantic Kernels for Question/Answer Classification.*
  In: Proceedings of the 45th Conference of the Association for Computational Linguistics (ACL 2007). A. Zaenen, A. van den Bosch (Eds.). ACL press, Prague, Czech Republic, June 2007.

- S. Quarteroni, A. Moschitti, S. Manandhar and R. Basili, *Advanced Structural Representations for Question Classification and Answer Re-ranking.*
  In: Advances in Information Retrieval, 29th European Conference on IR Research (ECIR 2007), Rome, Italy, April 2-5, 2007, Proceedings. G. Amati, C. Carpineto, G. Romano (Eds.), Lecture Notes in Computer Science Vol. 4425 Springer, Heidelberg, 2007.

Silvia Quarteroni
York, September 30th, 2007

Chapter 1

Introduction

"I think the problem is that the question was too broadly based"
(Adams, 1985)

"Forty two!" yelled Loonquawl. "Is that all you've got to show for seven and a half million years' work?"
"I checked it very thoroughly," said the computer, "and that quite definitely is the answer. I think the problem, to be quite honest with you, is that you've never actually known what the question is."
(Adams, 1979)

Using a computer to answer questions – any kind of questions – has been a human dream since the beginning of the digital era. In particular, the desire to interact with computers through dialogue as naturally as with humans has been one of the original motivations behind the creation of artificial intelligence: it suffices to think of the simulated human ELIZA (Weizenbaum, 1966), which emulated human conversation with surprising naturalness.

Both the themes of human-computer dialogue and automatic Question Answering are still very present in the minds of researchers in information retrieval and artificial intelligence.

Question Answering – in short, QA – is generally defined as the task which given a query in natural language, aims at finding one or more concise answers in the form of sentences (or phrases). For its high requirements in terms of precision and conciseness, Question Answering can be interpreted as a sub-discipline of information retrieval (IR) with the added challenge of applying techniques developed in the field of Natural Language Processing (NLP), such as the identification of the complex syntactic and semantic relationships present in text.

From this perspective, Question Answering systems move a step further in natural language understanding with respect to standard information retrieval systems, which have typical representatives in Internet search engines. By this we mean that information retrieval systems generally do not respond to a question but to a query, a set of words where syntactic structure is ignored. Furthermore, these do not return an answer, but a set of documents which are considered relevant to the query, i.e. which it is hoped will be useful to the user.

For instance, given a question such as "*When was Shakespeare born?*", a search engine would provide a set of documents containing the biographical details of the playwright and (possibly) also a large number of unrelated documents, containing, for example, biographical information about Shakespeare commentators or authors that are similar or contemporary to Shakespeare. A Question Answering system, on the other hand, should be able to present the user with the exact answer, "*Shakespeare was born in 1564*".

However, information retrieval technology remains a fundamental building block of Question Answering. In particular, open-domain Question Answering systems generally make use of information retrieval engines in order to narrow down the number of documents to be searched and processed in order to find an exact answer to a question. This step is achieved through the application of deeper linguistic techniques in order to filter out irrelevant documents, and of a consistent amount of question pre-processing and result post-processing.

Answering concise questions therefore becomes a problem of finding the best combination of word-level (IR) and syntactic/semantic-level (NLP) techniques, the former to produce as short a set of likely candidate segments as possible and the latter to pinpoint the answer(s) as accurately as possible.

This thesis contributes to the field of open-domain Question Answering by designing and deploying a model of a Web-based QA system offering several advanced features with respect to the state-of-the-art in both QA and IR. The salient contributions of this thesis are:

1. The adoption of advanced NLP techniques to produce answers to complex questions;

2. The personalization of results to the needs of individual users;

3. An interactive interface able to carry out natural language conversation.

By leveraging these features, the newly introduced model of QA clearly distinguishes the resulting system from the characteristics of traditional information retrieval systems.

Structure of this chapter

This chapter introduces the motivations and context behind the work presented in this thesis.
Section 1.1 gives a brief survey of the field of Question Answering, with a particular focus on open-domain applications. Section 1.2 outlines some open issues deriving from current limitations in Question Answering, as well as some related work attempting to solve such issues (when such related work exists).
Section 1.3 outlines the aims of this thesis towards the solution of the outlined issues, and enumerates the main contributions given by this thesis to current Question Answering. A general overview of the structure of this thesis is provided in Section 1.4.

1.1 A Long Researched Field

Research in Question Answering began in the early 1960s (Simmons, 1965) and since then, Question Answering has been the object of interest of a wider and wider community, from the fields of information retrieval, applied linguistics, human-computer interaction and dialogue.

This section gives a brief overview on the history of Question Answering and the motivations behind the work presented in this thesis.

1.1.1 Early Question Answering

Early work in the field of Question Answering concerned very limited domains and consisted in retrieving information from small databases, such as records of sport events (Green *et al.*, 1961).

One of the first generic Question Answering algorithms, presented in the 1970s by Simmons (1973), consisted of the following steps: first, taking the set of documents on which to perform QA and accumulating a database of semantic structures representing the meanings of the sentences forming such documents; then, a set of structures sharing several lexical concepts with the question was selected to form a list of candidate answers; finally, the question was matched against each candidate structure and the best matching structure was selected as the answer.

This very simple approach shows the important role of semantic processing that has characterized Question Answering from its beginning, exploiting information other than facts available in database systems, and distinguished it from information retrieval.

Question Answering in the late 1970s and until the end of the 1980s was tightly linked to human-computer dialogue systems, such as expert systems drawing information from structured knowledge bases. Indeed, the 1970s and 1980s saw the development of comprehensive theories in computational linguistics, which led to the development of ambitious projects in text comprehension and question answering.

One example of such a system was the Unix Consultant (UC) (Wilensky *et al.*, 1987), a system that answered questions pertaining to the Unix operating system. The system had a comprehensive hand-crafted knowledge base of its domain, and it aimed at phrasing the answer to accommodate various types of users. UC comprised a series of components among which a language analyzer that produced a representation of the content contained in an utterance, a goal analyzer that hypothesized the plans and goals under which the user was operating, and a domain planner that computed a plan to address the user's request. Finally, a language production mechanism expressed UC's response in English.

Although it appears that they never went past the stage of simple demonstrations, systems like the UC helped the development of theories on computational linguistics and reasoning.

Early Question Answering already denoted an interest towards the resolution of complex issues pertaining to human-computer conversation such as misconceptions and clarification. For instance, McCoy (1983) addressed the problem of correcting the user's misconceptions from the perspective of an expert system drawing from a knowledge base. The fact that her QA system drew from a small knowledge base which was structured according to an object taxonomy allowed it to detect misconceptions in users' questions about the attributes of its contents. Based on this, a series of response strategies could be deployed by the system for the sake of clarification.

Along the same lines, interesting work on cooperative natural language dialogue systems included efforts to go beyond question answering by "over-answering" questions, i.e. generating extended responses providing additional information to the user. Several types of extended responses were investigated, among which pointing out incorrect presuppositions (Kaplan, 1979). Another interesting approach was taken by Mays *et al.* (1982), where the system offered to "monitor" for information requested by the user as the system learns of changes in the knowledge base.

Wahlster *et al.* (1983) focused on over-answering yes-no questions, i.e. on generating extended responses that provide additional information to yes-no questions that pragmatically must be interpreted as wh-questions. The work attempted to build a natural language system able to elaborate on a response as a result of anticipating obvious follow-up questions, in particular by providing additional case role fillers, by using more specific quantifiers and by generating partial answers to both parts of questions containing coordinating conjunctions.

Notable early work aiming to go beyond simple QA included the attempt to address definitional questions, more complex than questions requiring factoids such as names and dates. In McKeown (1985), rhetorical schemas were used to generate definitions, while the information for the definitional text was found in a restricted knowledge base.

Open-domain QA, often called ODQA (Hori *et al.*, 2003), appeared in the late 1990s and soon became the standard in QA. In ODQA, the range of possible questions is not constrained, hence a much heavier challenge is placed on systems, as it is impossible to pre-compile all of the possible semantic structures appearing in a text.

One of the first attempts to perform open-domain QA led to the development of FAQFinder (Noriko, 2000), a system that linked the user's questions to a set of previously stored question-answer files. However, FAQFinder can be seen as an answer finding rather than a Question Answering system, as the answers were readily available instead of being created "on the fly" by the system.

AskJeeves (currently `www.ask.com`) also launched a Question Answering portal, equipped with a fairly sophisticated natural language question parser. However, AskJeeves did not provide direct answers to the asked questions: instead, it directed the user to the relevant Web pages, just as the traditional search engines do.

It is only with the TREC-QA campaigns that open-domain Question Answering systems have progressed in a major way.

1.1.2 TREC-QA

Question Answering research had a significant boost when it became the object of interest of the annual Text REtrieval Conferences (TREC, `http://www.trec.nist.gov`), a series of workshops promoted by the US National Institute of Standards (NIST) with the aim of advancing the state-of-the-art in text retrieval. Starting from TREC-8, a Question Answering track was added to the conferences, with the aim of providing a common evaluation framework for the development of open-domain Question Answering systems (Voorhees, 1999). TREC-8 QA was the first large-scale evaluation of open-domain Question Answering systems.

In TREC-8 QA, the data to be analysed to find answers came from several sources such as the Financial Times and various broadcasting services. The QA task consisted in answering 200 questions of the *factoid type*, i.e. questions that could be answered by a fact, such as a name or a date. The required answer format, to be returned by systems in one week's time, consisted partly of answers as long as a paragraph (i.e. up to 250 bytes) and partly of more direct responses (up to 50 bytes in length).

The relevance of answers was judged by human assessors, and the classic information retrieval metric of Mean Reciprocal Rank was used as a performance metric for evaluating results. The most accurate participant systems in TREC-8 found a correct response for more than two thirds of the questions (Voorhees, 1999).

The typical QA strategy adopted by such systems (see Srihari & Li, 1999) was articulated in three phases. The first phase attempted to classify the question according to the type of answer suggested by its question word (e.g. "*Who ... ?*" - person/organization). Then, systems retrieved document passages using state-of the-art text retrieval technology and the question as a query: such technologies were generally bag-of-words approaches (i.e. treating the question as an unordered bag of keywords to submit to the IR engine) for the 250 bytes answers and more sophisticated ones for 50 bytes answers.

Finally, shallow parsing was applied to find an entity of the same type as the one sought in the query; if such entity was found to be very close to the question words, it was returned by the system; otherwise, fallback techniques were applied.

This approach presents in a nutshell the structure of a modern QA system; as a matter of fact, the three phases performing question processing, document/passage retrieval and answer extraction characterize most current systems (Hovy *et al.*, 2000).

Since TREC-8 and until recently, Question Answering has moved increasingly towards exactness and precision: starting from TREC-11 (Voorhees, 2002), the tracks' requirements included returning *the exact phrase* containing the answer. This had the effect of considerably reducing the answer format, with the drawback of freeing answers from its context and uniquely focusing on questions with exact answers, hence ignoring the problem of multiple/alternative answers.

1.1.3 Recent Advances in Question Answering

Starting from 2003, TREC-QA campaigns have denoted interest towards non-factoid questions such as "definition" questions in TREC 2003 (Voorhees, 2003) and "Other" questions, requesting more information about a given topic, since TREC 2004 (Voorhees, 2004).

In the case of lists, the conceptual approach consisting in the search for exact answers is unchanged: according to the guidelines, a list question should be treated as a series of questions requesting factoids, and therefore answered by a set of factoids. Similarly, the criterion for judging the quality of list answers is basically the exactness of the factoids present in the list.

An additional observation is that, from the format of TREC question files, the required answer type (i.e. factoid, list or "Other") is known in advance, hence there is no need for systems to learn how to determine when multiple answers are desired.

Efforts have been made since TREC-QA 2004 (Voorhees, 2004) to address the issue of context management by the introduction of topics, or "targets", in the question sets. Since 2004, TREC-QA queries can contain references (such as anaphora) to their targets without such targets being explicitly mentioned in the query texts, thus requiring some form of reference resolution.

1.2 Some Open Issues in Question Answering

As previously pointed out, it is widely accepted that Question Answering represents a step beyond standard information retrieval, allowing a more sophisticated and satisfactory response to the user's information needs. However, and despite being a long-researched discipline, Question Answering still shares a series of unsolved issues with information retrieval, which are discussed below.

1.2.1 Lack of Personalization

In most state-of-the-art Question Answering systems, the results are created independently of the questioner's characteristics, goals and needs; in other words, there is a lack of *User Modelling*. This is a serious limitation: a primary school child and a History student may need different answers to the question: *When did the Middle Ages begin?* In the case of non-factoid questions, this limitation becomes even more evident: there are several ways to express definitions and describe processes, not all of which can be fully understood by any audience.

The need to personalize answers to definition questions and to adjust them to the user's needs has been highlighted starting from TREC-QA 2003 (Voorhees, 2003); however, it was then expeditiously solved by assuming one fixed user profile for all questions:

> The questioner is an adult, a native speaker of English, and an "average" reader of US newspapers. In reading an article, the user has come across a term that they would like to find out more about.
> They may have some basic idea of what the term means either from the context of the article (for example, a bandicoot must be a type of animal) or basic background knowledge (Ulysses S. Grant was a US president).
> They are not experts in the domain of the target, and therefore are not seek-

ing esoteric details (e.g., not a zoologist looking to distinguish the different species in genus Perameles)." (Voorhees, 2003).

User Modelling has been identified as a key technology in the MITRE roadmap for Question Answering (Maybury, 2002). However, there appear to be very few Question Answering systems to date where such technique is applied, especially in open-domain Question Answering (Komatani *et al.*, 2003; Hickl & Harabagiu, 2006).

1.2.2 Lack of Interactivity

Although Question Answering differs from standard information retrieval in the response format, both processes share a lack of interactivity. In the typical information-seeking session the user submits a query and the system returns a result; the session is then concluded and forgotten by the system.

However, users often issue queries not as standalone but in the context of a wider information need, for instance when researching a specific topic (e.g. "William Shakespeare"). In this case, efficient ways to address several related queries (birth-date, birthplace, famous characters, etc.) have been advocated to avoid users to enter successive related queries independently (Hobbs, 2002).

As mentioned above, recent editions of TREC-QA have approached the issue of context management by introducing "targets" in the question sets. Since TREC-QA 2004 (Voorhees, 2004), questions are grouped according to a common topic, upon which different queries (requiring factoid, list, or "Other" types of information) are formulated. Queries can currently contain elliptic and anaphoric references to their targets, as illustrated in Table 2.1.

It can be argued that the current TREC requirements only address one aspect of the complex issue of context management. Indeed, the problem of detecting that one query is related to a topic introduced by a previous one is solved by the presence of an explicit target. Moreover, reference resolution is not vital in order to achieve correct results; in fact, it would be sufficient to add the target keywords to the query keywords when accessing the IR engine in order to obtain a list of suitable candidate documents for answer extraction.

Recently, it has also been proposed that providing a Question Answering system with a dialogue interface would encourage and accommodate the submission of multiple related questions and handle the user's requests for clarification. An Interactive Question Answering workshop was organized within the HLT-NAACL conference (Webb & Strzalkowski, 2006) to set a roadmap for information-seeking dialogue applications of Question Answering.

Indeed, Interactive Question Answering (IQA) systems are still at an early stage and often relate to closed domains (Bertomeu *et al.*, 2006; Jönsson & Merkel, 2003; Kato *et al.*, 2006).

1.2.3 Complex Questions

Question Answering systems have long focused on *factoid* questions, i.e. questions concerning people, dates, numerical quantities etc., which can generally be answered by a short sentence or phrase (Kwok *et al.*, 2001).

Complex questions, or questions with complex answers, generally require definitions, descriptions or procedural explanations. These questions that cannot be solved as traditionally done for factoid questions, i.e. by relying on hand-written textual patterns or Named Entity recognizers, as pointed out in Kupiec (1999). For instance, while there are more or less sound criteria to determine that a string of text is a temporal expression (hence a good answer candidate to a time question) there are no fixed criteria to determine what makes a good answer to a definition question.

Among the first attempts to solve the problem of questions with complex answers is the work proposed by Buchholz & Daelemans (2001), where such answers are seen as consisting of several simple answers. Based on an analysis of TREC-8 results obtained using the Web and the SHAPAQA system, the authors compile a taxonomy of nine types of complex answers and propose a machine-learning approach to their solution. Of course, it can be noticed that relying on the presence of simple, atomic answers may not be sufficient to approach complex questions.

Answering complex questions has been identified as one of the main challenges in the Question Answering roadmap in Maybury (2002). Indeed, the introduction of TREC 2003 "definition" questions (Voorhees, 2003) and TREC 2004 "Other" questions (Voorhees, 2004) brought a consistent body of work on non-factoid questions. These include approaches to definition questions (Blair-Goldensohn *et al.*, 2003; Cui *et al.*, 2005), "how" (Yin, 2006) and "why" questions (Verberne *et al.*, 2007).

However, the problem of efficiently answering complex questions is still far from being solved, as it remains difficult to define evaluation criteria to assess the performance of complex Question Answering. The problem of evaluation is indeed one of the major open issues of current QA, as stated below.

1.2.4 System Evaluation

A number of remarks can be made on the TREC-QA approach regarding system evaluation. Roughly speaking, the common assumption in TREC-QA is that an ideal system is one that returns the "correct" answer in the shortest possible formulation. For instance, when "definition" questions were introduced in TREC 2003, the required answer format consisted of a list of information "nuggets" expressing key concepts about the entity being defined.

Borrowing from the evaluation of summarization systems, length was used as a crude approximation to precision, under the intuition that users would prefer the shorter of two definitions that contain the same concepts. The length-based measure gave a system an allowance of 100 (non white-space) characters for each correct nugget it retrieved. The precision score was set to one if the response is no longer than this allowance; otherwise the precision score was downgraded (Voorhees, 2003).

It can be argued that conciseness may not always be the best criterion to judge the quality of a complex answer, as can be illustrated by the following two definitions of the word "metaphor":

a) A figure of speech.

b) A figure of speech in which a word or phrase literally denoting one kind of object or idea is used in place of another to suggest a likeness or analogy between them (as in "drowning in money")[1].

According to TREC evaluation, the first definition would be preferred to the second one (which exceeds 100 non-whitespace characters); however, the second definition is the one that appears on the Merriam-Webster Online Dictionary ©, and it certainly provides a more complete description of what a metaphor is.

The "nugget" approach has been criticised in several works: for example, Lin *et al.* (2003) conducted a study revealing that users tended to prefer the result format of a Question Answering system in the form of a short passage providing some context to the sentence-level answer rather than more concise formats. Several systems approaching definition questions opt for a sentence-level answer format (Miliaraki & Androutsopoulos, 2004; Blair-Goldensohn *et al.*, 2003) on the grounds of similar arguments.

Based on these observations, we believe that the answers to complex questions such as: *What is a metaphor?* may benefit from longer formulations and be better understood with the inclusion of examples.

[1] Source: Merriam-Webster Online Dictionary©, `http://www.m-w.com/`.

Among other observations that can be made on the subject of Question Answering evaluation, we can also point out De Boni (2004)'s critical evaluation of the TREC-QA track, which points out how one of the prominent issues in TREC-QA is the lack of a proper definition of "answer" and "correct answer", even in the case of factoid questions. An example is the question *What river is called "Big Muddy"?* The only accepted answer was *Mississippi*, although *Mississippi River* could also be considered as acceptable.

In accordance with the above observations, we argue that the TREC-QA approach to evaluation, focusing on conciseness and strict criteria to assess precision, may be considered as generally suitable for factoid questions, but we believe that it is not the optimal strategy to assess the performance of systems answering complex questions discussed above.

1.3 Main Thesis Contributions

The identification of the issues discussed in Section 1.2 and the design and implementation of solutions to overcome such issues are the principal aims fulfilled by this thesis, which can be briefly summarized as follows.

1. **Standard Question Answering.** The first significant contribution of this thesis is the development of standard Question Answering techniques that deliver answers to both factoid and non-factoid questions based on the Web.

2. **Advanced Question Answering.** An in-depth study on complex Question Answering led to the development of classifiers for complex (i.e. non-factoid) answers. These exploit a variety of lexical, syntactic and shallow semantic features to perform textual classification using tree-kernel functions for Support Vector Machines.

3. **Personalized Question Answering.** The issue of personalization is addressed by the integration of a User Modelling component within the the Question Answering model. The User Model is able to filter and re-rank results based on the user's reading level and interests.

4. **Interactive Question Answering.** The issue of interactivity is addressed by the development of a dialogue model and a dialogue manager suitable for open-domain interactive Question Answering. The utility of such model is corroborated by the integration of an interactive interface to allow reference resolution and follow-up conversation into the core Question Answering system and by its evaluation.

5. **A Unified Model of Personalized, Interactive Question Answering.** A comprehensive model of personalized, interactive Question Answering has been defined, leading to a unified model of QA where User Modelling and dialogue cooperate.

6. **Deployment of a Personalized, Interactive QA System.** The development of a Web-based Question Answering system implementing the personalized, interactive Question Answering models above is the tangible outcome of the present thesis. The system, called YourQA, has been implemented in three versions (performing "standard", personalized and interactive QA). YourQA is used as a proof of concept system and for evaluating the QA models proposed in the thesis.

The contributions of this thesis to the fulfillment of the above aims are described in detail in Sections 1.3.1 to 1.3.6.

1.3.1 Standard Question Answering

The development of core Question Answering techniques, leading to the implementation of a Web-based QA system to serve as a baseline for personalization and interactivity, has been the objective of the first phase of this work.

The standard QA module designed for this purpose is structured according to the traditional three-layer structure of state-of-the-art QA systems, i.e. question processing, document retrieval and answer extraction (Kwok *et al.*, 2001).

In the question processing phase, particular attention is dedicated to the phase of question classification. Here, different types of expected answers are recognized based on the lexical and syntactic characteristics of questions, yielding to different approaches to answer extraction. Two different question classification models are adopted and compared for this purpose in Section 2.3.

The document retrieval phase is in charge of accessing an underlying Web search engine to gather a set of relevant documents for the question and of pre-processing these documents in order to prepare answer extraction. For this, the Google search engine (`http://www.google.com`) is adopted and the strategy to cope with real-time answer processing constraints without impairing the quality of retrieval is discussed in Section 2.4.

Finally, the answer extraction phase is the central phase of the core Question Answering module, where answers are sought in the retrieved documents in the form of sentences. Different techniques are applied depending on the expected answer type to pinpoint the correct answers and, based on such type, different answer formats are adopted. These are discussed in Section 2.5.

A detailed report of the standard QA module developed for this thesis is presented in Chapter 2. The answer extraction strategy is further enhanced by a study on advanced machine learning models for answer re-ranking as extensively reported in Chapter 3.

1.3.2 Advanced Question Answering

A significant contribution of this thesis to the efficient approach of complex questions consists in a thorough study of the impact of syntactic and shallow semantic information in two vital tasks in the QA process:

1. question classification;

2. answer classification and re-ranking.

Question classification is fundamental at the beginning of the QA process to estimate the expected answer type of a question and therefore to decide on the most appropriate answer extraction strategy. Symmetrically, answer classification and re-ranking are important in the terminal phase of answer extraction. Answer classification consists in assigning a category to a candidate answer in order to compare such category to the expected answer type of the question. In turn, the answer class provides criteria to refine answer extraction by re-ranking an initial list of candidate answers.

In this part of the thesis, which is the object of Chapter 3, we focus on complex – and especially definitional – Question Answering by studying various forms of representation of questions and answers based on lexical, syntactic and semantic information. In particular, we study new tree structures based on shallow semantics, which are named Predicate Argument Structures (PASs) (Kingsbury & Palmer, 2002) and new kernel functions to exploit the representational power of such structures with Support Vector Machines.

Our experiments, using such newly introduced data representations and learning functions, suggest that syntactic information helps tasks such as question and answer classification and that shallow semantics gives remarkable contribution when a reliable set of Predicate Argument Structures can be extracted, e.g. from answers.

The outcome of this research is applied to improve on the one hand the question classification performance of the question processing module in our core QA architecture. On the other hand, we drastically improve the performance of a baseline answer extractor by automatically re-ranking answers to complex questions (notably definitions and descriptions) based on the newly introduced data representations and machine learning algorithm, thus contributing to the solution of the problem of addressing complex questions.

Related Work

As mentioned earlier, the introduction of TREC 2003 "definition" questions (Voorhees, 2003) and TREC 2004 "Other" questions (Voorhees, 2004) brought a consistent body of work on non-factoid questions, such as definitions (Blair-Goldensohn *et al.*, 2003; Cui *et al.*, 2005), "why" (Verberne *et al.*, 2007) and "how" questions (Yin, 2006). However, as asserted in Section 1.2, the problem of efficiently answering such questions is still unsolved.

Our approach to complex QA derives from the tree kernel theory for Support Vector Machines introduced by Collins & Duffy (2002). In previous work, tree kernel functions have been successfully applied to a variety of tasks, such as question classification (Zhang & Lee, 2003) and relation extraction (Zelenko *et al.*, 2003; Moschitti, 2006). However, to our knowledge no study uses kernel functions to encode syntactic information in more complex tasks such as computing the relatedness between questions and answers in the purpose of answer re-ranking.

Moreover, the study of shallow semantic information, such as predicate argument structures annotated in the PropBank project (Kingsbury & Palmer, 2002), is relatively recent and approaches handling such information automatically still need investigation.

Hence, the technology we describe in Chapter 3 gives an important contribution to the advancement of the state-of-the art in complex Question Answering.

1.3.3 Personalized Question Answering

The aspect of personalization has up to now been rarely approached in Question Answering, especially in open domain QA. Although personalized QA has been advocated several times in the past, for instance in the roadmap by Maybury (2002) and in TREC (Voorhees, 2003), this thesis reports one of the first full-fledged applications of personalized open-domain QA.

In Chapter 4, personalization is demonstrated to be a useful approach for Question Answering through the following contributions:

1. Formulating personalization in Question Answering as a User Modelling problem, which consists in representing target users' characteristics, preferences, goals and needs in order to personalize an application (Kobsa, 2001). The contribution of the User Model to the overall Question Answering model consists in the definition of information filtering and re-ranking criteria based on the user's individual information needs.

2. Defining a suitable User Model to describe the information needs of users of an open domain QA system. Bearing in mind that the User Model can be customized to the domain at hand, we assert that two basic features need to be modelled in a Web-based application:

   (a) The user's general interests in terms of information *content*;

   (b) The user's level of reading comprehension in terms of information *presentation*.

3. Implementing a User Modelling component to interact with the previously developed standard Question Answering module. The user's interests are extracted from Web documents such as pages from the user's personal bookmarks or browsing history, as well as from any other type of textual document created or owned by him/her. For this purpose, key-phrase extraction is performed on representative user documents using the methodology exposed in Section 4.5.1.

   The presentation aspect of the User Model involves the user's reading ability. For this purpose, the reading levels of the documents retrieved from the Web for the user's query are estimated in order to retain for answer extraction only documents that are not too easy nor too complex to interpret.

4. Defining a methodology for the evaluation of personalized Question Answering by comparing the "standard" and the "personalized" versions of the QA system on the grounds of user satisfaction.

5. Conducting evaluation experiments to empirically assess the contributions of personalization to Question Answering using the above methodology.

The experiments reported in Chapter 5 reveal an important positive contribution of User Modelling to Question Answering.

Related Work

The technique of User Modelling for the personalization of Web applications is not new: introduced in Kobsa (2001), it has been applied in a variety of contexts, such as personalized search (Pitkow *et al.*, 2002; Teevan *et al.*, 2005), item recommendation (Ardissono *et al.*, 2001; Magnini & Strapparava, 2001; Miller *et al.*, 2003), learning environments (Person *et al.*, 2000; Romero *et al.*, 2003; Linton *et al.*, 2003) and cultural heritage (Graziola *et al.*, 2005).

Although such applications have several common aspects with Question Answering, and personalized QA has been advocated in several places (see Maybury, 2002), very little work seems to exist to date relating to User Modelling in QA applications. In Hickl & Harabagiu (2006), a novice versus expert model of the user is taken into account while extracting answers in an interactive QA system; however, as the primary scope of the study is not the User Modelling aspect, such model is not evaluated and remains quite simple, relying on a two-class stereotype rather than modelling individual user profiles. In Thai *et al.* (2006), a personalized QA system for the closed domain of business analysis is proposed, with the aim of taking into account context and recent user queries. Unfortunately, the personalization aspect does not appear to be implemented yet and its application relates to a closed domain.

Altogether, there does not seem to be a comprehensive study on how User Modelling can affect open domain Question Answering; ours is a first step towards that direction and is based in particular on the personalized search studies exposed in Pitkow *et al.* (2002) and Teevan *et al.* (2005), as explained in Chapter 4.

1.3.4 Interactive Question Answering

The solution to the problem of interactivity consists in developing a dialogue-based QA system. The general information flow in the system comprises:

1. A query formulation phase through a dialogue interface, helping the user to formulate his/her search needs;

2. A standard Question Answering phase during which answers are fetched and adapted to the user;

3. An answer presentation phase where the answer is provided to the user via the dialogue interface.

The aim of this aspect of the current research was to design an information-oriented dialogue management strategy suitable for Question Answering. As this is a relatively new application of dialogue systems, the resulting design and the consequent proof of concept constitute an important contribution of the thesis.

First, the requirements for modelling Interactive Question Answering are discussed, starting from the conversation phenomena occurring in generic dialogue and developing the desiderata for open domain, QA-oriented dialogue (see Section 5.1). Then, a theoretical dialogue model based on such requirements is studied and the main dialogue acts to be defined to achieve interactive QA are discussed in Section 5.2.

Furthermore, different dialogue management models are investigated in order to find the most suitable architectural design for the newly introduced dialogue model (Section 5.3). We opt for a chatbot-based interface able to maintain a notion of the recent conversational context and to resolve the most common types of anaphoric and elliptic expressions.

Since chatbot-based dialogue is a new application for Question Answering, the theoretical and design assumptions are first validated in the course of an exploratory Wizard-of-Oz study (Section 5.4). The encouraging results of such experiment lead to the implementation of an interactive version of YourQA interactive interface, as described in Section 5.5. A methodology for the evaluation of such interactive QA prototype is discussed in Section 5.6, and the results of such evaluation highlight how interactive QA is well received by users and has great potential for development in the future.

Related Work

As previously mentioned, recent Question Answering has been focusing more and more on the issue of interactivity. An Interactive Question Answering workshop has been organized within the 2006 HLT-NAACL conference (Webb & Strzalkowski, 2006) to set a roadmap for information-seeking dialogue applications of Question Answering. Indeed, Interactive Question Answering (often abridged to IQA) systems are still at an early stage and often relate to closed domains (Small *et al.*, 2003; Jönsson & Merkel, 2003; Kato *et al.*, 2006).

In line with most modern theories of conversational analysis (Sinclair & Coulthard, 1975), human-computer dialogue is represented in this thesis as a set of exchanges composed by dialogue acts. For the modelling of dialogue acts, inspiration is taken from several well-known dialogue annotation schemes, such as DAMSL (Core & Allen, 1997), TRAINS (Traum, 1996) and VERBMOBIL (Alexandersson *et al.*, 1997).

The approach to dialogue management presented in this thesis, which is based on chatbots, is innovative when compared to the usual finite-state, information-state or plan-based approaches to dialogue managers. A similar approach is reported in Galibert *et al.* (2005), although without a full evaluation, and in Basili *et al.* (2007), although for a closed domain.

An exploratory study is reported following the Wizard-of-Oz methodology, as common practice in dialogue system development (Dahlbaeck *et al.*, 1993). Similar evaluations are taken as inspiration for the design of the Wizard-of-Oz study, such as Bertomeu *et al.* (2006); Munteanu & Boldea (2000).

Finally, the evaluation of the final system is designed based on traditional dialogue evaluation frameworks such as PARADISE (Walker *et al.*, 2000) and the results of pioneering studies on Interactive QA such as Kelly *et al.* (2006).

1.3.5 A Unified Model of Personalized, Interactive Question Answering

Drawing from the experience collected during the research on personalization and interactivity, this thesis presents the creation of a comprehensive model of personalized, interactive Question Answering.

The model is centered on a standard Question Answering module which is responsible for efficiently extracting answers to both factoid and non-factoid questions using the Web. The standard QA module interacts with two satellite components: a User Modelling component and a dialogue component. The former is in charge of the personalization aspect and provides criteria to filter candidate answer documents and re-rank candidate answer passages based on their appropriateness to an individual model of the user's information needs; the latter is in charge of modelling the conversation context and ensuring a smooth interaction with the user.

Chapter 6 describes such unified framework, which has been implemented with the creation of the three prototype systems described in the following section.

1.3.6 Deployment of the YourQA System

The final contribution of this thesis is the deployment of a proof-of-concept system for the proposed approaches to Question Answering, in the form of an open-domain, personalized, interactive Question Answering system called YourQA.

The high-level architecture of YourQA, as represented in Figure 1.1, consists of three basic components: the Question Answering component, the User Modelling component and the dialogue component, fully described in Chapters 2, 4 and 5, respectively.

Three different versions of YourQA have been implemented and served as basis for the evaluation experiments reported throughout this thesis:

1. A "standard" version, i.e. a basic Web-based open-domain Question Answering system;
2. A "personalized" version, which constructs and applies User Models to personalize answers;
3. An "interactive" version, which is able to interact with the user through a chatbot interface, maintain the interaction context and resolve anaphoric and elliptic utterances.

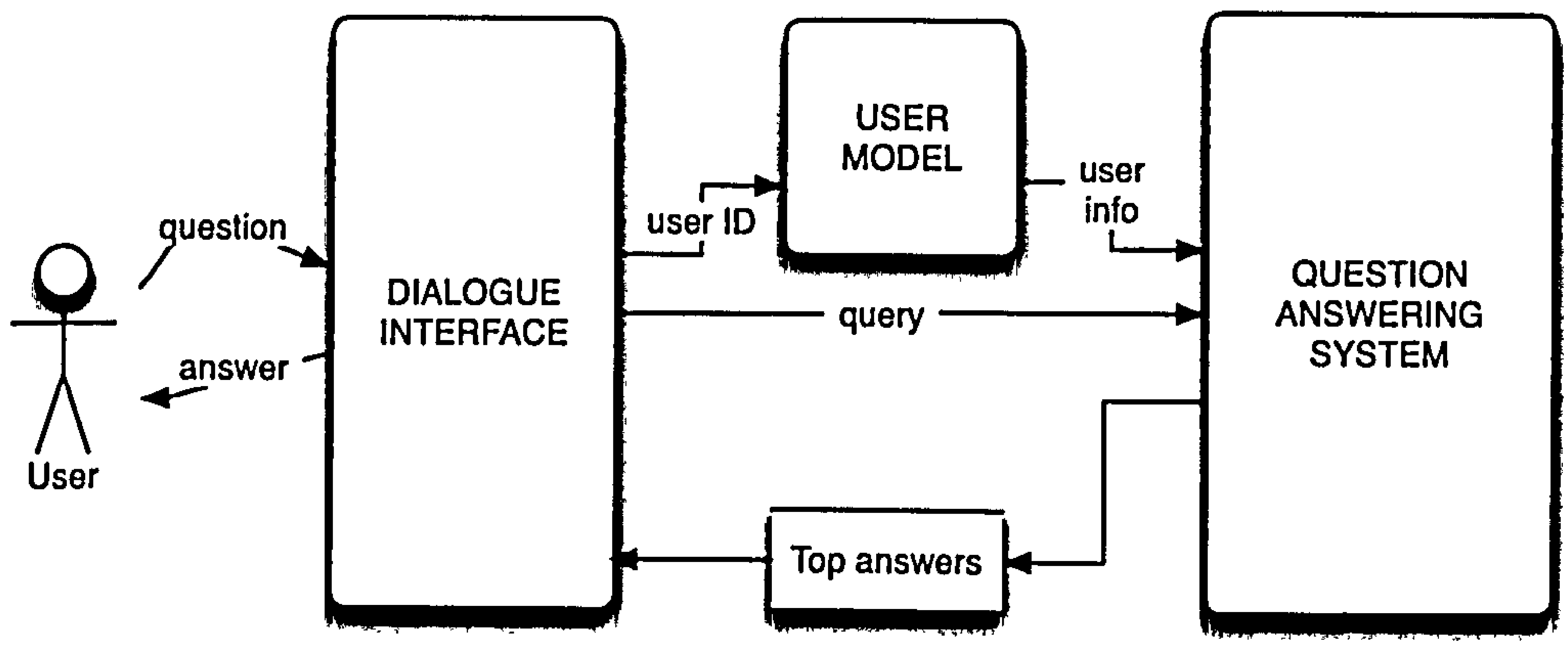

Figure 1.1: High level architecture of YourQA, a personalized interactive QA system

1.4 Thesis Overview

This thesis is organized in the following chapters. Chapter 2 provides a full description of the standard Question Answering techniques that we apply to achieve effective Web-based Question Answering.

Chapter 3 presents a deep study on advanced Question Answering techniques involving tree kernel approaches and Support Vector Machines for two vital tasks in core Question Answering: question classification and answer extraction/re-ranking.

Chapter 4 focuses on the aspect of personalization with a thorough description of the application of User Modelling to Question Answering and its evaluation.

Chapter 5 discusses the aspect of interactivity, designing a dialogue model for open-domain interactive QA as well as a dialogue management model suitable to implement the characteristics of such model. The implementation and evaluation of the interactive QA prototype designed as a proof-of-concept is also discussed.

Chapter 6 addresses future work along the lines of advanced techniques for complex questions, personalization and interactivity. A unified model Question Answering, joining the aspects of personalization and interactivity in a single framework, is also proposed.

Finally, Chapter 7 concludes the thesis with a summary of its salient contributions to current research in Question Answering.

Chapter 2

The YourQA Question Answering System

This chapter describes the architecture of the standard version of YourQA, a Web-based, open-domain Question Answering system able to address both factoid and non-factoid questions. In YourQA, the Question Answering process involves the three main phases of question processing, document retrieval and answer extraction which characterize most state-of-the-art QA systems.

The implementation of such architecture provides the baseline QA system against which we evaluate the advanced QA techniques reported in Chapter 3, the impact of personalization component described in Chapter 4 and the interactive QA component discussed in Chapter 5.

Structure of This Chapter

In this chapter, Section 2.1 traces an overview of standard Question Answering systems, which led to the design of the architecture for the standard QA version of YourQA, exposed in Section 2.2.

Section 2.3 is dedicated to question processing, which is centered around the classification of the question's expected answer type.

Section 2.4 is dedicated to the phase of document retrieval, which starts with the submission of a query to the underlying search engine and terminates with a list of candidate documents from which to extract answers.

Section 2.5 is dedicated to answer extraction, the final phase of Question Answering, focusing on how both factoid and non-factoid expected answer types lead to different approaches to the selection of answers.

Finally, Section 2.6 presents the result format of YourQA and the user interface of YourQA's standard version.

2.1 Overview of Standard Question Answering Systems

As Question Answering is a long researched field, our work on QA could draw from a well-established tradition. Moving from the foundation work in the 1960s and 1970s, where questions were drawn from closed and small databases, QA evolved towards its current standards starting from the late 1990s, when concise answers were required to be drawn from the Web to address open-domain questions.

In this section, we trace a more extensive overview of the history of research leading to state-of-the-art Question Answering than the one in Chapter 1. We focus particularly on current technologies and on their relation to the approaches taken for YourQA.

2.1.1 Early Question Answering Systems

As mentioned earlier, the first Question Answering appeared in the 1960s: Simmons (1965) reviews 15 different systems for QA. In early QA approaches, information was retrieved from databases regarding very limited domains: for instance, the BASEBALL system (Green *et al.*, 1961) attempted to retrieve answers about one season of baseball games in the American League.

Another foundational QA system was SHRDLU (Winograd, 1972), which answered questions about a world made of moveable blocks. The system had the ability of engaging in actual dialogue in which the human interlocutor could not only ask factoid questions such as "HOW MANY BLOCKS ARE NOT IN THE BOX?", but also yes/no questions, such as:

- Person: HAD YOU TOUCHED ANY PYRAMID BEFORE YOU PUT THE GREEN ONE ON THE LITTLE CUBE?
- Computer: YES, THE GREEN ONE.

and why-questions, as visible in the following exchange[1]:

- Person: WHEN DID YOU PICK IT UP?
- Computer: WHILE I WAS STACKING UP THE RED CUBE, A LARGE RED BLOCK AND A LARGE GREEN CUBE.

[1] Source: `http://hci.stanford.edu/~winograd/shrdlu/`

- Person: WHY?
- Computer: TO GET RID OF IT.
- Person: WHY DID YOU DO THAT?

Simmons (1973) presents one of the first generic QA algorithms, using the following approach:

1. Accumulate a database of semantic structures representing sentence meanings;
2. Select a set of semantic structures that appear relevant to the question (i.e. that share several lexical concepts with it) to form a list of candidate answers;
3. Match the question against each candidate;
4. Select the best matching semantic structure as the answer to return.

The concept of semantic structure is the key point in this algorithm and reflects the approach to QA of the 1970s, which was based on cognitive psychology. An important aspect of systems by then was the classification of questions, which was the first processing phase of QA algorithms.

Cognitive Science foundations of QA Schank & Abelson (1977) introduced the concept of "knowledge structures", conceptual nets linking together pieces of information about an event or story. The underlying "script" in a text was a "specific knowledge" that enabled to understand a situation without a great amount of textual processing. Indeed, such scripts enabled a sort of stereotyped sequence of actions that defined a well-known situation. Using such a representation, questions about events could be answered by identifying the corresponding knowledge structures and the use of such scripts.

Later on, Dyer (1983) introduced the idea of "Thematic Abstraction Units"; more abstract than scripts, they were used for story categorization. A story would be expressed in terms of plans, failed plans and recovery of plans. In order to represent the motivations and intentions of characters in a narrative, Dyer introduced I-links (intentional links), i.e. relationships between goals, plans and events such as *forced-by* or *achieved-by*. The approach to Question Answering in this model consisted initially in a classification of questions according to a predefined taxonomy; then the I-links in the Thematic Abstraction Units were traversed to find the appropriate answers.

QA from a database PLANES (Waltz, 1978) was a system able to answer questions from a database about aircrafts. The PLANES database was a relational database containing the records of maintenance and flight data of two types of planes during a period of two years. This kind of database ensured the presence of not too complex requests, although ellipsis and vague questions were possible and correctly handled most of the time.

PLANES used a set of augmented transition networks (ATNs), each of which recognized a phrase with a specific meaning. The processing of a user's request to the PLANES database included the following steps:

1. Parsing. This was done by matching the question against a set of subnets representing all the possible records in the database.

2. Query generation, i.e. translation of the semantic constituents of the user's question into a formal database query;

3. Evaluation, i.e. a database search to produce an answer;

4. Response generation, providing the system answer as a either number or list or in graphical form.

During parsing, the subnets (ATN phrase parsers) were applied subsequently against the input request. Whenever subnets matched a phrase, they pushed a value in canonical form into a stack structure called context register in order to store a history of the process and be able to resolve ellipsis and pronoun reference. Concept case frames were semantic sentence patterns of questions understood by the system used to complete a request when constituents of a sentence were missing.

The idea of question classification proposed by the initial models of QA described above had great influence on later systems, and notably on those developed for TREC-QA (see Section 2.1.2). However, the common limitation of early Question Answering systems – including also previously cited systems such as McCoy (1983); Mays *et al.* (1982); McKeown (1985); Wilensky *et al.* (1987) – consists mainly of the limited domains of their application. This allowed them to answer real user questions only on a small, structured world (such as the world of blocks in SHRDLU) or on a small range of facts (Woods *et al.*, 1972). Moreover, such abilities required a consistent effort in terms of knowledge representation and systems were rarely scalable.

Open-domain Question Answering

Open-domain Question Answering (ODQA) is Question Answering where the range of possible answers –and therefore of possible questions– is not constrained, hence users can ask any question, phrased in any way[2] and on any topic. In open-domain QA, knowledge is not annotated or encoded: answers must be sought and constructed from a collection of documents made up of "real" text, with the errors, omissions and complications that these imply. In some cases, the QA process can be supported by additional resources such as off-the-shelf databases like Amazon[3] for books and other media, the IMDB database for movies[4], Wordnet[5] (HERMJAKOB *et al.*, 2002) or Wikipedia[6] (Katz *et al.*, 2005), or locally compiled ones (e.g. the database of celebrities in Katz *et al.* (2004)).

As mentioned in Chapter 1, of the first approaches in the field of open-domain QA was FAQFinder (Noriko, 2000). The system linked the user's questions to a set of previously stored question-answer files; as a matter of fact, it was more an answer finding than a Question Answering system.

Also the open-domain QA portal launched by AskJeeves (`www.askjeeves.com`), cannot be seen as a full-fledged open-domain QA system. Indeed, AskJeeves only performed the question processing part of the QA task, while the answer extraction phase was not approached. As a matter of fact, it did not provide direct answers to the user's questions: instead, it directed him/her to the relevant Web pages, in the same way as standard search engines.

Section 2.1.2 illustrates the first major developments in open-domain Question Answering, which occurred with the TREC-QA campaigns.

2.1.2 TREC-QA Systems

The Text REtrieval Conferences (TREC) are a series of evaluation campaigns promoted by NIST (`http://trec.nist.gov`) with the aim of advancing the state-of-the-art in text retrieval. Starting from TREC-8 (Voorhees, 1999), a Question Answering track was added to the conferences, with the purpose of providing a common evaluation framework for the development of open-domain QA systems.

[2]as long as the question does not exceed a single sentence

[3]`http://www.amazon.com`

[4]`http://imdb.com`

[5]`http://wordnet.princeton.edu`

[6]`http://wikipedia.org`

TREC-8 QA

TREC-8 QA (Voorhees, 1999) was the first large-scale evaluation of open-domain QA systems. Twenty different organizations participated to the track, and the most accurate systems found a correct response for more than two thirds of the questions. The main characteristics of the track were the following:

- The data from which to extract answers came from a variety of sources, including newswire from the Financial Times, broadcasting services, etc.
- The track test-set consisted of 200 factoid questions, making the task very challenging;
- The required answer format consisted in answers partly as long as a paragraph (250 bytes) and partly more direct responses (50 bytes).
- The time available for returning results once the test-sets were available was limited to one week;
- The judgment on answer relevance was assigned by human assessors.

The typical strategy adopted by participant systems to answer questions was the following:

1. The system first tempted to classify the question according to the type of answer suggested by its question word (e.g. "Who ... ?" was classified as requiring a "person" or "organization").
2. Next, the system retrieved a portion of a document using an IR engine and the question as a query. The techniques were generally bag-of-words approaches for the 250 bytes answers and more sophisticated ones for 50 bytes answers.
3. Then, shallow parsing was applied to find an entity of the same type as the one sought in the query; if such entity was found to be very close to the question words, it was returned by the system. Otherwise, best-matching-passage techniques were used as a fall-back strategy.

This approach worked well for questions containing specific question words (such as *wh*-words); however, systems had difficulties with questions that did not contain a wh-word or worse did not seek answers of a particular entity type (e.g. *What is Head Start?*).

TREC-9 QA

The main difference between TREC-8 QA and TREC-9 QA (Voorhees, 2000) was the size of the document set from which to extract answers, which was considerably increased. The number of questions was also increased and the type of questions was considerably harder to approach as real user questions were used. Participants were required to return a ranked list of five pairs of the form `[document-id, answer-string]` such that each `answer-string` was supposed to contain the answer.

Also, 193 out of the 693 questions were variants of basic questions introduced to explore how the participant systems would handle semantic information. Hence, many of the 28 participant systems used the WordNet lexical database (Miller, 1995) as a source of related words for query expansion. Query expansion (Xu & Croft, 1996) aims at amplifying the recall of the QA system's underlying search engine, by incrementing the search sentence with strings containing synonyms or more generally words that are semantically related to the search nouns and verbs.

An example of this is the Falcon system (Wu *et al.*, 2000), which obtained the best results by answering about 65% of the TREC-9 questions. Falcon classified each question by expected answer type, but also included successive feedback loops to try progressively larger modifications of the original question until it found a satisfactory answer.

The Webclopedia (Hovy *et al.*, 2000) system used a classification of QA types to facilitate coverage, a robust syntactic-semantic parser to perform the analysis, and contained a matcher that combined word-level and parse-tree-level information to identify answer passages.

In general, TREC-9 participant systems had the major limitation of a restricted linguistic knowledge and a very poor semantic knowledge, which were highlighted by the presence of question variants.

TREC-QA 2001

In the 2001 edition of TREC-QA, answer format was limited to 50 bytes and the number of questions was 500 (Voorhees, 2001). The systems in TREC-10 mainly used analogous approaches to the ones in the preceding tracks: an information retrieval engine to choose a subset of relevant documents and a Named Entity tagger to analyze them and find a NE corresponding to their question type.

An interesting approach is that of the PIQUANT system (Chu-Carroll *et al.*, 2002), which used an architecture allowing for multiple answering agents to address the same question in parallel and for the results to be combined.

The main components of the architecture were a question analysis module, one or more answering agents to implement different answering strategies given the results of question analysis and a knowledge source, and an answer resolution component to combine the results issued from the different strategies.

This approach could enable the use of different knowledge source types (e.g. structured/unstructured) at the same time, as well as concurrent accesses to the same knowledge sources using different query criteria. This guaranteed a higher level of confidence concerning the answers provided by PIQUANT.

Context and clarification in QA TREC-10 QA was the first campaign to address the issue of several related questions on the same topic: the track included a context task that aimed at testing the systems' ability to track context through a series of questions. Systems were required to respond correctly to a kind of clarification dialogue where a full understanding of the question would have depended on understanding the previous questions and their answers. In order to test the ability to answer such questions correctly, 42 questions were prepared and divided into 10 series of related question sentences.

The follow-up resolution issue could be interpreted as a clarification problem, where the first question was followed by questions attempting to clarify it.

However, as underlined in De Boni & Manandhar (2003), the track did not approach the problem of recognizing whether the question currently under consideration was part of a previous series (i.e. clarifying previous questions) or the start of a new series, as the index of each question within its series clearly distinguished it as being a "fresh" question or a follow-up question.

Moreover, it must be pointed out that the context sub-task was resolved by participant systems by simply completing questions containing anaphoric references with keywords from the previous question. Hence, the problem of clarification tended to be expeditiously solved without any deep interpretation of the question itself (Voorhees, 2001).

An interesting aspect of the TREC-10 QA context sub-track is that questions contained various features that could be used for the detection of clarification dialogue. The use of pronouns and possessive adjectives, the absence of verbs, the repetition of proper nouns and the relevance of semantic relations between the words in close sentences were all useful hints. Based on such hints, De Boni & Manandhar (2003) proposed a clarification recognition algorithm that is adopted in this thesis to efficiently conduct interactive QA (see Chapter 5).

TREC-11 QA

With respect to the previous tracks, the major difference of TREC-11 QA guidelines was the answer format: a definite answer, i.e. a single noun or phrase, was required for the given questions. For this track, many systems deployed sophisticated techniques in order to improve the answer extraction phase, such as the use of planning and answer validation through logical forms (Nyberg *et al.*, 2002). However, no remarkable differences appeared between TREC-11 and previous tracks as far as our analysis is concerned.

TREC-QA 2003

TREC 2003 was particularly interesting as a significant move away from factoid question was attempted for the first time in TREC. Indeed, the main task of the QA track involved three types of questions: factoids, lists, and definitions. Each question was tagged according to its type, and the response format and evaluation methods differed for each type.

The required answer format for a *factoid* question was either exactly one `[doc-id, answer-string]` pair or the literal string "NIL". Each response was assigned one of the following four judgments:

- incorrect: the answer string does not contain a right answer or the answer is not responsive;
- not supported: the answer string contains a right answer but the document returned does not support that answer;
- not exact: the answer string contains a right answer and the document supports that answer, but the string contains more than just the answer or is missing bits of the answer;
- correct: the answer string consists of exactly the right answer and that answer is supported by the document returned.

The score for the factoid component of the main task was accuracy, i.e. the fraction of responses judged correct.

List questions were seen as a shorthand for asking the same factoid question multiple times. Hence, a system's response for a list question was an unordered set of `[doc-id, answer-string]` pairs such that each `answer-string` was considered an instance of the requested type and assessed using factoid metrics. Unlike in TREC 2001 and 2002, the 2003 list questions did not specify a target number of instances to return. Instead, systems were expected to return all of the correct, distinct answers contained in the document collection. The average F1-measure computed over these factoids was then used as the list question score for each system.

For *definition* questions, as in the list task, systems needed to return an unordered set of `[doc-id, answer-string]` pairs as a response. Each string was presumed to be a facet in the definition of the target. There were no limits placed on either the length of an individual answer string or on the number of pairs in the list, though systems were penalized for retrieving extraneous information.

Judging the quality of the systems' responses was done in two steps. In the first step, all of the answer-strings from all of the responses were presented to the assessor in a single (long) list. Using these responses and his own research done during question development, the assessor first created a list of "information nuggets" about the target.

At the end of this step, the assessor decided which nuggets were vital – nuggets that must appear in a definition for that definition to be good. In the second step, the assessor went through each of the system responses in turn and marked where each nugget appeared in the response.

The final accuracy for definition questions was measured using a F_β-score placing heavy emphasis on nugget recall ($\beta = 3$).

Finally, the scores of the three sessions (factoid, list and definition) were combined in the following weighted score:

$$WeightedScore = .5 \times FactoidAcc + .25 \times ListAvgF + .25 \times DefAvgF.$$

where $FactoidAcc$ was the accuracy obtained for factoid answers, $ListAvgF$ was the average F-measure obtained by the list answers, and $DefAvgF$ was the average F-measure obtained by the list answers.

TREC-QA 2004

TREC 2004 (Voorhees, 2004) was the first TREC-QA campaign to approach the issue of context management in the main QA track by the introduction of "targets" in the question sets. Since TREC 2004, questions are grouped according to a common topic, upon which different queries (requiring factoid, list, or "other" types of information) are formulated. A question series from TREC 2004 is illustrated in Table 2.1, where the common target

"Franz Kafka" is shared by five questions.

Also since TREC-QA 2004, queries can contain references (such as anaphora) to their targets without such targets being explicitly mentioned in the query texts, thus requiring some form of reference resolution. For instance, Series 22 in Table 2.1 contains two cases of pronominal anaphora (*he* in questions 22.2 and 22.4) and one case of determiner anaphora (*his* in question 22.3).

Table 2.1: Example of TREC 2004 question series

Series ID: 22	*Target:*	Franz Kafka
Question ID	*Type*	*Text*
22.1	FACTOID	Where was Franz Kafka born?
22.2	FACTOID	When was he born?
22.3	FACTOID	What is his ethnic background?
22.4	LIST	What books did he author?
22.5	OTHER	*more information on Franz Kafka*

The required system response for a factoid question was either exactly one `[doc-id, answer-string]` pair or the literal string "NIL", assessed as in TREC 2003. List questions were also assessed using the same methodology as in TREC 2003.

Other questions were evaluated using the same methodology as the TREC 2003 definition questions. A system's response for an "Other" question was an unordered set of `[doc-id, answer-string]` pairs as in the list component and each `answer-string` was presumed to be a facet in the definition of the series' target that had not yet been covered by earlier questions in the series.

System answers were limited to 7,000 non-whitespace characters in length and were assessed with the nugget-based methodology. Finally, the scores of the three sessions (factoid, list and other) were combined in the following weighted score:

$$WeightedScore = .5 \times FactoidAcc + .25 \times ListAvgF + .25 \times OtherAvgF.$$

In TREC-QA 2004, participant system approaches to factoid QA did not change much with respect to the strategies used in 2003. Most groups used their factoid-answering system for list questions, changing only the number of responses returned as the answer and, in the case of "Other" questions, similar techniques were used as those deployed for TREC 2003's definition questions.

The fact that factoid and list questions did not necessarily explicitly include the target of the question was a new difficulty in TREC-QA 2004, which makes this edition of particular interest.

To overcome such difficulty, in the document/passage retrieval phase, most systems simply appended the target to the query, which was possible in TREC as in all cases the target was the correct domain for the question, and most of the retrieval methods used treat the query as a simple set of keywords.

Another common approach was to replace all pronouns in the questions with the target. However, since many (but not all) pronouns in the questions did in fact refer to the target, this approach was not effective when the question used a definite noun phrase rather than a pronoun to refer to the target (e.g. "the band" when the target was "Nirvana").

Finally, other systems (see Ahn *et al.*, 2004) tried varying degrees of true anaphora resolution to appropriately resolve references in the questions. These approaches are particularly relevant to our work on interactive Question Answering (exposed in Chapter 5), where the target of questions is unknown and must be determined online by the system.

TREC-QA 2005

In TREC 2005 (Voorhees & Dang, 2005), the main Question Answering task was the same as in TREC 2004. A notable addition to the TREC 2005 tasks was a relationship task, where systems were given TREC-like topic statements that ended with a question asking for evidence for a particular relationship in the same format as for "Other" questions. Table 2.2 illustrates an example of a relationship question requiring information about entities involved in space exploration.

While the "topic" format of the relationship task was considerably different from the short questions appearing in the main task, the approaches used for the former did not differ highly from the ones taken for the latter's "Other" questions.

TREC-QA 2006

There were no notable modifications in the TREC-QA 2006 campaign apart from the introduction of a complex, interactive QA (ciQA) task (Kelly & Lin, 2007).

The ciQA task extended and refined the "relationship" task piloted in TREC 2005. Thirty complex relationship questions based on five question templates were investigated using the AQUAINT collection of newswire text. The interaction aspect of the task here relates to the fact that interaction forms were the primary vehicle for defining and capturing user-system interactions. However, this does not imply that actual natural language

Table 2.2: Example of TREC relationship question

Topic	The analyst is interested in cooperative international efforts in space exploration. Specifically, the analyst wants to know whether there was any international involvement in NASA's Cassini unmanned space probe mission.
Answer	The Cassini mission was jointly sponsored by the European Space Agency and the Italian Space Agency, along with NASA. ESA supplied the Huygens probe that will be released when Cassini reaches Saturn.
Evidence	`XIE19970408.0053`: Spacecraft to Be Launched to Saturn in October Cassini-Huygens is a joint project between NASA, the European Space Agency (ESA) and ASI (the Italian space agency). NASA has primary responsibility for the Cassini orbiter while ESA is responsible for the Huygens probe. `NYT19990816.0266`: support from ESA and ISA critical to success `APW19990818.0104`: Cassini Probe Gets by Earth

dialogue occurred between user and system, as "single-shot" interactions were all that was required.

In addition to the nugget evaluation score used since Voorhees (2003), nugget pyramids (Lin & Demner-Fushman, 2006) were implemented to obtain a more refined notion of nugget importance. In total, six groups participated in the ciQA task and contributed ten different sets of interaction forms. There were two main findings: baseline IR techniques are competitive for complex QA and interaction, at least as defined and implemented in this evaluation, did not appear to improve performance by much.

TREC campaigns are the obvious guidelines for the design of YourQA. Moreover, issues and limitations arising in TREC have been the sources of inspiration for the advanced techniques for non-factoid QA exposed in Chapter 3, for the personalized architecture described in Chapter 4 and also for the design of interactive QA in Chapter 5. All of these rely on the standard Question Answering component of YourQA, which is illustrated in detail in the next section.

2.2 High Level Architecture of YourQA's standard Question Answering Module

The standard Question Answering module in YourQA is articulated through three main phases, represented in Figure 2.1:

1. *Question processing*: this phase consists in analysing the user's question and transforming it into a search engine query;

2. *Document retrieval*: this phase consists in obtaining relevant documents from the Web and splitting them into their composing sentences;

3. *Answer extraction*: this phase consists in selecting final answers from the relevant document sentences.

The framework relies on the Web search engine Google (`www.google.com`) to retrieve documents that are relevant to the user's query.

The general architecture of YourQA's core QA component is greatly inspired by TREC-style Web-based systems, such as MULDER (Kwok *et al.*, 2001) and Webclopedia (Hovy *et al.*, 2000). As illustrated later, a significant difference is that, while such systems focused on factoid questions, YourQA also aims at addressing complex questions. A description of the salient characteristics of the two systems, with references to their similarities and differences with respect to YourQA is briefly discussed below.

MULDER MULDER (Kwok *et al.*, 2001) is a general-purpose Question Answering system available on the Web. The Question Answering process is divided in three phases:

1. *Pre-processing*: the natural language question is parsed and the parse tree is given to a classifier. Next, the query formulator translates the question into a series of queries which are fed in parallel to the underlying search engine, Google.

2. *Retrieval*: Google obtains relevant Web pages for the queries which are to be processed by the answer extraction module;

3. *Answer generation*: the answer extraction module extracts relevant snippets from the Web pages, generating a list of candidate answers. An answer selector scores and ranks the snippets and the sorted list of answers is displayed to the user.

The question processing/document retrieval/answer generation architecture in MULDER is also present in the general layout of the standard QA module of YourQA. Google is

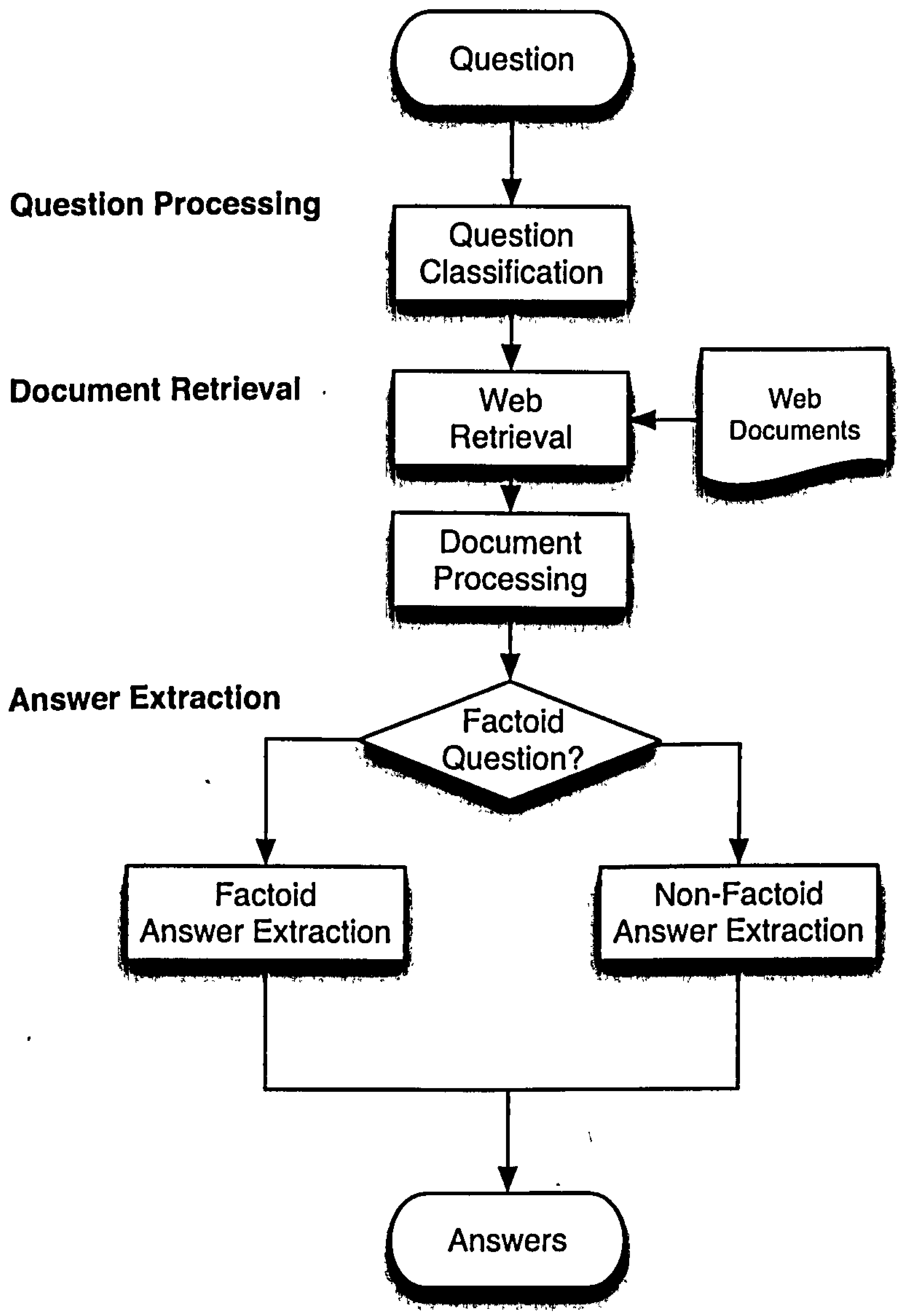

Figure 2.1: High level architecture of the standard version of YourQA

also the search engine of our choice, both because of its leading position in the search engine market and because of its available APIs.

Extracting relevant snippets from the retrieved documents and applying a ranking criterion is the approach that we are also pursuing. However, MULDER returns snippets in the same form as Google, i.e. phrases separated by dots.

YourQA provides answers in a different format as it returns actual sentences extracted from the source document, as explained in Section 2.6. We claim that this provides a more meaningful answer than that given by a sequence of incomplete sentences, as also suggested in Lin *et al.* (2003) (see Section 2.4).

We also have to observe that deep parsing (as performed in MULDER) is costly in terms of computation time, and may not always be useful. Indeed, Web documents are often written in a poor, ungrammatical style as opposed to news items or textual information explicitly selected to compose a data collection like the AQUAINT collection used for TREC-QA.

Also, given the variety of information available on the Web, deep analysis of answer candidates may not be necessary: it is often easy to find factoid information formulated in a way that is very close to the question formulation or that can be detected with lexical pattern matching. This is one reason that motivated our decision to rely on "shallow" NLP techniques rather than "deep" ones in the standard QA algorithm.

Webclopedia Webclopedia (Hovy *et al.*, 2000) is a Web-based QA system developed with the aim of finding the best combination of word-level (IR) and syntactic and semantic-level (NLP) techniques, the former to produce as short a set of likely candidate segments as possible and the latter to pinpoint the answer(s) as accurately as possible. Webclopedia deeply focuses on NLP, performing the following steps:

1. *Query analysis*: input questions are parsed to obtain a semantic representation.

2. *Query expansion*: in order to boost recall, WordNet 1.6 (Fellbaum, 1998) is used to expand query terms and place all the expanded terms into a boolean expression.

3. *Document retrieval*: the MG search engine (Bell *et al.*, 1995) is used, and the retrieved documents are ranked according to their ranking from query analysis.

4. *Document ranking*: the score of a document is computed as the ratio between the sum of scores of its words (based on their similarity with the query words) and the number of its words.

5. *Document segmentation*: each document is split into topical segments to be input to a candidate answer matcher, based on the assumption that important contextual information for pinpointing answers tends to occur within a local context.

6. *Segment ranking*: the resulting segments are re-ranked and returned to the user.

Webclopedia performs intensive linguistic analysis: it uses a syntactic-semantic parser to perform question analysis, and contains a matcher that combines word-level and parse-tree-level information to identify answer passages.

An interesting feature of the system is the introduction of a question taxonomy trying to account for the users' intentions (e.g. mapping: *Who was Christopher Columbus?* to type *why-famous* instead of *person*). This pragmatic approach provided inspiration for the creation of our question classifier (see Section 2.3.1).

2.3 Question Processing

We define question processing as the subtask of Question Aswering which starts by taking the user's question in natural language and terminates with the submission of a query to the underlying information retrieval engine.

Question processing in the standard YourQA component is centered on the task of question classification, which is exposed in Section 2.3.1. Then, the taxonomy developed for YourQA in order to address both factoid and non-factoid questions is reported in Section 2.3.2.

Two approaches to question classification, both of which have been implemented in YourQA, are presented in Section 2.3.3. These apply two different machine learning models, one based on SNoW and the other based on Support Vector Machines, to the task of question classification; a discussion of the obtained results concludes Section 2.3.3. Related work on question processing is discussed in Section 2.3.4.

2.3.1 Question Classification

Question classification (QC) is the task that maps a question into one of k expected answer classes. This is the first crucial task performed by a Question Answering system, as it constrains the search space of possible answers and contributes to selecting answer extraction strategies specific to a given answer class.

QC is formally defined as a multi-classification problem which consists in assigning an instance I (in our case, the question) to one of k classes. Such expected answer classes generally belong to two types: *factoid*, seeking short fact-based answers or *non-factoid*, seeking e.g. descriptions or definitions (see Li & Roth, 2002).

Previous Work on Question Classification

Question classification has been identified as one of the main bottlenecks in Question Answering: for instance, Moldovan *et al.* (2003) found that it accounted for 36.4% of the errors in an experiment on a state-of-the-art QA system.

Many systems use template based and pattern matching approaches to question classification. For instance, about 400 manually written templates are used in GuruQA (Prager *et al.*, 1999) while in Webclopedia (Hovy *et al.*, 2000) patterns are learned from search engine results on the Web. Textract QA (Srihari & Li, 2000) and YorkQA (De Boni, 2004), perform pattern matching based on *wh*-words. Alicante QA (Vicedo *et al.*, 2001) uses pattern matching to detect definition questions.

However, most accurate question classification systems apply supervised machine learning techniques to learn classifiers, e.g. Support Vector Machines (SVMs) (Zhang & Lee, 2003) or the Sparse Network of Winnows (SNoW) model (Li & Roth, 2005), where questions are encoded using various lexical, syntactic and semantic features.

The advantage of machine learning with respect to hand-written rules is that it is fully automatic, requiring no hand-written rules but only a set of questions classified according to answer type in order to be used for training and testing. Among the machine learning approaches to question classification, we focus on the SNoW model and Support Vector Machines.

Learning question classifiers with SNoW In Li & Roth (2002), a question classifier guided by a two-layered semantic hierarchy of answer types was learned. The first layer performed a coarse-grained classification of the question into six expected answer classes (henceforth called UIUC taxonomy): abbreviations (ABBR), descriptions (DESC), numeric expression (NUM), person (HUM), entity (ENTY) and location (LOC). The second layer took as input the results of the coarse classifier and mapped it into a fine-grained classification using 50 classes.

The learning architecture used by both classifiers is SNoW, based on the linear separation of the feature space by several lines. Given a confusion set and a question, SNoW's decision model outputs a ranked list of class labels as well as densities over each class; the top class, i.e. the one associated with the highest density, is the expected answer class of the question.

The optimal feature space used in Li & Roth (2002) included six primitive feature types, representing lexical, syntactic and semantic information. These were the question's bag-of-words, bag-of-POS tags, chunks and head chunks, along with semantic information such as Named Entities, hand-built lists of semantically related words, and semi-automatically built distributional similarity based categories. Using such features, the classification accuracy on the coarse-grained UIUC split reached as high as 98.8%; a further study using only the first four features (i.e. lexical and syntactic) showed that the coarse-grained accuracy reached 92.5% (Li & Roth, 2005).

Question classification using SVMs In Zhang & Lee (2003), five machine learning techniques for question classification were compared: Nearest Neighbors, Naïve Bayes, Decision Trees, SNoW and SVMs. Comparison was made through five experiments using the UIUC data-set introduced above. The results showed that SVMs could produce significantly better results than other algorithms both with only surface text features (bag-of-words, word n-grams) and when taking advantage of the syntactic structure of questions, i.e. their parse trees.

An accuracy of 90% on the coarse-grained classification was achieved on the UIUC data-set by the use of a tree kernel function to compute matches between syntactic parse tree structures. Full details on tree kernel functions and our application of such functions to question classification are given later in this chapter and in Chapter 3.

2.3.2 The YourQA Question Taxonomy

Although the most intuitive approach to question classification appears to be to distinguish among different *question* classes, the most efficient method is to classify *expected answer types* (Moldovan *et al.*, 2003). For instance, classifying a question as a "what" question does not help to decide what kind of answer will best respond to it, as the word "what" can relate to a person, a year, a location, etc. Classifying based on expected answer types allows instead to map a "what" question to different answer classes based on the question's lexical, syntactic and/or semantic features.

Among the question taxonomies developed for QA systems, one of the most well known is certainly the UIUC taxonomy, used in e.g. Li & Roth (2002). Such taxonomy partitions expected answer types according to two levels of granularity, the first of which (coarse-grained taxonomy) encompasses six classes: abbreviations (ABBR), descriptions (DESC), numeric expression (NUM), person (HUM), entity (ENTY) and location (LOC). A fine-grained taxonomy further distinguishes about 50 more specific classes within the individual coarse-grained question types.

Our examination of the question test-sets from TREC-8 QA to TREC-12 QA[7], a corpus containing 3204 questions, led us to the observation that it would be difficult or rather impossible to assign a number of test questions from recent editions of TREC-QA to any of the coarse-grained taxonomy types in the UIUC taxonomy, unless such expected type were explicitly mentioned. For instance, the "list" question type, a representative of which is TREC question number 1923: "*What governments still officially recognize and support International Labor Day?*", has been introduced from TREC-11. An automatic classifier based on the UIUC taxonomy would need the additional information that the question must be answered in the form of a list of entities as currently specified in TREC campaigns; clearly, it is not possible in a Web QA system to have this kind of information.

Moreover, questions that require non-factoid answers, such as lists, descriptions, explanations, would have all been assigned to the "DESC" type in the coarse-grained taxonomy, making the classification too generic for the purpose of a QA system aiming at addressing complex answers. On the other hand, applying the fine-grained taxonomy would have implied a focus on too specific, potentially un-necessary sub-classes of factoid questions. This in turn might have resulted in less accurate classifiers when dealing with Web QA.

This observation motivated our interest in the design of an independent question taxonomy for YourQA, in which we balance the presence of factoid and non-factoid question types. The resulting taxonomy, henceforth named the "YourQA taxonomy", is a coarse-grained taxonomy which consists of eleven question types: PERSON, LOCATION, TIME, QUANTITY, ORGANIZATION, OBJECT, LIST, DEFINITION, HOW, WHY, WHY-FAMOUS. The above types can be grouped into two macro-categories: the "factoid" group, encompassing the former six types, and the "non-factoid" group, encompassing the latter five.

The YourQA taxonomy is briefly described and exemplified in Table 2.3. While the UIUC taxonomy distinguishes a fine-grained repartition of the non-factoid classes (specifying the coarse-grained class "DESC") into the definition, description, manner and reason types, the YourQA non-factoid group maintains the DEFINITION class, merges description and manner into the "HOW" type as these are often difficult to distinguish in the TREC corpus, and separates the "reason" subclass into the "WHY" and "WHY-FAMOUS" classes, which we interpret as asking for different types of information (a specific reason in the first case, relevant information about an entity in the second case). Moreover, the "LIST" type is added to accommodate list questions.

[7] publicly available at `http://trec.nist.gov`

Table 2.3: YourQA's eleven class taxonomy

	Question Class	Expected answer	Example
Factoid	PERSON	A Named Entity of type human	*"Who killed Lee Oswald?"*
	LOCATION	A geographical location	*"Where is Inoco based?"*
	TIME	A temporal expression	*"When was the submarine invented?"*
	QUANTITY	A numerical quantity	*"How fast can a Corvette go?"*
	ORGANIZATION	A group, e.g. team, company	*"What company manufactures Sinemet?"*
	OBJECT	A generic entity	*"What is Grenada's main commodity export?"*
Non-factoid	LIST	A list of items	*"What were Columbus' three ships?"*
	DEFINITION	A definition or description	*"What is platinum?"*
	HOW	An explanation	*"How did Socrates die?"*
	WHY	A generic cause	*"Why does the moon turn orange?"*
	WHY-FAMOUS	Relevant information about an entity	*"Who was Gandhi?"*

The MONEY type, a further subclass of the QUANTITY type, has been created during the system implementation for the purpose of improving the accuracy of answer extraction when searching for expressions of currency and rate. However, the identification of the MONEY subclass only occurs estimated in a second time after the machine-learning based question classification has yielded QUANTITY as en expected answer type. For full details, please refer to Section 2.5.1.

2.3.3 An Experimental Approach to Question Classification

In order to design efficient question classifiers for the Web-based QA system to be modelled and to evaluate their accuracy, a comparative study was designed. Two question classification tasks were defined: the first one –henceforth "UIUC task"– was a well-known one, which consisted in classifying the UIUC corpus (available at: `http://l2r.cs.uiuc.edu/~cogcomp/Data/QA/QC/`) according to the six-class UIUC taxonomy; the second one –henceforth "YourQA task"– consisted in classifying the TREC-8 to TREC-2004 test-set questions according to the eleven-class YourQA taxonomy.

Moreover, two different machine learning based question classifiers were implemented in YourQA: one applied the SNoW model and the other applied SVMs. In both cases, we opted for the use of lexical and syntactic features and did not rely on manually or semi-automatically constructed lists of relevant words (as opposed to e.g. Li & Roth (2005)) since the classifiers were trained to address questions submitted to an open-domain QA system. The remainder of this section illustrates in detail the two classifiers along with their performance. We start by illustrating the results of classification using the SVM model applied first to the UIUC task and then to the YourQA task.

Classification using the SVM model

The SVM classification model used for YourQA is described in full detail in Section 3.4.1 and in Quarteroni *et al.* (2007); Moschitti *et al.* (2007). The question multi-classifier combines the output of the individual question classes' binary SVMs according to the ONE-vs-ALL scheme, where the final output class is the one associated with the most probable prediction.

The performance of the multi-classifier and the individual binary classifiers was measured with accuracy and F1-measure, respectively. The data used for the first SVM experiment consists of the UIUC dataset, which contains the 6,000 questions available at: `http://l2r.cs.uiuc.edu/~cogcomp/Data/QA/QC/`; these are divided into a test set composed by the 500 TREC 2001 questions (Voorhees, 2001) and a training set composed of the remaining 5,500 questions. The UIUC dataset is manually annotated according to the coarse-grained question taxonomy defined in Li & Roth (2002) – i.e. ABBR, DESC, NUM, HUM, ENTY and LOC.

Question representation was based on a variety of features, which include the question's syntactic parse tree (PT), part-of-speech tags (POS), and bag-of-words (BOW). The features were combined by summing the output of a linear kernel function computed over the BOW and POS features with that of a tree kernel function computed over the question's PT, in a study reported in Section 3.4 and published in Quarteroni *et al.* (2007); Moschitti *et al.* (2007).

To collect statistically significant information, we ran 10-fold cross validation on the entire dataset, thus obtaining 10 different random splits. Based on such experiment, the features which gave the best performance were the question's syntactic parse tree (PT), and bag-of-words (BOW), that yielded an accuracy of 86.1%±1.1. This result is reported in Table 2.4, along with other feature combinations tested during the experiment; a more detailed discussion of the contributions of the single features is provided in Section 3.4.

Features	Accuracy (cross-validation)
PT	84.8±1.2
BOW	84.7±1.2
POS	32.4±2.1
PT+BOW	**86.1±1.1**
PT+BOW+POS	84.7±1.5

Table 2.4: Accuracy of the SVM question classifier with various combinations of the bag-of-word (BOW), parse tree (PT) and Part-Of-Speech (POS) features when applied to the UIUC corpus and taxonomy.

Table 2.5 illustrates the accuracy (in terms of F1) of the individual binary classifiers for the UIUC corpus and taxonomy. The most accurate binary classification is the one carried out for NUM, which generally exhibits easily identified cues such as "how much/-many". The ENTY type, which is more generic, is the hardest to classify, while LOC and HUM appear to be well-classified, also thanks to their regular patterns ("where" and "who" identifiers). ABBR, the second most poorly classified type, exhibits a high standard deviation in cross validation as there are only 95 total instances in the whole UIUC data-set, leaving little significance to the classification results. A more detailed discussion appears in Section 3.4.1 where the learning models and UIUC dataset are presented more thoroughly.

Based on such results, the best performing learning features used for the UIUC exper-

Question Class	F1 (cross-validation)
ABBR	78.5±7.0
DESC	84.6±2.3
ENTY	75.7±1.3
HUM	86.8±2.0
LOC	88.9±1.5
NUM	94.2±1.4
Multi-Classifier Accuracy	86.1±1.3

Table 2.5: Performance of the best SVM classifier by question class for the UIUC corpus and taxonomy (results are presented as average± standard deviation).

iment, namely the question's bag-of-words and parse tree, were applied to the YourQA task in a second experiment. As mentioned above, the YourQA dataset contains 3204 questions (the TREC 8-12 questions) and the YourQA taxonomy consists of 11 classes. The overall accuracy in the YourQA experiment, obtained via cross-validation, was 83.43% ±3.85 (see Table 2.6, last row).

As for the individual binary classifiers, the performances of which are reported in Table 2.6, we can see that the TIME and QTY types, which correspond roughly to the NUM types in the UIUC taxonomy, are very well classified. This result is consistent with what observed during the UIUC experiment. The PLACE type, often characterized by distinctive words such as "Where" or generic location words such as "river", "city", etc., is the second best classified question type. The PERSON classifier is seventh in order of accuracy, which could be due to the fact that in the YourQA split it can often be confused with the WHY-F class. The HOW type is quite well classified while the LIST and WHY classes appear to be more difficult to classify. This can be explained by the fact that both types appear rarely in the corpus and hence are more difficult to learn. Finally, the worst performing individual classifier is the one for ORG, and this seems to be for several reasons: on the one hand, there are very few questions requiring organizations in the YourQA corpus; moreover, these are often lexically difficult to distinguish from the PERSON type.

The loss in accuracy in the YourQA task can be explained when considering that on the one hand the amount of available training data is lower, and more importantly the task is intrinsically more complex as it consists in classifying using an eleven-class taxonomy instead of a six-class one.

Question Class	F1 (cross-validation)
TIME	90.77±3.35
DEF	88.75±5.99
ORG	29.29±9.74
LIST	69.96±6.50
QTY	85.18±3.78
OBJ	64.69±8.41
WHY	40.00±12.65
WHY-F	79.69±12.57
HOW	81.01±7.24
PLACE	83.34±3.05
PERSON	79.69±5.49
Multi-Classifier Accuracy	83.43±3.85

Table 2.6: Performance of the best SVM classifier by question class for the YourQA corpus and taxonomy (results are presented as average± standard deviation).

Classification using the SNoW model

Second, we applied the SNoW learning algorithm to both the UIUC and YourQA tasks. The most effective features among those we tested were the following:

1. Word unigrams (bag-of-words), bigrams and trigrams,
2. POS tag unigrams (bag-of-POS tags), bigrams and trigrams,
3. Bag-of-Named Entities[8].

When applied to the UIUC task, the SNoW algorithm gave an accuracy of 84.1%±1.7, obtained via ten-fold cross-validation. The SNoW algorithm was then applied to the YourQA task, achieving an accuracy of 79.3%±2.5. As in the SVM experiment, also in the SNoW experiment the accuracy is lower when using the YourQA corpus and taxonomy than when using the UIUC corpus and taxonomy (see Table 2.7).

For both the YourQA and UIUC experimental settings, having fixed the corpus and taxonomy, the paired t-test comparing the results of classification when using the SNoW and SVM models gives a statistically significant difference in favor of the SVM model (i.e. $p < 0.05$). Hence, we can say that the best learning model of classification found in the SVM experiment performs significantly better than the best learning model found in the SNoW experiment.

[8]extracted using Lingpipe (`http://www.alias-i.com/lingpipe/`)

Model	Accuracy (YourQA, cross-val.)	Accuracy (UIUC, cross-val.)
SVM	83.4±3.9	86.1±1.4
SNoW	79.3±2.5	84.1±1.7

Table 2.7: Question classification results for the UIUC and YourQA tasks when using the SVM resp. SNoW classification model. Results are expressed as average accuracy ± standard deviation.

While it makes sense to compare the accuracy of classification with respect to the learning models (i.e. to compare the SNoW model and the SVM model) in statistical terms, given our experimental setting it is impossible to fix one learning model and compare the accuracy of the two corpora, which contain different instances partitioned according to different taxonomies. Hence, we can only explain the loss in accuracy of both the SNoW and SVM models for the YourQA corpus and taxonomy in a qualitative way, by recalling the smaller amount of data in the YourQA corpus and by the fact that the multi-classification involves eleven classes, i.e. almost twice as many as in the UIUC corpus.

2.3.4 Related Work on Question Processing

In addition to question classification, another widely used question processing technique applied in the context of TREC-based systems is query expansion (see Wu *et al.*, 2000; Hovy *et al.*, 2000).

One way to perform query expansion is by mining a lexical database such as WordNet (Miller, 1995) for words related to the search keywords. Such related words are then used in place of the original keyword and a modified search string is submitted to the search engine in addition to the original one.

Although lexical databases can be a precious resource to increase retrieval recall, they inevitably diminish the result precision and therefore must be used with moderation and efficiently tuned. This is the key reason behind the fact that the core QA model proposed in YourQA does not include a query expansion phase.

In particular, we motivate our choice by two main reasons. The first reason is that it does not seem sufficient to select the most similar words or concepts (e.g. WordNet synsets) for each individual question keyword and replace the latter by such related words to perform query expansion. In fact, the semantic notion of similarity and relatedness always depends on the context, hence on the other words. We argue that an efficient model taking such context into account in the open domain still needs to be developed and goes beyond the scope of this thesis.

The second reason is that the QA system proposed in this thesis is Web-based: as opposed to TREC-QA systems, where the underlying search engine has a limited number of documents from which to extract information, the Web is substantially larger and more varied in size, hence the probability of finding the required information formulated in the same form in which it is sought is much higher and question expansion would introduce too much noise.

2.4 Document Retrieval

In YourQA, the document retrieval phase starts with the user's query and terminates with a list of documents ready to be processed for answer extraction.

In the standard version of YourQA, the document retrieval phase is minimal, consisting of a Web-based retrieval step and a document processing step; this phase acquires more relevance in the personalized Question Answering model, as explained in Section 4.4.1.

2.4.1 Web-based Retrieval

As in MULDER, (Kwok *et al.*, 2001), the Google search engine is accessed during the document retrieval phase (in the implementation, YourQA uses the Java Google APIs available at: `http://code.google.com/apis/`). For this purpose, the user's question is used as a query, and the top documents returned by the search engine (currently the top 20) are obtained as results for such query.

Among the information provided by the format of a Google search result as returned from the Google APIs, we retain the following for the purpose of answer extraction:

- URL of the result document;
- Title of the result document;
- Google rank of the result document.

For each Google result, the document retrieval module retrieves the corresponding document from the URL in order to conduct fine-grained analysis and extract answers. For this purpose, the quick and light-weight Lynx text browser (available at: `http://lynx.browser.org/`) is used and the result documents are saved with their original file name.

A trade-off between redundancy and speed

The great redundancy of information present in the Web is nowadays a well-established resource for Question Answering systems, both open-domain (starting from Kwok *et al.* (2001)) and closed-domain (see Fujii & Ishikawa (2002), where a QA system draws information from an encyclopedia generated from the Web). Dumais *et al.* (2002) summarizes the two main advantages of such redundancy as the possibility of enabling simple query reformulations and the facilitation of the answer mining process. Indeed, the presence of different phrasings of the same concepts on the Web alleviates the task of query expansion and answer extraction.

However, redundancy is also clearly an issue when it comes to system processing speed; therefore, there needs to be a threshold on the number of search engine results to be considered for real-time Web QA.

The work in Dumais *et al.* (2002) reports the evaluation of the accuracy of a Web-based QA system with respect to a varying number of initial relevant snippets (i.e. text summaries) collected from the top hits of the underlying search engine. Their use of snippets in place of actual documents is motivated by processing speed and complexity. The experiment results show that accuracy improves sharply as the number of snippets increases from 1 to 50, and more slowly between 50 and 200, eventually falling off after 200 snippets.

In YourQA, although the use of Google APIs makes it possible to exploit search engine result snippets (which appear as answer hints on a typical Google result page) in order to locate the answer's neighborhood, we retrieve and process the whole Google documents. Indeed, Google result snippets are often juxtaposed extracts of text containing single query keywords or subsets of the former, located at different positions in the original documents (and visualized with interposed "..."), or incomplete sentences (see Figure 2.2).

Moreover, as the purpose of search engines such as Google is to return relevant *documents*, the criteria applied to return results relate to the document level, hence the compact format of Google result snippets does not guarantee that the corresponding portion of text from which such snippets are extracted is indeed compact.

In order to make the QA system responsive in real time, and exploiting the observation that, thanks to the high relevance of Google results, actual users rarely need to seek information beyond the first search engine result page[9] (i.e. the first 10 results in the case

[9]Data based on a 2006 search engine user study conducted by iProspect (`www.iprospect.com`) revealed that 62% of search engine users click on a result within the first page of results, while 90% of them click on a result within the first three pages of results.

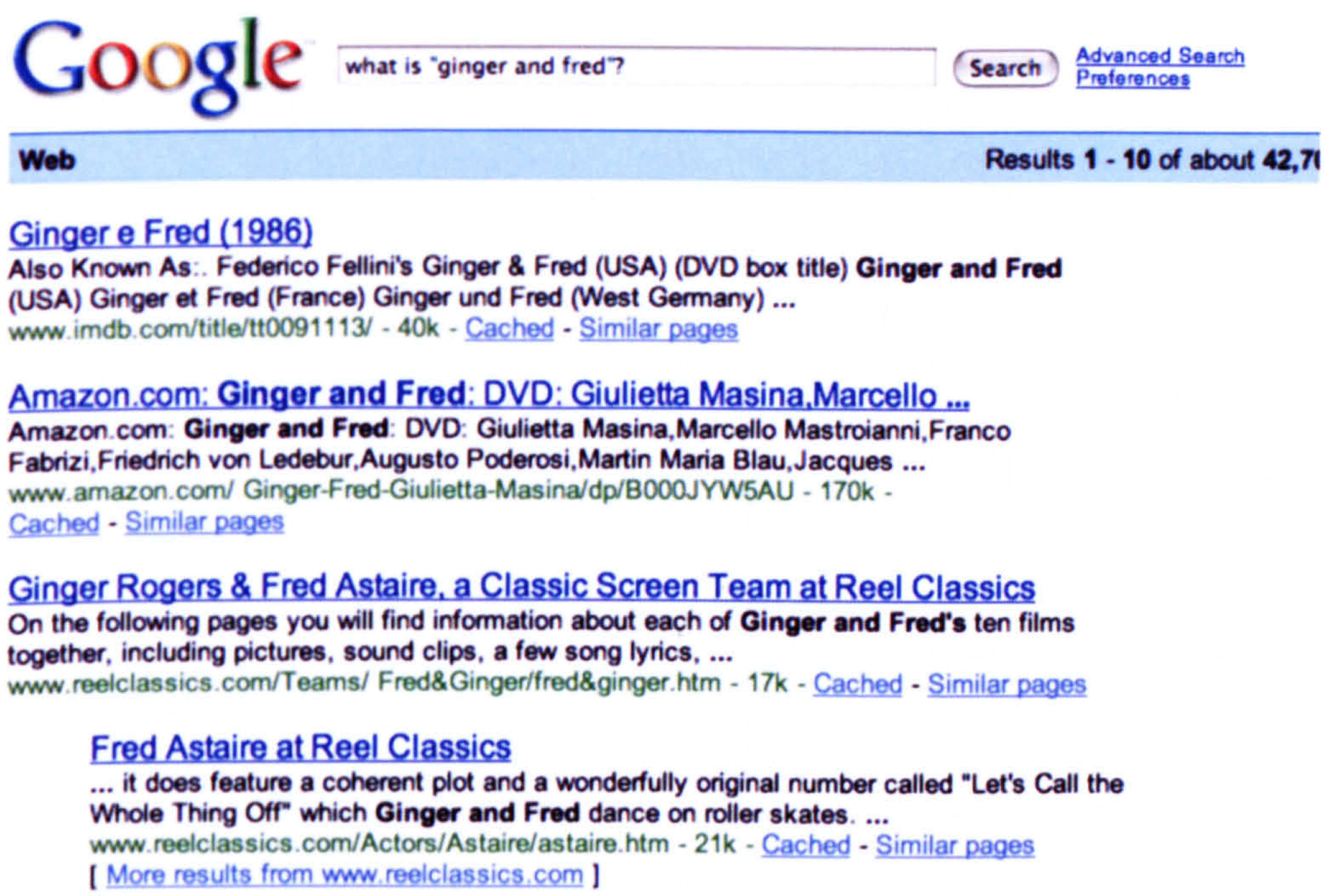

Figure 2.2: Extract of a Google result page

of Google), we limited the number of retrieved Google documents to 20. This appears as a good compromise between the study in Dumais *et al.* (2002), which showed a good accuracy of answers when using up to 50 document snippets and the fact that in our case what is processed are not snippets but actual Web pages.

2.4.2 Document Processing

Document processing in the model of Question Answering proposed in this thesis aims at preparing the extraction of answers in the format of *sentences*. In the case where the expected answer type is a factoid, answer extraction is narrowed down to the phrase/word level, however the answer format still consists of a sentence where such factoids are highlighted.

Sentence-format answers distinguish YourQA from current TREC-QA requirements, which have consisted of text snippets of decreasing sizes in the past years and now demand (as explained in Section 2.1.2):

- the "exact" answer, i.e. a phrase or word, for factoid questions;

- a set of relevant information nuggets for the "Other" answer types.

We argue that, in the case of non-factoid questions such as definition questions, it makes sense that users receive answers in a form where the syntactic and semantic relationships between the relevant words are explicitly present instead of as a set of juxtaposed keywords as this can lead to ambiguities. This holds especially for answers describing events and furthermore for definitions, which generally appear on dictionaries in the form of actual sentences; in this, we agree with Miliaraki & Androutsopoulos (2004).

Moreover, recent user behaviour studies showed that even in the case of factoid Question Answering systems, the most eligible result format consisted in a paragraph where the sentence containing the answer was highlighted (Lin *et al.*, 2003).

Finally, it must be pointed out that a context-sensitive approach to Question Answering, returning a sentence rather than pinpointing a phrase or word, improves the confidence of finding correct answers with respect to the application of deep natural language understanding techniques in order to "spot" the exact answer (as in Hovy *et al.* (2000); Harabagiu *et al.* (2000)).

Moving from these considerations, the document processing step, carried out after document retrieval, consists in splitting each document into sentences in order to compute the degree of match between the user's query and each sentence within such document.

To perform sentence splitting, manual patterns are applied and each document is then represented as an array of sentences. Once this step is terminated, the answer extraction phase can take place as described in the following section.

2.5 Answer Extraction

In YourQA, answer extraction takes as input the expected answer type as estimated by the question classification module and the set of candidate answers, i.e. sentences extracted from the documents retrieved for the question by the document retrieval component and subsequently split during document processing.

Based on the outcome of the question classifier, the answer extraction module determines whether the expected answer belongs to the factoid group, i.e. PERS, ORG, LOC, QTY, TIME or MONEY (the latter is a further specialization of the QTY type inferred using hand-written rules, as reported below). If this is the case, the required factoid contained in each candidate answer sentence is pinpointed down to the phrase or word level using factoid QA techniques. Otherwise, other similarity criteria are adopted to compute the similarity between the candidate answers and the original question.

Algorithm 1 YourQA's answer extraction algorithm

1. Compute the number of common keywords between the question q and the candidate answer a;

2. `if` the expected answer type EAT of q is *factoid* **`then`**
   `if` $EAT \in \{PERS, ORG, LOC\}$ **`then`**
   use the Named Entity recognizer to detect NEs of the required type;
   `else` //($EAT \in \{QTY, TIME, MONEY\}$)
   use specific hand-written rules to detect phrases of the required format;
   `else` //(EAT is *non-factoid*)
   compute additional similarity metrics to induce a ranking in candidate answers.

3. Combine the criteria computed in Step 2 to the similarity in Step 1 to induce a ranking over candidate answers;

4. Select the top n candidate answers and return them to the user.

YourQA's answer extraction algorithm is formalized in Algorithm 1 and summarized in Figure 2.3.

The first, common similarity metric applied in both the factoid and non-factoid case is the bag-of-words similarity, described below.

Bag-of-word similarity

The bag-of-word similarity between the question q and a candidate answer a, $bow(q, a)$, is the number of matches between the question keywords q_i, with $i < |q|$, and the candidate answer keywords a_j, with $j < |a|$, normalized by dividing by the number of question keywords, $|q|$:

$$bow(q, a) = \frac{\sum_{i<|q|, j<|a|} match(q_i, a_j)}{|q|} \tag{2.1}$$

The following subsections explain in detail the answer extraction strategies adopted *in addition to the bag-of-words criterion* in both the factoid and non-factoid case.

2.5.1 Factoid Answers

If the expected answer type is a factoid, we distinguish between two cases: if the type is a person (PERS), organization (ORG) or location (LOC) – which correspond to the types of entities recognized by the Named Entity (NE) recognizer used in YourQA, Lingpipe (`http://www.alias-i.com/lingpipe/`) – we run the NE recognizer to spot all

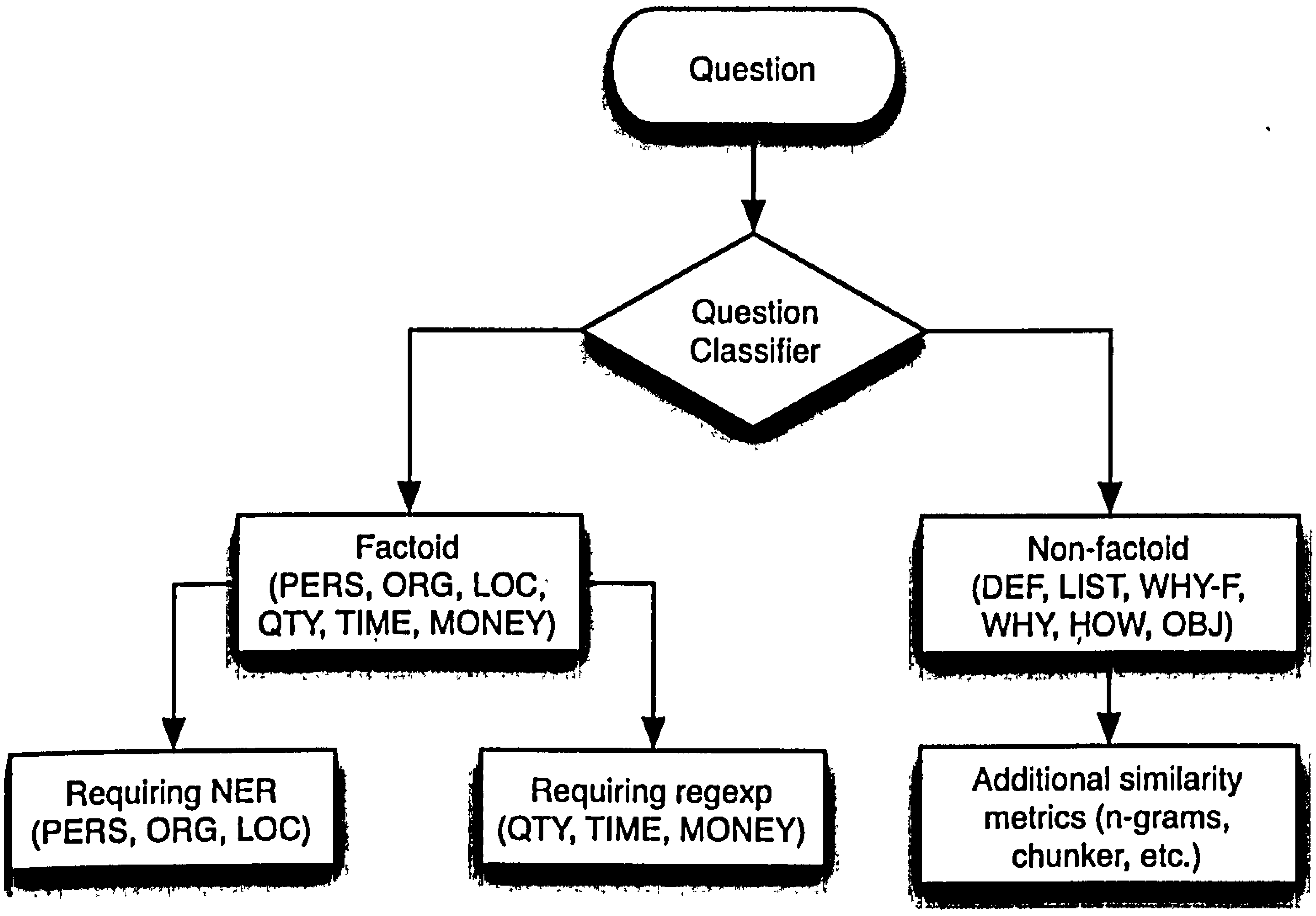

Figure 2.3: Answer Extraction phase

entities of the required type in the sentence. Otherwise, we must refer to other answer pinpointing strategies.

PERS, ORG, LOC

If the expected answer type corresponds to the NE classes recognized by Lingpipe, we perform NE recognition on the candidate answer sentences. If a word labelled with the required NE type is found, it is assigned a score which depends on the distance between such word and the closest question keyword found in the sentence.

Candidate answer sentences are therefore compared based on the following criteria:

1. number of common keywords between question and sentence;
2. distance between the closest named entity of the required type and the question keywords in the sentence;
3. Google rank of the document containing the sentence.

Hence, candidate answers are reordered based on the additional criterion of the named entity distance, which is applied as a secondary criterion to the bag-of-words score; the Google rank of the original answer document is used as a tie-breaker. The accuracy of answer extraction therefore inevitably relies on the precision and recall of the NE recognizer.

QTY, TIME, MONEY

If the expected answer is a factoid but belongs to a type which cannot be spotted by LingPipe, we refer to a dozen rules based on regular expressions and the sentence POS tags; these have been written manually for each expected answer type. Some examples of patterns are reported in Table 2.8.

<table>
<tr><th>Time expression pattern</th><th>Match example</th></tr>
<tr><td>(day s*,{0,1}){0,1} month (s*d{1,2},{0,1}){0,1} (s*d{2,4}){0,1}</td><td>Mon May 12, 99</td></tr>
<tr><td>(s*d{2,4}s*)(-|to)(s*d{2,4})s*</td><td>(1976- 1998)</td></tr>
<tr><td>d{1,2}/d{1,2}/d{2,4}</td><td>12/11/82</td></tr>
<tr><td>((A|a)fter|(B|b)efore|(U|u)ntil|(D|d)uring|(I|i)n)s+d{3,4}</td><td>Until 1997</td></tr>
<tr><th>Money expression pattern</th><th>Match example</th></tr>
<tr><td>d+(([,.]{0,1})d+)*(hundred|...|billion|Kb|Mb)*[pcK]{0,1}[?$]</td><td>10 million $</td></tr>
<tr><td>d+(([,.] {0,1})d+)*(hundred|...|billion|Kb|Mb)* *(euro|dollars|...)</td><td>10.2 M dollars</td></tr>
</table>

Table 2.8: Sample patterns used during answer extraction in YourQA

The presence or absence of a sentence substring matching the given rules is once again taken as an additional similarity criterion between the question and each candidate answer; the re-ranking criteria thus become:

1. number of common keywords between question and sentence;
2. presence of an expression matching the rules written for the required answer type;
3. Google rank of the document containing the sentence.

2.5.2 Non-factoid Answers

We assign to the non-factoid group the purely non-factoid answer types, i.e. WHY, HOW, WHY-F, DEF, LIST as well as the OBJ type which is too generic to be grasped by a factoid answer approach.

In these cases, we aim at more sophisticated sentence similarity metrics than the simple bag-of-word metric applied previously; however, we cannot benefit from Named Entity recognition and the design of hand-written rules seems very complex, especially in a Web context where there can be endless ways of phrasing definitions, reasons etc.

The solution adopted in YourQA is a blend of several similarity metrics, which are combined with the bag-of-words similarity metric using a weighted sum.

N-gram similarity

In many cases, the presence of question keywords in a candidate answer is not a sufficient criterion to establish a strong similarity between the question and such answer: it may be advisable to verify that the keywords are close enough or in the same order in both sentences. This is why we resort to n-gram similarity, which is a function of the number of common keyword n-grams between the question and answer. We define:

$$ng(q,a) = \frac{|commonN(q,a)|}{|ngrams(q)|} \tag{2.2}$$

where $commonN(q,a)$ is the number of shared n-grams between q and a and $ngrams(q)$ is the set of question n-grams. In the current version of YourQA, bigrams are used, i.e. $n = 2$.

Chunk similarity

Sentence chunks can be defined as groups of consecutive, semantically connected words in the sentence, which can be obtained using a shallow parser (in our case, the one provided by the OpenNLP chunker[10]). While any sequence of n tokens taken from a text can be said to be an n-gram, a chunk is a group of tokens bearing semantic information and hence potentially much more informative. For example, in the sentence: "Shallow parsing is an analysis of a sentence which identifies the constituents, but does not specify their internal structure.", the bigram "shallow parsing" is a valid chunk, while "parsing is" is not.

The chunk similarity, $chk(q,a)$, is a function of the number of common chunks between q and a, $|commonC(q,a)|$. The similarity is then normalized by dividing $|commonC(q,a)|$ by the total number of chunks in q, $|chunks(q)|$:

$$chk(q,a) = \frac{|commonC(q,a)|}{|chunks(q)|} \tag{2.3}$$

where $commonC(q,a)$ is the number of shared chunks between q and a and $chunks(q)$

[10] `http://opennlp.sourceforge.net`

is the set of question chunks.

Head NP-VP-PP similarity

This is a variation of the chunk similarity metric, where we focus on word groups composed by a noun phrase, a verb phrase and a prepositional phrase (labelled NP, VP and PP, respectively, by the chunker). The idea is to match the group formed by the semantically most important word composing the NP (also called head-NP) and by the following VP[11] and PP chunk (the first one in case several PPs occur after the VP). As an approximation of the semantic head of a NP, we apply the algorithm designed in Collins (1999) for obtaining syntactic NP heads. The head NP-VP-PP similarity is defined as:

$$hd(q,a) = \mu \times HNPmatch(q,a) + \nu \times VPmatch(q,a) + \xi \times PPmatch(q,a) \quad (2.4)$$

where:

- $VPmatch(q,a)$ is computed by identifying the VPs in q and a which share the maximum number of tokens; such optimal VPs are henceforth called called $maxVP_q$ and $maxVP_a$ and VPmatch(q,a) is their number of shared tokens between $maxVP_q$ and $maxVP_a$,
- HNPmatch(q,a) is the number of common tokens between the HNPs associated with $maxVP_q$ and $maxVP_a$, respectively,
- PPmatch(q,a) is the number of common tokens between the PPs associated with $maxVP_q$ and $maxVP_a$, respectively,

and μ, ν and ξ are carefully chosen weights. The current version of YourQA uses $\mu = \nu = .4, \xi = .2$.

WordNet similarity

As an additional semantic similarity metric, we use the Jiang-Conrath distance (Jiang & Conrath, 1997), which is defined over a lexical database. Given a database D, the Jiang-Conrath metric combines:

- $P(q_i)$ and $P(a_j)$, i.e. the probabilities of occurrence in D of the i-th word in the question and the j-th word in the answer, respectively;

[11] VPs are lemmatized.

- the probability of occurrence in D of $mscs_{q_i,a_j}$, i.e. q_i and a_j's *most specific common super-ordinate*[12].

The final distance metric is:

$$d(q_i, a_j) = IC(q_i) + IC(a_j) - 2 \cdot IC(mscs_{q_i,a_j}),$$

where $IC(n) = log^{-1}P(n)$ is the *information content* of word n and $P(n)$ is approximated by the frequency in D of n. For an evaluation of various semantic distance metrics and the advantages of use of the Jiang-Conrath metric, we refer the reader to Jiang & Conrath (1997). As the underlying database, WordNet 2.0 (Miller, 1995) is used. The resulting similarity metric, named WordNet similarity, is therefore defined as:

$$wn(q, a) = 1 - \frac{\sum_{i<|q|, j<|a|} d(q_i, a_j)}{|q|}. \tag{2.5}$$

Combined non-factoid similarity

The similarity formula resulting from the combination of the similarity metrics above is:

$$sim(q, a) = \alpha \times bow(q, a) + \beta \times ng(q, a) + \gamma \times chk(q, a) + \delta \times hd(q, a) + \epsilon \times wn(q, a) \tag{2.6}$$

Based on empirical observation of YourQA's results, the α, β, γ and δ coefficients have been tuned to their current values of $\alpha = .6, \beta = .2, \gamma = \delta = .1$ and $\epsilon = 0$.

Based on this similarity, the non-factoid re-ranking criteria are:

1. The combined similarity between question and candidate answer, $sim(q, a)$, as computed in (2.6);

2. The Google rank of the document containing the candidate answer in case of a tie.

The use of the original search engine rank (in this case, Google) as a secondary criterion for answer ranking may appear debatable. On the one hand, the information retrieval algorithms applied by search engines aim at returning relevant information at the *document* level. This means that search engine results are ranked based on a variety of metrics that take into account the whole document, hence relevant information may be spread across the document rather than being concentrated in one sentence or passage.

[12]The most specific common super-ordinate or hypernym between two nodes n_1 and n_2 in a lexical hierarchy is a node in the hierarchy which satisfies two conditions: a) it is an ancestor for both the considered nodes; b)no deeper node in the hierarchy satisfies condition a). For instance, $mscs(cat, panther) = feline$.

Moreover, it could be argued that there is no guarantee that a document returned by a search engine because of its relevance to the query actually contains the desired answer. This is especially true in the case of factoid questions, where the information required is very specific.

On the other hand, it is also true that modern search engines apply sophisticated retrieval techniques and ranking algorithms such as PageRank (Page *et al.*, 1998; Langville & Meyer, 2006), yielding documents which are highly likely to contain answers. This is indeed the working hypothesis on which we found our Web-based QA research. Based on these observations, we argue that the original rank given by the search engine is a reliable indicator of the informative content of the corresponding document.

Furthermore, *ceteris paribus*, we assert that it is more likely that, if one *document* D_1 has been judged more relevant to a given question q than another document D_2, the closest answer *sentence* to address q extracted from D_1 is more useful than the one extracted from D_2.

These are our arguments to choose search engine rank as a secondary ranking criterion for our candidate answers.

2.6 Result Format

This section illustrates how the answers produced by the answer extraction algorithm in Section 2.5 are returned to the user. Although different interfaces exist for different versions of the YourQA system (see Chapter 6), the result format described below is common to all of them.

As illustrated in Figure 2.4, a result (i.e. an answer) in YourQA is an object composed by two elements:

1. A *header* containing useful information about the answer's document of origin (described in Section 2.6.1);

2. A *passage* centered around the closest sentence to the question (described in Section 2.6.2).

These are described below.

2.6.1 Answer Header

Each answer passage is preceded by a header providing useful information about the answer. The mandatory objects appearing in the header are:

1. Title: GradeSaver: ClassicNote: About Pride and Prejudice, **URL:** http://www.gradesaver.com/classicnotes/titles/pride/about.html, **Google Rank:** 6, **file:** about.html

About Pride and Prejudice.

Pride and Prejudice, published in 1813, is Jane's Austen's earliest work, and in some senses also one of her most mature works.

Austen began writing the novel in 1796 at the age of twenty-one, under the title First Impressions.

Figure 2.4: Top result to the question: "When was *Pride and Prejudice* published?"

1. The rank assigned by the answer extraction algorithm,
2. The title of the original Web-page,
3. A clickable URL of such page, which the user can load for more information.

Optional information in the answer header consists in the original Google rank of the passage or, in the case the personalization module is active (see Chapter 4), the weight of the answer with respect to the user profile (as computed in Section 4.4.2).

2.6.2 Answer Passage

The answer passage is centered around the closest sentence to the query as detected by the procedure in Section 2.5. Such sentence is highlighted in the text (currently in boldface, as illustrated in Figure 2.4) and enclosed by a context composed of up to two preceding and two following sentences (depending on the total length of the resulting passage: the maximum length limit for the answer passage is currently fixed to 750 bytes).

In the answer passage, several types of information are highlighted:

1. The answer sentence is visualized in boldface;
2. The question's keywords matched in the answer *passage* are in color throughout the passage (in Figure 2.4, the words *Pride* and *Prejudice* appear in navy blue);
3. Matched question keywords in the answer *sentence* are also visualized in color (in Figure 2.4, *Pride*, *Prejudice* and *published* appear in purple), to indicate that these made the current sentence a top answer within the document where it appears.
4. Additional keywords and expressions are highlighted depending on the expected answer type, as explained below.

Factoid Passage Format

For questions where the expected answer is a factoid, the answer format is refined in order to highlight factoids of interest within the returned passage. The expected types of interesting factoids are estimated based on the top two expected answer types as predicted by the question classification module, which we name EAT_1 and EAT_2.

In some cases, for instance in the presence of query adverbs such as "Where" and distinctive NPs such as "Who", we judge it useful to highlight the presence of locations and names, respectively, in the answer passages. This decision overrides the output of the Question Classifier which, as illustrated in Section 2.3, has a high accuracy (around 80%) but is not extremely precise due to the difficulty of distinguishing between eleven classes.

Table 2.9 summarizes the rules of attribution of interesting factoids according to the above two criteria. Rule 11 shows an example of a case where overriding occurs for queries starting with the "Where" adverb: in this case, the expected answer type is set to "Location" regardless of the output of the QC. A similar case happens for the "Who" adverb: it suffices that one of the top two predictions is labelled PERS to fix "Name" as the expected answer type (see Rules 1 and 2).

It may be worth reminding that, as explained in Section 2.5, the expected factoids in the last column of Table 2.9 are located using the following strategies:

- Named Entity recognition for the Location, Name and Organization types;
- Manual regular-expression patterns in the case of Date and Numerical Quantity types.

Factoid term coloring Factoid terms corresponding to the desired type are visualized in color within the result passages, where different colors correspond to different types of terms. For instance, as visible in Figure 2.4, time patterns such as "in 1813" are visualized in light blue.

Despite being rather an implementation matter, a summary of the scope of term coloring within factoid answer passages may be worth mentioning. This is reported in Table 2.10.

2.6.3 Standard User Interface

The standard version of YourQA has the characteristics and behavior of a typical Web-based Question Answering system. As in traditional QA systems, a Question Answering

Rule	EAT_1	EAT_2	Query adverb	Expected factoid type
1	PERS	ϕ	"Who"	Name
2	ϕ	PERS	"Who"	Name
3	QTY	ϕ	ϕ	Numerical Quantity
4	TIME	ϕ	ϕ	Date
5	ORG	ϕ	ϕ	Organization
6	ϕ	ORG	ϕ	Organization
7	OBJ	ϕ	ϕ	Organization
8	ϕ	OBJ	ϕ	Organization
9	PLACE	ϕ	ϕ	Location
10	ϕ	PLACE	ϕ	Location
11	ϕ	ϕ	"Where"	Location

Table 2.9: Result passages: expected factoid estimation. ϕ indicates that the value in the corresponding field is not relevant. Notice that the Organization type is among the most difficult to classify and typically mistaken with the Object type. Hence, when the EAT is "OBJECT" and organizations are identified within the passage by the NE recognizer, we find it useful to highlight such entities (Rules 5 to 8).

Factoid type	Scope of coloring
Name	Answer sentence
Location	Answer sentence
Organization	Answer sentence
Date	Answer sentence; if no temporal expression found in answer sentence, whole answer passage
Numerical Quantity	Answer sentence; if no quantity expression found in answer sentence, whole answer passage

Table 2.10: Factoid type and corresponding coloring scope

session in this version of YourQA consists of a single question-answer pair and no notion of context is maintained. In the user interface (see Figure 2.5), users type their question in a text field, submit it using a button and results are loaded into a Web page accessible through a link (see Figure 2.6).

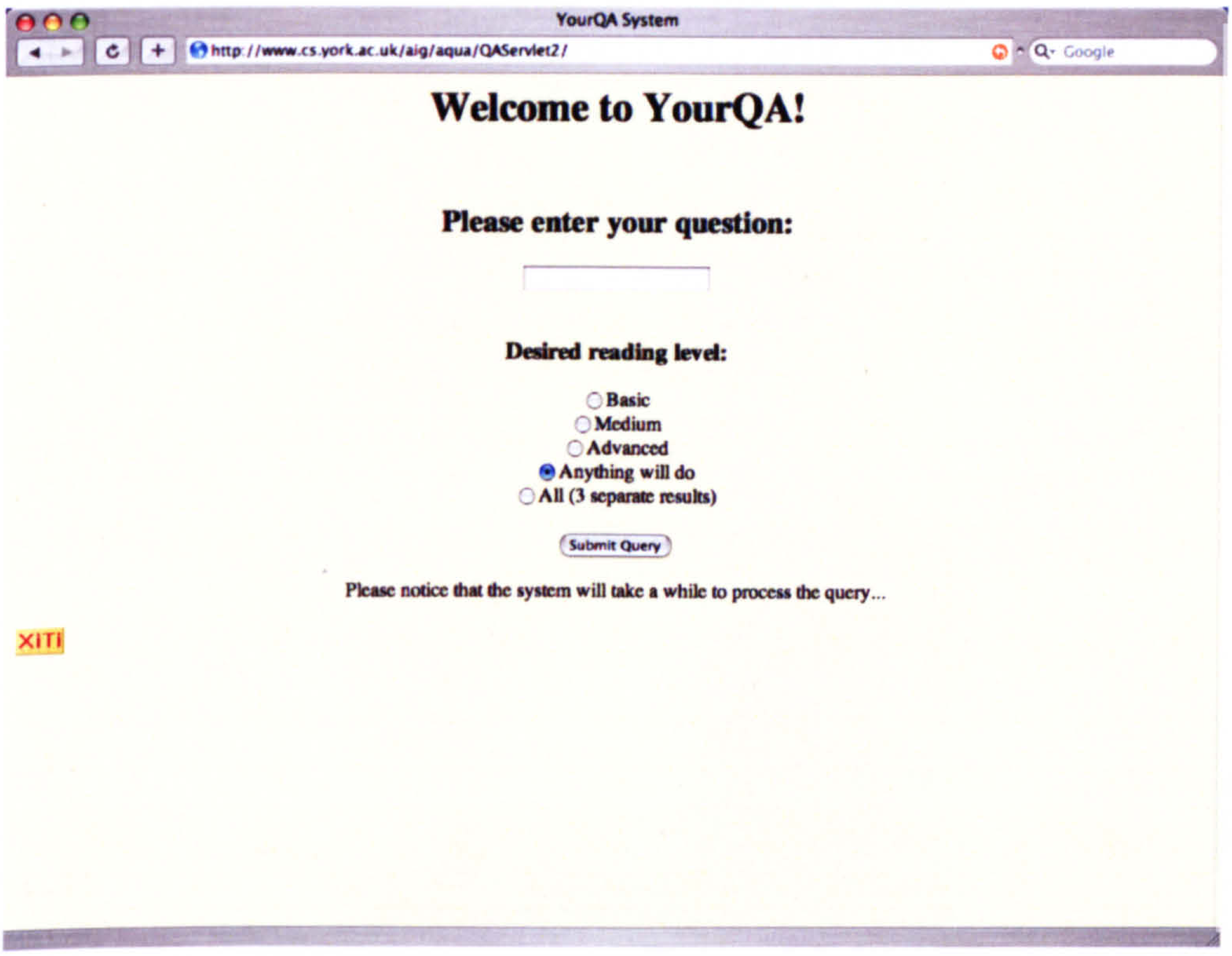

Figure 2.5: Standard YourQA interface: entry page

When clicking such link, the user finds an HTML page carrying the list of top answers in the form of short paragraphs. In the current implementation of YourQA, the top results are represented as HTML list elements and returned in an HTML page (see Figure 2.7). Such result page is structured in the following way:

1. A title, containing the original question (e.g. "*When was Pride and Prejudice published? – results*");
2. A summary of the query, where the question keywords used to produce answers are highlighted;
3. The expected answer type (e.g. "*Expected answer type => [TIME HOW]*");

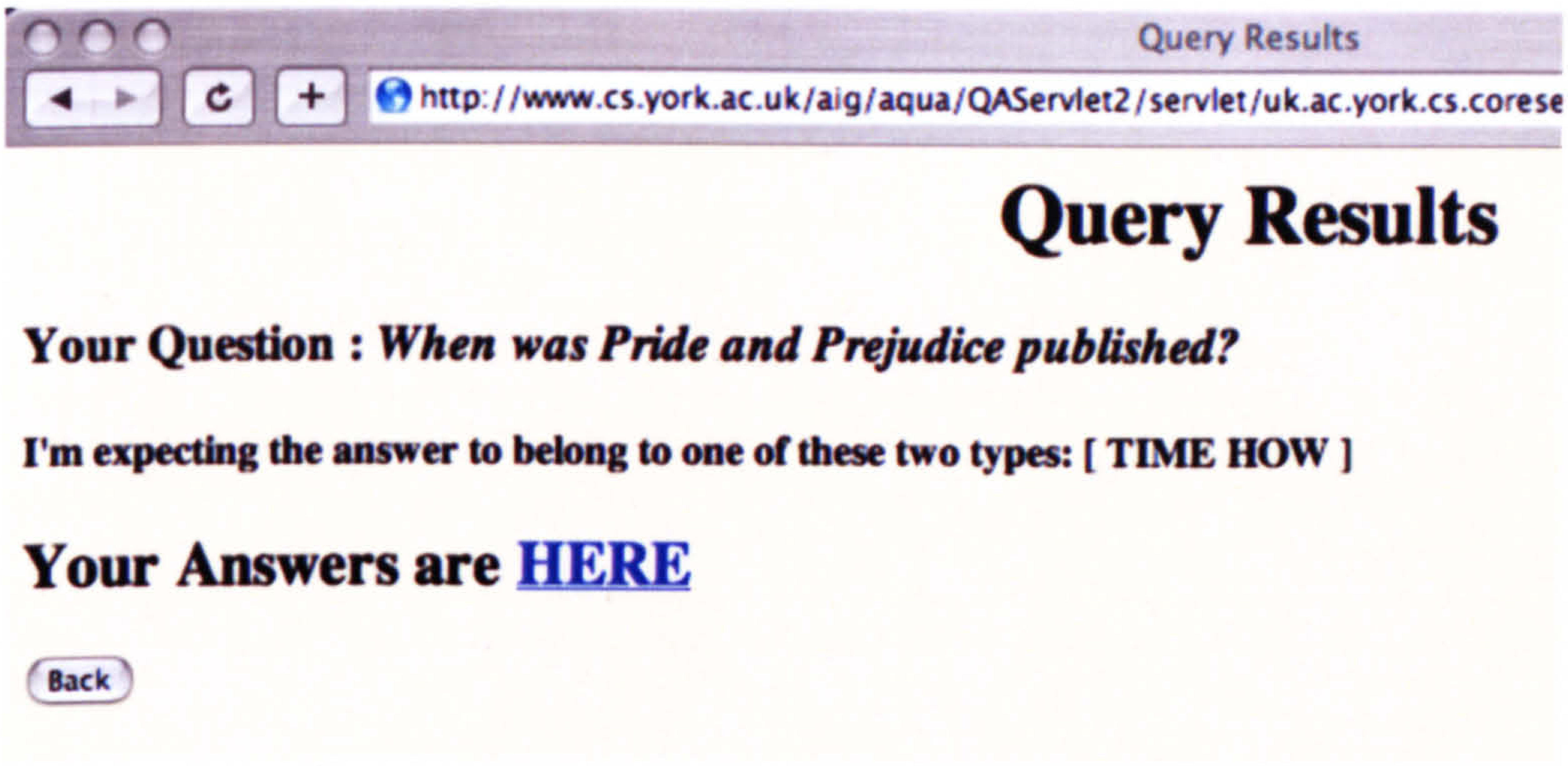

Figure 2.6: Standard YourQA interface: result page

4. A legend explaining the color coding of the results;

5. The ordered list of answers, formatted as illustrated in Sections 2.5.1 and 2.5.2.

The motivation behind this choice of result layout is that such format makes the results accessible from different types of desktop and Web interfaces: up to date, three servlet versions and one applet version exist for the YourQA system, all of which access results in the format described above. These are described in Chapters 4 and 5.

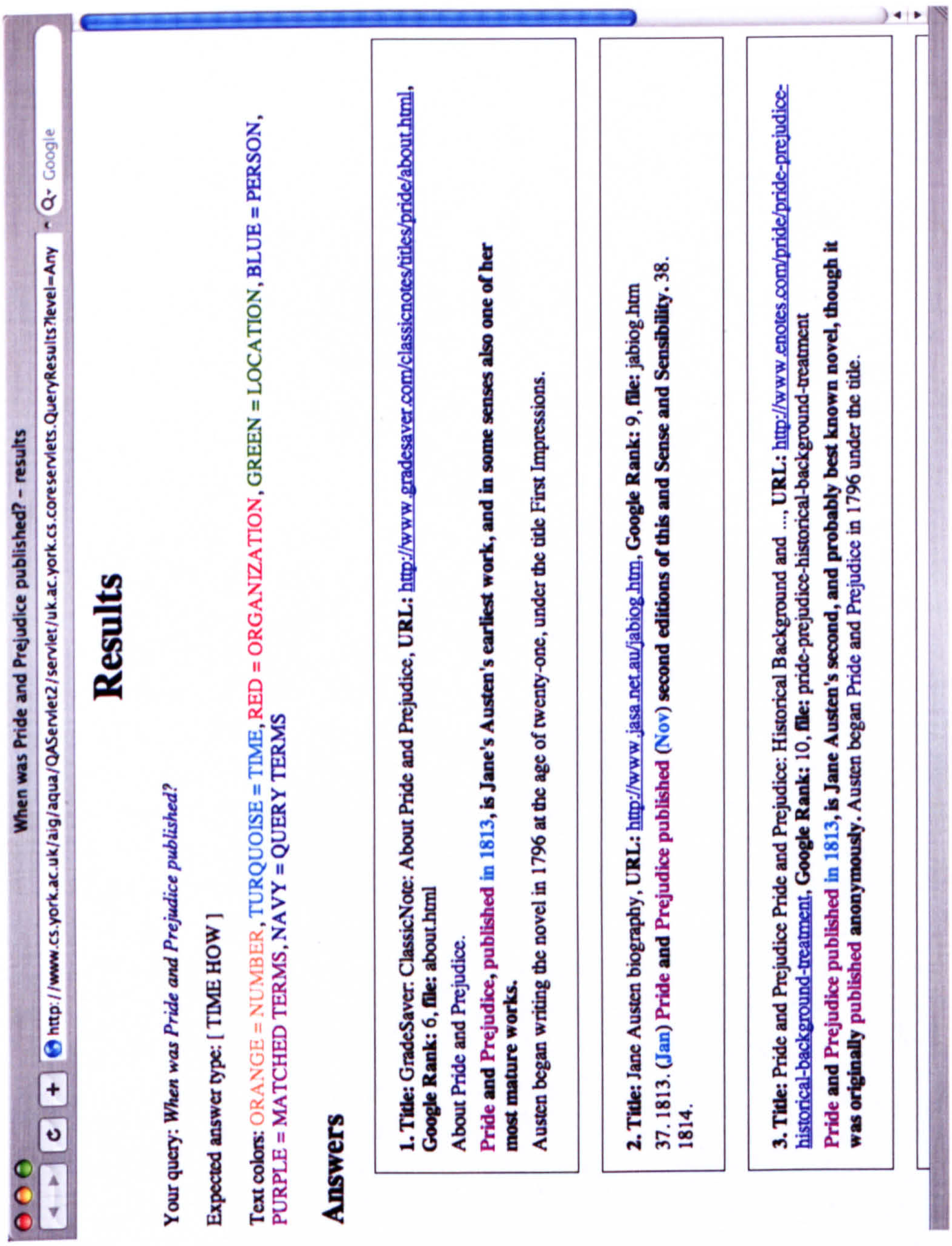

Figure 2.7: YourQA's result for the question: *"When was Pride and Prejudice published?"*

Chapter 3

Advanced Question Answering

As explained in the previous chapter, question classification and answer classification and re-ranking are vital tasks in a Question Answering system.

Question processing is often centered on question classification, which selects one of k expected answer classes. Most accurate models apply supervised machine learning techniques to implement classifiers, e.g. SNoW (Li & Roth, 2005), where questions are encoded using various lexical, syntactic and semantic features.

In the answer extraction phase, answer classification is often used as a method to detect predefined types of answers such as definitions; a further answer re-ranking phase is optionally applied. Here, too, the syntactic structure of a sentence appears to provide more useful information than a bag of words (Chen *et al.*, 2006), although the correct way to exploit such structure is still an open problem.

An effective way to integrate syntactic structures in machine learning algorithms is the use of tree kernel (TK) functions (Collins & Duffy, 2002), which have been successfully applied to question classification (Zhang & Lee, 2003) and other tasks, e.g. relation extraction (Zelenko *et al.*, 2003; Moschitti, 2006). In more complex tasks such as computing the relatedness between questions and answers in answer re-ranking, to our knowledge no study uses kernel functions to encode syntactic information.

Moreover, the study of shallow semantic information such as predicate argument structures annotated in the PropBank project (Kingsbury & Palmer, 2002) is relatively recent and approaches handling such information automatically still need investigation. We argue that semantic structures can be used to characterize the relation between a question and a candidate answer.

In this chapter, we extensively study new structural representations, encoding parse trees, bag-of-words, POS tags and predicate argument structures (PASs) for question classification, answer classification and answer re-ranking. We present tree representations for both simple and nested PASs, i.e. PASs whose arguments are other predicates. Moreover, we introduce kernel functions to exploit PASs, which are automatically derived using the Semantic Role Labeling system described in (Moschitti *et al.*, 2005, 2008).

Our experiments using SVMs and the above kernels and data are reported in Section 3.4. The main findings of such experiments are the following:

1. Our approach reaches state-of-the-art accuracy on question classification;

2. PropBank-based predicate-argument structures are not effective for question classification.

3. However, predicate-argument structures show promising results for answer classification when applied on a corpus of answers found by YourQA to TREC-QA 2001 description questions. The latter are the 138 TREC-QA 2001 questions labelled as "DESC" according to the previously introduced UIUC taxonomy[1], also used in Li & Roth (2002).

4. The best SVM answer classifier increases the ranking accuracy of our QA system by about 25% in terms of MRR.

This chapter is structured as follows: Section 3.2 introduces advanced models to represent syntactic and semantic information in a QA context. Section 3.3 explains how such information is exploited in an SVM learning framework by introducing novel tree kernel functions. Section 3.4 reports our experiments on question classification, answer classification and answer re-ranking. Finally, Section 3.5 concludes on the utility of the new structure representations and sets the basis for further work.

3.1 Advanced Models for Sentence Representation

Traditionally, the majority of information retrieval tasks have been solved by means of the so-called *bag-of-words* approach augmented by language modelling (Allan *et al.*, 2002). However, when the task requires the use of more complex semantics the above approach does not appear to be effective, as it is inadequate to perform fine-level textual analysis. To overcome this, QA systems use linguistic processing tools such as syntactic parsers to produce sentence parse trees. In our studies, reported in Quarteroni *et al.* (2007); Moschitti *et al.* (2007), we exploited two sources of syntactic information: deep syntactic parsing and shallow semantic parsing.

[1]available at: `http://l2r.cs.uiuc.edu/~cogcomp/Data/QA/QC/`

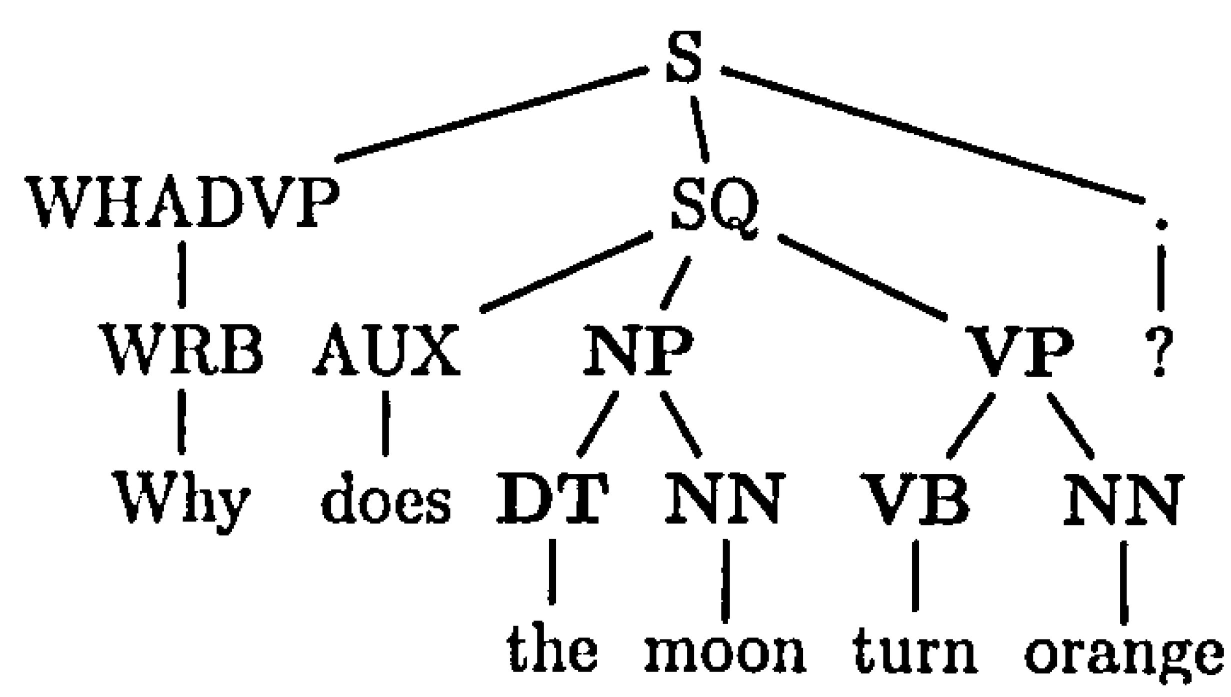

Figure 3.1: Parse tree of a question from TREC-QA 2001

While the former is a fully exploited technology able to derive syntactic parse trees from a sentence (Collins & Duffy, 2002; Charniak, 2000), the latter has recently been the object of a consistent body of work. Shallow semantic parsing aims at detecting and labelling a proposition with the relations between its components, i.e. predicates and arguments.

3.1.1 Syntactic Structures

The syntactic parse tree of a sentence is a hierarchical representation of the syntactic relationships between its words. In such tree, each node with its children is associated with a grammar production rule, where the symbol at the left-hand side corresponds to the parent and the symbols at the right-hand side are associated with the children. The terminal symbols of the grammar are always associated with the leaves of the tree. As an example, the parse tree for a question from TREC-QA 2001 is reported in Figure 3.1.

Parse trees have often been used in natural language processing applications requiring the use of grammatical relations, e.g. extraction of subject/object relations. It has been shown (Zhang & Lee, 2003; Moschitti, 2006) that syntactic information outperformed bag-of-words and bag-of-n-grams on question classification in QA. Indeed, the advantage of computing parse tree-based sentence similarity with respect to purely lexical approaches is that trees provide structural relations hard to compute otherwise.

For instance, let us consider question q: "*Why does the moon turn orange?*" and the sentences:

- s_1:"*The moon turns orange during an eclipse.*"
- s_2:"*The orange moon turns around the Earth.*"

From a bag-of-words point of view, there is no reason to prefer s_1 to s_2 as an answer; however, when we analyze the parse trees of q, s_1 and s_2 (see Figures 3.2 and 3.3), there

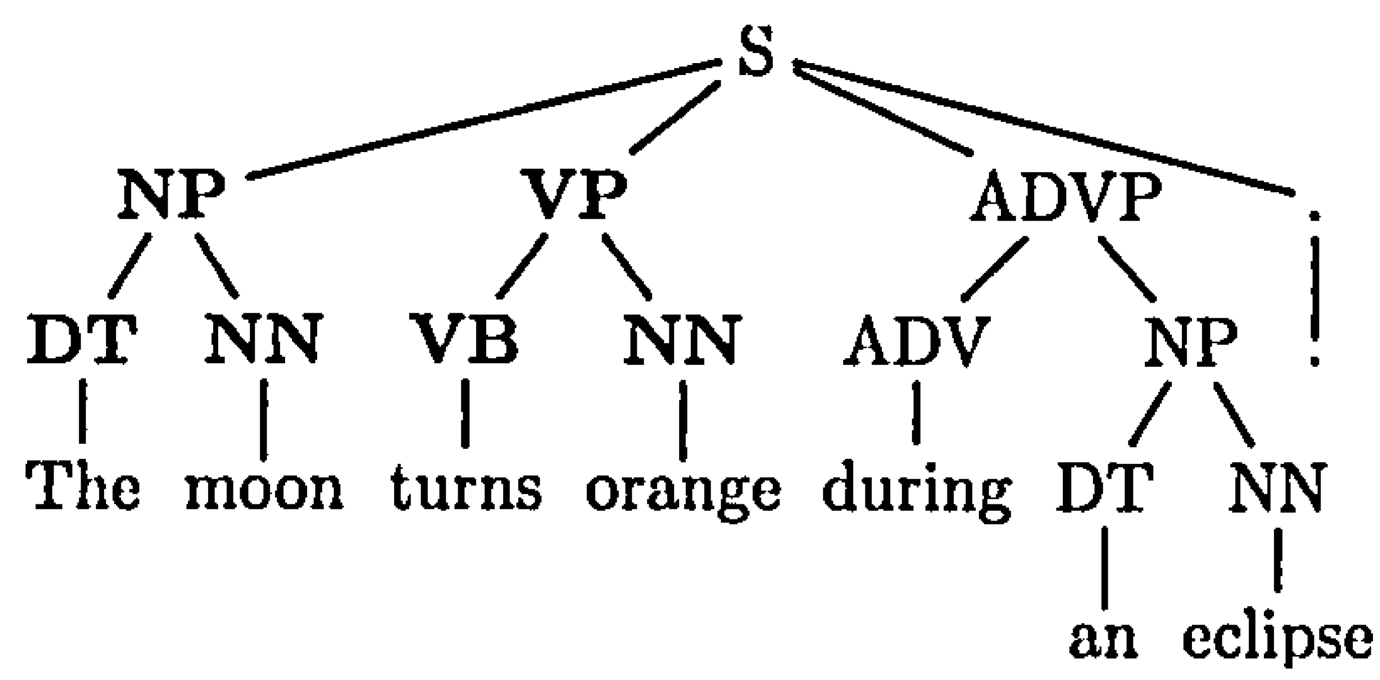

Figure 3.2: Parse tree of s_1:"*The moon turns orange during an eclipse.*"

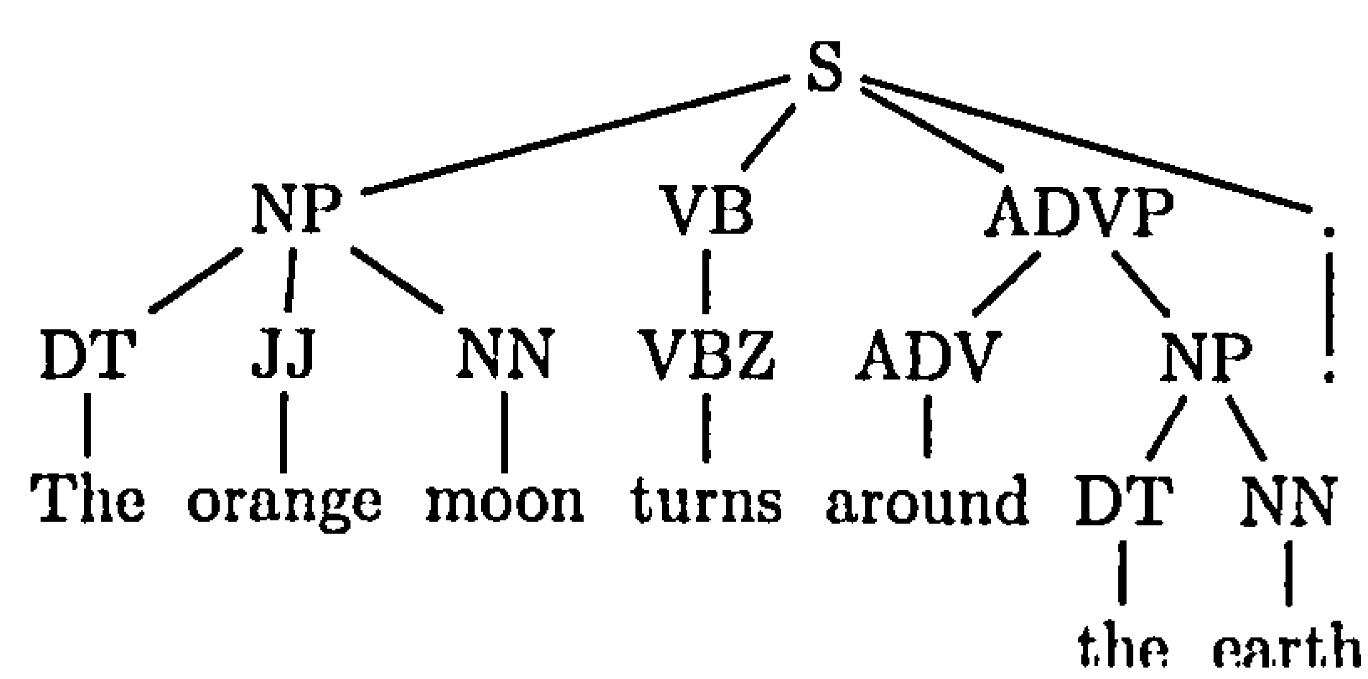

Figure 3.3: Parse tree of s_2:"*The orange moon turns around the Earth.*"

is more overlap between the parse tree of q and s_1 than between the parse tree of q and s_2.

A successful example of the use of parse trees for QA is their application in a question classification task; Zhang & Lee (2003) question showed that parse trees combined with the question words outperformed bag-of-words and bag-of-n-grams on the six class coarse-grained taxonomy defined in Li & Roth (2002).

However, when approaching complex QA tasks, the use of parse trees has some limitations. For instance, in definitional QA candidate answers can be expressed by long and articulated sentences or even paragraphs. Since the information encoded in a parse tree is intrinsically sparse, it does not contribute well to computing the similarity between such answers; shallow semantics however, being a more "compact" source of information, could prevent the sparseness of deep structural approaches and the noise of bag-of-word models.

3.2 Encoding Shallow Semantic Structures

As mentioned above, shallow semantic representations seem a promising research direction to cope with the data sparseness problem. Initiatives such as PropBank (PB) (Kings-

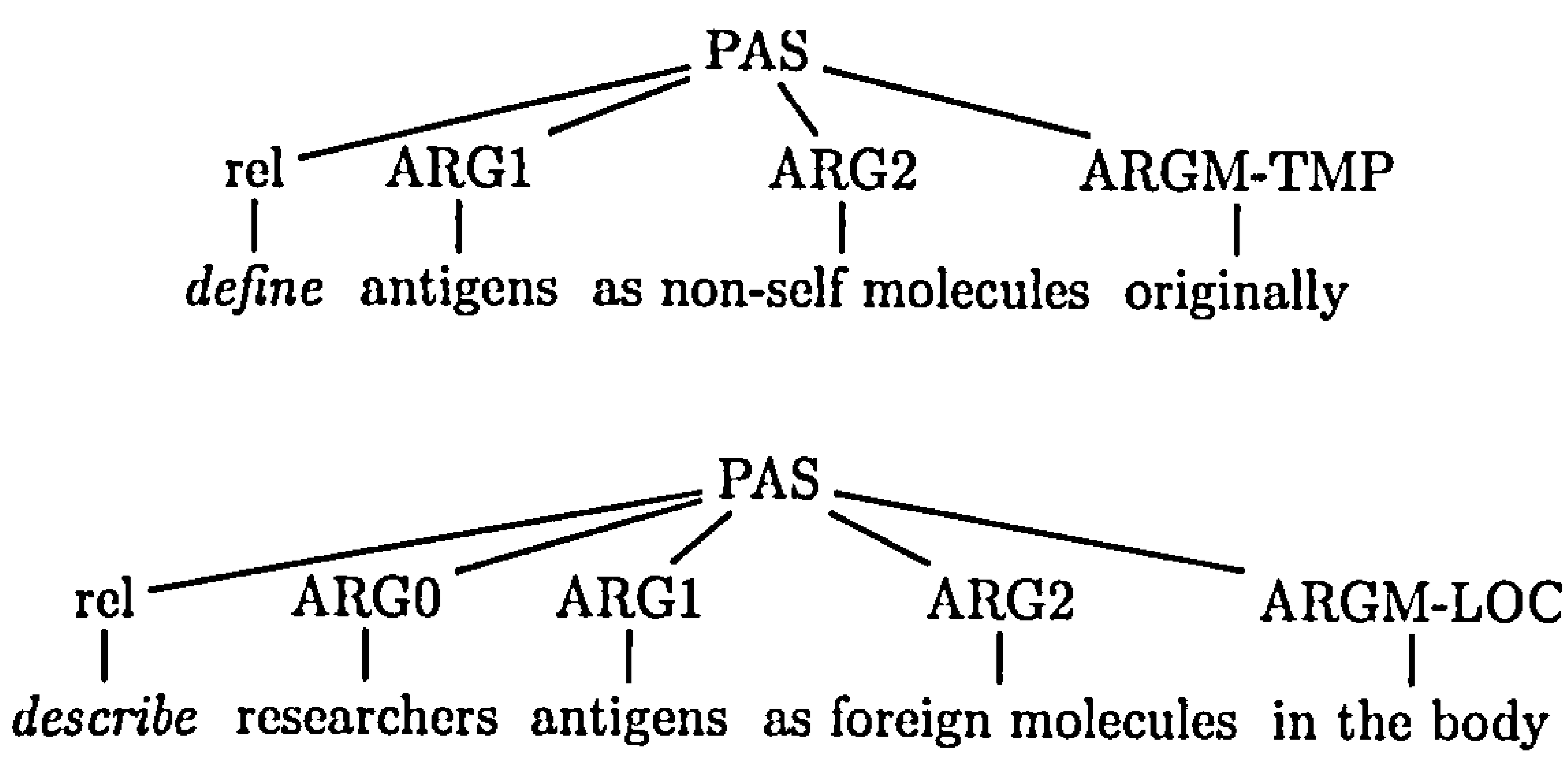

Figure 3.4: Predicate argument structures of two sentences expressing similar semantics.

bury & Palmer, 2002) have made possible the design of accurate automatic Semantic Role Labeling (SRL) systems (Carreras & Màrquez, 2005). The PB corpus contains 300,000 words annotated with predicate-argument information on top of the Penn Treebank 2 Wall Street Journal texts. For each predicate, the expected arguments are labelled sequentially from $ARG0$ to $ARG5$, $ARGA$ and $ARGM$, where the latter two refer to action verb subjects and verb modifiers (e.g. "manner"), respectively.

Attempting an application of Semantic Role Labeling to Question Answering hence seems natural, as pinpointing the answer to a question relies on a deep understanding of the semantics of both. Let us consider the PB annotation:

(1) `[`$_{ARG1}$ `Antigens] were [`$_{AM-TMP}$ `originally] [`$_{rel}$ `defined] [`$_{ARG2}$ `as non-self molecules].`

Such annotation can be used to design a shallow semantic representation that can be matched against other semantically similar sentences, e.g.:

(2) `[`$_{ARG0}$ `Researchers] [`$_{rel}$ `describe] [`$_{ARG1}$ `antigens] [`$_{ARG2}$ `as foreign molecules] [`$_{ARGM-LOC}$ `in the body].`

For this purpose, we can represent the above annotated sentences using the tree structures described in Figure 3.4.

Furthermore, we can improve such representation by substituting the arguments with their most important word – often referred to as the semantic head – as in Figure 3.5. In this compact representation, hereafter Predicate-Argument Structures (PAS), arguments are replaced with their most important word – often referred to as the semantic head. This reduces data sparseness with respect to a typical BOW representation. It seems intuitive that data sparseness can be remarkably reduced by using this shallow representation instead of the BOW representation.

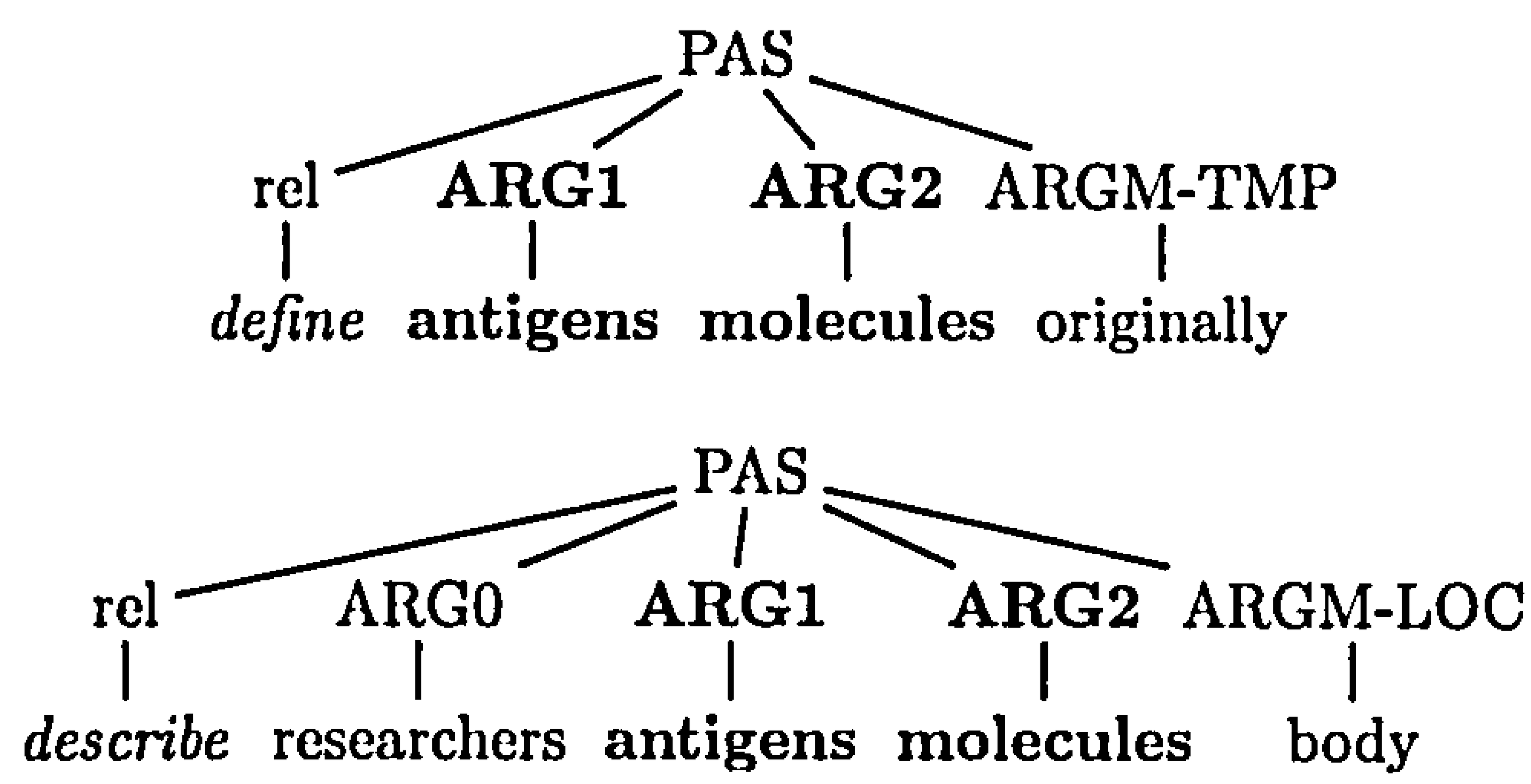

Figure 3.5: Compact predicate argument structures of two different sentences. Arguments ARG1 and ARG2 are associated with the same terminal words.

3.2.1 Nested Predicate Argument Structures

It can be argued that sentences rarely contain a single predicate; it happens more generally that propositions contain one or more subordinate clauses. For instance, let us consider a slight modification of the first sentence: *"Antigens were originally defined as non-self molecules which bond specifically to antibodies*[2]*."* Here, the main predicate is "defined", followed by a subordinate predicate "bond". The SRL system outputs the following *two* annotations:

(3) `[`$_{ARG1}$ `Antigens] were [`$_{ARGM-TMP}$ `originally] [`$_{rel}$ `defined] [`$_{ARG2}$ `as non-self molecules which bond specifically to antibodies].`

(4) `Antigens were originally defined as [`$_{ARG1}$ `non-self molecules] [`$_{R-A1}$ `which] [`$_{rel}$ `bond] [`$_{ARGM-ADV}$ `specifically] [`$_{ARG2}$ `to antibodies].`

giving the PASs in Figure 3.6.

As visible in the first tree in Figure 3.6, when an argument node corresponds to an entire subordinate clause, we label its leaf with PAS, e.g. the leaf of ARG2. Such PAS node is actually the root of the subordinate clause in the second tree of Figure 3.6. Taken as standalone, the individual PASs do not express the whole meaning of the sentence; it is more accurate to define a single structure encoding the dependency between the two predicates as in Figure 3.7. We refer to nested PASs as PASNs.

It is worth to note that semantically equivalent sentences syntactically expressed in different ways share the same PB arguments and the same PASs, whereas semantically different sentences result in different PASs. For example, the sentence: *"Antigens were*

[2]This is an actual answer from YourQA to the TREC 2001 question: "*What are antibodies?*".

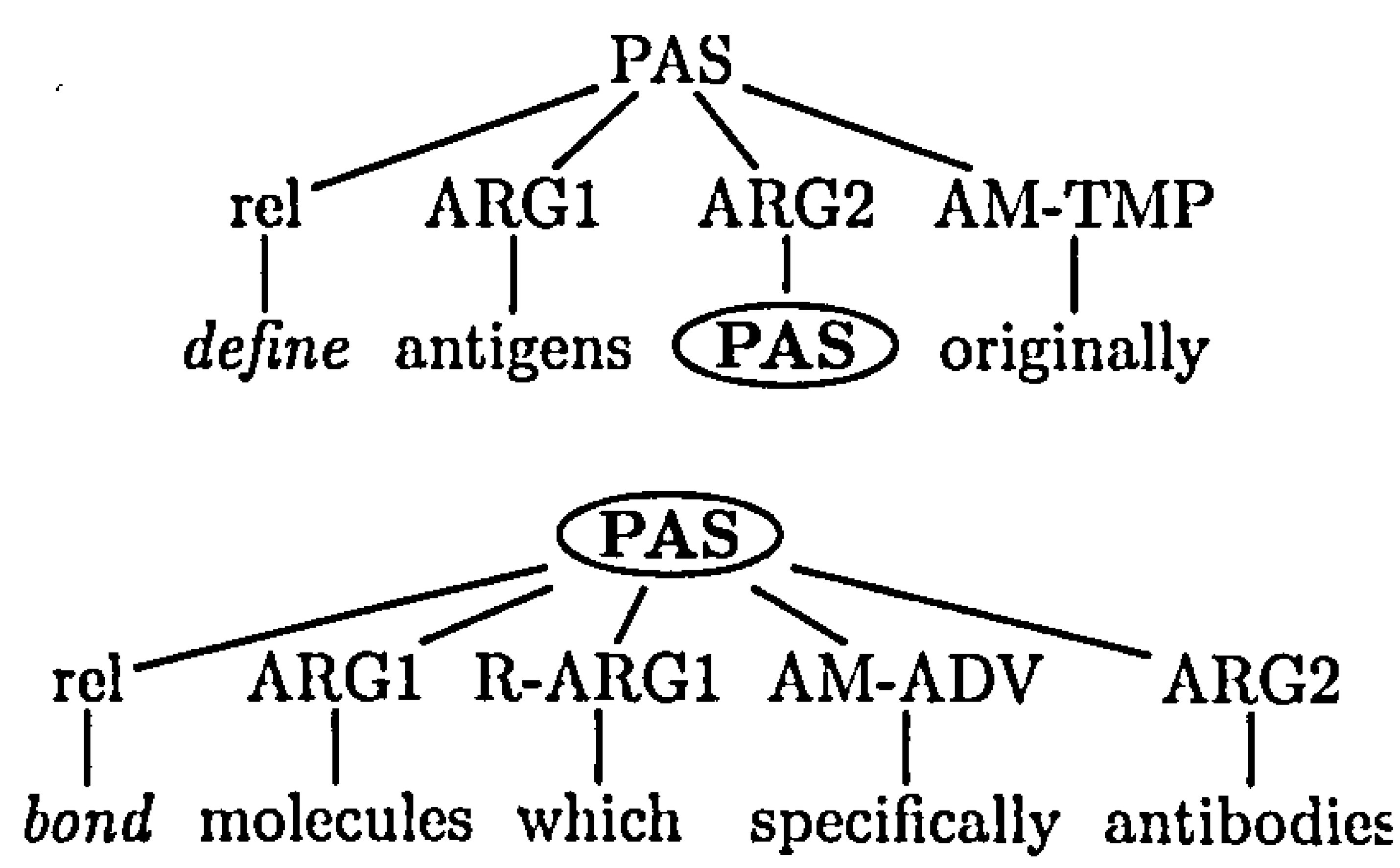

Figure 3.6: Two PASs composing a PASN

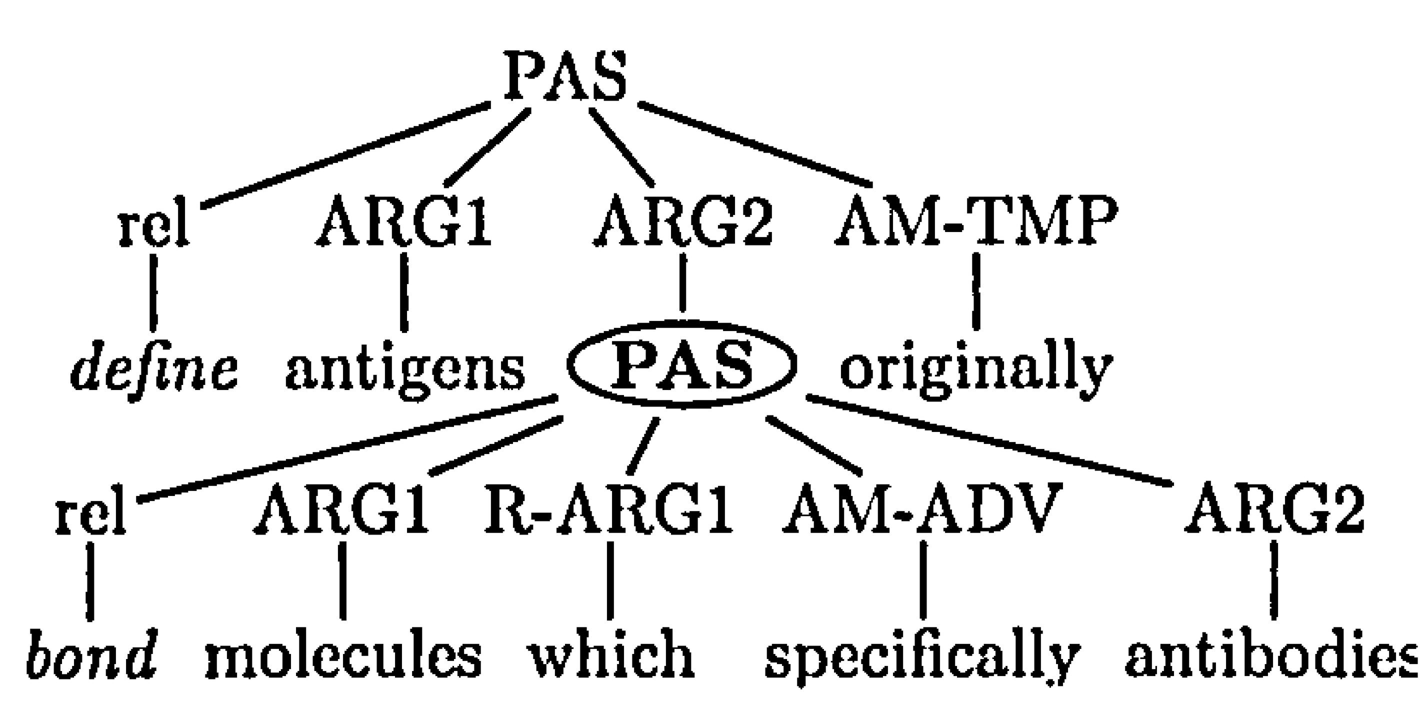

Figure 3.7: Example of a PASN

originally defined as antibodies which bond specifically to non-self molecules", uses the same words as (4) but has different meaning. Its PB annotation:

(5) `Antigens were originally defined as` [$_{ARG1}$ `antibodies`] [$_{R-A1}$ `which`] [$_{rel}$ `bond`] [$_{ARGM-ADV}$ `specifically`] [$_{ARG2}$ `to non-self molecules`]

clearly differs from (4), as ARG2 is now *non-self molecules*; consequently, the PASs are also different.

Once we have assumed that parse trees and PASs can improve on the simple BOW representation, we face the problem of representing tree structures in learning machines. Section 3.3 introduces a viable approach based on tree kernels.

3.3 Syntactic and Semantic Kernels for Text

As mentioned above, encoding syntactic/semantic information represented by means of tree structures in the learning algorithm is problematic. A first solution is to use all its possible substructures as features. Given the combinatorial explosion of considering subparts, the resulting feature space is usually very large.

To manage such complexity we can define kernel functions that implicitly evaluate the scalar product between two feature vectors without explicitly computing such vectors. A tree kernel (TK) function that computes the number of common subtrees between two syntactic parse trees has been given in Collins & Duffy (2002). Below, we report such function.

3.3.1 Collins & Duffy's Syntactic Tree Kernel

Given two trees T_1 and T_2, let $\{f_1, f_2, ..\} = \mathcal{F}$ be the set of their substructures (fragments) and let $I_i(n)$ be equal to 1 if f_i is rooted at node n, 0 otherwise. We define

$$K(T_1,T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1,n_2) \quad (3.1)$$

where N_{T_1} and N_{T_2} are the sets of nodes in T_1 and T_2, respectively and

$$\Delta(n_1,n_2) = \sum_{i=1}^{|\mathcal{F}|} I_i(n_1) I_i(n_2).$$

The latter is equal to the number of common fragments rooted in nodes n_1 and n_2.

Algorithm 2 The Syntactic Tree Kernel

1. if the productions at n_1 and n_2 are different then $\Delta(n_1, n_2) = 0$;
2. if the productions at n_1 and n_2 are the same, and n_1 and n_2 only have leaf children (i.e. they are pre-terminal symbols) then $\Delta(n_1, n_2) = 1$;
3. if the productions at n_1 and n_2 are the same, and n_1 and n_2 are not pre-terminals then

$$\Delta(n_1, n_2) = \prod_{j=1}^{nc(n_1)} (1 + \Delta(c_{n_1}^j, c_{n_2}^j)) \tag{3.2}$$

We can compute $\Delta(n_1, n_2)$ as reported in Algorithm 2. Here, $nc(n_1)$ is the number of children of n_1 and c_n^j is the j-th child of node n. Note that, since the productions are the same, $nc(n_1) = nc(n_2)$.

As proved in Collins & Duffy (2002), Algorithm 2 allows the evaluation of Equation 3.1 in $O(|N_{T_1}| \times |N_{T_2}|)$. A decay factor λ is usually added by changing the formulae in 2. and 3. to:

2. $\Delta(n_1, n_2) = \lambda$,
3. $\Delta(n_1, n_2) = \lambda \prod_{j=1}^{nc(n_1)} (1 + \Delta(c_{n_1}^j, c_{n_2}^j))$.

A normalization in the kernel space, i.e. $K'(T_1, T_2) = \frac{K(T_1,T_2)}{\sqrt{K(T_1,T_1) \times K(T_2,T_2)}}$, ensures a similarity score between 0 and 1.

To illustrate the algorithm, Figure 3.8 shows two parse trees T1 and T2 and the substructures they have in common. It is worth to note that the fragments of the above Syntactic Tree Kernel (STK) are such that any node contains either all or none of its children. Consequently, [NP [DT]] and [NP [NN]] are not valid fragments.

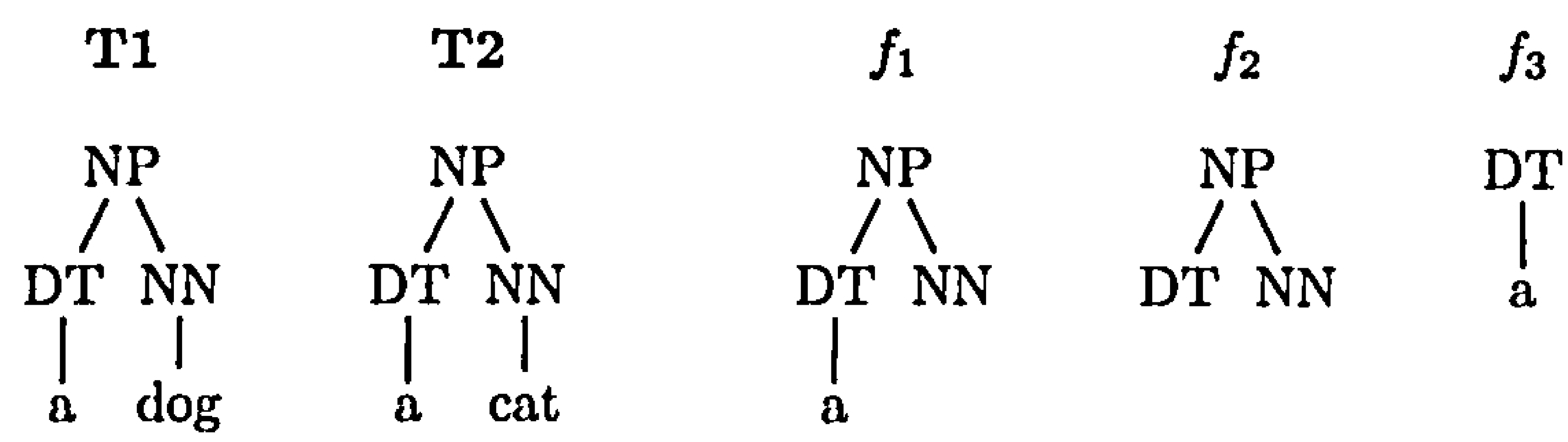

Figure 3.8: Two parse trees, T1 and T2, with their fragments f_1, f_2 and f_3 derived by the STK function

This limitation makes it unsuitable to derive important substructures from the PAS trees defined above, as many important subparts would be neglected. For instance, al-

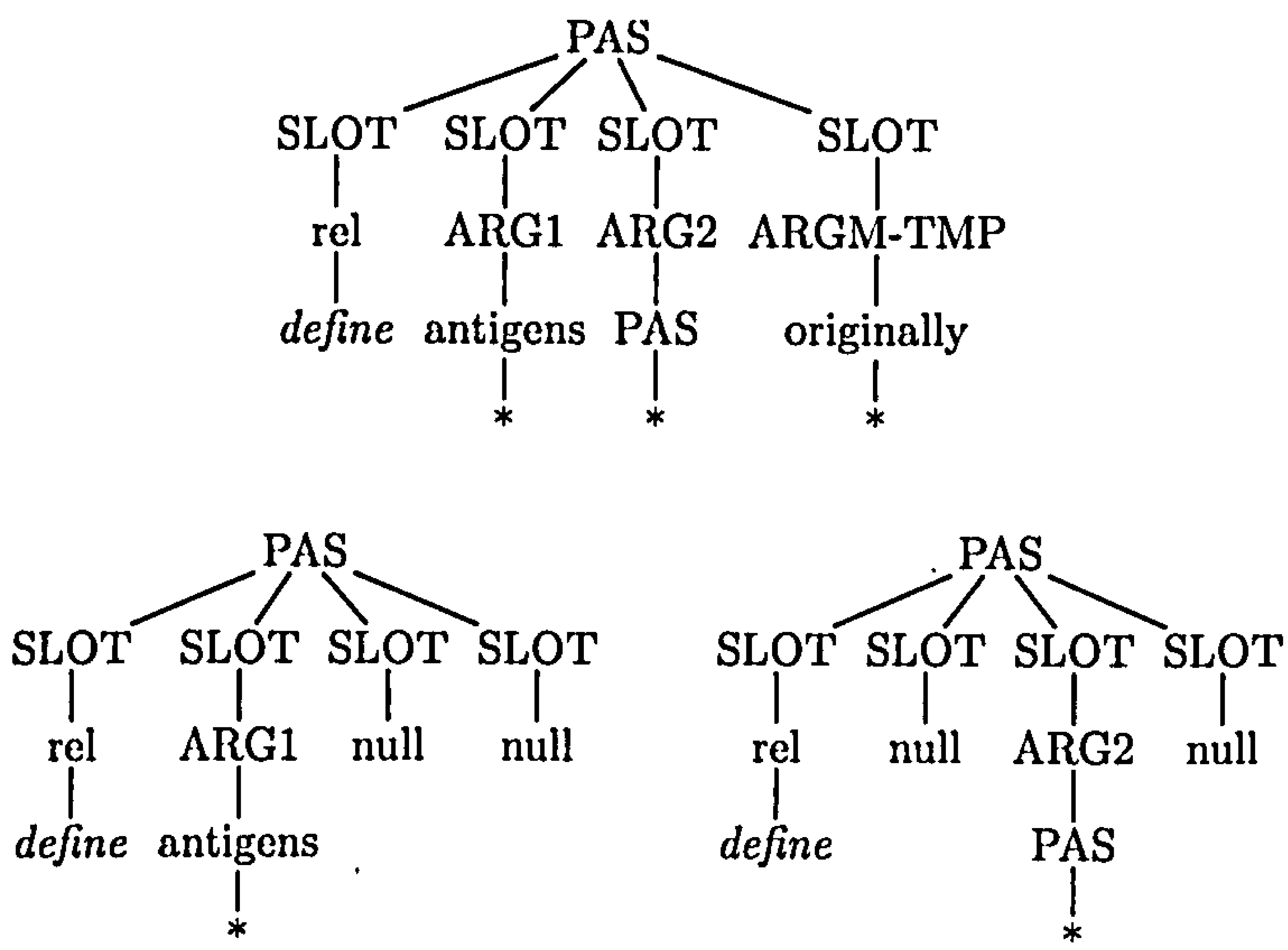

Figure 3.9: A PAS with two of its fragments.

though the two PASs of Figure 3.5 share most of the subtrees rooted in the *PAS* node, Collins and Duffy's kernel would compute no match. This problem was solved in Quarteroni *et al.* (2007) by designing the Shallow Semantic Tree Kernel (SSTK) which allows to match portions of a PAS.

The SSTK, described in the following section, is able to evaluate meaningful substructures for PAS trees. Moreover, moving from the observation that as a single PAS may not be sufficient for text representation, Section 3.3.2 proposes a kernel that combines the contributions of different PASs.

3.3.2 The Shallow Semantic Tree Kernel

The SSTK is based on two ideas: first, the PAS is changed, as shown in Figure 3.9 (top) by adding SLOT nodes. These accommodate argument labels in a specific order, i.e. a fixed number of slots is provided, possibly filled with *null* arguments, that encode all possible predicate arguments. For simplicity, the figure shows a structure of just 4 arguments, but more can be added to accommodate the maximum number of arguments a predicate can have. Leaf nodes are filled with the wildcard character $\star$ but they may alternatively accommodate additional information.

The SLOT nodes are used in such a way that the adopted TK function can generate fragments containing one or more children like for example those shown in Figure 3.9. As previously pointed out, if the arguments were directly attached to the root node, the

Algorithm 3 The Shallow Semantic Tree Kernel

0. if n_1 (or n_2) is a pre-terminal node and its child label is *null*, $\Delta(n_1, n_2) = 0$;
1. if the productions at n_1 and n_2 are different then $\Delta(n_1, n_2) = 0$;
2. if the productions at n_1 and n_2 are the same, and n_1 and n_2 only have leaf children (i.e. they are pre-terminal symbols) then $\Delta(n_1, n_2) = 1$;
3. if the productions at n_1 and n_2 are the same, and n_1 and n_2 are not pre-terminals then

$$\Delta(n_1, n_2) = \prod_{j=1}^{nc(n_1)} (1 + \Delta(c^j_{n_1}, c^j_{n_2})) - 1. \tag{3.3}$$

kernel function would only generate the structure with all children (or the structure with no children, i.e. empty).

The second key idea of the SSTK is that, as the original tree kernel would generate many matches with slots filled with the null label, a new step 0 is set:

0. if n_1 (or n_2) is a pre-terminal node and its child label is *null*, $\Delta(n_1, n_2) = 0$;

and, in step 3, one unit is subtracted from $\Delta(n_1, n_2)$:

3. $\Delta(n_1, n_2) = \prod_{j=1}^{nc(n_1)} (1 + \Delta(c^j_{n_1}, c^j_{n_2})) - 1.$

The above changes generate a new Δ which, when substituted (in place of the original Δ) in Equation 3.1, gives the Shallow Semantic Tree Kernel (illustrated in Algorithm 3).

A. Moschitti proposed the following to show that the SSTK is effective in counting the number of relations shared by two PASs (Moschitti *et al.*, 2007):

Proposition 1 *The new Δ function applied to the modified PAS counts the number of all possible k-ary relations derivable from a set of k arguments, i.e. $\sum_{i=1}^{k} \binom{k}{i}$ relations of arity from 1 to k (the predicate being considered as a special argument)*[3].

[3]For the interested reader, the proof is the following:

Proof A kernel applied to a tree and itself computes all its substructures, hence if we evaluate SSTK between a PAS and itself we must obtain the number of generated k-ary relations. The above claim can be proved by induction.

Base case ($k = 0$): a PAS with no arguments is used, i.e. all its slots are filled with null labels.

Let r be the PAS root; since r is not a pre-terminal, step 3 is selected and Δ is recursively applied to all of r's children, i.e. the slot nodes. To the latter, step 0 assigns $\Delta(c^j_r, c^j_r) = 0$. As a result,

$$\Delta(r, r) = \prod_{j=1}^{nc(r)} (1 + 0) - 1 = 0$$

TK functions can be applied to sentence parse trees, therefore their usefulness for text processing applications, e.g. question classification, is evident. On the other hand, the SSTK applied to one PAS extracted from a text fragment may not be meaningful since its representation needs to take into account all the PASs that it contains.

Such problem can be addressed by using a kernel defined on *multiple* PASs. Let P_t and $P_{t'}$ be the sets of PASs extracted from the text fragment t and t'. The $K_{\texttt{all}}$ kernel is defined as follows:

$$K_{\texttt{all}}(P_t, P_{t'}) = \sum_{p \in P_t} \sum_{p' \in P_{t'}} SSTK(p, p'). \tag{3.4}$$

During the experiments in Section 3.4 the $K_{\texttt{all}}$ kernel is used to handle predicate argument structures, while the TK kernel in Equation 3.1 is used to process parse trees and a linear kernel is used to handle POS and BOW features.

3.4 Experiments

The purpose of our experiments was to study the impact of the shallow semantic representations introduced earlier (i.e. PASs and PASNs) for QA tasks. We focused our attention on the two critical phases of question classification and answer re-ranking for Web-based QA systems.

In the question classification task, we extended previous studies, by testing a set of previously designed kernels available in the literature, e.g. Zhang & Lee (2003); Moschitti (2006) and their combinations with the new Shallow Semantic Tree Kernel (see

and the base case holds.

General case: r is the root of a PAS with $k+1$ arguments. Then:

$$\Delta(r,r) = \prod_{j=1}^{nc(r)} (1 + \Delta(c_r^j, c_r^j)) - 1 = \prod_{j=1}^{k}(1 + \Delta(c_r^j, c_r^j)) \times (1 + \Delta(c_r^{k+1}, c_r^{k+1})) - 1.$$

For k arguments, it can be assumed by induction that

$$\prod_{j=1}^{k}(1 + \Delta(c_r^j, c_r^j)) - 1 = \sum_{i=1}^{k} \binom{k}{i},$$

i.e. the number of k-ary relations. Moreover, $(1 + \Delta(c_r^{k+1}, c_r^{k+1})) = 2$, thus:

$$\Delta(r,r) = \sum_{i=1}^{k} \binom{k}{i} \times 2 = 2^k \times 2 = 2^{k+1} = \sum_{i=1}^{k+1} \binom{k+1}{i},$$

i.e. all the relations until arity $k+1$. □

Quarteroni *et al.*, 2007).

In the answer re-ranking task, we approached the problem of detecting description answers, among the most complex in the literature (Cui *et al.*, 2005; Kazawa *et al.*, 2001). We define description answers in the same way as Li & Roth (2002) as answers containing definitions, reasons or manners (e.g. "Why does the moon turn orange?"). As will be explained later, we experiment with answers to the description questions appearing in the test-set of the TREC-QA 2001 campaign.

In the course of our experiments, we adopted the following data representations:

BOW: bag-of-words,

POS: bag-of-POS tags,

PT: parse tree,

PAS: predicate argument structure,

PASN: nested predicate argument structure.

As mentioned earlier, BOW and POS are processed by means of a linear kernel, PT is processed with TK, PAS and PASN are processed by SSTK.

Moreover, various combinations of the above kernels were tested, by summing the individual models, exploiting the property that such operation always produces a valid kernel (Shawe-Taylor & Cristianini, 2004).

The above kernels were implemented in the SVM-light-TK software[4], which encodes tree kernel functions in SVM-light (Joachims, 1999).

3.4.1 Question Classification

As a first experiment, we focused on question classification, for which benchmarks and baseline results are available (Zhang & Lee, 2003; Li & Roth, 2005).

As defined in Section 2.3.1, question classification is a multi-classification problem which consists in assigning an instance I to one of n classes, which generally belong to two types: factoid, seeking short fact-based answers (e.g. name, date) or non-factoid, seeking e.g. descriptions or definitions (see e.g. the UIUC taxonomy (Li & Roth, 2005)).

We designed a question multi-classifier by combining n binary SVMs according to the ONE-vs-ALL scheme, where the final output class is the one associated with the most probable prediction.

[4]available at `ai-nlp.info.uniroma2.it/moschitti/`

While the PTs were derived by using the Charniak parser, the PASs were automatically derived by a state-of-the-art SRL system which achieves a 76% F1-measure (Moschitti *et al.*, 2005).

The SVM-light software allowed us great flexibility and advanced features in order to configure our classifiers; among these, we took advantage of what is henceforth called "cost-factor" parameter, which allows to adjust the rate between Precision and Recall of the classifier during learning based on the development set. Intuitively, by allowing to vary the importance of precision with respect to recall, the cost-factor parameter allows the classifier to privilege positive examples over negative ones: a cost-factor of 2 implies that a correctly classified positive instance is twice as important as a correctly classified negative one.

Trying a few cost-factor values would enable us to check whether the differences in terms of F1-measure obtained for different kernel combinations (e.g. a model combining BOW and PT vs a model consisting of BOW only) would be preserved. Constant behavior of the F1 curves with respect to several cost-factor parameter values would strenghten the validity of our findings.

As benchmark data, we used the UIUC dataset; as introduced in Section 2.3.1, the dataset is manually partitioned according to the coarse-grained question taxonomy defined in Li & Roth (2002) – i.e. ABBR, DESC, NUM, HUM, ENTY and LOC. Moreover, a manual split of the dataset is available at: `http://l2r.cs.uiuc.edu/~cogcomp/Data/QA/QC/`, contains 5,500 training and 500 test instances; the test set is composed of the 500 TREC 2001 test questions (Voorhees, 2001). We refer to this manual split as "UIUC split" throughout this section.

The performance of the multi-classifier and the individual binary classifiers was measured respectively with accuracy and F1-measure. To collect statistically significant information, we ran 10-fold cross validation on the 6,000 questions.

Results

Table 3.1 shows the accuracy of different question representations on the UIUC split (Column 1) and the average accuracy $\pm$ the corresponding confidence limit (at 90% significance) on the cross validation splits (Column 2). Table 3.2 shows the accuracy of the individual binary classifiers for each question class[5].

The analysis of such experimental data suggests the following observations.

[5] These values are the same as those reported in Table 2.5

Learning Models	Accuracy	Accuracy (cross-val.)
PT	90.4	84.8±1.2
BOW	90.6	84.7±1.2
PAS	34.2	43.0±1.9
POS	26.4	32.4±2.1
PT+BOW	**91.8**	**86.1±1.1**
PT+BOW+POS	91.8	84.7±1.5
PAS+BOW	90.0	82.1±1.3
PAS+BOW+POS	88.8	81.0±1.5

Table 3.1: Accuracy of the question classifier with different feature combinations

UIUC split results Our first finding is that the STK on PT and the linear kernel on BOW produce a very high result, i.e. about 90.5%. This is higher than the best outcome derived in Zhang & Lee (2003), i.e. 90%, obtained with a kernel combining BOW and PT. When our BOW is combined with STK, it achieves an even higher result, i.e. 91.8%, very close to the 92.5% accuracy reached in Li & Roth (2005) by using complex semantic information derived manually from external resources.

Our higher results with respect to Zhang & Lee (2003) are explained by a highly performing BOW, the use of parameterization and most importantly the fact that our model is obtained by summing two separate kernel spaces (i.e. the linear kernel for the BOW feature and the tree kernel for the PT feature; both kernels are normalized separately), as mixing BOW with tree kernels does not allow SVMs to exploit all its representational power.

Secondly, model PT+BOW shows that syntactic information can be beneficial in tasks where text classification is vital, such as QA. Here, syntax can give a remarkable contribution in determining the class of a question; moreover, the lexical information (BOW) has a limited impact due to the little number of words forming a question..

Thirdly, the PAS feature does not provide improvement. This is mainly due to the fact that at least half of the training and test questions only contained the predicate "to be", for which a PAS cannot be derived by our PB-based shallow semantic parser. Also, PT probably covers most of the question's semantic information encoded by PAS.

Cross-validation results The 10-fold cross-validation experiments confirm the trends observed in the UIUC split: the best model is PT+BOW, which achieves an average accuracy of 86.1%. This value is lower than the one recorded for the UIUC split: the explanation is that the UIUC test set, which contains the TREC 2001 questions, is not consistent with the training set; indeed, it includes a larger percentage of easily classified question types, e.g. the numeric (22.6%) and description classes (27.6%) while their percentage in training is 16.4% and 16.2%, respectively.

This shows the importance of cross-validation results which, given the very low values the standard deviation, also suggest that the superior accuracy of the PT+BOW over the BOW model is statistically significant.

Individual binary classification Finally, for individual binary classification, the most accurate is the one carried out for NUM, which generally exhibits easily identified cues such as "how much/many". The more generic ENTY type proves hardest in both the UIUC and cross-validation experiments, while LOC and HUM remain well-classified in both cases also thanks to their regular patterns ("where" and "who" identifiers).

A difference in the UIUC and cross-validation experiments can be noticed in the DESC class, where clearly the F1 in cross validation is lower because of the less favorable splits. ABBR, the second most poorly classified type in both experiments, exhibits a high standard deviation in cross validation as there are only 95 total instances in the whole data-set, leaving little significance to the classification results.

On its own, the POS feature did not prove very effective in the task, and this was reflected in the combined feature experiments (runs PT+BOW+POS and PAS+BOW+POS), showing that the information provided by the POS tags was subsumed by the other features.

Question Class	P (UIUC)	R (UIUC)	F1 (UIUC)	F1 (cross-val.)
ABBR	87.5	77.8	82.4	78.5± 7.0
DESC	95.8	99.3	97.5	84.6±2.3
ENTY	73.6	83.0	78.0	75.7±1.3
HUM	89.6	92.3	90.9	86.8±2.0
LOC	86.6	85.2	85.7	88.9±1.5
NUM	99.0	86.7	92.5	94.2±1.4
Multi-Classifier Accuracy			91.8	86.1±1.3

Table 3.2: Performance of the best SVM classifier by question class (± standard deviation).

3.4.2 Answer Classification

Question classification does not allow to fully exploit the PAS potential since questions tend to be short and with few verbal predicates (i.e. the only ones that the SRL system we are using can extract). A different scenario is answer classification, i.e. deciding if a passage/sentence correctly answers a question. Here, the semantics to be generated by the classifier are not constrained to a small taxonomy and answer length may make the PT-based representation too sparse.

In Moschitti *et al.* (2007), we learned answer classification with a binary SVM that determines if an answer is correct for the target question, i.e. the question it is supposed to answer. Hence, in this experiment the classification instances are $\langle$*question, answer*$\rangle$ pairs with the constraing that in each pair, $answer$ must be a sentence identified as a candidate answer for $question$.

Each pair component can be encoded with PT, BOW, PAS and PASN representations (processed by the previously discussed kernels). While TREC questions generally contain at most one predicate, the opposite holds for answers; we could therefore experiment with PASN as well.

As test data, we collected the 138 TREC 2001 test questions labelled as "description" (i.e. "DESC") according to the UIUC taxonomy and for each, we obtained a list of answer paragraphs extracted from Web documents using YourQA. The number of answers we obtained for each question varied depending on the number of answers extracted by YourQA from the top Web documents returned by Google. Each paragraph sentence was manually evaluated based on whether it contained an answer to the corresponding question. Moreover, to simplify the classification problem, we isolated for each paragraph the sentence which obtained the maximal judgment according to the human annotator (in case more than one sentence in the paragraph had the same judgment, we chose the first one).

We collected a corpus containing 1309 sentences, 416 of which answered the question either concisely or with noise; the 416 pairs formed by these answers and their corresponding questions were labelled as positive instances ("+1"). The rest, containing sentences that were either irrelevant to their corresponding question or contained hints relating to the question but could not be judged as valid answers, were labelled as negative instances ("-1").

For instance, given the question "*What are invertebrates?*", the sentence "*At least 99% of all animal species are invertebrates, comprising ...*" yielded a pair labelled "-1", while "*Invertebrates are animals without backbones.*" yielded a pair labelled "+1".

Results

To test the impact of our models on answer classification, we ran 5-fold cross-validation, with the constraint that two pairs $\langle q, a_1 \rangle$ and $\langle q, a_2 \rangle$ associated with the same question q

could not be split between training and testing. Hence, each reported value is the average over 5 different outcomes[6]. The experiments were organized as follows.

Impact of BOW and PT First, we examined the contributions of BOW and PT representations as they proved very important for question classification. Figure 3.10 reports the plot of the F1-measure of answer classifiers trained with all combinations of the above models according to different values of the cost-factor parameter, adjusting the rate between Precision and Recall. We see here that the most accurate classifiers are the ones using both the answer's BOW and PT feature and either the question's PT or BOW feature, i.e. Q(BOW) + A(PT, BOW) resp. Q(PT) + A(PT, BOW) combinations.

When PT is used for the answer the simple BOW model is outperformed by 2 to 3 points. Hence, we infer that both the answer's PT and BOW features are very useful in the classification task. However, PT does not seem to provide additional information to BOW when used for question representation. This can be explained by considering that answer classification (restricted to description questions) does not require question type classification since its main purpose is to detect question/answer relations. In this scenario, the question's syntactic structure does not seem to provide much more information than BOW.

Impact of PAS and PASN Secondly, we evaluated the impact of the newly defined PAS and PASN features combined with the best performing previous model, i.e. Q(BOW) + A(PT, BOW). Figure 3.11 illustrates the F1-measure plots again according to the cost-factor parameter. We observe here that model Q(BOW) + A(PT, BOW, PAS) greatly outperforms model Q(BOW) + A(PT, BOW), proving that the PAS feature is very useful for answer classification, i.e. the improvement is about 2 to 3 points while the difference with the BOW model, i.e. Q(BOW) + A(BOW), exceeds 3 points.

The Q(BOW) + A(PT, BOW, PASN) model is not more effective than Q(BOW) + A(PT, BOW, PAS). This suggests either that PAS is more effective than PASN or that when the PT information is added, the PASN contribution fades out.

To further investigate the previous issue, we finally compared the contribution of the PAS and PASN when combined with the question's BOW feature alone, i.e. no PT is used. The results, reported in Figure 3.12, show that this time PASN performs better than PAS. This suggests that the dependencies between the nested PASs are in some way captured by the PT information. Indeed, it should be noted that we join predicates only in case one is subordinate to the other, thus considering only a restricted set of all possible

[6]The standard deviations ranged approximately between 2.5 and 5.

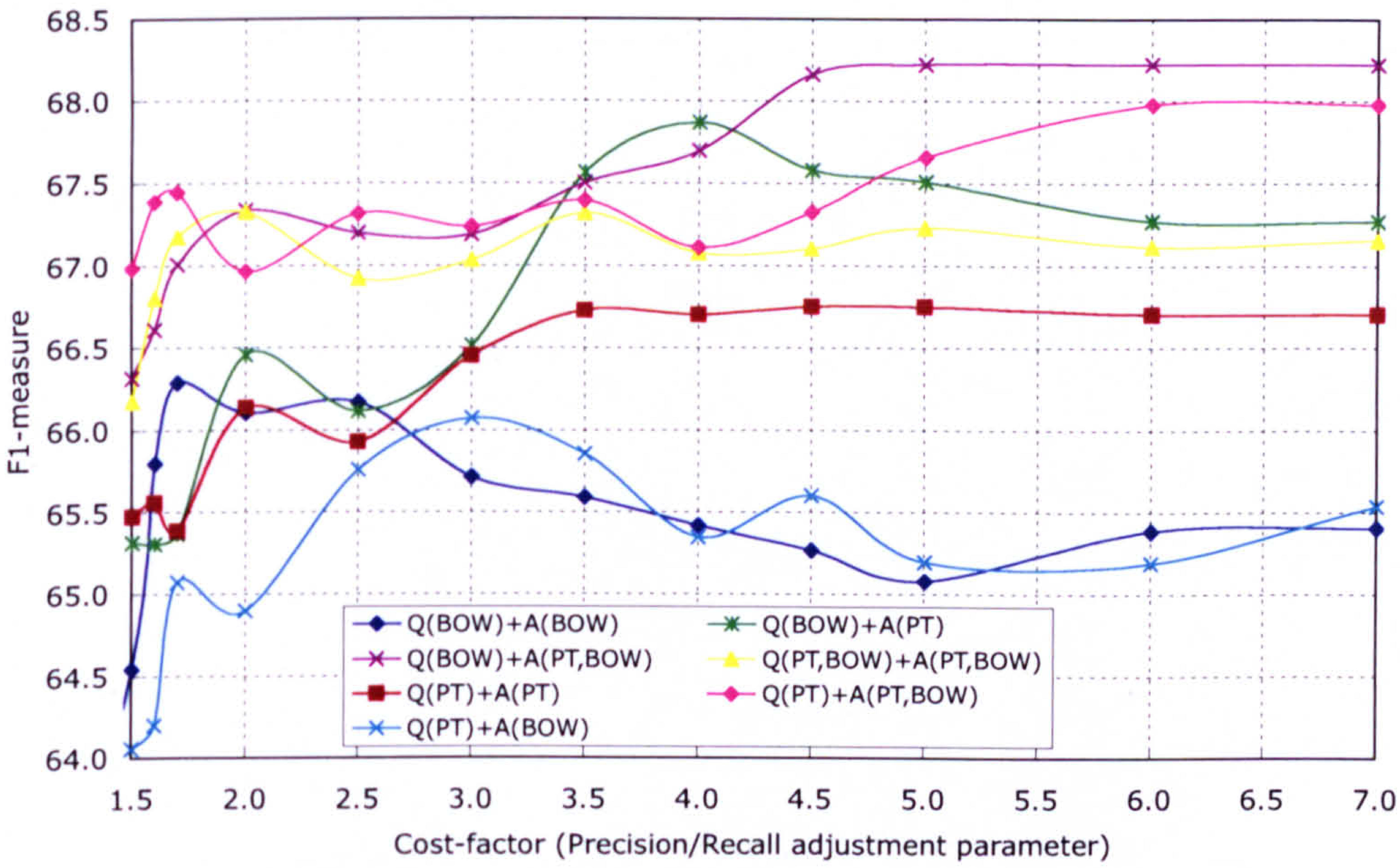

Figure 3.10: Impact of the BOW and PT features on answer classification

predicate dependencies. However, the improvement over PAS confirms that PASN is the right direction to encode shallow semantics from different sentence predicates.

3.4.3 Answer Re-ranking

The output of the answer classifier can be used to re-rank the list of candidate answers of a QA system, following Algorithm 4. The algorithm starts from the top answer in the list, and evaluates its correctness with respect to the question. If the answer is classified as correct its rank is unchanged; otherwise it is pushed down in the list, until a lower ranked incorrect answer is found.

We used the answer classifier with the highest F1-measure on the development set according to different cost-factor values[7]. We applied such model to the Google ranks and to the ranks of YourQA.

Table 3.3 illustrates the results of the answer classifiers derived by exploiting Google and YourQA ranks: the top N ranked results returned by Google resp. YourQA are considered as correct definitions and the remaining ones as incorrect for different values of

[7]However, by observing the curves in Fig. 3.11, the selected parameters appear as pessimistic estimates for the best model improvement: the one for BOW is the absolute maximum, but an average one is selected for the best model.

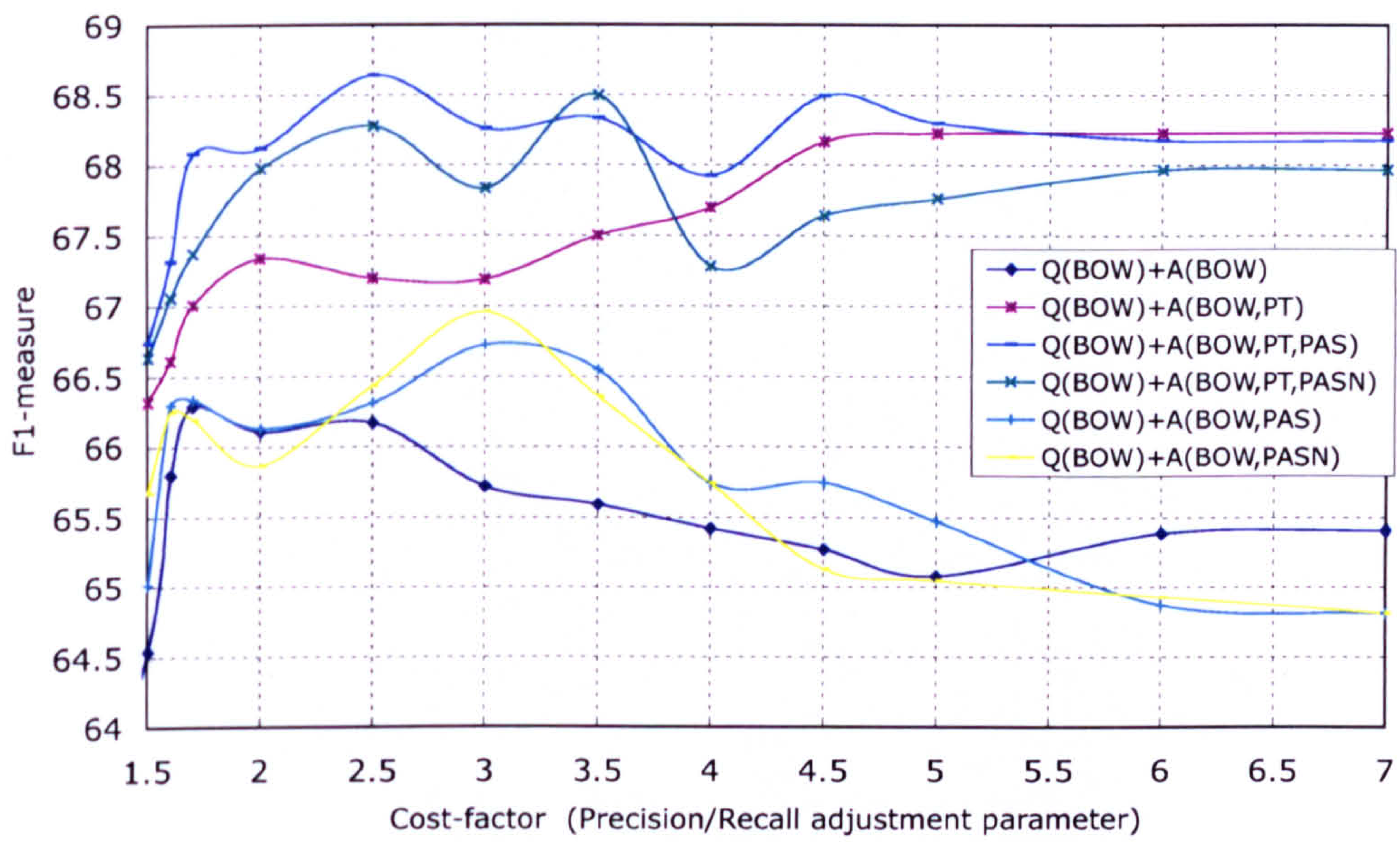

Figure 3.11: Impact of the PAS and PASN features combined with the BOW and PT features on answer classification

N. The correctness of such results is measured according to the "gold standard" consisting in the human annotator's judgment. We show N = 5 and the maximum N (all), i.e. all the available answers. Each measure is the average of the Precision, Recall and F1-measure from cross validation. These results show that F1-measure of Google and YourQA are greatly outperformed by the answer classifier when it comes to detecting definition answers.

To conclude, we implemented the simple re-ranking algorithm described previously and assessed its performance with the MRR metric[8] adopted in TREC 2001. Indeed,

[8]The Mean Reciprocal Rank is defined as: $MRR = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{rank_i}$, where n is the number of questions

Algorithm 4 Answer re-ranking algorithm

1. Start from the top answer in the list;
2. If the current answer is classified as a correct definition, leave it unchanged;
3. Else, shift it down in the ranking until a lower answer ranked as incorrect is encountered;
4. Stop at the bottom of the list;

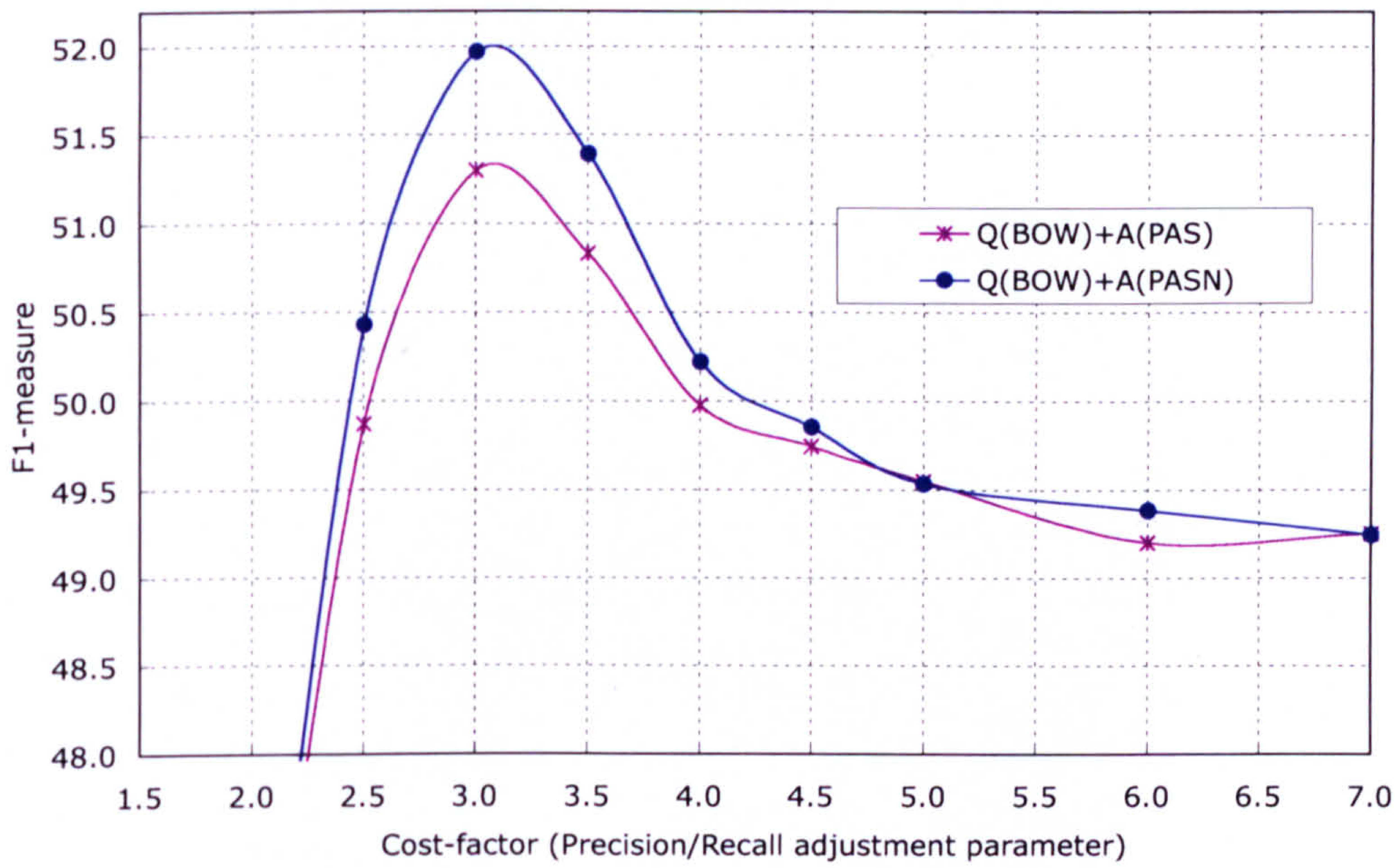

Figure 3.12: Comparison between PAS and PASN when used as standalone features for the answer on answer classification

Baseline	P	R	F1-measure
Google@5	39.22±3.59	33.15±4.22	35.92±3.95
YourQA@5	39.72±3.44	34.22±3.63	36.76±3.56
Google@all	31.58±0.58	100	48.02±0.67
YourQA@all	31.58±0.58	100	48.02±0.67

Table 3.3: Baseline classification accuracy of YourQA and Google

although since the TREC 2003 definition track (Voorhees, 2003) answers were expected in the form of bags of information "nuggets", we still believe it is meaningful to return definitions in the form of single-snippets – and consequently evaluating them according to MRR, as discussed in Chapter 1.

Table 3.4 reports the MRR achieved by Google, YourQA alone and YourQA after re-ranking (Re-ranker). We note that Google is outperformed by YourQA since its ranks are based on whole documents, not on single passages. Thus Google may rank a document containing several sparsely distributed question words higher than documents with several words concentrated in one passage, which are more interesting. When the answer

and $rank_i$ is the rank of the first correct answer to question i (i.e. labelled as "+1" in the human annotation)

	Google	YourQA	Re-ranker
MRR	48.97±3.77	56.21±3.18	81.12±2.12

Table 3.4: MRR of YourQA, Google and the best re-ranker

classifier is applied to improve the YourQA ranking, the MRR reaches 81.1%, rising by about 25%.

Finally, it is worth to note that the answer classifier based on the model Q(BOW) + A(BOW, PT, PAS) (and parameterized as described) gave a 4% higher MRR than the one based on the simple BOW features. As an example of such improvement, for question "What is foreclosure?", the sentence: "Foreclosure means that the lender takes possession of your home and sells it in order to get its money back." was correctly classified by the best model, while BOW failed.

3.4.4 Related Work

Unfortunately, there do not seem to be any results concerning a Web-based answer classifier for the same question set and few are available on the performance computed over description questions alone on the NIST corpus; for instance, NTT's system achieved an MRR of .247 on description questions using a heuristic searching for appositives (Kazawa *et al.*, 2001).

Interesting related work on definition answer re-ranking (Xu *et al.*, 2005) was conducted by comparing SVM classifier predictions to induce a ranking with the Ranking SVM algorithm (Joachims, 1999). The study was conducted at both passage and sentence level and by using both TREC data and a company's intranet data. This is indeed a research direction that we intend to explore in future work as our re-ranking algorithm is now based on the output of our binary SVM answer classifier.

In Chen *et al.* (2006), answer ranks were computed based on the probabilities of bigram language models generating candidate answers. This approach achieved an F_5 of .531 on the 50 TREC 2003 definition questions. Language modelling was also applied to definitional QA in Cui *et al.* (2005) to learn soft pattern models based on bigrams. Our approach is different from the above in that we attempt to capture structural information, and this has proven to be very effective in our experiments, yielding a very high MRR.

3.5 Conclusion

In this chapter, we have introduced new structures to represent textual information in three question answering tasks: question classification, answer classification and answer re-ranking. We have described tree structures (PAS and PASN) to represent predicate-argument relations, which we automatically extract using the SRL system in (Moschitti *et al.*, 2005). We have also introduced two functions, $SSTK$ and $K_{\texttt{all}}$, to exploit their representative power.

Our experiments with SVMs and the above models suggest that syntactic information helps tasks such as question classification whereas semantic information contained in PAS and PASN gives promising results in answer classification.

In the course of future work, we aim to study ways to capture relations between predicates so that more general semantics can be encoded by PASN. Forms of generalization for predicates and arguments within PASNs like LSA clusters, WordNet synsets and FrameNet (roles and frames) information also appear as a promising research area.

Chapter 4

Personalized Question Answering

A common problem in Question Answering and information retrieval (especially when Web-based) is information overload, i.e. the presence of an excessive amount of data from which to search for relevant information. This results in the risk of high recall but low precision of the information returned to the user, as underlined in Belkin & Croft (1992).

In the open domain in particular, this problem affects the relevance of results with respect to the users' needs, as queries can be ambiguous and even answers extracted from documents with relevant content may be ill-received by users if such documents are formulated in a language unsuitable to them.

While the need for personalization has been addressed by the information retrieval community for a long time (Belkin & Croft, 1992; Kobsa, 2001), very little effort has been carried out up to now in the Question Answering community in this direction. Indeed, personalized Question Answering has been advocated in TREC-QA starting from 2003 (Voorhees, 2003):

> "Without any idea of who the questioner is and why he or she is asking the question it is essentially impossible for a system to decide what level of detail in a response is appropriate – presumably an elementary-school-aged child and a nuclear physicist should receive different answers for at least some questions.

However, the issue was solved rather expeditiously by designing a scenario where an "average news reader" (hence one particular user type) was imagined to submit the 2003 task's *definition* questions:

> [...] The questioner is an adult, a native speaker of English, and an "av-

erage" reader of US newspapers. In reading an article, the user has come across a term that they would like to find out more about.
They may have some basic idea of what the term means either from the context of the article (for example, a bandicoot must be a type of animal) or basic background knowledge (Ulysses S. Grant was a US president).
They are not experts in the domain of the target, and therefore are not seeking esoteric details (e.g., not a zoologist looking to distinguish the different species in genus Perameles)." (Voorhees, 2003).

It is clear that the problem of personalization in Question Answering is just postponed by such a solution; it remains the case that different users have different information needs and we believe that information retrieval systems – among which Question Answering systems – should at least provide personalization as an optional feature. We argue that personalization is a key issue to make Question Answering closer to the user's actual information requirements, and plays an important role among the current directions for improving Question Answering technology.

In this chapter, we present an adaptation of User Modelling (Kobsa, 2001) to the design of personalized Question Answering. User Modelling has up to now been mainly applied in the context of information retrieval (Teevan *et al.*, 2005), intelligent tutoring (Virvou & Moundridou, 2001) or cultural heritage (Stock & AlFresco Project Team, 1993). We show how a model of the user's reading abilities and personal interests can also be used to efficiently improve the quality of the information returned by a Question Answering system.

Structure of This Chapter

This chapter is structured as follows. Section 4.1 introduces the design of a personalized Question Answering system characterized by a User Modelling component.

Section 4.2 briefly discusses previous approaches to User Modelling to contextualize the approach taken in YourQA. Section 4.3 outlines the attributes of the YourQA User Model, designed for open-domain, Web based Question Answering, which accounts for the user's reading level and interests.

Section 4.4 shows how the reading level of Web pages obtained during the document retrieval phase can be estimated and used to filter out unsuitable documents from the final answers. Moreover, it illustrates how the user's interests can be matched to candidate answer documents during answer extraction to achieve personalization.

Section 4.5 describes the implementation of the personalized version of YourQA, performing the tasks discussed in the previous sections. Section 4.6 introduces a methodology for user-centered evaluation of personalized Question Answering and reports the positive results of such methodology.

Finally, Section 4.7 concludes on the application of User Modelling to Question Answering and discusses further work.

4.1 High Level Architecture of a Personalized Question Answering System

The salient feature of the personalized version of YourQA with respect to a traditional QA system such as the standard version of YourQA illustrated in Chapter 2, is the presence of a User Modelling component (as illustrated by the schema in Figure 4.1). The task of such component is to construct, maintain and update a User Model (UM), i.e. a representation of the user.

The concept of User Modelling as an approach to the representation of users has been introduced in (Kobsa, 2001) for generic applications. User Modelling has been applied in information retrieval to the tasks of personalized search (Teevan *et al.*, 2005) and item recommendation (Miller *et al.*, 2003), where the user's interests are represented and used in order to re-rank the results of his/her query.

The User Model in YourQA contains two types of information: on the one hand, an estimation of the user's age and reading level; on the other, a representation of his/her topics of interest.

As illustrated by the schema in Figure 4.1, the interaction of the User Model with the core Question Answering module happens in two phases: first, the UM provides criteria to filter out unsuitable documents for the user during the document retrieval phase (see Section 4.4.1). Secondly, the UM provides criteria to re-rank candidate answers based on profile relevance during answer extraction (see Section 4.4.2).

Before describing in detail the attributes of the User Model developed in YourQA and how they are created, applied and updated, we trace in Section 4.2 a brief history of previous work on User Modelling from its origins in the late 1970s to current approaches.

4.2 Previous Work on User Modelling

Seminal work in User Modelling is usually traced back to the works of Perrault *et al.* (1978) and Rich (1979a,b). Here, User Modelling was performed by the application system, and often no clear distinction could be made between system components that served

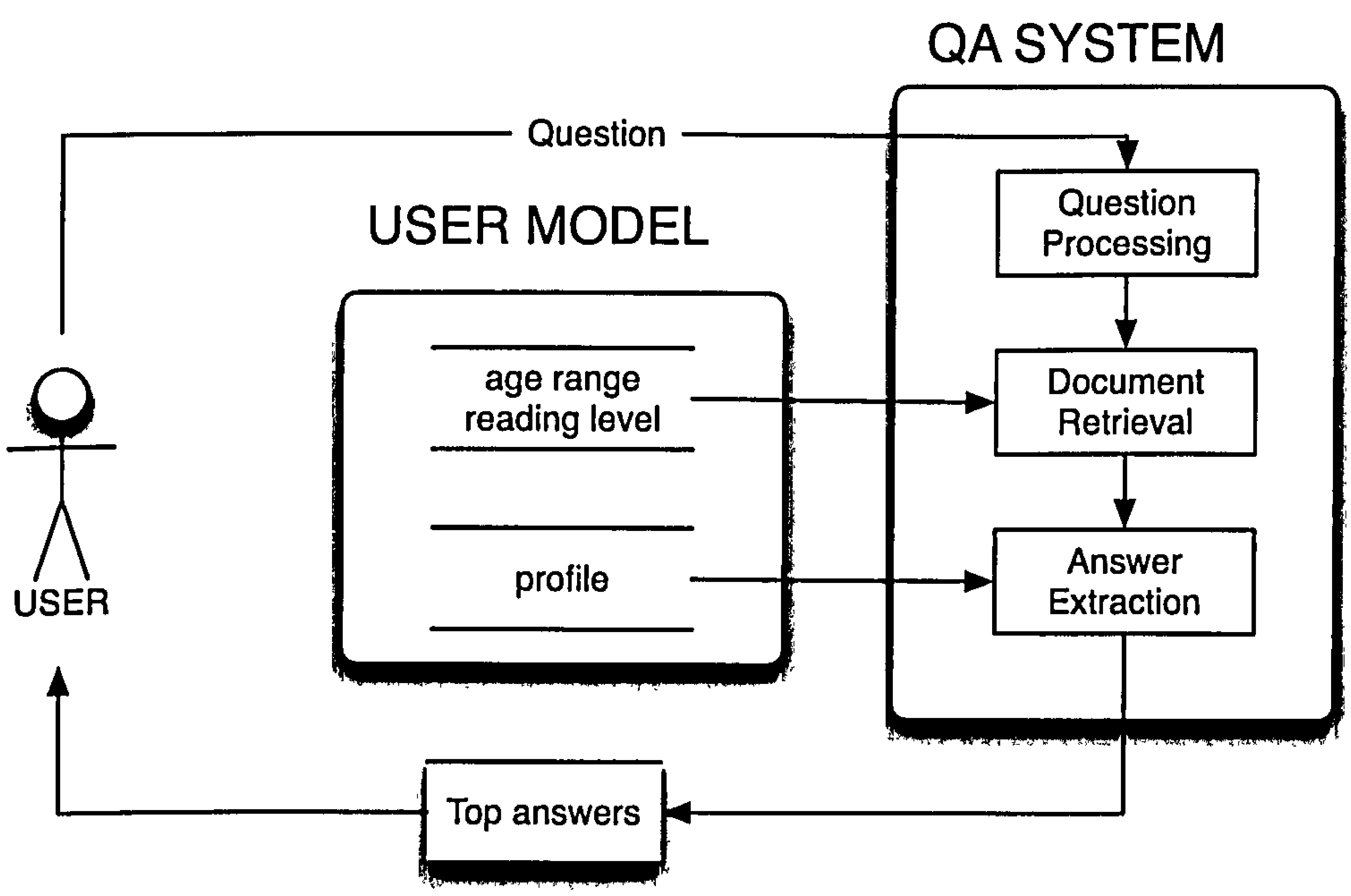

Figure 4.1: Personalized Question Answering Architecture

User Modelling purposes and components that performed other tasks. From the mid-eighties onwards, such a separation was increasingly made, but no efforts are reported on rendering the User Modelling component reusable for the development of future user-adaptive systems.

In 1986, Tim Finin published his "General User Modelling System" GUMS (Finin, 1986). This software allowed programmers of user-adaptive applications to define stereotypical users and hierarchies of stereotypes. For each stereotype, Prolog facts could be defined, describing stereotype members, while Prolog rules prescribed the system's reasoning about them. Although GUMS was never used together with an application system, it set the framework for the basic functionality of future 'general' (i.e., application-independent) User Modelling systems.

Early nineties: shell systems

Kobsa (1990) seems to be the first author who used the term "User Modelling shell systems", borrowed from the field of expert systems. This referred to the User Modelling aims of software decomposition and abstraction to support modifiability and reusability. The decisions as to what important structures and processes should go into User Modelling shell systems were mostly based on intuition and/or experience; the most important requirements for User Modelling shell systems were:

- *Generality*: shell systems were required to be usable in as many application and content domains as possible, and within these domains for as many User Modelling tasks as possible.
- *Expressiveness*: shell systems were expected to be able to express as many types of assumptions about the user as possible at the same time.
- *Strong Inferential Capabilities*: shell systems were expected to perform all sorts of reasoning that are traditionally distinguished in artificial intelligence and formal logic, such as reasoning in a first-order predicate logic, reasoning with uncertainty, and conflict resolution when contradictory assumptions are detected.

The reason for assigning such importance to these requirements reflects the affinity of User Modelling research of the early days to artificial intelligence, natural-language dialogue and intelligent tutoring. In the following years, different aspects of User Modelling were in focus, as described further.

4.2.1 Current Applications of User Modelling

Starting from the late 1990s, the value of Web personalization was increasingly recognized in the area of electronic commerce. Indeed, Web personalization allows product offerings, sales promotions, product news, ad banners, etc. to be targeted to each individual user taking the user's navigation data, purchase history and other previous interactions with the electronic merchant into account. From a more general perspective, personalization allows the relationship with customers on the Internet to migrate from anonymous mass marketing and sales to one-to-one marketing.

The major fields of application of contemporary User Modelling systems are heavily Web based. These include personalized IR, item recommenders, learning environments, natural language applications and cultural heritage. For each application field, we briefly mention some examples.

Personalized Information Retrieval

Personalized search systems and item recommenders are perhaps the fields of application that mostly resemble Question Answering. Several such systems exist which model users at different levels and for different kinds of personalization.

For instance, WBI (Barrett *et al.*, 1997) is a Web proxy that intercepts the HTTP stream for observation and alteration. Applications of WBI include personal history, page watching and recommendation of possibly useful links.

Another interesting application is Metiore (Bueno & David, 2001), a prototype for providing access to scientific publications in a laboratory. Metiore constructs an explicit individual User Model for representing the user's activities during information retrieval and then proposes him/her potentially interesting publications.

Finally, also summarization is an application benefitting from User Modelling: Alfonseca & Rodriguez (2003) propose a User Model that produces individual, ad hoc summaries of documents on the Web developed according to user's interests and available time (and reading efficiency).

Item Recommenders

UM is extremely important in the field of item recommenders, which aim to suggest the users to search for a particular product according to the user's own preferences or the preferences of the group/stereotype the user has been associated with. Among the types of items for which UM-based recommenders are used we can mention:

- News: SiteIF (Magnini & Strapparava, 2001) is a personal agent for a bilingual news Web site that learns user's interests from their requested pages. It also tries to anticipate what pages would be interesting for the user to read.

- Movies: in MOVIELENS (Miller *et al.*, 2003) predictions on possibly interesting movies are made based on explicit ratings provided by users, implicit ratings derived from navigational data, and implicit ratings derived from purchased products.

- Books: the COGITO project (Abbattista *et al.*, 2002) has a personalization component able to discriminate between interesting and uninteresting items for the user. The architecture of COGITO contains an "Item Recommender" module that uses supervised machine learning to induce a classifier based on book information extracted from Web pages and on individual user preferences over book categories. This information is used to suggest particular book-titles to the users.

Learning environments

One of the main applications of User Modelling takes place in learning environments: during the last ten years, there has been a significant interest towards Web-based Intelligent Tutoring Systems (ITS), Collaborative Learning Environments and in general the possibility of knowledge transmission in an ideal one-to-one relationship between the user/student and the provider/instructor (Linton *et al.*, 2003). Since the ideal aim is tailoring the information to the individual user – in this case, the student – it is essential to construct a good estimation of his/her characteristics, goals and needs.

An illustrative application of UM to learning is the AHA! system (Romero *et al.*, 2003), a Web-based Adaptive Educational Hypermedia System. AHA! can adapt a course to each specific student by building an individual User Model. Based on the UM, it can for instance adapt the content of a hypermedia page to the knowledge or current objective of the user, or suggest the most relevant links to follow.

Also in HAPPYneuron (Habieb-Mammar *et al.*, 2003), adaptive interfaces are created to suit the individual users' profiles and provide user-tailored courses. Another example is aLF (active Learning Framework) (Gaudioso & Boticario, 2002), a Web-based collaborative environment designed for Web learning communities, where the administrator can personalize the information received by each participant by adding information to the individual User Models.

In the Collaborative Learning Environment in Linton *et al.* (2003), a special module called student model module observes each learner and estimates his or her degree of understanding with respect to each topic.

Dialogue Systems

As mentioned earlier, dialogue systems are among the earliest applications of User Models. In particular, fields of application include task-oriented dialogue with systems such as TRAINS (Allen *et al.*, 2000) for route planning and VERBMOBIL (Alexandersson *et al.*, 1997) for automatic translation. Both applications build models of users and their plans and goals in order to be adaptive.

Another interesting approach to personalized dialogue is in the field of Embodied Conversational Agents (ECAs): Cavalluzzi *et al.* (2003) propose an ECA that interacts with the user to provide advice in a domain that is influenced by affective factors.

Finally, also command & control systems such as DeepListener (Horvitz & Paek, 2001) cope with uncertainty about user's utterances and goals with a User Model.

Cultural Heritage

User Modelling has also been widely applied in cultural heritage applications. A representative case is ALFRESCO (Stock & AlFresco Project Team, 1993), an interactive system built for accessing images and information about Italian 14th century frescoes. The system aims not only at providing information, but also at promoting other masterpieces that may attract the user. ALFRESCO is not simply an item recommender system: the system can show images, give some punctual answer to a question by replying with instances such as the title and location of a painting, or dates etc., or give a more complex description of a fresco or some other entities, through natural language.

4.3 User Model Design

As User Models are inherently tied to the application for which they are designed, it does not make sense to design a User Model independently of a concrete application task. In Section 4.3.1, we outline some approaches to User Model construction in fields related to Question Answering that inspired the approach taken in YourQA.

4.3.1 Building a User Model

For the construction of User Models, a traditionally used technique has been stereotyping, a means of providing quick although not always precise assessment on the user. Stereotyping consists in associating the user to a group and then attaching the properties of the group to the user. For instance, in KNOME Chin (1994), the User Modelling component of the Unix Consultant, the UM employs a double-stereotype system in which one set of stereotypes represents the user's expertise and another represents the difficulty level of the information.

In current UM applications, there are several other channels through which to collect useful information to construct a User Model. We categorize them below according to the applications for which they are designed, as this influences greatly what aspects of a user must be modelled and with what relevance.

Learning Environments

The ideal aim in learning environments is tailoring the information to suit the learner's characteristics, goals and needs. Some of the source types used in actual systems include:

- explicit information: the student can provide information on himself and his own estimated level, for instance by means of an access questionnaire.
- the *stereotype* approach (i.e. classification of users into a predefined set of users with analogous features) can be used by analyzing the type of course, year and background of the student (as in Virvou & Moundridou (2001)). This enables to obtain an estimation of the user's level of knowledge with respect of a particular subject.
- the *overlay* model: in this case, students' knowledge is seen as a subset of the system's knowledge; the User Model is therefore built on the basis of the latter (Virvou & Moundridou, 2001).
- previous usage: usage information can be collected to build a model of the user's knowledge level, learning speed, lacks and preferences. This can be useful information for the teacher in order to modify the content of adaptive online courses and thus to improve their performance.

  For instance, the AHA! system (Romero *et al.*, 2003) stores usage information in three Web log files which are converted in three tables of a database: 1) Times: contains information about the XML pages (content, question, etc.) and the exact time in which the student has accessed to them. 2) Levels: contains information about the knowledge level (high, medium, low) that the student has in each concept. 3) Success: contains information about the success or failure of the students in the questions (tests or activities).
- *cognitive styles*: in Habieb-Mammar *et al.* (2003), students are first given a series of training exercises evaluating the way they approach problems; based on these,

rules are applied in order to present courses using the most suitable media.

This procedure is called supervised cognitive training: a Web site provides exercises that train and evaluate cognitive abilities. Normalized data is stored into a database for each variant of exercise and family of population distinguishing gender, educational level and age. Comparing the trainee's results and normalized data, the user's cognitive profile is progressively built; this enables the system to advise the user in his choice of exercises.

Natural language applications

In natural language applications, and dialogue applications in particular, several types of indicators are used to collect evidence to build a UM:

- explicit/implicit natural language input: information can be gathered from explicit utterances of the users but also implicitly inferred by keywords in the text they type.
- dialogue history: analyzing past interactions can lead to the construction of a persistent User Model which can be continuously enriched and updated.
- collaborative approaches: in STyLE-OLM (Dimitrova, 2003), the User Model is built via a collaborative interaction between user and computer diagnoser during a dialogue game. The user's beliefs and some possible explanations of misconceptions by the user are collected during the dialogue and become part of the model. A similar approach was previously taken by KNOME (Chin, 1994), the User Modelling component of UC, a natural language consultation system for the UNIX operating system. During the course of an interactive session between a dialogue agent and the a user, KNOME inferred the user's level of expertise from the dialogue and maintained a model of the user's knowledge of the UNIX domain.

Web applications

Web applications, whether directed to personalizing the layout of specific pages or to recommend specific items (e.g. in e-commerce applications), make use of the following elements to build User Models:

- personal data: users are often encouraged to build their own profile, providing personal data such as age, profession, interests (Magnini & Strapparava, 2001). This information can be used to classify the users according to different stereotypes;

- content of requested Web pages: based on Web page contents, similar pages can be retrieved in the future, and the topics contained can give precious details on the interests of users (Teevan *et al.*, 2005).

- browsing behavior: clicks, pointing, eye gaze, and other non-textual information can be collected and give insights on the user's level of confidence with a particular topic, and in general on the browsing style and speed (Shen *et al.*, 2005).

- history: the whole recording of interactions can be mined to extract various types of information and to refine the UM.

- available time: not only the user's interests, but also the amount of time and the reading efficiency when browsing are used for instance to present different information and different summaries to different users (Alfonseca & Rodriguez, 2003).

While some of the approaches to User Model construction are difficult to apply to the field of open-domain Question Answering – in particular those that are too related to the type of application for which they are designed (e.g. beliefs or success in performing a task)– others are more generic (personal data, past interactions with the system, explicit feedback from users).

The following section explains the attributes of YourQA's User Model as well as how the User Modelling component constructs and updates it.

4.3.2 The User Model in YourQA

As a target domain which would be generic enough to be a proof-of-concept of the utility of personalized Question Answering and at the same time a concrete, task-oriented application of User Modelling, we chose the education domain. The User Model in YourQA represents students searching for information on the Web for their assignments.

Two basic aspects compose the user representation in such model: on the one hand, the user's interests in terms of answer contents; on the other, the user's preferences in terms of answer presentation.

These are modelled using three attributes:

- Age range, $a \in \{7-10, 11-16, adult\}$; the first two ranges correspond to the primary and secondary school age in Britain, respectively;

- Reading level, $r \in \{basic, medium, advanced\}$;

- Profile, p, a set of textual documents, bookmarks and Web pages of interest.

Analogous UM components can be found in the SeAn (Ardissono *et al.*, 2001) and SiteIF (Magnini & Strapparava, 2001) news recommender systems, where information such as age and browsing history, respectively are part of the User Model. Komatani *et al.* (2003) model users of a closed-domain, information-seeking dialogue system according to three dimensions, which include their skill and knowledge level of the system and topic and their degree of hastiness.

More generally, our approach is similar to that of personalized search systems such as Teevan *et al.* (2005), which constructs User Models based on the user's documents and Web pages of interest.

Although the reading level can be modelled separately from the age range, for simplicity we here assume that these are paired. The reading level attribute is used to modify the presentation aspect of the QA system; during the document retrieval phase, an estimation of the suitability of documents to the user's age and reading abilities is used to filter out undesired documents, as explained in Section 4.4.1.

The profile attribute is in turn applied during answer extraction in order to select answers from documents having topics in common with the topics extracted from the set of documents in the user profile, as explained in Section 4.4.2.

The first issue in any personalization application is how to create an efficient model of a previously unknown user (Lashkari *et al.*, 1994). Several applications, such as the SeAn news recommender (Ardissono *et al.*, 2001), construct an initial User Model based on a form filled in by the user, indicating e.g. his/her age.

While approaches such as stereotyping (Chin, 1994) are suitable for applications with a specific domain, they are not appropriate for the open domain. Recently, personalized information retrieval applications have been designed which extract information unobtrusively from documents on the user's desktop (Teevan *et al.*, 2005) or from their browsing habits Pitkow *et al.* (2002).

In our implementation, which is intended to be a proof-of-concept rather than a final model of personalization, we opt for a compromise solution, where information is elicited and obtained from the users in very little time at the moment of creation of a query. As described in Section 4.5, users are invited to specify the desired reading level of their results and a small number of documents of interest.

However, this need not be the case as the techniques explained in the two following sections can be used to estimate both the reading level and the interests of the user in a virtually unobtrusive way.

4.3.3 Reading Level Component

The reading level component of the User Model can be estimated on the basis of the textual documents (including Web pages) owned or written by the user. The process of reading level estimation of a document is described below.

Reading Level Estimation

Among the most widely used approaches to reading level estimation are models based on sentence length, such as "Flesch-Kincaid" (Kincaid *et al.*, 1975), Fry (Fry, 1969) or SMOG (McLaughlin, 1969). The key idea behind these approaches is that the the readability of text is inversely proportional to its length, hence such approaches assess readability using variations of sentence length-based metrics.

However it can be noticed that in Web documents, sentences are generally short and more concise than in printed documents, regardless of the complexity of the text. Hence the discriminative power of the above metrics can be affected by the fact that the difference in length between complex documents and simple ones is often not as wide as for the printed text.

As opposed to the previous approaches, the language modelling approach which has been adopted in YourQA and is illustrated below accounts especially for lexical information.

The language modelling technique has been proved in Collins-Thompson & Callan (2004) to be at least as effective as the Flesch-Kincaid approach when modelling the reading level of subjects in primary and secondary school age.

One remark about the above approach to textual readability is that the latter is not modelled at a conceptual level: thus, complex concepts explained in simple words might be classified as suitable even for a basic reading level.

While this aspect may appear as a weakness of the current approach, we must point out that from an initial analysis, we have observed that in most Web documents, lexical, syntactic and conceptual complexity are usually consistent within documents. Hence, we argue that it makes sense to apply a reasoning-free technique without impairing readability estimation.

We model reading level estimation as a multi-classification task which consists in assigning a document d to one of k different classes, each of which represents one reading level. In order to represent the three different age ranges defined in the corresponding attribute of the User Model, we define the three following classes:

1. *basic*, representing a document suitable for ages $7 - 11$;

2. *medium*, representing a document suitable for ages 11 – 16;

3. *advanced*, representing a suitable for adults.

We then approach reading level estimation as a supervised learning task, where representative documents for each of the three classes are collected as labelled training instances and used to classify previously unseen documents according to their reading levels.

Our training instances consist of about 180 HTML documents, which originate from a collection of Web portals where pages are explicitly annotated by the publishers according to the three reading levels above. The three Web document sets representing the 7–11, 11–16 and adult age ranges contain 33,154, 33,407 and 35,024 words respectively. Examples of such Web portals include:

1. BBC education (`http://bbc.co.uk/schools`),

2. Think Energy (`http://www.think-energy.com`),

3. Cassini Huygens resource for schools (`http://www.pparc.ac.uk/Ed/ch/Home.htm`),

4. Magic Keys storybooks (`http://www.magickeys.com/books/`),

5. NASA for kids (`http://kids.msfc.nasa.gov`).

While the first three provide contents suitable for all three age ranges, the fourth one is especially useful for the 7 – 11 age range and the last one for the 7 – 11 and 11–16 age ranges.

The readability judgments of the Web portals are our gold standard for learning reading level classification; the fact that our training instances are labelled by an external and trusted source contribute to the objectivity and soundness of our approach.

As a learning model, we use the Smoothed Unigram Model, which is a variation of a Multinomial Bayes classifier (Collins-Thompson & Callan, 2004) based on the representation of the data known as unigram language modelling.

Given a set of documents, a unigram language model represents such set of as the vector of all the words appearing in the component documents associated with their corresponding probabilities of occurrence within the set. For generality, and to account for data sparseness, we use word stems in place of the individual words, as obtained by applying the Porter Stemmer (Porter, 1980).

In the test phase of the learning process, given an unclassified document D, a unigram language model is built to represent the single document D (as done for the training documents). The estimated reading level of D is the language model lm_i maximizing the likelihood $L(lm_i|D)$ that D has been generated by lm_i (In our case, three language models lm_i are defined, where $i \in \{basic, medium, advanced\}$.) . Such likelihood is estimated using the function:

$$L(lm_i|D) = \sum_{w \in D} C(w, D) \cdot log[P(w|lm_i)] \tag{4.1}$$

Moreover, w is a word in the document, $C(w, d)$ 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). We evaluate and validate the language modelling approach to readability in the experiments reported in Section 4.6.1.

Related Work

Within computational linguistics, several applications have been designed to address the needs of users with low reading skills. The computational approach to textual adaptation is commonly based on *natural language generation*: the process "translates" a difficult text into a syntactically and lexically simpler version.

In the case of PSET (Carroll *et al.*, 1999) for instance, a tagger, a morphological analyzer/generator and a parser are used to reformulate newspaper text for users affected by aphasia.

Another example of research in this direction research is Inui *et al.* (2003)'s lexical and syntactical paraphrasing system for deaf students. In this system, the judgment of experts (teachers) is used to learn selection rules for paraphrases acquired using various methods (statistical, manual, etc.).

In the SKILLSUM project (Williams & Reiter, 2005), used to generate literacy test reports, a set of choices regarding output (cue phrases, ordering and punctuation) are taken by a micro-planner based on a set of rules.

The approach presented in this thesis is conceptually different from these: exploiting the wealth of information available by using the Web as a source, the QA system can afford to choose among the documents available on a given subject those which best suit the given readability requirements.

4.3.4 Profile Component

The user's interests are estimated based on the profile component of the User Model, which, as anticipated in Section 4.1, is defined as a set of both textual documents and Web pages of interest.

Information extraction from the user's documents as a means of representation of the user's interests, such as his/her desktop files, is a well-established technique for personalized information retrieval: Teevan *et al.* (2005) experiment with an index of various amounts of users' data to create a personalized search model, while in the Outride system (Pitkow *et al.*, 2002) a browser plugin accesses links and bookmarks building a model of the user's browsing preferences.

Our use of Web pages and bookmarks for estimating user's interests is also inspired by news recommender systems such as SiteIF (Magnini & Strapparava, 2001), a personal agent for a bilingual news Web site that learns user's interests from the Web pages they requested in the past. This approach has its origins in Web proxying based adaption, where the HTTP stream of interaction with a browser system is intercepted for observation and alteration, as in WBI (Barrett *et al.*, 1997).

Both the collected textual documents and Web documents form the profile document set called p, which is used to perform the estimation of the user's interests.

Profile Estimation

Profile estimation is based on key-phrase extraction, a technique which has been previously employed in several natural language tasks, including topic search, document clustering and summarization (Frank *et al.*, 1999; D'Avanzo *et al.*, 2004). Key-phrase extraction can be defined as a classification task where the aim is to extract the most important words or phrases to represent the semantics of a given text.

Unlike text categorization, where a fixed set of domain-specific key-phrases must be attached by the classifier to each instance, key-phrase extraction has the advantage that it does not require key-phrases to be known in advance. This makes such technique suitable for open-domain applications such as the one at hand.

While key-phrase extraction seems to be an innovative technique for User Modelling, there is evidence from previous work of the use of alternative content-based techniques for UM creation. For instance, in Magnini & Strapparava (2001), documents passed over are processed and relevant senses (disambiguated over WordNet (Miller, 1995)) are extracted and then combined to form a semantic network. A filtering procedure dynamically predicts new documents on the basis of the semantic network.

With respect to this approach, our approach is lighter in the sense that we do not need to access an external lexical database such as WordNet, with a variable (at times inexistent) coverage of different topics. Moreover, we are dealing with open-domain QA, that involves a greater number of senses.

As a key-phrase extractor, we use Kea (Witten *et al.*, 1999). One of the reasons for our choice lie in the fact that Kea is domain independent, a requirement in open-domain QA applications. Moreover, in a comparative study with other key-phrase extractors (Frank *et al.*, 1999), Kea has been shown to be very robust across different document sizes and domains. As our QA system deals with Web pages, which can have different lengths and structures, such robustness is a definite advantage.

Kea proceeds as follows: first, it splits each document into phrases and then takes short subsequences of these initial phrases as candidate key-phrases. Then, for each candidate phrase ϕ and each document D in a set of documents S, Kea uses two attributes to determine whether or not ϕ is a key-phrase with respect to D in S:

1. O, the index of ϕ's first occurrence in D;

2. T, the $TF \times IDF$ score[1] obtained by ϕ with respect to D in S.

T and O are assumed independent following Naïve Bayes; the probability that ϕ is a key-phrase for D in S is therefore:

$$P(key_D^\phi|T,O) = \frac{P(T|key_D^\phi) \cdot P(O|key_D^\phi) \cdot P(key_D^\phi)}{P(T,O)} \quad (4.2)$$

where:

- $P(T|key_D^\phi)$ is the probability that ϕ has $TF \times IDF$ score T, given that ϕ is a key-phrase within D;
- $P(O|key_D^\phi)$ is the probability that ϕ has offset O, given that ϕ is a key-phrase within D;
- $P(key_D^\phi)$ is the a priori probability that ϕ is a key-phrase within D;
- $P(T,O)$ is a normalization factor[2].

[1] $TF \times IDF$ score is a measure of salience of term contained in a document within a given collection. The $TF \times IDF$ of a term t in document D belonging to collection S is measured as follows:

$$TF \times IDF(t,D,S) = P(t \in D) \times -logP(t \in [S/D]).$$

[2] Currently $P(T,O) = 1$.

All these probabilities are estimated by counting the number of times the corresponding event occurs in the training data (Frank *et al.*, 1999).

Based on the probabilities computed using formula (4.2), Kea outputs for each document D in the set a ranked list where the candidate key-phrases are in decreasing order. The top k phrases are selected as document key-phrases for document D: after experimenting with several values, we fixed the k threshold to six.

The internal representation of a profile document set P representing an individual user in YourQA is in the form of a two-dimensional array of key-phrases, where each row corresponds to a profile document and each column is associated with the rank of the corresponding key-phrase in the list of key-phrases.

As an illustrative example, a basic user profile, created from two documents about Italian cuisine and the animation picture "Akira", respectively, might result in the array:

$$P = \begin{bmatrix} pizza & lasagne & tiramisu \\ akira & anime & katsuhiro\_otomo \end{bmatrix} \quad (4.3)$$

A further treatment on the outcome of key-phrase extraction is the stemming of key-phrases, which is carried on by using the Porter Stemmer (Porter, 1980).

The profile resulting from the extracted key-phrases is the base for all the subsequent QA activity: any question the user will submit to the Question Answering system is answered by taking such profile into account.

4.4 User Modelling for Personalized Question Answering

The interaction between the User Modelling component and the core Question Answering component modifies the standard Question Answering process presented in Chapter 2 at several stages, resulting in a new personalized QA algorithm (Algorithm 5). While question processing remains unchanged, the User Model affects the document retrieval phase and the answer extraction phase, the personalized versions of which are described below.

It may be worth highlighting that this model of personalization affects the results to all types of questions, regardless of their expected answer classes. Thus, both factoid and non-factoid questions can receive personalized answers according to the proposed algorithm.

The need to personalize answers to non-factoid questions may appear as the most in-

Algorithm 5 Personalized QA algorithm

- **Question Processing:**

  1. The query is classified and the two top expected answer types are estimated;
  2. The query is submitted to the underlying search engine;

- **Document Retrieval:**

  1. The top n documents are retrieved from the underlying search engine and split into sentences;
  2. The retrieved documents' reading levels are estimated;
  3. Documents having a different reading level from the user are discarded; if the remaining documents are insufficient, part of the incompatible documents having a close reading level are retained;
  4. From the documents remaining from reading level filtering, topics are extracted using Kea;
  5. The remaining documents are split into sentences;

- **Answer Extraction:**

  1. Document topics are matched with the topics in the User Model that represent the user's interests;
  2. Candidate answers are extracted from the documents and ordered by relevance to the query;
  3. As an additional answer relevance criterion, the degree of match between the candidate answer document topics and the user's topics of interest is used and a new ranking is computed on the initial list of candidate answers.

tuitively justified, for instance to account for ambiguous questions yielding answers about different domains. For instance, Google will respond to the question: "What is *Ginger and Fred*?" with documents relating to a film, a building and a dancing couple. Acronyms such as "UM" can refer to several different entities ("University of Michigan", "United Methodists", "User Modelling", etc.) and the query "Where is the UM conference this year?" would thus have several possible answers from which to choose.

However, personalization can also affect the factoid domain; as previously mentioned, it is quite intuitive that depending on the age and reading abilities of the reader, the answer to "When did the Middle Ages begin?", which is clearly a temporal (hence factoid) question, can be different. While a child might be content with the answer "The middle ages start in 476 AD with the fall of the Roman Empire", an adult interested in history might prefer an answer highlighting how it makes little sense to fix a unique starting date to the Middle Ages and that indeed there are several events that may be seen as the beginning of the medieval period.

4.4.1 Document Retrieval

In the standard QA algorithm, the document retrieval phase consisted in retrieving the top search engine documents and splitting them into sentences (see Section 2.4). When the User Modelling component is active, two additional retrieval steps take place: first, the documents' reading levels are estimated using the method described in Section 4.3.3; the documents having an incompatible reading level with respect to the User Model are discarded. Finally, the key-phrases for the remaining documents are extracted using Kea, as explained below.

Reading Level Filtering

The first step carried out during personalized document retrieval is the estimation of the reading level of each document returned by Google in response to the query. Such estimation is conducted via language modelling following the technique in Section 4.3.4. The documents having an incompatible reading level with the user are discarded so that only those having the same estimated reading level as the user are retained for further analysis.

As there can be queries for which the number of retrieved documents having the requested reading level is lower than the number of documents returned by the system (currently five), this condition is relaxed so that part of the documents having other reading levels are accepted in the set of candidate documents for answer extraction. In particular, if the user's reading level is *advanced*, medium reading level documents are considered and, in case the threshold number of documents is not met, basic documents complete the set. If the requested reading level is *medium*, documents having a basic readability are used to complete the set; finally, if the requested reading level is *basic*, medium documents are accepted in the set.

In all cases, due to the absence of other criteria at this stage of the QA algorithm, the choice of which documents to retain for a given reading level is determined by the search engine rank of the former (a higher rank determines preference).

The subsequent QA phase of answer extraction therefore begins with the documents from the reading level filtering phase.

Key-phrase Extraction

Once a working subset of the retrieved documents has been collected, key-phrases are extracted from the documents using the same approach as for the UM profile (see Section 4.5.1): Kea is applied over the set of retrieved documents to extract the top k key-phrases for each document. These are represented by the system as a two-dimensional array similar to the one created for the UM profile (see 4.3), which we call *Retr*.

As an illustrative example, we report part of the *Retr* array for the query: "What is *Ginger and Fred*[3]?" in (4.4). Notice that also in this case, key-phrases are stemmed using the Porter Stemmer (Porter, 1980). In the array, each row represents a retrieved document and each column represents a key-phrase rank; for instance, key-phrase *movi* located in cell $(1, 1)$ is the first ranked key-phrase extracted from a document about Fred Astaire and Ginger Roger's movies.

$$Retr = \begin{bmatrix} & & & \vdots & & \\ movi & item & fred\_astaire & film & ginger\_rogers & photo \\ build & resid & gehri & tower & project & gehry\_residence \\ fred & ginger & film & music & movi & review \\ film & fred & ginger & fellini & ginger\_and\_fred & dvd \\ gehri & build & citi & histor & destruct & ruin \\ & & & \vdots & & \end{bmatrix} \tag{4.4}$$

These modifications to the standard document retrieval phase allow the answer extraction phase to take advantage of the different parameters of the User Model. Section

[3]Notice that, as visible from the key-phrases, *Ginger and Fred* may refer to a famous dancing couple, the "dancing buildings" by architect F. Gehry, and a film directed by F. Fellini.

4.4.2 explains in detail how the UM profile is used to determine the final answer ranking.

4.4.2 Answer Extraction

As illustrated in Section 2.5, in the standard version of YourQA the primary passage ranking criterion is the similarity of the passage's central sentence with the question, and the secondary criterion is the Google rank of the document from which the passage has been extracted.

The personalized version of YourQA applies an additional ranking criterion giving priority to answers from documents having common key-phrases with the user's profile documents, as described below.

Relevance Computation

For each document composing the UM profile set and the retrieved document set, a ranked list of key-phrases is available from the previous steps. Both key-phrase sets are represented by the User Modelling component of YourQA as arrays, where each row corresponds to one document and each column corresponds to the rank within such document of the key-phrase in the corresponding cell.

The arrays of UM profile key-phrases and of retrieved document key-phrases are named P and $Retr$, respectively. We call $Retr_i$ the document represented in the i-th row in $Retr$ and P_n the one represented in the n-th row of P. Notice that, while column index reflects a ranking based on the relevance of a key-phrase to its source document, row index only depends on the name of such document (hence it does not determine a rank based on relevance to the question).

Given k_{ij}, i.e. the j-th key-phrase extracted from $Retr_i$, and P_n, i.e. the n-th document in P, we call $w(k_{ij}, P_n)$ the *relevance* of k_{ij} with respect to P_n. We define $w(k_{ij}, P_n)$ as:

$$w(k_{ij}, P_n) = \begin{cases} \frac{|Retr_i|-j}{|Retr_i|}, & k_{ij} \in P_n \\ 0, & otherwise \end{cases} \quad (4.5)$$

Here, $|Retr_i|$ is the number of key-phrases of $Retr_i$. The total relevance of $Retr_i$ with respect to P, $w_P(Retr_i)$, is defined as the maximal sum of the relevance of its key-phrases, obtained for all the rows in P:

$$w_P(Retr_i) = max_{n \in P} \sum_{k_{ij} \in Retr_i} w(k_{ij}, P_n). \quad (4.6)$$

Keeping the relevance computation separated across the single documents (rows) in the profile is a strategy to prevent errors and ambiguities. Without this precaution, we might have for instance a profile about programming and Greek islands resulting in a high weight for a document about the Java island.

Final Answer Ranking

Having computed a relevance score for each document retrieved for the query, the personalized version of YourQA uses the following answer ranking criteria:

1. Similarity of the answer passage to the question;
2. Relevance of the passage's source document with respect to the UM profile;
3. Google rank of the source document.

In Figure 4.2 for instance, the answer is targeted at a user interested in architecture; this is why a result about a building obtains the top ranking.

1. Title: Frank Gehry's "Ginger and Fred" in Prague - Essay by Josef Pesch, **URL:** http://lava.ds.arch.tue.nl/gallery/praha/tgehryen.html, **Google Rank:** 3, **weight:** 1.0

The Californian architect Frank O Gehry and his Czech co-architect. Vladimir Milunic have designed an impressive building to fill a space left empty in the centre of Prague after World War II bombing. **It is a 'dancing building' and was named "Ginger & Fred" in an allusion to the American film icons.** The building is part of the tradition of deconstructive architecture (also known as catastrophe architecture): Gehry's postmodern signature is undeniably visible - and stands in marked contrast to the building's historic setting.

Figure 4.2: First answer to the question: "What is *Ginger and Fred?*"

Table 4.1 compares the results of the query: "UM conference" when no profile is used and when a profile containing the key-phrase "User Modelling" is active. In the second case, the profile key-phrases disambiguate the query and contribute to a higher ranking of answers related to User Modelling (potentially more interesting to the user). Considering that in a QA system the list of answers is rarely supposed to include more than five results, filtering based on the UM can dramatically improve the relatedness of answers to the user profile. A full experimental evaluation of the usefulness of profile-based answer extraction is reported later in Section 4.6.2.

Let us point out that in personalized QA as much as in personalized IR, a key issue in pursuing user adaptivity is that this must not be at the cost of objectivity. We believe that this is the case in our approach for two main reasons. First, due to the limited number of key-phrases extracted from documents, when common key-phrases are found between one document in the UM set and one in the $Retr$ set, it appears worthwhile to point out to the user that such document is very relevant to his/her profile. Second, the compatibility

Rank	Profile OFF	Profile ON
1	University of Miami	**User Modelling**
2	University of Montana	**User Modelling**
3	undetermined	University of Miami
4	United Methodism	University of Montana
5	University of Miami	undetermined
6	University of Michigan	United Methodism
7	University of Michigan	University of Michigan
8	University of Michigan	University of Miami
9	**User Modelling**	University of Michigan
10	**User Modelling**	University of Michigan

Table 4.1: Example of personalized answer re-ranking. Meaning of "UM" in the 10 top answers to the query: "UM conference" when using no profile (left column) and when using a profile containing the key-phrase "User Modelling" (right column)

of a given document with respect to a given User Model is always a secondary ranking criterion to the semantic similarity to the query; profile match is only considered in case of a tie between candidate answers.

4.4.3 Related Work

The approach to User Modelling presented in this chapter can be seen as a form of implicit (or quasi-implicit) relevance feedback, i.e. feedback not explicitly obtained from the user but inferred from latent information in the user's documents.

Indeed, we take inspiration from Teevan *et al.* (2005)'s approach to personalized search, computing the relevance of unseen documents (such as those retrieved for a query) as a function of the presence and frequency of the same terms in a second set of documents on whose relevance the user has provided feedback.

More specifically, for each of the $|N|$ documents retrieved for a query, and for each term $t_i \in N$, the number of documents $\in N$ containing t_i, i.e. n_i, is computed. The relevance of term t_i with respect to the current user is then (Teevan *et al.*, 2005):

$$w(t_i) = log\frac{(r_i + 1/2)(N - n_i + 1/2)}{(n_i + 1/2)(R - r_i + 1/2)}, \tag{4.7}$$

where R is the number of documents for which relevance feedback has been provided (i.e. documents which have been indexed), and r_i is the number of documents which contain t_i among the R examined.

We interpret R as the set of documents composing the User Model profile, while

N evidently corresponds to the set of documents retrieved by YourQA during document retrieval. Moreover, instead of handling all the terms contained in the user's documents (which can be costly and introduce noise), we use the information deriving from key-phrase extraction and only analyse the terms contained in the key-phrase arrays P and $Retr$.

The relevance formula in Teevan *et al.* (2005), which is reported in (4.7), computing the log product between $\rho_i = \frac{(r_i+1/2)}{(R-r_i+1/2)}$ and $\nu_i = \frac{(N-n_i+1/2)}{(n_i+1/2)}$, accounts for the frequency of a term within the document in which it is contained and across the documents in the considered document set. This product can be seen as a $TF \times IDF$ measure, as ρ_i accounts for term frequency in documents for which relevance feedback has been given, and ν_i accounts for inverse document frequency.

In YourQA, such frequency-related information is already available from key-phrase extraction, as Kea is based on $TF \times IDF$. Indeed, Kea classifies a term as a key-phrase for a document if it occurs frequently in such document (high TF) and not too frequently in the other documents under exam (low DF). We assume that if a term t_i is a key-phrase for some document in P, then $\rho_i \approx \frac{1}{R}$. Similarly, $\nu_i \approx N$ if t_i is a key-phrase for some document in $Retr$. Hence, when $t_i \in P$, $w(t_i) \approx \omega = log\frac{N}{R}$, i.e. we can approximate $w(t_i)$ with a constant. This yields the relevance formula:

$$w(t_i) = \begin{cases} \omega, & t_i \in P \\ 0, & otherwise \end{cases} \tag{4.8}$$

The final relevance formula in (4.5) is a refined version of the one in (4.8), where the relevance is normalized and sensitive to key-phrase rank.

4.5 Personalized User Interface of YourQA

As a proof of the utility of personalized Question Answering, a personalized version of YourQA has been implemented with the purpose of collecting usage data and experimenting with different User Models. The personalized version described here has also been used to perform a thorough evaluation of both reading level-based and profile-based evaluation according to the evaluation methodology designed in Section 4.6.

The dynamics of use of the personalized prototype are described in the remainder of this section: interaction starts with the creation or reload of a user profile, which can be saved and updated. Then, the QA session takes place and personalized results are returned to the user.

4.5.1 Session Setup

Before entering their questions, users can provide a small amount of information about themselves via the interface in a similar fashion as in e.g. the SeAn news recommender (Ardissono *et al.*, 2001). This is the information processed to construct, save and update their User Model. When accessing YourQA, the user has three options (see Figure 4.3):

A) Create a new profile from documents of interest and/or browser bookmarks; in this case, key-phrase extraction is used to obtain a list of key-phrases from the text documents or bookmarked Web pages;

B) Load a previously saved profile; in this case, the list of key-phrases contained in the loaded user profile are obtained;

C) Decide not to use a profile; in this case, no key-phrases are extracted.

In cases A) and B), the key-phrases corresponding to the user's profile are shown to him/her, who can then exclude those he/she finds unsuitable or incorrect (see Figure 4.4). The profile resulting from the remaining key-phrases is the base for all the subsequent QA activity: any question the user will submit to YourQA will be answered by taking such profile into account.

The user can click on the "Save as..." button (see Figure 4.4) in order to remember a newly created profile or the current updates (i.e. selected/deselected key-phrases) and reload the profile in the future.

Providing documents of interest is a way to solve the cold start problem of creating a profile from a previously unseen user (Lashkari *et al.*, 1994). While defining a complete profile can be time consuming for the user, simply asking for Web pages of interest or mining his/her bookmarks folder appears to be a fairly unobtrusive and effortless way to collect initial information. Enabling the user to modify and save a profile, in addition to the implicit updates consisting in the user's evolving bookmarks and documents of interest, makes the UM component dynamic.

4.5.2 Session Execution

Once a profile has been chosen, the actual Question Answering session can start, with the user entering a question in the dedicated text field. By default, the personalized prototype developed for YourQA performs no filtering based on reading levels. However, the user has the option to activate the filtering based one of the reading levels specified in the UM (i.e. basic, medium or advanced). Alternatively, and just for demonstrative purposes, the

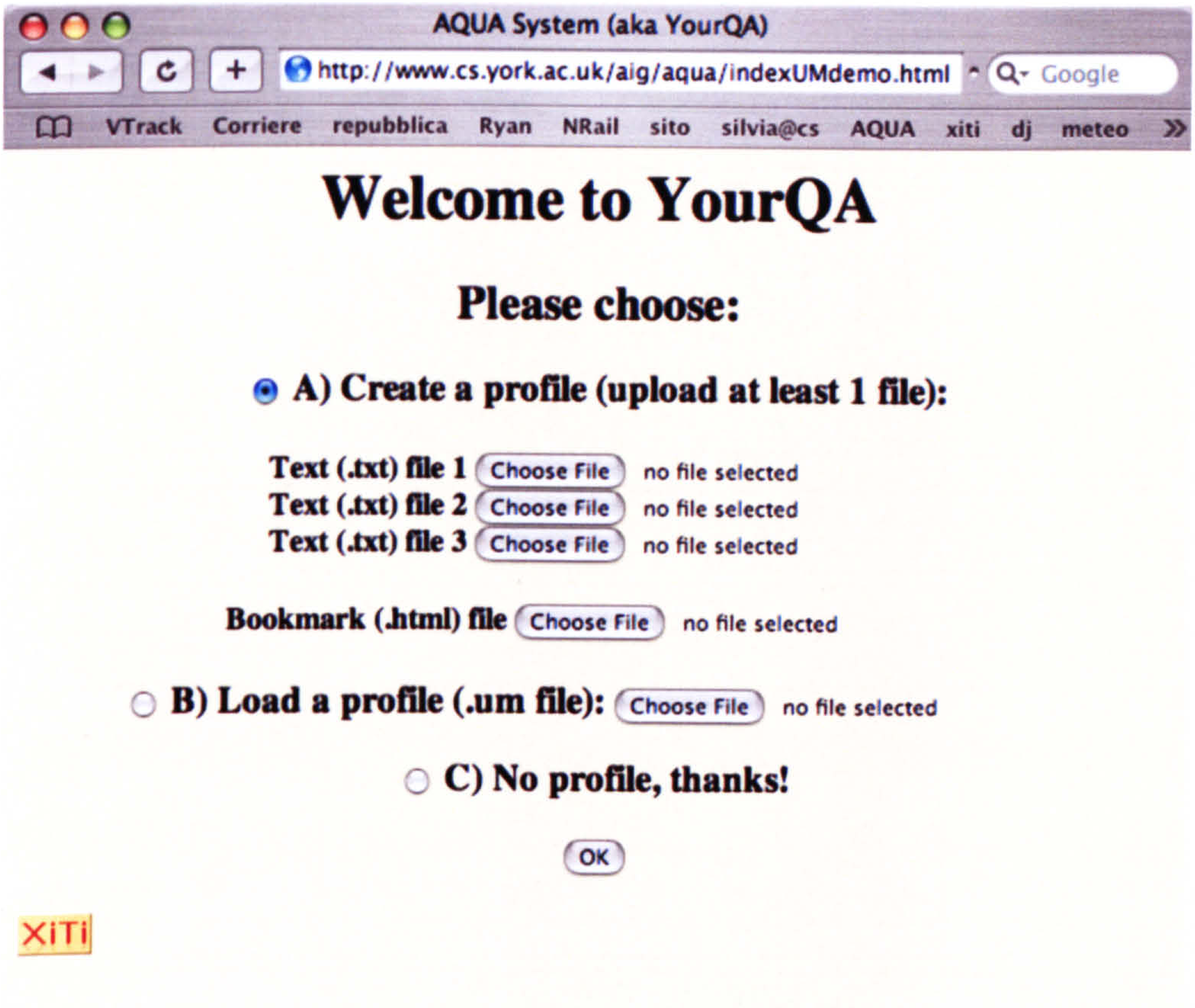

Figure 4.3: Profile creation

prototype allows to perform three separate answer extraction sessions, hence returning a different set of results for each reading level. An illustration of the query submission phase is reported in Figure 4.4.

The core Question Answering session continues as exposed in Sections 4.4.1 and 4.4.2; the result page and result format (an example of which is given in Figure 4.2) are the same as in the standard QA case, as described in Chapter 2.

4.6 Evaluation

Empirical evaluations of adaptive systems are rare. Many of them include a simple evaluation study with small sample sizes (often containing one instance) and without any statistical method. We designed the evaluation of Personalized Question Answering in

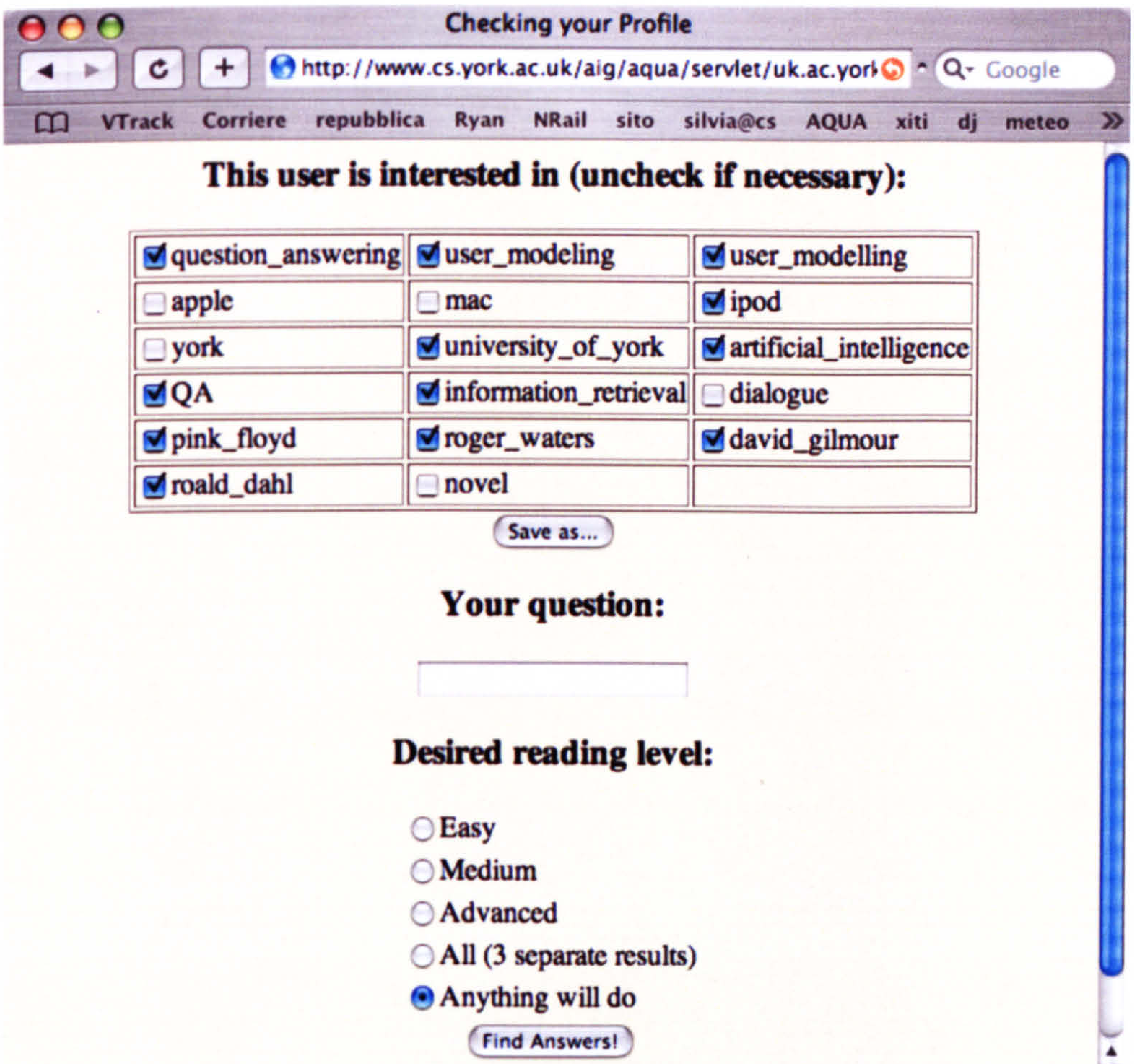

Figure 4.4: Profile modification and save, query submission

order to separately assess the contributions of the reading level attribute and of the profile attribute of the User Model. The motivation for this choice was on the one side to obtain a qualitative measure of each of the UM components, so that these may be used separately for the purpose of different applications; on the other, this evaluation strategy minimizes the introduction of biases and interaction effects.

It must be added that since the reading level parameter and the profile parameter relate to the different aspects of information presentation and content, different types of evaluation can be found in the literature to assess their performance.

Sections 4.6.1 and 4.6.2 focus on the evaluation methodology for the reading level and the profile attribute of the User Model, respectively, along with their salient results.

4.6.1 Reading Level Evaluation

The evaluation of reading level estimation was conducted according to two criteria: the first criterion was an objective assessment of the robustness of the unigram language models created to represent the User Model's reading level. The second approach was user-centered and consisted in assessing the agreement of users with the system's estimation.

Robustness of the Unigram Language Models

The robustness of the unigram language models was computed by running 10-fold cross-validation on the set of documents used to create such models.

First, we randomly split all of the documents used to create the language models into ten different folds of the same size. Then, the accuracy of reading level estimation was computed in two ways:

(a) Within each fold, the ratio of correctly classified documents with respect to the total number of documents was computed separately for each reading level. Then, the average between the three reading level estimation accuracies of each fold was used as accuracy of the fold. The final accuracy was thus the average accuracy of the different folds. The results of this experiment are reported in Table 4.6.1 (Column 1) and show an average accuracy of 91.49% with a standard deviation of $\pm$ 6.54.

(b) The ratio of correctly classified documents with respect to the total number of documents was computed for each fold regardless of the reading level. Such ratio was used as accuracy for the fold and the average accuracy was computed for the ten folds as before. The results of this second experiment are reported in Table 4.6.1 (Column 2) and show an average accuracy of 94.23% with a standard deviation of $\pm$ 1.98.

A high level of accuracy is important to ensure the consistency of reading level estimation. These results prove that unigram language models are good predictors of the basic, medium and advanced reading levels. However, this does not prove a direct effect on the user's perception of such levels. The following experiment takes charge of the user-centric aspect of reading level evaluation.

User Agreement with Reading Level Estimation

The metric used to assess the users' agreement with the system's reading level estimation was called *Reading level agreement (A_r)*. Given the set $\mathcal{R}$ of results returned by the

Fold id	Accuracy (a)	Accuracy (b)
$fold_0$	95.24	95.65
$fold_1$	100	97.22
$fold_2$	95.83	97.01
$fold_3$	87.96	95.51
$fold_4$	89.83	93.65
$fold_5$	91.67	93.84
$fold_6$	88.39	92.9
$fold_7$	95.24	93.02
$fold_8$	94.44	93.12
$fold_9$	76.28	91.07
Average	**91.49 ± 6.54**	**94.23 ± 1.98**

Table 4.2: Reading level accuracy evaluation: 10-fold cross-validation results (individual folds and average ± standard dev.)

system for a reading level r, it is the ratio between $suitable(\mathcal{R})$, i.e. the number of documents $\in \mathcal{R}$ rated by the users as suitable for r, and the total number of documents in $\mathcal{R}$:

$$A_r = \frac{suitable(\mathcal{R})}{|\mathcal{R}|}.$$

A_r was computed for each reading level.

The reading level agreement experiment was performed as follows:

- **Participants.** The involved participants were 20 subjects aged between 16 and 52. All had a self-assessed good or medium English reading level, and came from various backgrounds (University students/graduates, professionals, high school) and mother-tongues.

- **Materials.** The evaluation was performed by the 20 participants on the results returned by YourQA for 24 questions some of which are reported in Table 4.3. For each question, the results were returned in three different answer groups, corresponding to the basic, medium and advanced reading levels.

  As can be seen in Table 4.3, the answers to these questions include factoids (such as *"Who painted the Sistine Chapel?"*), lists (*"Types of rhyme"*), and definitions (*"What is chickenpox?"*); some of the answers can be controversial, such as: *"What is Shakespeare's most famous play?"*

- **Procedure.** Each evaluator had to examine the results returned by YourQA to 8 of the 24 questions. For each question, he/she had to assess the three sets of answers corresponding to the reading levels, and specify for each answer passage whether he/shee agreed that the given passage was assigned to the correct reading level.

Table 4.3 reports some sample questions along with their agreement scores. It 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.

Query	A_{adv}	A_{med}	A_{bas}
Who painted the Sistine Chapel?	0.85	0.72	0.79
Who was the first American in space?	0.94	0.80	0.72
Who was Achilles' best friend?	1.00	0.98	0.79
When did the Romans invade Britain?	0.87	0.74	0.82
Definition of metaphor	0.95	0.81	0.38
What is chickenpox?	1.00	0.97	0.68
Define german measles	1.00	0.87	0.80
Types of rhyme	1.00	1.00	0.79
Who was a famous cubist?	0.90	0.75	0.85
When did the Middle Ages begin?	0.91	0.82	0.68
Was there a Trojan war?	0.97	1.00	0.83
What is Shakespeare's most famous play?	0.90	0.97	0.83
Average	**0.94**	**0.85**	**0.72**

Table 4.3: Examples of queries and reading level agreement

4.6.2 Profile Evaluation

In designing an evaluation method for the profile component of the User Model, the aim was to assess whether user-adaptive answer filtering would be positive in terms of answer usefulness and, in any case, whether it would be perceived at all.

Since to our knowledge there is very little if any published work on the evaluation of personalized QA, we drew our evaluation guidelines from general work on user-adaptive system evaluation (Chin, 2001) and from the closest domain to QA for which this exists: personalized search (Brusilovsky & Tasso, 2004).

As personalized search is a form of IR, its typical evaluation metrics are precision and recall, where precision is measured in terms of user satisfaction. An example of such evaluation is the one for UCAIR (Shen *et al.*, 2005), a search engine plugin which re-orders the list of results with respect to the user's information need model (this is based on his/her browsing actions, i.e. clicking on a link, hitting the "back" button, etc.). Here, the baseline system for evaluation is the underlying search engine, and the application's performance metric is result precision at different recall levels.

In the evaluation of YourQA, the impact of the User Model profile was tested by using as a baseline the standard version of YourQA, where the User Modelling component is inactive.

Two experiments were conducted to evaluate personalization, involving a total of 22 participants. The second experiment was designed to correct and improve the power of the first one, however the procedures were very similar, as described below.

4.6.3 First Experiment

The first experiment involved twelve adult participants, seven male and five female, from different backgrounds and occupations. The design of the experiment consisted of the following two phases.

First phase: profile design

In the first phase, participants were invited to explore the Yahoo! Directory (`http://dir.yahoo.com`) and provide 2-3 categories of their interest. Moreover, they were invited to brainstorm as many key-phrases as they wanted relating to each of their chosen categories.

Key-phrases were used to create offline individual profiles to be loaded into memory in the following phase. For each profile domain, related queries were elaborated in such a way that the system's answers would be different when the UM-based filtering component was active, and entered in one set. The design of "artificial" queries for the users, by which their interaction with the system was controlled instead of leaving them free to formulate their own queries, was necessary to ensure that the final answer ranking would be affected by the use of the profile, allowing to measure a difference with respect o the "profile off" case.

Second phase: Question Answering

Participants were then assigned an instruction sheet with three tasks. Each task started with one of three fixed queries to be typed in YourQA, chosen from the previously compiled query set with specific criteria:

Q_A: related to one of his/her interest domains. This was answered using his/her profile.

Q_B: related to a different interest domain of the same user, answered using a baseline QA (i.e: YourQA *without* User Modelling);

Q_C: related to another user's profile, with no overlap with the current user's profile. This was answered using the baseline QA system. Note that, since by construction there is no common key-phrase between the current profile and the retrieved documents, by (4.5) *the output is the same as if the profile was used.*

The roles of the three questions above are the following.
Q_A tests YourQA's personalization abilities. Hence, it was chosen for each user so that the final list of answers would be affected by the UM component.
Q_B represents the baseline QA system: its role is to compare YourQA to a state-of-the art system under the same experimental conditions.
Q_C is an additional, "control" baseline query whose role is to check if there is a bias in the user towards questions relevant to his/her profile. Also, since the same queries were used as Q_A and Q_C for different users, we could compare the answers given to each query when the UM profile was active and when not.

Examples The queries formulated for the profile evaluation were mostly non-factoid queries: several questions invoked definitions of terms or expressions which could have several meanings depending on the field of interests. For instance, the question: "What is boot camp?" could refer to both computing and the military domain and its answer rankings varied based on whether or not the user submitting it to YourQA was the one having specified mac applications in their interests. Another example was the question: "What is Apollo?" for which results about NASA missions would be ranked highest for a user interested in space exploration than for a user with no such interests. Several of the questions invoked disambiguating acronyms, such as: "What is a RPG?" or "What is EOS?" which referred to role-playing games and digital photography.

For each query, the top five answers were computed in real time by the QA system by switching off reading level filtering to minimize biases.

Questionnaire Regardless of the application, a common UM evaluation practice is the use of questionnaires, where a sample of individuals is selected to represent the potential user population and is submitted with a series of questions about a recently experimented prototype. For example, the adaptive learning environment in Habieb-Mammar *et al.*

(2003) is assessed according to different aspects of the application, such as usability and satisfaction.

In the evaluation of the profile component, we defined a user satisfaction questionnaire to be filled in by the users as follows. As soon as each query's results were available, the users had to answer the following four questions on the experiment instruction sheet:

- For each of the five results *separately*:

  TEST1: *This result is useful in answering the question*: Yes / No

  TEST2: *This result is related to my profile*: Yes / No

- Finally, for the five results taken as a whole:

  TEST3: *Finding the information I wanted in the result page took*:
  (1) Too long, (2) Quite long, (3) Not too long, (4) Quite little, (5) Very little

  TEST4: *For this query, the system results were sensitive to my profile*:
  Yes / No / Don't know

The questionnaire provides a qualitative assessment of the effects of User Modelling, which are tested at user level to eliminate the nuisance introduced by cross-user evaluation (Chin, 2001). Each question relates to a separate factor:

- TEST1 measures the perceived usefulness of each result in answering the corresponding query. This measurement corresponds to the standard user-centered precision metric applied by other personalized IR applications, such as Shen *et al.* (2005).
- TEST2 measures the perceived relatedness of the answer content with respect to the profile.
- TEST3 measures the user's satisfaction with respect to the time taken browsing results. This is another typical user-centered evaluation metric (Walker *et al.*, 2000).
- TEST4 measures the perceived profile sensitivity in answering the query overall (i.e. not with respect to the individual answers).

Interaction logs Users interacted with YourQA on a workstation equipped with MORAE^{TM} (`www.techsmith.com/morae`), a commercial, non-intrusive software able to record the user's activity while carrying on a task. Interaction logs were recorded to measure the time taken to find information and to complete and understand user comments and questionnaire answers.

Results

The qualitative information collected from the questionnaire was the following.

Answer usefulness (TEST1) Table 4.4 reports the average and standard deviation of the number of answers judged useful for each query (answers to TEST1). These were compared by carrying out a one-way analysis of variance (ANOVA) and performing the Fischer test using the usefulness as factor (with the three queries as levels) at a 95% level of confidence. We chose this test rather than a paired t-test, as the factor under exam has 3 levels (the 3 queries), to ensure the error robustness and sensitivity of the test.

The test revealed a significant difference in the specific contrast between Q_A and Q_C (linear F= 5.86, degrees of freedom = 1,11, $p = 0.034$), suggesting that users are positively biased towards questions related to their own profile when it comes to perceived utility.

However, we did not find a significant difference overall, hence not between Q_A and Q_B, therefore we cannot prove that there is a significant impact in utility when the UM is active. We believe that this may be due to the fact that our study involved a limited number of users and that their judgments may have been "distracted" by other aspects of the system, such as the response time.

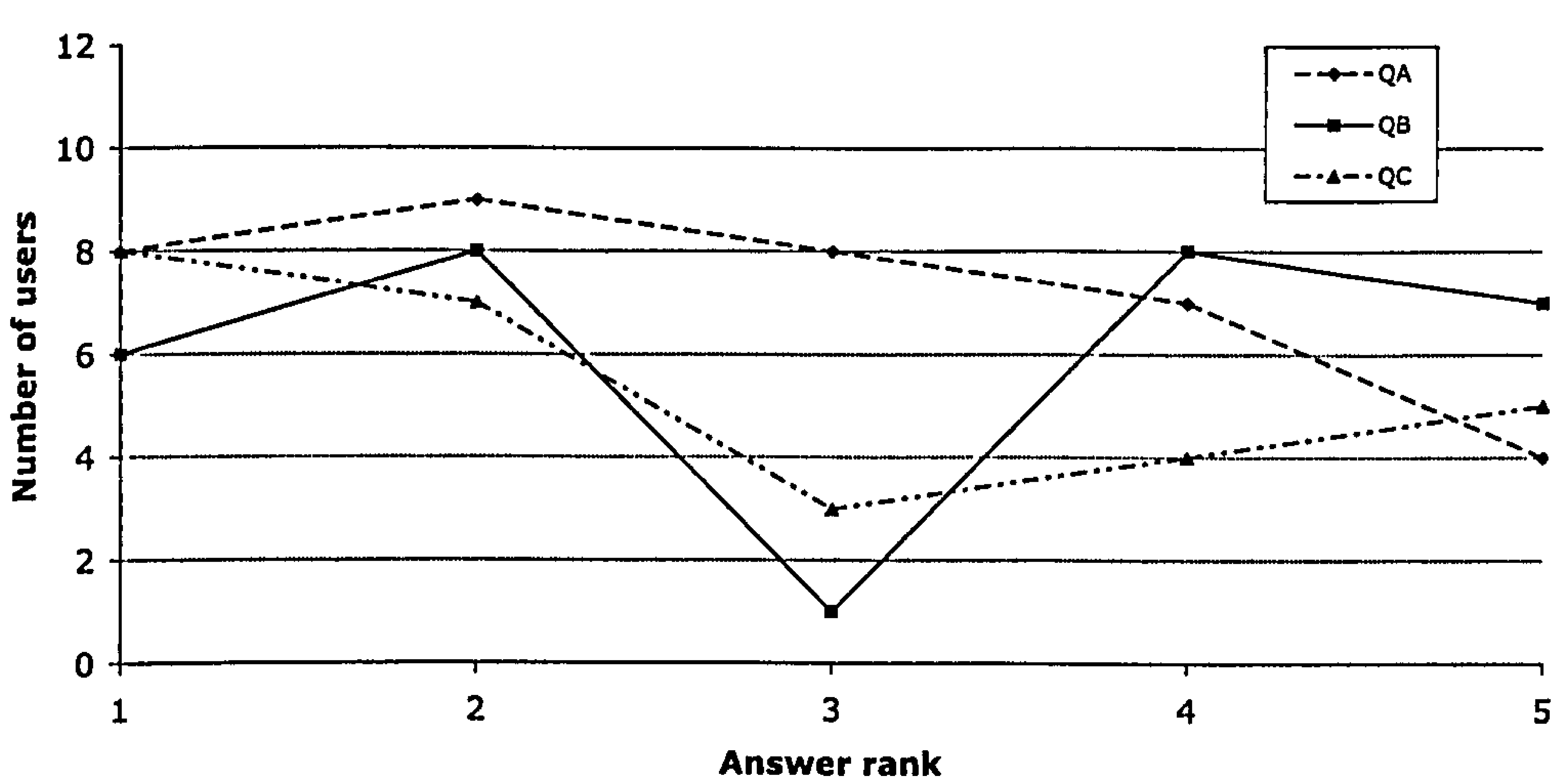

Figure 4.5: First evaluation: perceived answer usefulness (number of users vs rank)

To further investigate perceived utility, we counted *for each query q* the number of answers judged useful by the user to which q had role Q_A (i.e. was addressed by the personalized QA system). Moreover, we counted the occurrences of the same answers in the list of results to the user for which q had role Q_C (i.e. when results were not affected by personalization). These counts are visualized in Figure 4.6.

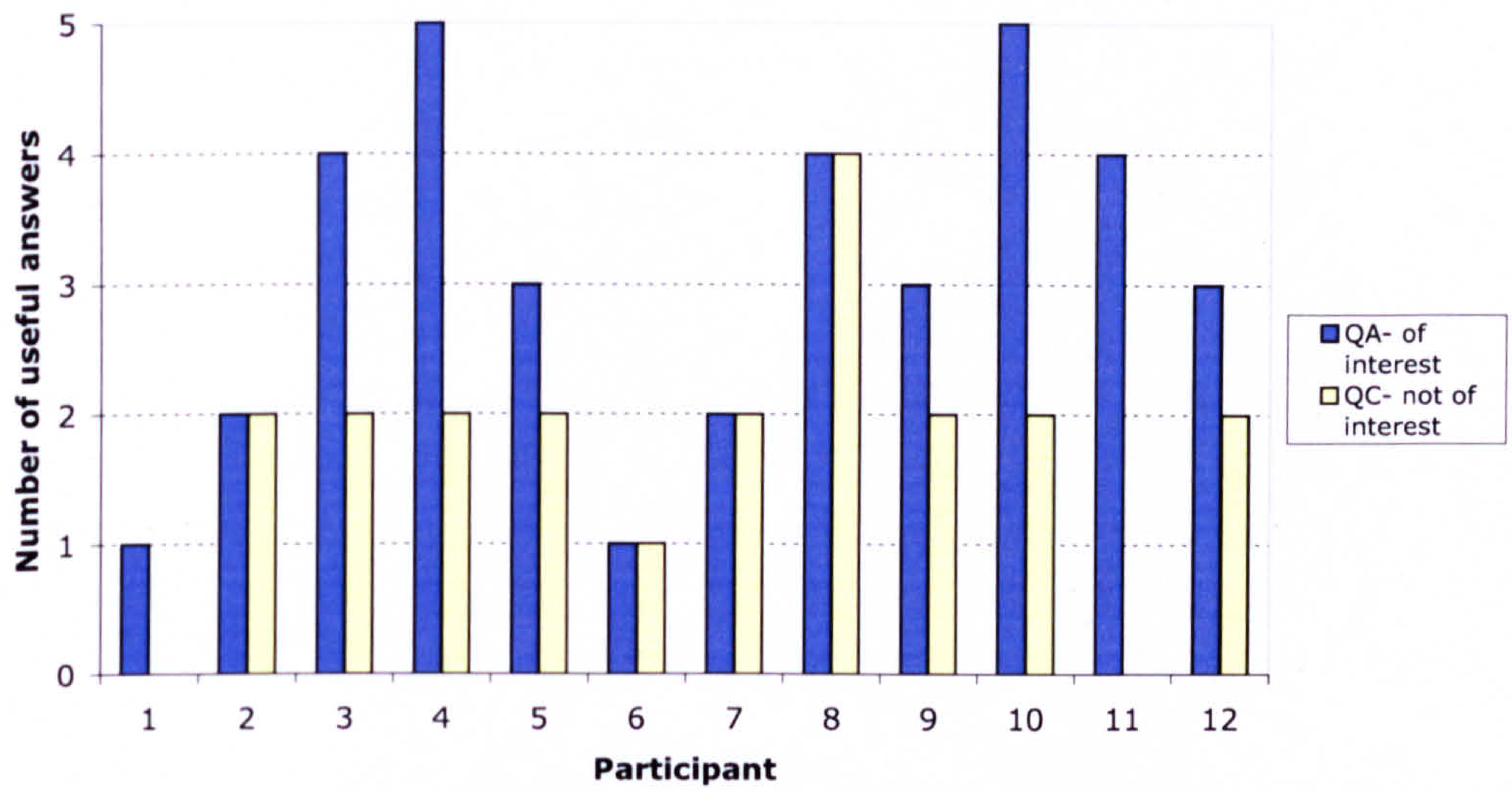

Figure 4.6: Occurrence of useful answers with respect to role of the question. For each of the 12 questions q_i on the "Participant" axis, bar "QA" shows how many of the five answers were judged as useful by the participant having a profile related to q_i, while bar "QC" shows how many of such answers were returned to the participant having an unrelated profile to q_i.

The paired t-test showed a statistically significant difference ($p = 0.006$), so we can say that for the same questions, useful answers are more likely to occur when profile filtering is active.

The latter measurement can be seen as a form of recall, and related to the evaluation conducted for the COGITO item recommender (Abbattista *et al.*, 2002). COGITO's performance metrics include recall (the fraction of positive examples classified as positive) and precision (the fraction of examples classified as positive that are actually positive). The perceived answer usefulness metric reported above can be viewed as a form of precision at a fixed level of recall (i.e. the so-called $P@5$ metric).

As a final remark, the number of users finding each single answer to Q_A useful started high for the first result and tended to decrease with the answer rank, as visible in Figure 4.5. In contrast, the usefulness of answers to Q_B and Q_C exhibited a more random allure.

However, when we performed the Friedman test on these values, we did not find a significant difference, probably because the data came from five measurements (i.e.

ranks) only. In the following experiment, we elicited Likert scale answers instead of Yes/No answers for a more fine-grained analysis, as explained below.

Answer relatedness (TEST2) To analyze the answers to TEST2, which measured the perceived relatedness of each answer to the current profile, we computed the ANOVA table on the data in Table 4.4, row 2. We used the number of answers judged as related as the independent variable and the three queries as factors. This time, the results showed an overall significant difference (F= 11.9, d.f. = 1,11, $p < 0.001$).

These results confirm that answers obtained without using the users' profile were perceived as significantly less related to those obtained using their own profile, i.e. there is a significant difference between Q_A and Q_B. As expected, the difference between Q_A and Q_C (where the question is unrelated to the profile) is even more significant.

Once again, we observed that the perceived relatedness of the results to Q_A tended to be higher for the first ranked answers and to slightly decrease with the answer rank (see Figure 4.7).

For Q_B, the result relatedness was generally lower and seemed to follow a more irregular pattern; this makes sense as the profile ranking was not active.

For Q_C, the result relatedness was much lower and again did not exhibit a descending pattern across the rank as the relatedness for Q_A did. However, from Friedman's ANOVA we can only call Q_A's descending pattern a trend, as $0.05 < p = .098 < 0.1$ (F=8.2, d.f.=4).

Table 4.4: Perceived answer usefulness and relatedness to the user profile

Measurement	Q_A	Q_B	Q_C
Perceived usefulness	0.6±1.42	0.5±1.57	0.45±0.29
Perceived relatedness	0.7±1.38	0.5±1.98	0.22±1.88

Time spent looking for answers (TEST3) In formulating TEST3, we assumed that profile-based QA would help users find interesting information more quickly. However, the time question proved problematic: we noticed from user comments and interaction logs that such time was often mistaken with the perceived duration of the document retrieval phase. Another factor making time difficult to interpret is the fact that the system was previously unknown, hence examining the results to the first query took longer than the following ones.

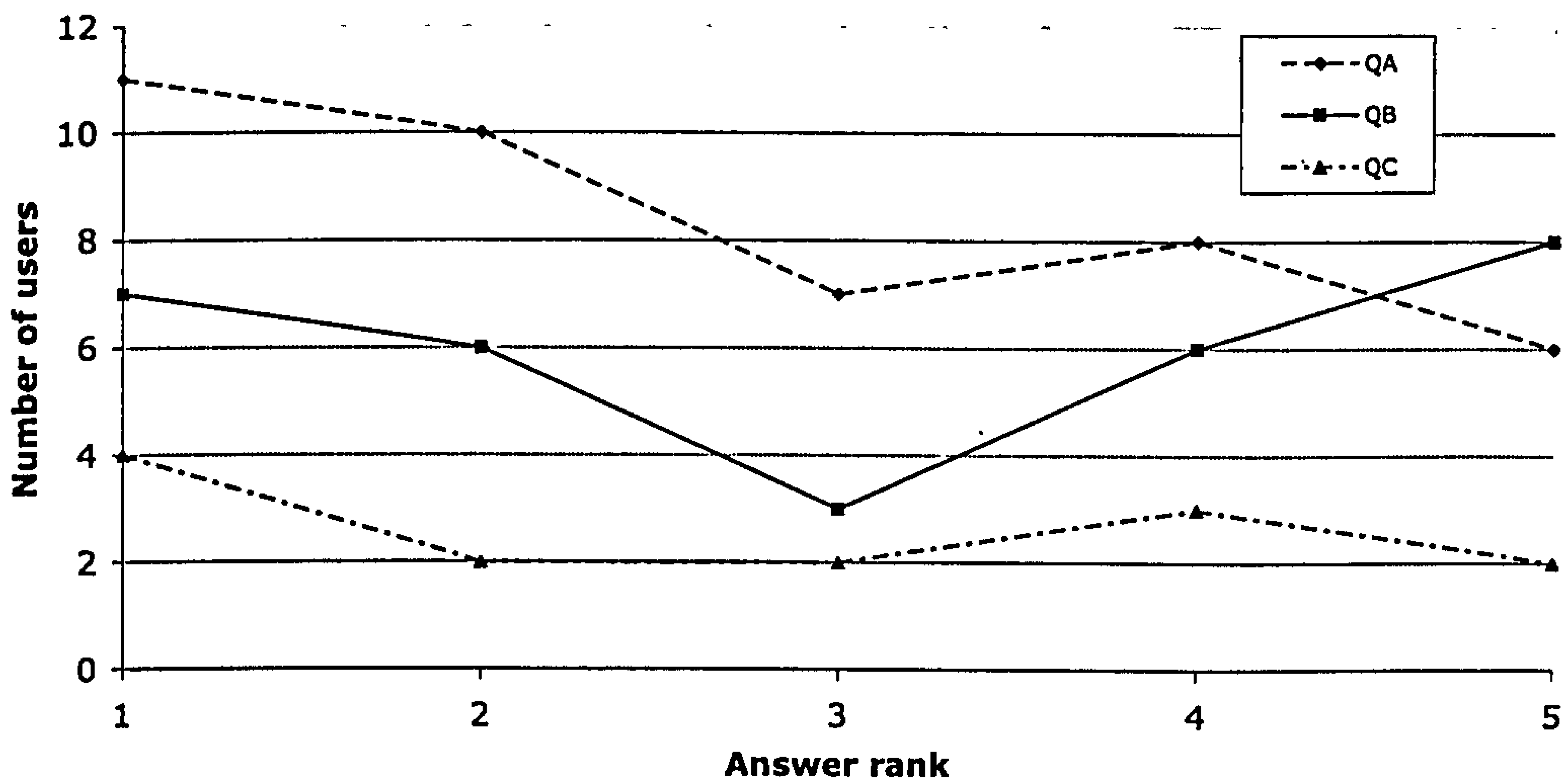

Figure 4.7: First evaluation: perceived answer relatedness to profile (number of users vs rank)

Furthermore, several users mistook the time spent looking for information with the time spent actively browsing the result page and clicking on the result links to read interesting information; to these users, a longer browsing time probably meant better fitness of the answers to the profile. Hence, we decided not to consider time as a source of information in the first evaluation.

Profile sensitivity (TEST4) One surprising result from the questionnaire was that although Q_C was selected for each user in order to be as unrelated as possible to his/her categories, users did not always realize that their profile had no role in answering such query (perhaps the wording: *The system's answers were sensitive to my profile* was ambiguous).

In any case, the perceived relatedness to the user's profile of the answers as a whole, i.e. the profile sensitivity of the system in answering the query altogether, was sensibly higher for Q_A (0.92±0.27) than for Q_B (0.5±0.52) and Q_C (0.28±0.47), as shown in Figure 4.8.

We computed the ANOVA table using, as a variable, the number of users agreeing that the system had been sensitive to their profile in answering the current query and the three queries as factors. This gave a significant difference between each query (F= 22, d.f.=1,11, $p < 0.001$), confirming that users perceived the sensitivity of the system to their own profile when the UM component was active.

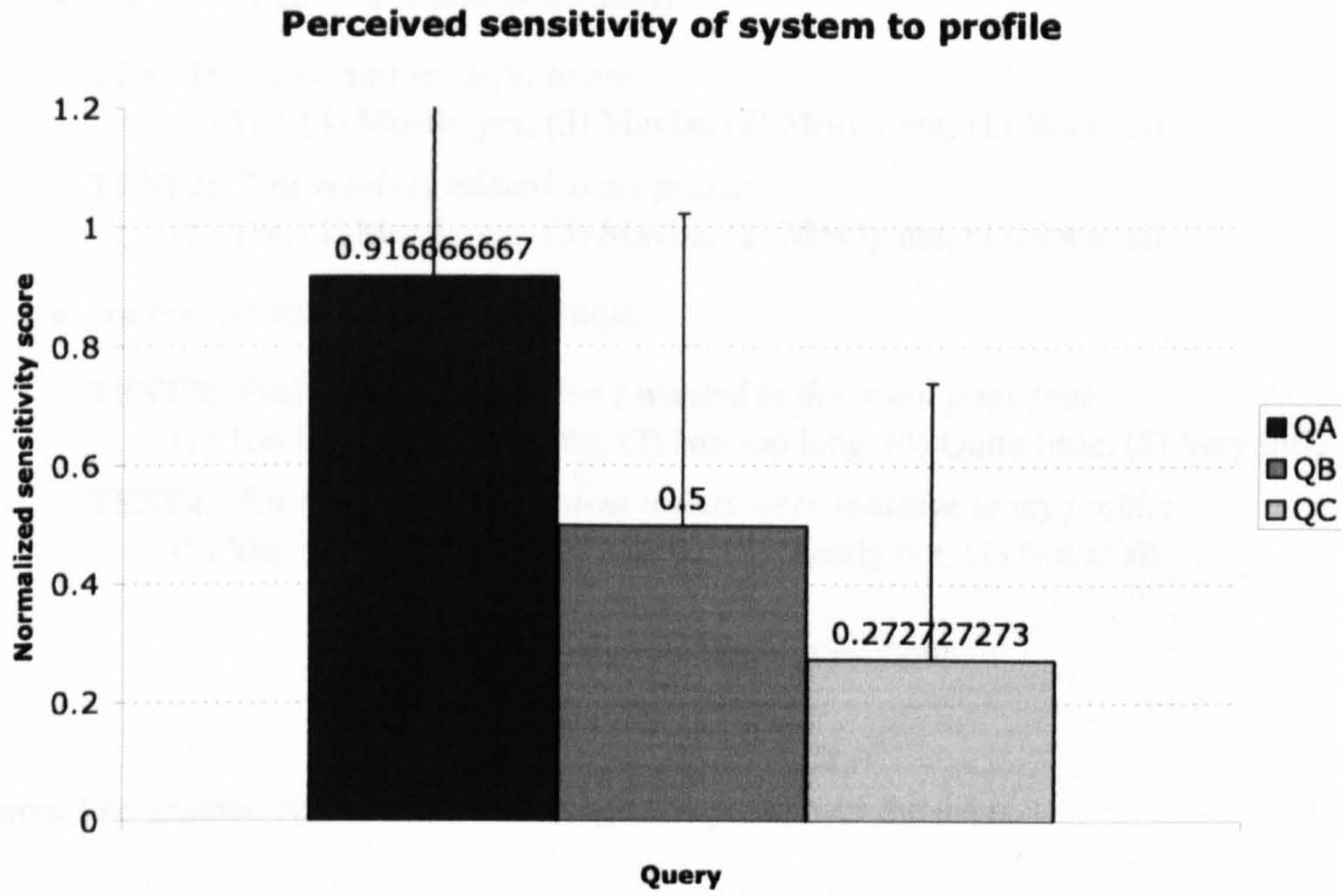

Figure 4.8: First evaluation: perceived system sensitivity to profile (percentage of agreement with system sensitivity vs role of the query)

Final Evaluation

The final evaluation built on the experience of the initial evaluation and amended some of its limitations. This time, ten participants from various backgrounds took part in the experiment, six women and four men; all were adults. While the general design was unmodified, the questionnaire was changed so that users had to answer questions TEST1, TEST2 and TEST4 via a 5-point Likert scale going from 5="Yes" to 1="Not at all".

Moreover, the text of TEST1 was modified into: *This result is useful to me*, so that usefulness was referred to the users' profile rather than to the actual content of the answer. This change was made to address the fact that during the first experiment users did not know what to reply when they were unfamiliar with the subject of the questions or didn't know the correct answer. The final evaluation questionnaire is reported in Figure 4.9.

The adoption of a Likert scale made it possible to compute the average and standard deviations of the user comments with respect to each answer among the top five returned by the system. It was therefore possible to replace the initial binary measurement of perceived usefulness, relatedness and sensitivity in terms of total number of users with a

- For each of the five results *separately*:

  TEST1: *This result is useful to me*:
  (5) Yes, (4) Mostly yes, (3) Maybe, (2) Mostly not, (1) Not at all

  TEST2: *This result is related to my profile*:
  (5) Yes, (4) Mostly yes, (3) Maybe, (2) Mostly not, (1) Not at all

- For the five results taken as a whole:

  TEST3: *Finding the information I wanted in the result page took*:
  (1) Too long, (2) Quite long, (3) Not too long, (4) Quite little, (5) Very little

  TEST4: *For this query, the system results were sensitive to my profile*:
  (5) Yes, (4) Mostly yes, (3) Maybe, (2) Mostly not, (1) Not at all

Figure 4.9: Final evaluation: questionnaire

more fine-grained one in terms of average computed over the users.

Results

The final experiment results, summarized in Table 4.5, are discussed below.

Table 4.5: Second evaluation: summary of results (average ± standard dev.)

Measurement	Q_A	Q_B	Q_C
Perceived usefulness (TEST1, rank average)	3.6±0.4	2.3±0.3	3.3±0.3
Perceived relatedness (TEST2, rank average)	4.0±0.5	2.2±0.3	1.7±0.1
Perceived time (TEST3)	3.1±1.1	2.7±1.3	3.4±1.4
Perceived sensitivity (TEST4)	3.9±0.7	2.5±1.1	1.8±1.2

Answer usefulness (TEST1) The first row of Table 4.5 reports the average and standard deviation of the perceived answer usefulness for each query (answers to TEST1). These results, as visible from Figure 4.12, show a remarkable difference between the perceived usefulness for question Q_A with respect to question (Q_B).

The results were compared by carrying out a one-way analysis of variance (ANOVA) and performing the Fischer test using the usefulness as factor (with the three queries as levels) at a 95% level of confidence. This time, the test on perceived usefulness revealed an overall significant difference (linear F=3.811, degrees of freedom = 1,9, $p = 0.035$),

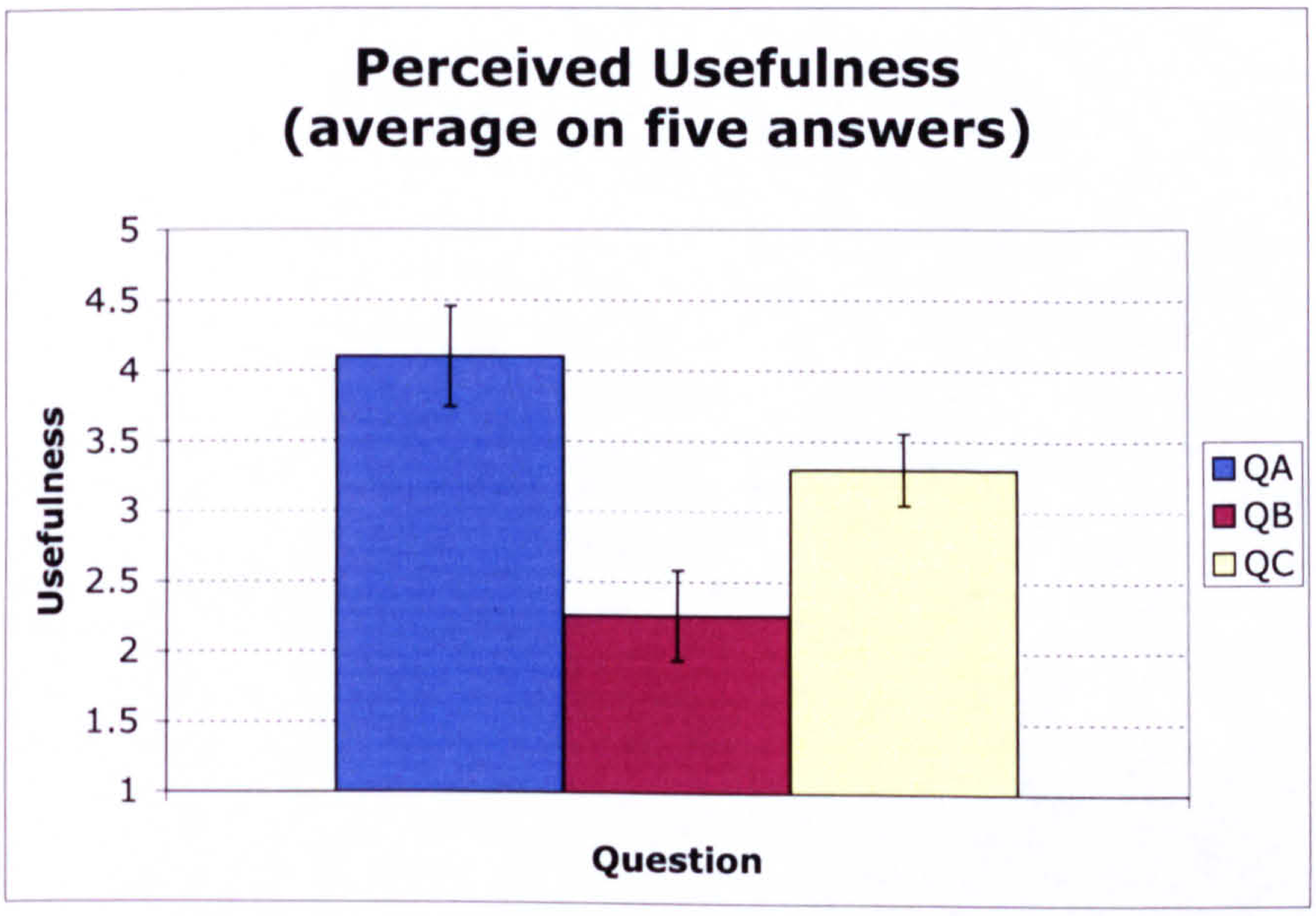

Figure 4.10: Second evaluation: perceived system usefulness to profile (average computed over the top five answers)

confirming that users are positively biased towards questions related to their own profile when it comes to perceived utility.

Figure 4.11 shows the average usefulness plotted over the top five answer ranks; once again, these results indicate that when personalization is active, the usefulness of answers to the question related to each user's profile (Q_A) started high and gradually decreased. It is slightly less the case for the unrelated question (Q_C) and especially for question Q_B, which is related to the user's profile but for which no personalization is applied.

Answer relatedness (TEST2) To analyze the answers to TEST2, which measured the perceived relatedness of each answer to the current profile, we computed the ANOVA table on the data in Table 4.5, row 2. We used the average relatedness of the answers computed across the users as the independent variable and the three queries as factors. This time, the results showed an overall significant difference (F= 15.33, d.f. = 1,9, $p < .0001$).

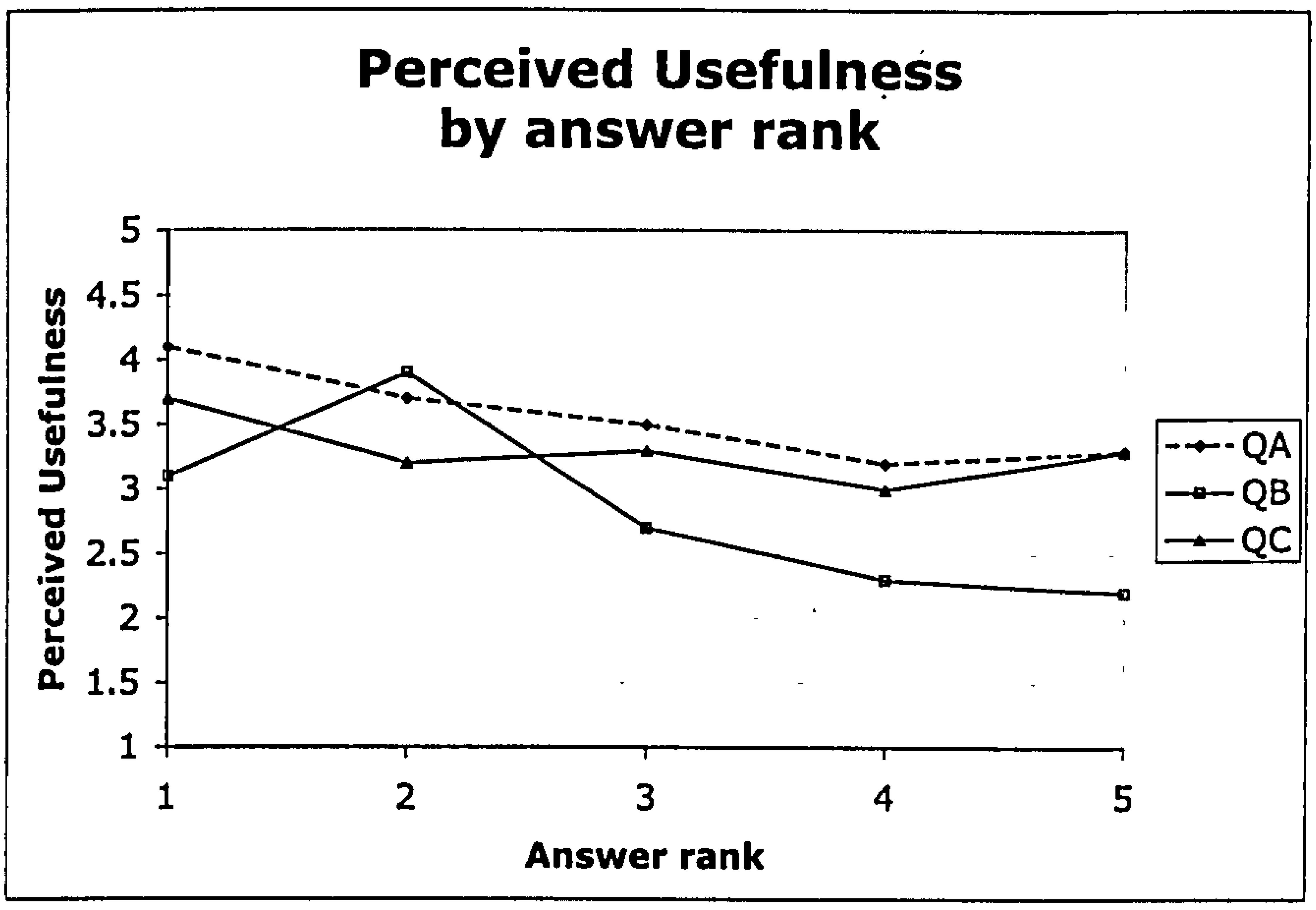

Figure 4.11: Second evaluation: perceived system usefulness to profile with respect to answer rank

These results, visualized in Figure 4.12, confirm that answers obtained without using the users' profile were perceived as significantly less related to those obtained using their own profile, i.e. there is a significant difference between Q_A and Q_B. As expected, the difference between Q_A and Q_C (where the question is unrelated to the profile) is even more significant.

Once again, we observed that the perceived relatedness of the results to Q_A tended to be higher for the first ranked answers and to slightly decrease with the answer rank (see Figure 4.13). For Q_B, the result relatedness was generally lower and seemed to follow a more irregular pattern; this makes sense as the profile ranking was not active. For Q_C, the result relatedness was much lower and again did not exhibit a descending pattern across the rank as the relatedness for Q_A did.

Time spent looking for answers (TEST3) From the first evaluation, we knew that the time question is problematic as the perception of time is influenced by a variety of fac-

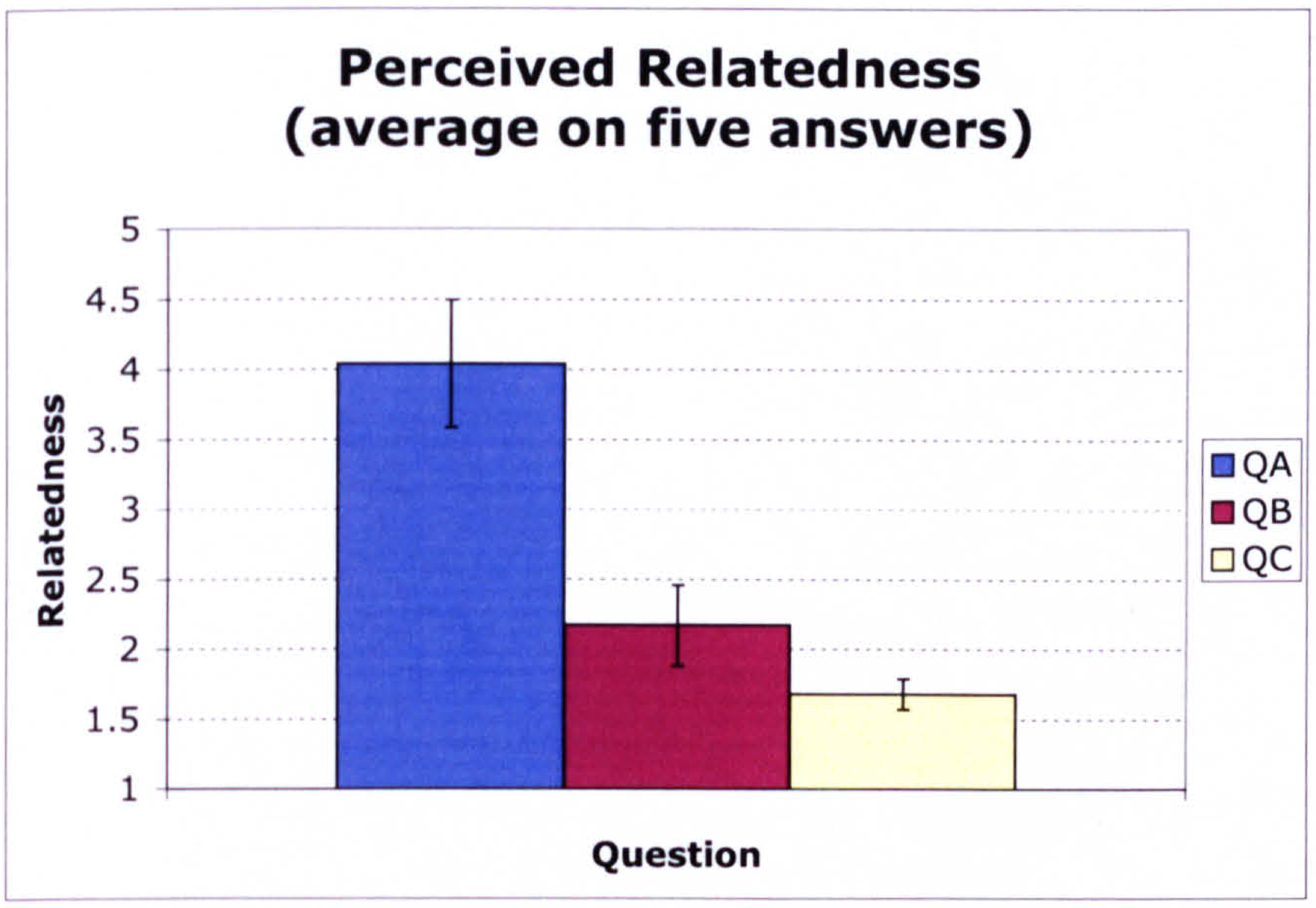

Figure 4.12: Second evaluation: perceived system relatedness to profile (average computed over the top five answers)

tors, such as familiarity with the system and personal interest towards the results. For this reason, although we performed the perceived time evaluation once again in the second evaluation, we do not believe that this result can be used to prove claims about personalization efficiency.

The average perceived time needed for users to locate suitable answers to their queries is plotted in Figure 4.14, which shows that, for questions related to the users' profile, the time required to locate answers was perceived to be slightly shorter when the personalization component was on (Q_A obtained an average of 3.1$\pm$1.1, while Q_B obtained 2.7$\pm$1.3).

However, for question Q_C, which was unrelated to their profile, users found the time of interaction even shorter, although with a considerable standard deviation (3.4$\pm$1.4). A possible explanation to this result may be the fact that users spent less time investigating results to questions which they perceived to be unrelated to their profiles; in general

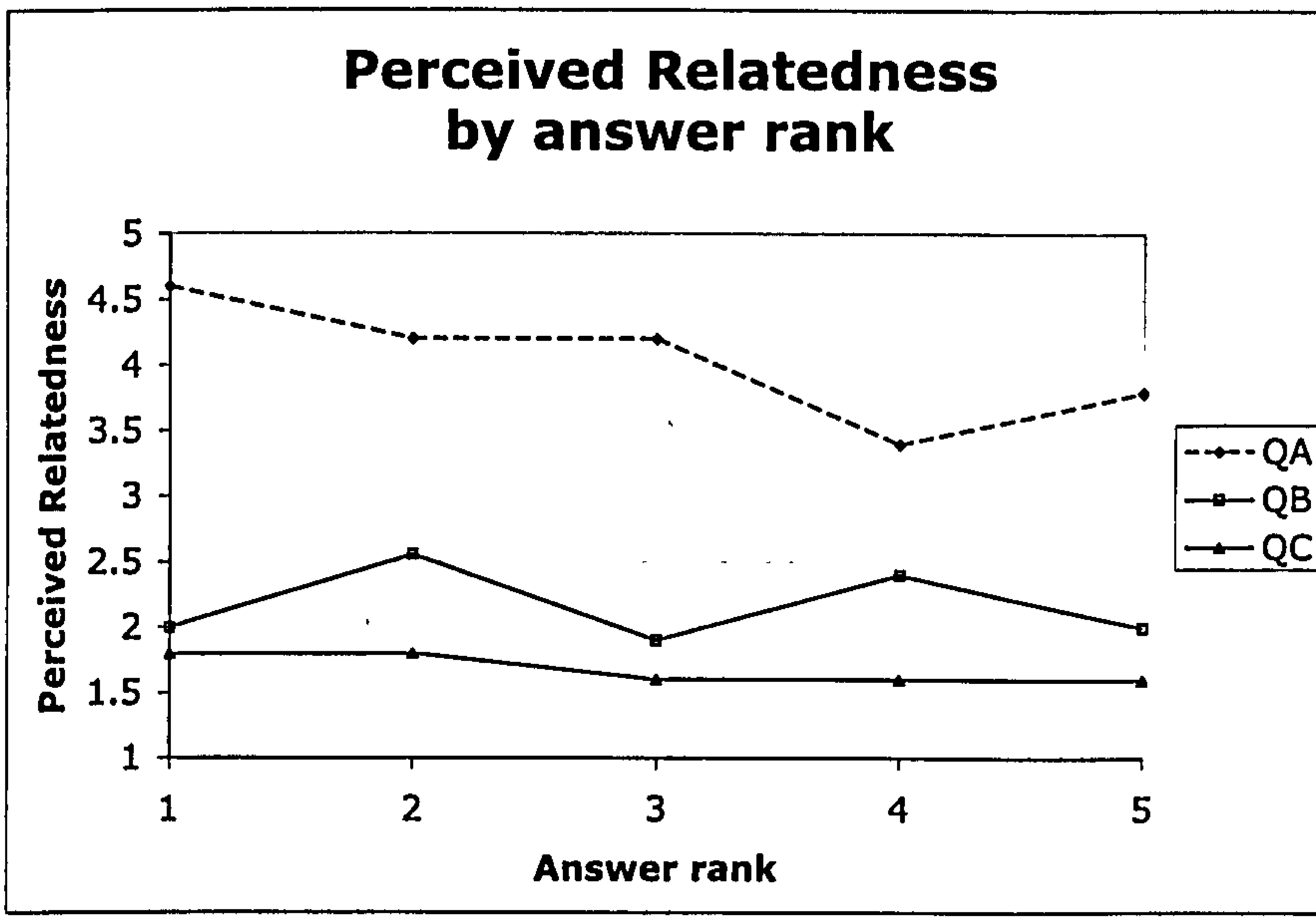

Figure 4.13: Second evaluation: perceived system relatedness to profile with respect to answer rank

however, we can say that there appears to be no relationship between the perception of browsing time and personalization.

This result is confirmed by the fact that the ANOVA table computed using average perceived time as variable and the three questions as factors did not give any significance, nor did any of the paired t-tests computed over each result pair.

Profile sensitivity (TEST4) For the second evaluation, we clearly specified on each user's profile summary that they should assume during the experiments that their interests were exclusively the ones specified during the first step of topic elicitation. This was to reduce the occurrence of biases from other interests which users may not have specified in the elicitation phase.

The average sensitivity of the five answers altogether computed over the ten participants for each query is plotted in Figure 4.15. This shows a considerable difference in perceived sensitivity between the answers to question Q_A (3.9±0.7) and those to question

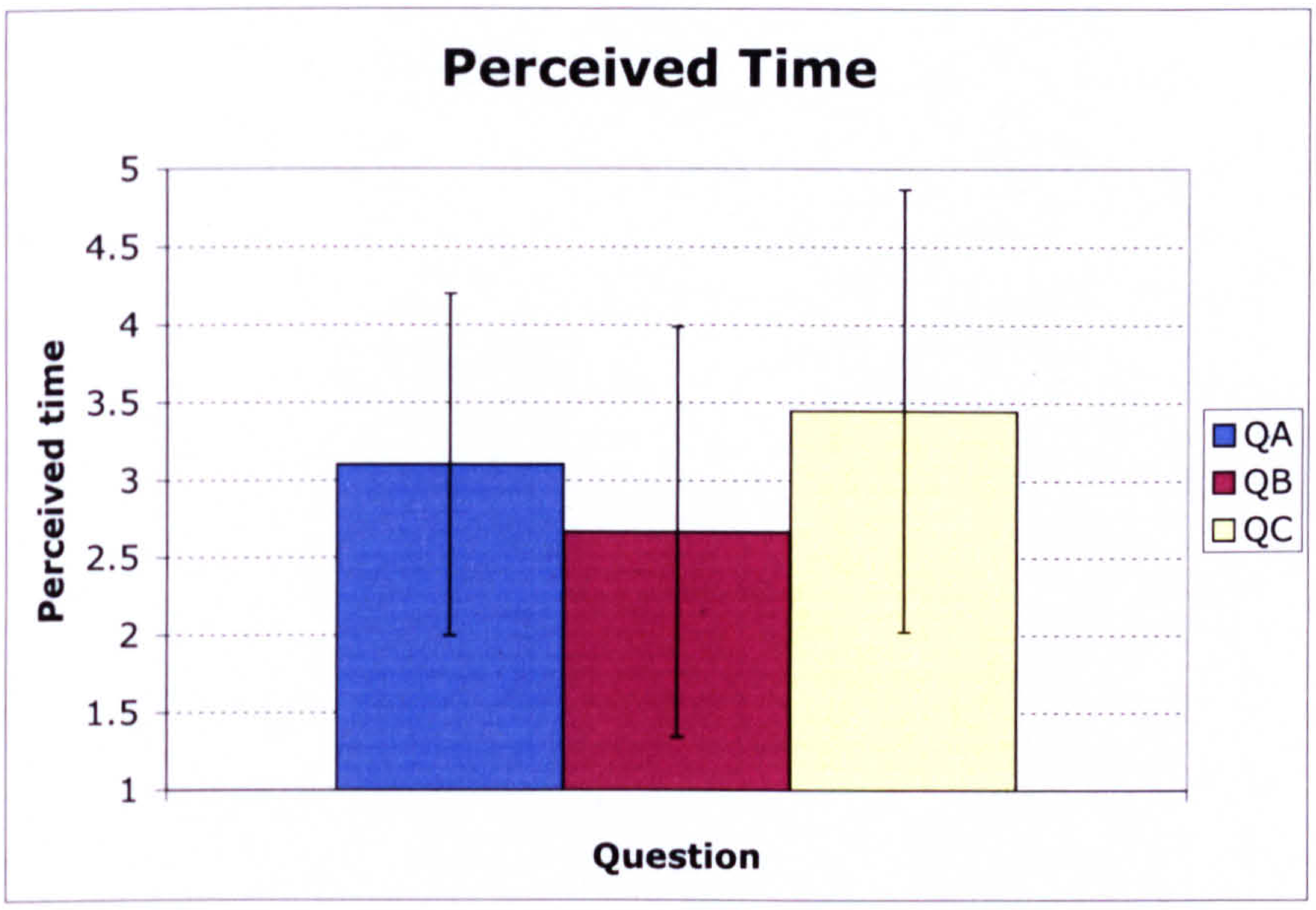

Figure 4.14: Second evaluation: perceived result browsing time

Q_B (2.5±1.1) and Q_C (1.8±1.2).

As we used a Likert scale during the second evaluation, we were able to perform an analysis of variance on the sensitivity results. The ANOVA table showed an overall significant difference (F=10.64, d.f.=1,9, $p < 0.0004$).

To conclude, our experience with profile evaluation shows that personalized QA techniques yield answers that are indeed perceived as more satisfying to users in terms of usefulness and relatedness to their own profile. This is a very positive result which makes it encouraging to explore more refined models of the users interests and also the assessment of profiles based on automatically extracted keywords from user documents, which was not the case in the course of our exploratory evaluations.

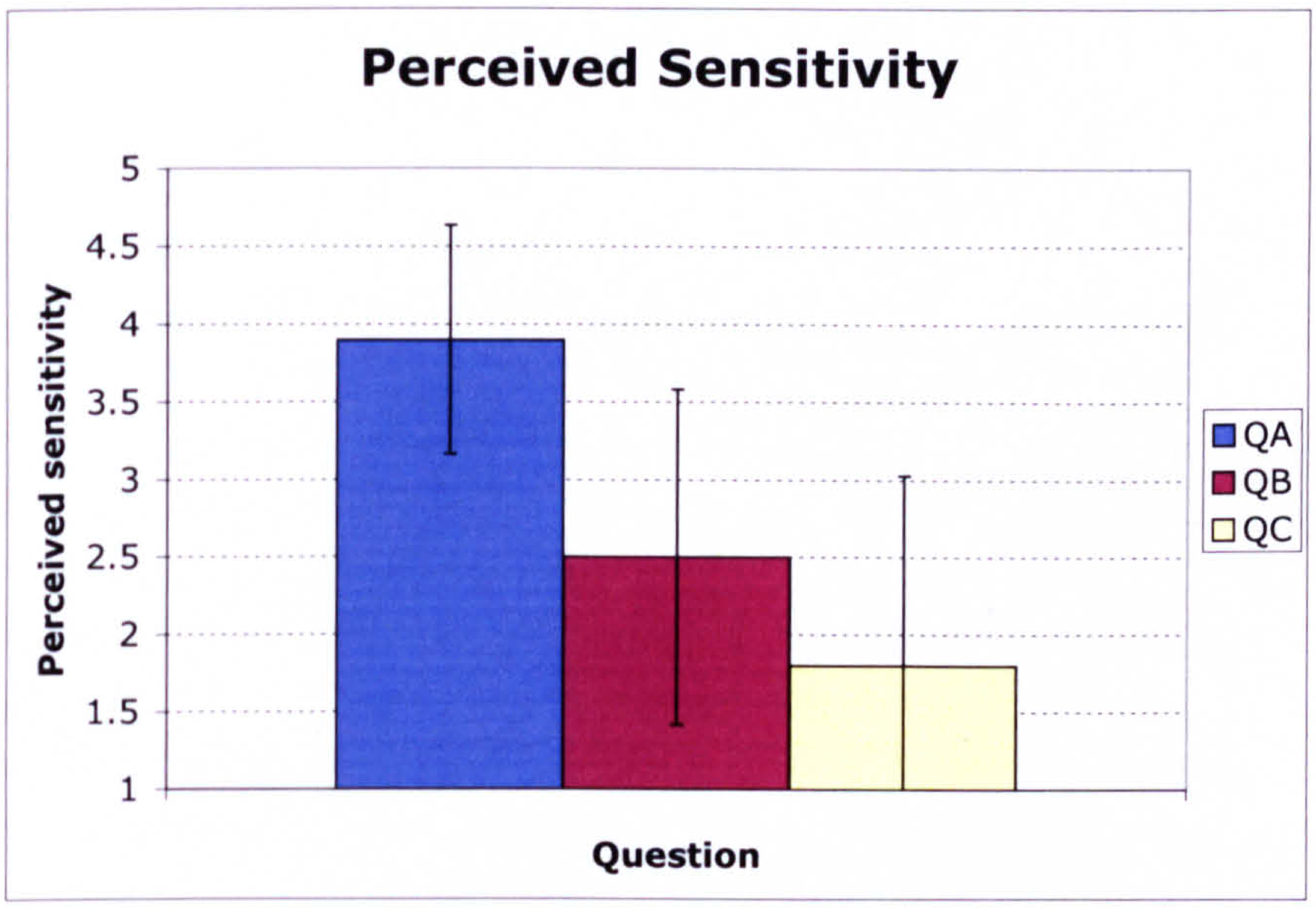

Figure 4.15: Second evaluation: perceived sensitivity to profile

4.7 Conclusions

In this chapter, we have presented an efficient and light-weight method to personalize the results of a Web-based Question Answering system based on a User Model representing individual users' reading level, age range and interests. Although the intuition behind the use of User Modelling for personalizing a Web application is not new (Brusilovsky & Tasso, 2004), this chapter describes to our knowledge the first fully implemented and evaluated application of User Modelling for Question Answering.

We show how the User Model components can be estimated automatically and fairly unobtrusively from the user's documents and how they can be used to filter and re-rank the answers to their queries. Moreover, we introduce a user-centered evaluation methodology for personalized Question Answering which assesses independently the effects of the two main User Model components: reading level and personal interests.

The results of our experiments show on the one hand the efficiency of the language modelling techniques used for reading level estimation; on the other, we find a statistically significant improvement when filtering answers based on the users' profile in terms of both perceived answer usefulness and profile relatedness.

4.7.1 Future Work

Future work in personalized Question Answering will further integrate the User Modelling component and the standard Question Answering component of YourQA to allow dynamic updates of the UM based on previous information-seeking history.

A first step towards an increasingly dynamic User Model consists in the analysis of the interaction logs obtained by the system, upon which key-phrase extraction can be performed to update the user's interests.

Inspiration for this phase can be taken from collaborative filtering in the field of item recommenders, where the user contributes to a model of his own preferences concerning a particular item and provides explicit feedback information to the system when obtaining results. For instance, in GroupLens (Resnick *et al.*, 1994) collaborative filtering is based on explicit user ratings, ratings implicitly extracted from navigation behavior, and data from transaction history. In our case, user ratings could be based on Web pages where users navigate following the links on the YourQA answer passages.

In a second phase, the development of dynamic User Models involves the integration between a dialogue interface and the User Modelling component. In Chapter 5, we describe an interactive interface for YourQA, which is based on a chatbot. We believe that the individual dialogue history stored for each user of the interactive version of YourQA can be an even more effective source of information about the user than the history of the standard version. As a matter of fact, the dialogue history can be periodically mined to obtain two types of information: first, the user's reading level, which can be assessed based on the user utterances, i.e. the lexicon used in his/her queries; secondly, the user's interests, which can be inferred by key-phrase extraction conducted on the user's utterances, and in particular on his/her queries.

Moreover, interesting developments involved here include the adaptation of the dialogue management strategy on the basis of the dialogue conducted so far, and using dialogue as a tool for incrementally acknowledging user preferences.

Chatbots and User Modelling: the case of intelligent tutoring Previous work highlighting the contribution of dialogue interfaces to the construction of User Models is represented particularly in the field of intelligent tutoring systems.

In STyLE-OLM (Dimitrova, 2003), dialogue games are used as a means of intelligent tutoring. Here, the User Model incorporates user's beliefs - domain propositions that can be correct, erroneous, and incomplete - and some possible explanations of what might have caused erroneous beliefs based on erroneous reasoning rules, such as misclassification and misattribution. At the end of the interaction, a formal mechanism based on modal logic combines the beliefs in the commitment stores and elicits a resultant User Model.

An interesting case is the system described in Linton *et al.* (2003), where an Intelligent Tutoring System (ITS) is integrated in a Collaborative Learning Environment (CLE). The issue of adding an ITS module to a CLE is that learners are engaged in a deliberative discussion, and it is by examining their contributions to such discussion that the system can infer the current level of knowledge and understanding of each user.

To understand what underlies learners' utterances, a chat tool with sentence openers (i.e. phrases that comprise the first few words of a sentence) is used. To enter an utterance in the chat tool, learners must first select the most suitable sentence opener; they may then input the remainder of the utterance in their own words. The sentence opener reveals the speaker's intention or speech act; these speech acts are correlated with deliberative discussion and with individual understanding of the subject matter.

A special module called student model module observes each learner and estimates his degree of understanding with respect to each topic. Four indicators of learner understanding are examined: first is the pure volume of the learner's contribution to a topic; a second indicator of understanding is response latency, or the amount of thinking time required to generate an utterance. A third indicator of learner understanding, or lack of understanding, is the use of specific speech acts, e.g., "I'm not so sure". Finally, the fourth indicator is the learner's use of the system's whiteboard tool to design portions of the solution to the domain exercises, as with conventional intelligent tutoring systems.

Chapter 5

Interactive Question Answering

Interactive Question Answering (IQA) is the third salient contribution of the present thesis after the complex Question Answering techniques and the design of a User Modelling component for personalization.

Our research on IQA is motivated by the commonly observed behavior of users of information retrieval systems: these often issue queries not as standalone questions but in the context of a wider information need, for instance when researching a specific topic (Hobbs, 2002).

As described in Section 2.1.2, efforts have been carried out in recent editions of TREC-QA in order to approach the issue of context management by the introduction of "targets" in the question sets, i.e. topics about which different related queries are formulated. Such queries can contain elliptic or anaphoric references to their targets without such targets being explicitly mentioned in the query texts, hence some form of reference resolution is required to address them.

However, as also pointed out in De Boni & Manandhar (2003), it can be observed that the current TREC requirements only address one aspect of the complex issue of context management. Indeed, the problem of detecting that one query is related to a topic introduced by a previous one is solved by the presence of an explicit target.

Recently, a new research direction has been proposed, which involves the integration of Question Answering systems with dialogue interfaces in order to encourage and accommodate the submission of multiple related questions and handle the user's requests for clarification in a less artificial setting (Maybury, 2002).

Furthermore, an Interactive Question Answering workshop has been organized within the HLT-NAACL conference (Webb & Strzalkowski, 2006) to set a roadmap for information-seeking dialogue applications of Question Answering (QA). However, as pointed out in

Section 1.2.2, Interactive Question Answering systems are still at an early stage (such as Wizard-of-Oz studies) or applied to closed domains (Small *et al.*, 2003; Jönsson & Merkel, 2003; Kato *et al.*, 2006; Basili *et al.*, 2007).

Structure of this Chapter

In this chapter, we report the design, implementation and evaluation of a dialogue manager to achieve Interactive Question Answering in the open domain. With the extension of the dialogue component, YourQA can be considered a full-fledged Interactive Question Answering system.

Section 5.1 introduces the requirements for modelling Interactive Question Answering, starting from the conversation phenomena occurring in generic dialogue and developing the desiderata for open domain, QA-oriented dialogue itself. Section 5.2 introduces a dialogue model for Interactive QA systems, based on such requirements; Section 5.3 draws on related work to define the dialogue management model for YourQA's dialogue component.

Section 5.4 describes an exploratory Wizard-of-Oz study conducted to confirm our design assumptions. The implementation and evaluation YourQA's interactive interface are described in Sections 5.5 and 5.6. Finally, Section 5.7 draws conclusions on the design and implementation of the Interactive Question Answering version of YourQA.

5.1 Desiderata for Interactive Question Answering

Interactive Question Answering dialogue can be considered as a form of inquiry oriented dialogue, which has been defined by Larsson (2002) as dialogue whose sole purpose is the transfer of information, and in which no dialogue participant assumes non-communicative actions outside the dialogue.

This type of task-oriented dialogue is often referred to using the broader category of information-seeking dialogue (McGlashan *et al.*, 1992; Carlson, 1996; Oppenheimer *et al.*, 2001). Two roles are modelled in inquiry oriented dialogue: inquirer (generally the user), looking for information on a given topic, and expert (generally the system), interpreting the inquirer's needs and providing the required information.

Although we agree with Dahlbaeck *et al.* (1993) that attempting to perfectly emulate human dialogue using a machine is an unrealistic and perhaps unimportant goal, we believe like Karis & Dobroth (1991) that computer-based information-seeking dialogue can benefit greatly in terms of usability from such an attempt. Hence, the design of task-oriented dialogue systems cannot happen without an accurate analysis of the conversational phenomena observed in human-human dialogue.

This section outlines the main features to be represented in information-seeking human-computer dialogue, drawing from the conversation phenomena observed in human dialogue and consequently focusing on the desiderata for Interactive QA dialogue.

Such observations are the basis for outlining a dialogue model suitable for Interactive QA and ultimately for defining the dialogue management model which bridges the gap between the theoretical model and the information-seeking task to perform.

5.1.1 Salient Features of Human Information-Seeking Dialogue

Several extensive reports have been written focusing on different features of human dialogue, e.g. Churcher *et al.* (1997) and Lewin *et al.* (2000). For the purpose of describing information-seeking dialogue, we focused on the following features:

- *Overall structure*: As observed by Sinclair & Coulthard (1975), human dialogues usually have an opening, a body and a closing. Based on actual human conversations, the authors elaborate a hierarchical discourse grammar representing dialogue as a set of *transactions*, composed by *exchanges*, in turn made of *moves*, whose elementary components are *speech acts*.

  In this framework, which has dominated the computational approaches to dialogue to the present, utterances are considered as dialogue acts as they aim at achieving an effect such as obtaining information, planning a trip, driving an unmanned vehicle, etc.

- *Mixed initiative*: initiative refers to who is taking control of the interaction. When one of the interlocutors is a computer system, the literature typically distinguishes between mixed-, user-, and system-initiative (Kitano & Ess-Dykema, 1991).

  In mixed-initiative dialogue, the system must be able to take control in order to confirm given information, clarify the situation, or constrain user responses. The user may take the initiative for most of the dialogue, for instance by introducing information that has not been specifically asked or by changing the subject and therefore the focus of the conversation, as it often happens in human interaction (Hearst *et al.*, 1999).

- *Over-informativeness*: human dialogues often involve more information than required (Churcher *et al.*, 1997). This usually enables dialogue to be more pleasant as the users do not need to ask for all desired pieces of information. For instance, given the question: "Do you have the time?", one would rather reply with the time than with "yes" or "no".

- *Contextual interpretation*: human interaction relies on the conversation participants sharing common notion of context and topic (Grosz & Sidner, 1986). Such common context is used by participants to issue and correctly interpret rhetorical phenomena such as ellipsis, anaphoric and deictic references (such as "he/his" or "this/that"), not to mention more complex phenomena such as reprise and sluicing (Purver *et al.* (2002) provides an extensive review of these).

- *Grounding*: it has been observed that, to prevent or recover from possible misunderstandings, speakers engage in a collaborative, coordinated series of exchanges that instantiate new mutual beliefs and make contributions to the common ground of a conversation. This process is known as grounding (Cahn & Brennan, 1999).

Section 5.1.3 underlines the fundamental issues implied by accounting for such phenomena when modelling human-computer dialogue. Particular attention is given to the context of information-seeking dialogue.

5.1.2 Previous Work on Information-Seeking Dialogue

Information-seeking dialogue systems have been a subject of research for a long time and are now a well-established technology. Typical applications of information-seeking dialogue are timetable enquiries (Sturm *et al.*, 1999), leisure activity search (Rajman *et al.*, 2004; Alexandersson & Becker, 2001) and calculation of service prices (McGlashan *et al.*, 1992).

For example, WAXHOLM (Carlson, 1996) is a spoken dialogue system giving information on boat traffic in the Stockholm archipelago. Dialogue management in WAXHOLM is based on grammar rules and lexical semantic features; topic selection is accomplished based on probabilities calculated from user initiatives.

SmartKom (Alexandersson & Becker, 2001) is a multimodal dialogue system that combines speech, gesture and mimics input and output. One of its applications is an intelligent telephone booth with which tourists can book tickets, get information about local activities, attractions etc.

Below, we discuss three more examples of information-seeking dialogue; these contribute to our analysis of the issues of dialogue modelling, reported in the following section.

SUNDIAL SUNDIAL (McGlashan *et al.*, 1992) aimed at building real-time integrated computer systems able to maintain cooperative dialogues with users over telephone lines and in several languages. The main objectives of SUNDIAL were to be able to deal with the dialogue phenomena as described in Section 5.1.1, to be capable to predict about the user response and eventually to build a generic, application and language independent system.

Here, the system architecture is articulated in three main components: a linguistic interpreter, interpreting speech input and converting it into text; a dialogue manager, which provides an interpretation of user utterances within the dialogue context, and plans a representation for the following system utterances; finally, an utterance generator, made of a message generation module and a speech synthesis module.

WHEELS WHEELS (Goddeau *et al.*, 1996) provides a spoken language interface to a database of approximately 5,000 classified advertisements for used automobiles. The task is to assist the user in narrowing the list of ads to a small number, which can then be read or faxed to the user.

WHEELS is implemented using the infrastructure of the GALAXY system (Seneff *et al.*, 1998), a distributed framework for organizing conversational systems to optimize resource-sharing and extensibility. Using this framework, the WHEELS domain is organized as a server process: the input to the server is a semantic frame representation of the input utterance produced by the speech recognizer and language analyser which run in separate processes. The output of the server includes a spoken response and an optional tabular representation containing information about cars of interest.

ARISE ARISE (Automatic Railway Information System for Europe) is a multilingual spoken dialogue system providing train timetable information over the phone (Sturm *et al.*, 1999). The ARISE system is based on a modular architecture containing six components: a continuous speech recogniser, a natural language understanding component, a mixed-initiative dialogue manager, a database retrieval component, a generation component, and a synthesizer.

A two-level dialogue strategy is employed by the Dialogue Manager (DM): a mixed initiative approach, where the user can provide information at any time, is combined with a system-directed approach when the system detects a problem during the dialogue. A mix of implicit and explicit confirmation is used, based on how confident the system is as to whether an item has been correctly understood.

The DM first fills in a semantic frame representing the task: this is done by interpreting the utterance in the context of the ongoing dialogue, common sense knowledge, task domain knowledge, and dialogue history. The DM then either prompts for missing information or sends a database query. Before the query is sent off, interpretative and history management rules are applied to determine whether new information is contained in the query and whether this information contradicts information given before. If so, it can either keep the original information, replace it with the new one or engage in a confirmation sub-dialogue. Moreover, when errors occur, these exceptions are handled in an explicit way, which means that the DM provides the user with clear hints on answering possibilities.

5.1.3 Issues in Modelling Information-Seeking Dialogue

Based on the analysis of previous information-seeking applications and on the salient features of information-seeking dialogue, we outline below the main issues in modelling such kind of dialogue, with an eye on the relevance of these issues to Interactive Question Answering.

Ellipsis

Ellipsis is an omission of part of the sentence, resulting in a sentence with no verbal phrase (or "fragment"). For instance, in a flight reservation system, an ellipsis example could be:

- *System*: There is a flight that leaves at 11 AM and arrives at 2 PM.
- *User*: And at 1 PM?

Ellipsis is an issue that also affects Question Answering dialogue. Consider the exchange:

- *User*: When was Shakespeare born?
- *System*: In 1564.
- *User*: Where?

For the interpretation of an elliptic sentence, the conversational context must be taken into account (and modelled efficiently). An example of approach to ellipsis resolution is SHARDS (Ginzburg & Gregory, 2001), which provides a grammatical framework for the interpretation of elliptic fragments based on a version of HPSG. SHARDS provides a procedure for computing the content values of such fragments based on contextual information contained in a discourse record.

Anaphoric references

An anaphor is an abbreviated linguistic form whose full meaning can only be recovered by reference to the context; the entity to which the phenomenon of anaphora refers is called referent, and the mention of the entity to which an anaphor refers is called the antecedent. The following exchange contains an example of anaphoric reference:

- *User*: When was Shakespeare born?
- *System*: In 1564.
- *User*: Whom did *he* marry?

where "he" is the referent and "Shakespeare" is the antecedent. Common forms of anaphora are third person pronoun and third person adjective anaphora, where such pronouns and adjectives as "he/she/it/they" or "his/her/its/their" are used in place of the entities to which they refer. Such entities can be single or compound nouns (such as William Shakespeare), or even phrases (*"The Taming of the Shrew"*).

Reference resolution is a common problem in natural language tasks (Mitkov, 2002); the difficulty of the problem is related to the number of anaphoric expressions encountered, which in turn is related to the length of the texts to be processed.

Among the strategies for anaphora resolution there exist both rule-based (Lappin & Leass, 1994) and statistical (Ge *et al.*, 1998) approaches. In an information-seeking dialogue application, where brief utterances are generally exchanged, the difficulty of anaphora resolution is less than in other information extraction tasks such as document summarization (Steinberger *et al.*, 2005). For this reason and because of the need for real-time resolution, rule-based approaches seem to offer the most appropriate solution in a real-time dialogue application.

Grounding and Clarification

While formal theories of dialogue assume complete and flawless understanding between speakers, in practice there exists a need for grounding (Cahn & Brennan, 1999). Such need arises frequently in task-oriented dialogue: the system should be able to understand when it makes sense to ask the user for explicit confirmation of a given utterance, when implicit confirmation should be sought to avoid lack of fluidity in dialogue (e.g. "Where would you like to search for a *Chinese* restaurant?"); finally when no confirmation at all should be asked (e.g. "Did you say yes?" would be a strange question from the system).

A typical Question Answering scenario where requests for confirmation should be modelled is upon resolution of anaphora. Consider the exchange:

- *User*: When did Bill Clinton meet Yasser Arafat in Camp David?
- *System*: In 2000.
- *User*: How old was *he*?

The user's question contains two Named Entities of type "person": hence, resolving the *he* pronoun can yield two candidate referents, i.e. Bill Clinton and Yasser Arafat. Having resolved the anaphoric reference, the system should decide whether to continue the interaction by tacitly assuming that the user agrees with the replacement it has opted for (possibly "*he* = Bill Clinton") or to issue a grounding utterance ("Do you mean how old was *Bill Clinton*?") as a confirmation.

On the other hand, users can request clarification upon system utterances. Purver *et al.* (2002) conducted a study to categorize different kinds of clarification requests and estimate their frequencies of occurrence based on texts in the British National Corpus.

Their salient findings were that clarification requests occurred in about 4% of the sentences in the corpus and that most of these were conventional reprise fragments. These are elliptical sentences reporting part of the interlocutor's utterance, such as a):"We will be 23 at the party" / b): "23?") or conventional requests (such as: a) . . . / b): "Sorry?".

While clarification requests can be interpreted in various ways, the "causal" reading (corresponding to e.g. "Do you mean X?") was found to be the most frequent one, hence the most useful to be encoded in the system when recovering from a clarification session.

Turn-taking

According to conversation analysis, the nature by which a conversation is done is through turns, or pairs of utterances often called *adjacency pairs* (Schegloff & Sacks, 1973; Sacks *et al.*, 1974). Information-seeking dialogue can thus be modelled as a sequence of ⟨*request, response*⟩ pairs.

In natural dialogue, there is very little overlap between when one participant speaks and when the other(s) do: the resulting discourse is very fluid. To ensure such fluidity, the computer's turn and the human's turn must be clearly determined in a dialogue system. While this is an important issue in the case of spoken dialogue, where a synthesizer must output a reply to the user's utterance, it does not appear to be very relevant to textual dialogue systems, since when the system is ready to reply, producing its reply takes only an instant and hence it is virtually impossible that the user inputs text at the same time as the system.

Adjacency pairs and insertion sequences

Adjacency pairs are intuitive structures having the form question-answer, greeting-greeting, request-acceptance. They are usually employed as a paradigm to partition the discourse in well-distinguished portions, following the theory of conversational turn-taking exposed above (Sacks *et al.*, 1974).

Moreover, natural conversation exhibits insertion sequences (Churcher *et al.*, 1997), where pairs are embedded one into another, for instance when the system asks a question and the user issues a request for clarification. These must not be interpreted by the system as responses to the previous system utterance but as a signal to start a clarification session. Consider the following ticket reservation example:

- *System*: I found a return ticket for £39. Confirm purchase?
- *User*: Does it require a discount card?

Handling this type of conversation is possible when the recent dialogue context can be memorized; a typical dialogue context implementation is in the form of a stack (Grosz & Sidner, 1986), where recent utterances can be pushed and popped according to the current dialogue move.

Conversational fillers

In human dialogue, phrases like "A-ha" or "exactly" are prompted in order to fill the pauses of the conversation and acknowledge the interlocutors, for instance upon receipt of information or for grounding purposes.

Although a minor issue in implementing information-seeking dialogue, recognizing conversational fillers requires dialogue management systems with a wide-coverage of user utterances, as does including them into the system's response.

5.1.4 Summary of Desiderata for Interactive Question Answering

Based on the phenomena and issues observed above, we summarize our desiderata for Interactive Question Answering as follows:

- *Context maintenance*: maintaining the conversation context and topic to allow the correct interpretation of the user's utterances (in particular of follow-up questions requests for clarification);
- *Utterance understanding:* this includes follow-up/clarification detection and the handling of phenomena like ellipsis and anaphoric expressions;

- *Mixed initiative*: users should be able to take the initiative during the conversation, for example by issuing clarification requests and quitting the conversation when they desire to do so;
- *Follow-up proposal*: an IQA system should be able to encourage users to provide feedback about satisfaction with the answers received and also to keep the conversation with the user active until he/she has fulfilled their information needs;
- *Natural interaction:* a wide coverage of the user's utterances is required to enable smooth conversation, as well as the generation of a wide range of utterances to encourage users to keep the conversation active.

5.2 A Dialogue Model for Interactive Question Answering

The phenomena and issues observed in Section 5.1 led us to the design of a representative dialogue scenario for YourQA's prospective Interactive Question Answering task. Such scenario is discussed in Section 5.2.1.

5.2.1 Dialogue Scenario

In YourQA's dialogue scenario, a typical QA session consists of the interaction reported in Algorithm 6.

This scenario basically consists of a sequence of adjacency pairs of the form *greeting-greeting, question-answer.* Nested dialogue pairs such as the system's request for confirmation and the user's reply or acknowledgment can be found within the question-answer pairs. Moreover, at any time the user can issue a request for clarification when he/she does not understand the system's utterance, in which case the system replies with a clarification.

Having outlined the ideal dialogue scenario for interactive QA, the issue now consists in implementing such conversation in an actual dialogue system. Sections 5.2.2 and 5.2.3 introduce the main conversation acts, or dialogue moves, which are required by such scenario in order to elaborate a suitable dialogue management model.

Algorithm 6 Dialogue scenario

1. An optional reciprocal greeting;
2. A direct question q from the user;
3. q is analyzed to detect whether it is related to previous questions or not;
4. (a) If q is unrelated to the preceding questions, it is submitted to the QA component;
   (b) If q is related to the preceding questions (follow-up question), and is elliptic, the system uses the previous questions to complete q with the missing keywords and submits a revised question q' to the QA component;
   (c) If q is a follow-up question and is anaphoric, i.e. contains references to entities in the previous questions, the system tries to create a revised question q'' where such references are replaced by their corresponding entities, then checks whether the user actually means q'';
   If the user agrees, query q'' is issued to the QA component. Otherwise, the system asks the user to reformulate his/her utterance until finding a question which can be submitted to the QA component;
5. As soon as the QA component results are available, an answer a is provided;
6. The system enquires whether the user is interested in a follow-up session; if this is the case, the user can enter a query again. Else, the system acknowledges;
7. Whenever the user wants to terminate the interaction, a final greeting is exchanged.

5.2.2 Towards a Dialogue Taxonomy

Several theories of discourse structure exist in the literature and have led to different models of dialogue. Among these, a widely used representation of dialogue consists in the speech act theory, introduced by Austin (1962), which focuses on the communicative actions (*speech acts*) performed when a participants speaks.

Austin (1962) distinguished between *constative* utterances and *performatives*, such as greetings or apologies, which perform an action rather than expressing something about the state of the world. According to Austin, performatives can be characterized in terms of three kinds of verbal *acts*: *locution*, which relates to the literal use of an utterance, *illocution*, relating to what the speaker intends to perform, and *perlocution*, relating to what is achieved.

Cohen & Perrault (1979) contributed to the speech act theory the notion of illocutionary speech acts as plan operators that affect the beliefs of the speaker and hearer. They introduced a planning system (i.e., a formal language for describing states and events in the world, including people's beliefs), and a definition of how speech acts change the state of the world and speakers' separate beliefs and mutual beliefs.

Allen & Perrault (1980) reinterpreted Cohen's speech act definitions in terms of STRIPS plan operators (Fikes & Nilsson, 1971): For example, the definition of the illocutionary act of informing is:

INFORM (Speaker, Hearer, P)
`Preconditions:` Speaker know P (P is true & speaker believe P) `Effects:` Hearer know P `Body:` Hearer believe Speaker want(hearer know P)

hence the operators are represented as consisting of a header,a set of preconditions and effects, and finally a body which lists actions and goals to be achieved for the action to be performed. Litman and Allen's further work (see Litman & Allen, 1990) extended the above work to better account for various dialogue phenomena, including clarification subdialogues.

Based on speech act theory, several annotation schemes of speech acts – also called *dialogue moves*– have been developed for task-oriented dialogue:

- The DAMSL generic annotation scheme (Core & Allen, 1997), based on three layers of dialogue functions:

1. forward communicative functions, proposing something to the interlocutor ("directive", "offer", "commit", ...),
2. backward communicative functions, relating to previous utterances ("accept", "acknowledge", "answer", "re-phrase", ...),
3. utterance features ("task management", "conventional form", ...);

- The HCRC annotation scheme (Kowtko & Isard, 1993; Anderson & Bader, 1991). This was designed for a cooperative application, where the goal was that an instruction giver (having a path on his/her map) would help an instruction follower (having the map only) to reconstruct a path on a given map.
  The scheme is based on 12 main dialogue moves, such as "instruct", "clarify", "query", "acknowledge", "reply", and "check".

- The LINDA/LINLIN scheme (Dahlbaeck & Jonsson, 1998), developed for annotating information retrieval dialogues. The taxonomy involves:

  1. initiative moves such as "question" and "update";
  2. response moves such as "answer";
  3. dialogue management moves, such as"greeting", "farewell" and "discourse continuation".

- The VERBMOBIL scheme (Alexandersson *et al.*, 1997) was developed for the translation of spontaneous speech-to-face dialogues. The annotation scheme contains 45 different illocutionary acts grouped in three main sets:

  1. the acts aiming at dialogue control ("greet", "bye", "thank", ...),
  2. the acts aiming at task management ("init", "defer", "close"),
  3. the acts aiming at task promotion ("request", "suggest", "inform", "feedback", ...).

- The TRAINS conversation act typology (Traum, 1996) distinguishes between four types:

  1. turn-taking acts ("take-turn", "release-turn", ...),
  2. grounding acts ("ack", "repair", ...),
  3. core speech acts ("inform", "request", ...),
  4. argumentation acts ("clarify", ...).

While the level of granularity and the range of moves of most of the schemes above was determined by the application of the dialogue system, as pointed out in Larsson (1998) there appears to be a number of generic common dialogue moves, which include:

- Core speech acts (TRAINS): these include initiatives and responses;
- Conventional (DAMSL) or discourse management (LINLIN) moves: opening, continuation, closing, apologizing;
- Feedback (VERBMOBIL) or grounding (TRAINS) moves: to elicit and provide feedback;
- Turn-taking moves (TRAINS), relating to sub-utterance level.

5.2.3 YourQA's Dialogue Moves

Inspired by the previous general observations about annotation schemes, and aiming at implementing the scenario in Algorithm 6, we developed a set of dialogue moves to represent interactive QA. These are listed in Tables 5.1 and 5.2.

Table 5.1: User dialogue moves

User move	Description
greet	conversation opening
ack	user acknowledgment of system's utterance
ask(q)	user asks question q
usrReqClarif	user's clarification request
quit	conversation closing

Table 5.2: System dialogue moves

System move	Description
greet	conversation opening
ack	system acknowledgment of user's utterance
sysReqClarif	system's clarification request
ground(q)	grounding move concerning question q
answer(a)	system answers with answer a
follow-up	system's proposal to continue the conversation
quit	conversation closing

In our annotation, the core speech acts are represented by the *ask* and *answer* moves. The *ask* move has as parameter the user's question, as this is to be processed in order to identify whether it is a follow-up or multiple question and then must be submitted to the underlying QA system.

The *answer* move is parameterized by the contents of the answer returned by the QA system, which must be submitted to the dialogue interface which visualizes it to the user.

Amongst discourse management moves, we find *greet*, *quit* in both the user and system moves, and *follow-up* proposal from the system. The user feedback move is *usrReqClarif*, mirrored by the system's *sysReqClarif* move. Currently, these two moves are not parameterized as the request for reformulation is supposed to be regardless of the content of the previous utterances. Hence, the system will utter a *sysReqClarif* move without explicitly mentioning the former user utterance (using template expressions such as "Sorry, I don't understand what you just said. Can you please reformulate?"). Similarly, the user is expected to use generic clarification request formulae, such as "I don't understand", rather than "What do you mean by ... ?".

A common feedback move to both user and system is *ack*, while the *ground* and *clarify* moves are only in the system's range. The *ground* move takes q, the current query as resolved by the system as a parameter and seeks for confirmation from the user that the latter actually intends to ask q.

Finally, we do not annotate the scenario above with turn-taking moves as these are at a sub-utterance level.

Following such dialogue move taxonomies, the dialogue scenario in Algorithm 6 can be annotated as in Algorithm 7.

Given the dialogue model formalized in this section, we now discuss the choice of a dialogue manager to implement such moves.

5.3 A Dialogue Manager Model for Interactive Question Answering

The dialogue manager is the implementation strategy bridging the gap between the theoretical dialogue model of Interactive Question Answering outlined in Section 5.2 and the actual task of Interactive QA.

Broadly speaking, dialogue management models are attached to two categories: on the one side, pattern-based approaches, on the other plan-based approaches (Cohen, 1996;

Algorithm 7 Dialogue scenario annotated with dialogue moves

1. A reciprocal greeting (*greet* move);
2. A direct question q from the user (*ask(q)* move);
3. q is analyzed to detect whether it is related to previous questions or not;
4. (a) If q is unrelated to the preceding questions, it is submitted to the QA component;
   (b) If q is related to the preceding questions (follow-up question), and is elliptic, i.e. contains no verb (*"Why?"*), the system uses the previous questions to complete q with the missing keywords and submits a revised question q' to the QA component;
   (c) If q is a follow-up question and is anaphoric, i.e. contains references to entities in the previous questions, the system tries to create a revised question q'' where such references are replaced by their corresponding entities, then checks whether the user actually means q'' (move *ground(q")*);
   If the user agrees, query q'' is issued to the QA component. Otherwise, the system asks the user to reformulate his/her utterance (move *sysReqClarif*) until finding a question which can be submitted to the QA component;
5. As soon as the QA component results are available, an answer a is provided (*answer(a)* move);
6. The system enquires whether the user is interested in a *follow-up* session; if this is the case, the user can enter a query (*ask* move) again. Else, the system acknowledges (*ack*);
7. Whenever the user wants to terminate the interaction, a final greeting is exchanged (*quit* move).

At any time the user can issue a request for clarification (*usrReqClarif*) in case the system's utterance is not understood.

Xu *et al.*, 2002). The following sections report a brief critical overview of these, underlining their issues and advantages when addressing interactive QA.

5.3.1 Pattern Based Approaches: Dialogue Grammars and Finite-State Approaches

Finite-State (FS) approaches provide the simplest method for implementing information-seeking dialogue management. Here, the dialogue manager is represented as a Finite-State machine, where each state models a separate phase of the conversation, and each dialogue move encodes a transition to a subsequent state (Sutton, 1998). Hence, from the perspective of a state machine, speech acts become state transition labels.

When the state machine variant of a dialogue grammar is used as a control mechanism for a dialogue system, the system proceeds as follows:

1. first, it recognizes the user's speech act from the utterance,
2. then, it makes the appropriate transition,
3. finally, it chooses one of the outgoing arcs to determine the appropriate response to supply.

The advantage of state-transition graphs is mainly that users respond in a very predictable way, as the system has the initiative for most of the time. However, an issue with FS models is that they allow very limited freedom in the range of user utterances: since each dialogue move must be pre-encoded in the models, there is a scalability issue when addressing open domain dialogue.

Form-filling approaches to dialogue management To compensate the lack of flexibility of FS approaches, a number of systems have taken *form-filling* (or *frame-based*) approaches, based on the structure of the topics in the discourse. In this framework, the details of the topic are slots to fill (as in a relational database), and slots in turn can become the topics of lower-level tables and have attributes to themselves. Examples of such systems systems are Hulstijn's theater booking system (Hulstijn, 1996), *InfoVox*, a restaurant search system (Rajman *et al.*, 2004), and IM2.MDM, a multimodal meeting database (Bui & Rajman, 2004).

The main advantage of form-filling approaches with respect to simple finite-state ones is that users can supply more information than requested (for instance, relating to more than one task) in the same turn, thus removing the system's need to generate some of the upcoming utterances.

A few general issues with FS models and dialogue grammars still remain open. First, dialogue grammars require that the communicative action(s) being performed by the speaker in issuing an utterance be identified. In the past, this has been a difficult problem for people and machines, for which prior solutions have required plan recognition (Cohen, 1996). Moreover, these models typically assume that only one state results from a transition; however, utterances are multifunctional. An utterance can be, for example, both a rejection and an assertion, and a speaker may expect the response to address more than one interpretation.

Plan-based approaches, offering a more sophisticated strategy for dialogue management, are discussed below.

5.3.2 Plan Based Approaches and the Information State

Plan-based theories of communicative action and dialogue (Cohen, 1996) assume that the speaker's speech acts are part of a plan, and the listener's job is to uncover and respond appropriately to the underlying plan rather than just to the utterance. VERBMOBIL and TRAINS are examples of plan-based approaches, as briefly described below.

VERBMOBIL The VERBMOBIL project (Alexandersson *et al.*, 1997) is a speech-to-speech translation system made of over 40 modules for both speech and linguistic processing, although we will only consider the dialogue module. The system mediates a dialogue between two persons, with no constraints except to use the ca. 2500 words the system recognizes. In VERBMOBIL, dialogue structure is articulated into *turns* and *utterances*, from which dialogue acts are extracted.

The most interesting features of the VERBMOBIL system are the inference mechanisms, integrating data in representations of different contexts of the dialogue. Inferences can be of two types:

- *Plan based* inferences: in VERBMOBIL, dialogue is organized according to three phases: an opening phase, a negotiation phase and a closing phase. The task of determining the current phase is attributed to the plan recognizer, which builds a tree-like structure called the intentional structure and performs inference to "guess" the user's plan.

- *Thematic* inferences: the thematic structure is used to solve anaphoric expressions like "next" or "this"; it also checks if time expressions are correctly recognized.

A different type of inference is used to generate predictions about what is said next. Dialogue act predictions are based solely on conditional frequencies of dialogue act sequences as observed on the annotated VERBMOBIL corpus.

TRAINS: A simplified approach to planning The TRAINS project (Allen *et al.*, 2000) simulates a route planning dialogue system capable of planning relatively simple trips. The user is given a map on a screen showing cities, routes and the locations of a given set of trains, and a verbal description of the destination; the goal is to find the optimal path.

The aspects of the TRAINS world to be modelled require a few predicates, and individual beliefs of the hearer are not modelled. Instead, there is a shared set of beliefs that includes all the information that has been conveyed in the dialogue and private information that only the system currently knows. Instead of different individual plans, there only is a single shared plan, even though this does not prevent the system from needing to know the user's intentions lying under his/her utterances.

As the system also needs a plan for its own utterances and to determine the routes to propose to the user, this is done by domain-specific reasoners.

The Information State approach

Within plan-based approaches, a well-established dialogue management model is the Information State (IS) approach, inspired by the Dialogue Game Board theory in Ginzburg (1996). Here the conversation is centered on the notion of Information State (IS), which comprises the topics under discussion and common ground in the conversation and is continually queried and updated by rules fired by participants' dialogue moves.

Ginzburg & Sag (2004) describes the notion of context in information-state dialogue as follows. Dialogue participants have an individual Dialogue Game Board (DGB), the structure of which involves three main components:

1. FACTS: set of commonly agreed upon facts;

2. QUD: questions under discussion at a given time;

3. LATEST-MOVE (L-M): the latest dialogue move.

When an utterance occurs in dialogue, while speakers incorporate their own utterances right away in the DGB, hearers put the utterance in QUD-MAX, a structure that evaluates which is the question under discussion with maximum priority. The L-M must be updated after each utterance and the old L-M must be grounded appropriately according to the procedure in Algorithm 8.

Algorithm 8 Context management in the Information State (Ginzburg & Sag, 2004)

If an utterance u by user A has occurred,

- Try to understand u, according to DGB;
- If u is not understood:
  1. Set u aside;
  2. make $content(u, A, \mu(u))?$ QUD-maximal, where $content(u, A, \mu(u))?$ is the content question in A's utterance u whose meaning is $\mu(u)$
  3. provide a $content(u, A, \mu(u))?$-specific utterance, i.e. an utterance attempting to clarify u.

The IS theory, implemented for instance in the TRINDI project (Larsson & Traum, 2000), has been applied to a range of closed-domain dialogue systems, including travel information, route planning (Bos *et al.*, 2003; Larsson *et al.*, 2000) and command-and-control interfaces (Lemon *et al.*, 2001). However, there does not appear to be an implementation of the IS approach that is suitable for dialogue in the open domain.

5.3.3 Discussion

Although it provides a powerful formalism, the IS infrastructure was too complex for our Interactive QA application. We believe that the IS approach is primarily suited to applications requiring a planning component such as in closed-domain dialogue systems and to a lesser extent in an open-domain QA dialogue system.

Moreoever, the Interactive QA task is an information-seeking one where transactions are generally well-structured and not too complex to detect (see also Jönsson (1993)). Hence, this shortcoming of pattern-based dialogue models does not appear to greatly impact on the type of dialogue we are addressing.

Finally, as pointed out in Allen *et al.* (2000), there are a number of problems in using plan-based approaches in actual systems:

- The *knowledge representation* problem: beliefs, intentions and plans are hard to represent;
- The *knowledge engineering* problem: it is problematic to define the information required to cover the full range of situations and possible utterances;

- The *computational complexity*: planning algorithms are usually too complex to respond in real time;
- The *noisy input* problem: errors in parsing and interpretation can affect the quality of interaction.

These observations suggest that, on the one hand, several reasons make plan-based approaches appear as an unpractical solution. However, while FS models seem to be an eligible alternative for interactive QA, they also come with a number of limitations among which the limited management of context and the lack of coverage of the user utterances.

The ideal dialogue management module for Interactive QA seems to lie somewhere in between the FS and IS models, as proposed in Section 5.3.4.

5.3.4 Chatbot-based Interactive Question Answering

As an alternative to the FS and IS models, we studied conversational agents based on AIML interpreters. AIML (Artificial Intelligence Markup Language) was designed for the creation of conversational robots ("chatbots") such as ALICE[1]. These are based on the pattern matching technique, which consists in matching the last user utterance against a range of dialogue patterns known to the system. A coherent answer is created by following a range of "template" responses associated with such patterns.

In AIML, ⟨*pattern,template*⟩ pairs form "categories", an example of which is the following ALICE greeting category:

Category 9 Example of an ALICE category

```
<category>
<pattern>WHO ARE YOU</pattern>
<template>I am ALICE, nice to meet you!</template>
</category>
```

As its primary application is small talk, chatbot dialogue appears more natural than in FS and IS systems. Moreover, since chatbots support a limited notion of context, they offer the means to handle follow-up recognition and other dialogue phenomena not easily covered using standard FS models. Below, we outline the features of information-seeking dialogue that can be handled by such approach.

[1] http://www.alicebot.org/

Advantages of Chatbot Dialogue

Chatbot dialogue seems particularly well suited to handle the dialogue phenomena introduced in Section 5.1.1, as discussed in detail below:

- *Overall structure*: As chatbot dialogue is articulated in ⟨*pattern,template*⟩ pairs, it seems ideal to model the conversation scenario proposed in Section 5.2.1, which is annotated using adjacency pairs.

  Moreover, in such scenario the conversation initiative as well as the decision when to issue a request for information is left entirely at the user's command: this is mirrored by the user-driven structure of chatbot conversation, where any system utterance will occur only in response to an utterance from the user. However, as pointed out below, there is also room for system initiative in clarifying the current conversation as required in some information-seeking dialogue situations.

- *Mixed initiative*: as mentioned earlier, the system must be able to constrain user responses and to take control at times during the conversation in order to confirm given information or clarify the situation.

  The patterns used by a chatbot system can be oriented to Question Answering conversation so that the user is encouraged to formulate information requests rather than engage in small talk, as in the following category:

  ```
  <category>
  <pattern>HELLO *</pattern>
  <template>Hello, what is your question?</template>
  <category>
  ```

  where a user utterance starting with "Hello" triggers a template inviting the user to formulate a specific question.

  On the other hand, the user may take the initiative for most of the dialogue, for instance by ignoring the system's requests for feedback and directly formulating a follow-up question. The following interaction is possible:

  - *User*: "What is a thermometer?"
  - *System*: "I found the following answers: ...
    Are you happy with these answers?"
  - *User*: "How does it measure the temperature?"

  This triggers an ⟨*ask,answer*⟩ adjacency pair with a new conversation focus.

- *Contextual interpretation*: Contextual interpretation of the user's utterances can be handled by a follow-up question resolution module designed to take care of ellipsis and anaphoric references in questions, such as the one described in Section 5.5.1. Section 5.5.2 shows that such resolution strategy can be implemented by coding a set of AIML categories allowing to invoke an external follow-up question detection and resolution module and to integrate its output in the conversation.

- *Grounding and clarification*: The management of misunderstandings is possible thanks to the *usrReqClarif* and *sysReqClarif* moves.

  The *sysReqClarif* move is fired when the current user utterance is not recognized as a question according to the set of question patterns known to the system. For example, in the following category:
  ```
  <category>
  <pattern>I NEED *</pattern>
  <template>
  Is that a question you want me to look up?
  </template>
  <category>
  ```
  when the user formulates a question starting with "I need" (e.g. "I need information about Shakespeare"), the system prompts for a clarification. If the user confirms that his/her utterance is a question, the system will proceed to clarify it and answer it; otherwise, it will acknowledge the utterance.

  Symmetrically, as the structure of conversation is user-driven, the user can enter a request for clarification of the system's latest utterance (move *usrReqClarif*) at any time he/she finds the latter unclear.

- *Over-informativeness*: AIML makes it is possible to code categories enabling the user to respond to the system by providing more information than a simple acknowledgment. For instance, the following exchange is possible:

  - *User:* "What is a candy thermometer?"
  - *System:* ...
  - *User:* "How does *it* measure the temperature?"
  - *System:* "Do you mean how does *a thermometer* measure the temperature?"
  - *User:* "No, how does *a candy thermometer* measure the temperature?".

Here, the system incorrectly resolves the referent "it" using the antecedent "thermometer", which is incomplete; the user addresses the system's request for clarification and rectifies the system's incorrect assumption at the same time.

Previous Work on Chat-based Dialogue

Chatbots have rarely been used for information-seeking dialogue in the literature: starting from the CONVERSE system, which was not task-oriented, we illustrate some approaches whose application resembles more closely our open-domain QA task.

CONVERSE CONVERSE (Levy *et al.*, 1997) was a machine dialogue system funded by Intelligent Research of London which won the Loebner prize[2] in 1997. It covered about 80 topics, which were appropriate to its persona as a young female New York-based journalist.

Among the resources of CONVERSE were the WordNet lexical database (Miller, 1995) and a dictionary of proper names. The chatbot could store the personal information it elicited from a user and incorporate it later into the conversation. CONVERSE's control structure was a simple blackboard system in which the generation decisions were made numerically based on weights assigned by the closeness of fit of the input to the expected input.

The system had only limited recovery mechanisms if it was not able to find a topic relevant to the input, and relied on seizing control of the conversational initiative as much as it could. Since this system models only small talk conversation, the dialogue had no application goals of any kind.

Information-seeking chat Stede & Schlangen (2004) describes a type of dialogue called "information-seeking chat", applied to a tourist information application for the city of Potsdam. This genre is distinguished from standard information-seeking dialogue by its more exploratory and less task-oriented nature; hence, while it is still more structured than general free conversation, it also uses much more mixed-initiative dialogue than traditional task-oriented dialogue.

The information-seeking chat application is deployed in a closed domain (tourist information): a declarative domain model called topic map (similar to an ontology) serves

[2]The Loebner Prize is awarded to the most convincingly human system as assessed in a test where six computer programs, four human subjects and ten human interrogators are involved. The participants (humans and machines) engage in conversation and interrogators must detect which of them are actual humans.

both as a representation of the domain knowledge and as a repository for the discourse history.

The system relies on a simple taxonomy of dialogue moves and a dialogue management (DM) strategy, where the main task of the dialogue manager is to guide the user through the pre-defined topic map. In the dialogue model, Information State update is realized by updating the weights of elements in the domain that are then used to propose elements of such domain to the user.

REQUIRE: Closed-domain QA Another recent example of chatbot deployment for a QA task is REQUIRE (Basili *et al.*, 2007), an interactive system for domain specific dialogue. The framework is demonstrated within a sexual health information service in Italian.

REQUIRE is a domain-driven dialogue system, whose aim is to support the specific tasks evoked by Interactive Question Answering scenarios. Its dialogue management model is based on a Finite State architecture; its speech acts (or dialogue moves) include a *clarification* act where the Dialogue Manager asks the user about a topic as a "request of information about something", an *explanation* act where the system helps the user to disambiguate previously introduced terms upon user request, and a *disambiguation* act which is a particular case of explanation where the user's request for explanation is unclear.

Since the underlying IR engine returns several candidate answers, a planner must decide which interactions are useful to focus on the subset of relevant ones. Hence the transitions among states in the FS machine are determined not only by the speech acts detected using AIML patterns, but also by the outcome of the planner which designs an appropriate sequence of interactions to reach the correct response(s) while at the same time minimizing the number of clarifications asked to the user.

RITEL: Open-domain QA An example of chat interface to an open-domain Question Answering system is the RITEL project (Galibert *et al.*, 2005). RITEL aims at integrating a spoken language dialogue system and an open-domain Question Answering system to allow a human to ask general questions and refine the search interactively.

In Galibert *et al.* (2005), the dialogue interface of RITEL is described as an ELIZA variation. However, the RITEL project currently seems at an early stage and no thorough description is available about its dialogue management model. Effort has mostly been focused on collecting an interaction corpus and the system only delivers answers to few questions; moreover, the QA knowledge base seems to be a closed-domain database.

To assess the utility of a chatbot-based dialogue manager in an open-domain QA application, we conducted an exploratory Wizard of Oz experiment, described in Section 5.4.

5.4 A Wizard-of-Oz Experiment

Wizard-of-Oz (WOz) experiments are usually deployed for natural language systems to obtain initial data when a full-fledged prototype is not yet available (Dahlbaeck *et al.*, 1993; Bertomeu *et al.*, 2006). They consist in "hiding" a human operator behind a computer interface to simulate conversation with the user, who believes to be interacting with a fully automated prototype.

5.4.1 Design

In addition to the general assumption that a chatbot would be sufficient to successfully conduct a QA conversation, we intended to explore whether a number of further assumptions were founded in the course of our experiment:

- First, users would use the system to obtain information, thus most of their utterances would be questions or information requests.
- Then, users would easily cope with the system's requests to rephrase their utterances should the system fail to understand their previous utterances.
- Finally, as the YourQA passage answer format (Section 2.5) provides more information than explicitly requested, which has been shown an effective way to reduce the number of user clarification requests (Kato *et al.*, 2006; Hickl & Harabagiu, 2006), such requests would be few.

Task Design

We designed six tasks, to be proposed in groups of two to six or more subjects so that each would be performed by at least two different users. The tasks reflected the intended typical usage of the system and were the following:

T_1 "Find out who painted Guernica and ask the system for more information about the artist".

T_2 "Find out when Jane Austen was born".

T_3 "Ask what are Shakespeare's most famous plays".

T_4 "Look for the definition of open source software".

T_5 "Ask about the price of the iPod Shuffle and then about the PowerBook G4".

T_6 "Find out what types of cloud there are".

Users were invited to test the supposedly completed prototype by interacting with an instant messaging platform, which they were told to be the system interface.

Role of the Wizard

Since our hypothesis was that a conversational agent is sufficient to handle Question Answering, a set of AIML categories was created to represent the range of utterances and conversational situations handled by a chatbot.

An example of these is presented in Category 10, where the user's utterances matching the regular expression "`I * question`" are addressed with a suggestion to formulate a question using a randomly chosen phrasing.

Category 10 A greeting category used in YourQA

```
<category>
<pattern>I * QUESTION </pattern>
<template>
<random>
<li>Let's see if I can answer you.  </li>
<li>Fire away!  </li>
<li>What is your question?  </li>
<li>Go ahead!  </li>
</random>
</template>
</category>
```

The role of the wizard was to choose the appropriate pattern within the ones in the available set of categories, and type the corresponding template into the chat interface. If none of the categories appeared appropriate to handle the situation at hand, the wizard would create one to keep the conversation alive and preserve the illusion of interacting with a machine.

Since the capabilities of the system at the time only permitted a slow real-time response, answers in HTML format were collected for the scenarios above to be proposed to the user via links in the chat interface. For instance, the wizard would answer a question by writing: *"The answers to your question are available at the following link: <url>"*.

The wizard would ask if the user had any follow-up questions after each answer (*"Are you happy with this answer?"* or *"Can I help you further?"*). For follow-up questions and information requests that could not be reduced to any of the available results, the wizard would return a link to the Google result page for the query.

User Feedback Collection

To collect user feedback, two sources were used: *chat logs* and a *post-hoc questionnaire.*

Chat logs provide objective information such as the average duration of the dialogues, the situations that fell above the assumed requirements of the chat bot interface, how frequent were the requests for repetition, etc.

The questionnaire, submitted to the user immediately after the WOz experiment, enquires about the user's experience using a 5-point Likert scale. This measure 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 assessing user-centered information retrieval among 20 tested.

Inspired by the WOz experiment in Munteanu & Boldea (2000), the WOz questionnaire consisted of the six questions in Figure 5.1.

Q_1: Did you get all the information you wanted using the system?

Q_2: Do you think the system understood what you asked?

Q_3: How easy was it to obtain the information you wanted?

Q_4: Was it difficult to reformulate your questions when you were invited to?

Q_5: Do you think you would use this system again?

Q_6: Overall, are you satisfied with the system?

Figure 5.1: Wizard-of-Oz experiment questionnaire

Questions Q_1 and Q_2 assess the performance of the system and were rated on a scale from 1= "Not at all" to 5="Yes, Absolutely"; in alternative to using one of the five values, users could respond with "Undecided". Questions Q_3 and Q_4 focus on interaction difficulties, especially relating to the system's requests to reformulate the user's question. Questions Q_5 and Q_6 relate to the overall satisfaction of the user. The questionnaire also contained a text area for optional comments.

5.4.2 Results

The WOz experiment involved one wizard and seven users, three women and four men, aged between 17 and 53. All had a self-assessed medium-good level of understanding of English although not all were native speakers; all were regular users of search engines. Five out of seven worked in scientific domains, one was a high-school pupil, another a private business owner.

The users interacted with the wizard via a popular, freely available chat application which all of them had used before. All but one believed that the actual system's output was plugged into the interface. The average dialogue duration was 11 minutes, with a maximum of 15 (2 cases) and a minimum of 5 (1 case) [3].

Chat logs

From the *chat logs*, we observed that, as predicted, all dialogues were information seeking: none of the users asked any question about the system or its capabilities.

One unpredicted result was that users often asked two things at the same time (e.g. *Who was Jane Austen <u>and</u> when was she born?*). To account for this case, we decided to handle multiple questions in the final prototype, as described in Section 5.5.

The wizard often had to ask the users for reformulations. This occurred when the user asked a multiple question, such as: *I would like to purchase a PowerBook G4, what price is it going to be?*, or: *Who painted Guernica and what's his biography?*. The *sysReqClarif* dialogue move proved very useful in this occasion, and clarification requests such as: *Can you please reformulate your question?* or: *In other words, what are you looking for?* were widely used.

Users seemed to enjoy "testing" the system and accepted the invitation to produce a follow-up question in 55% of the cases where this was proposed.

Questionnaire

The values obtained for the user satisfaction questionnaire are reported in Table 5.3.

From these results, users appear to be generally satisfied with the system's performances. None of the users had difficulties in reformulating their questions when this was requested: Q_4 obtained an average of 3.8±.5 standard deviation, where "3 = Neutral" and "4 = Easy".

[3] The two "non-scientist" users took slightly longer than the others to perform the task, i.e. 13 vs 10 minutes: however this depends on many factors such as length of the results and should not be considered significant.

Q	Description	Result
Q_1	Did you get all the information you wanted using the system?	4.3±.5
Q_2	Do you think the system understood what you asked?	4.0
Q_3	How easy was it to obtain the information you wanted?	4.0±.8
Q_4	Was it difficult to reformulate your questions when you were invited to?	3.8±.5
Q_5	Do you think you would use this system again?	4.1±.6
Q_6	Overall, are you satisfied with the system?	4.5±.5

Table 5.3: Wizard-of-Oz questionnaire results: mean± standard deviation

For the remaining questions, satisfaction levels were high, i.e. above 4. Users generally thought that the system understood their information needs (Q_2 obtained a score of 4) and were able to obtain such information (as shown by the results for Q_1). Global satisfaction reached the highest result registered in the questionnaire, a very encouraging 4.5±.5.

User comments

Our main observation from the user comments in the questionnaire was that users expected the tool to be robust: *[...] As it is a computer tool, the user ought to bear in mind some questioning pattern. Nevertheless, the tool coped quite well with (possibly unexpected) language twists such as: "Was Guernica painted by Pablo Picasso"?.*

However, users did not assume that it would behave exactly like a human and seemed to receive system grounding and clarification requests well, e.g.: *[...] on references to "him/it", pretty natural clarifying questions were asked.* This was a particularly encouraging result as we were in doubt that anaphora resolution might not be well received by users.

5.5 Implementation

The dialogue manager and interface were implemented based on the scenario in Section 5.2.1 and the successful outcome of the Wizard-of-Oz experiment.

5.5.1 Dialogue Management Algorithms

As chatbot dialogue follows a pattern-matching approach, it is not constrained by a notion of "state". When a user utterance is issued, the chatbot's strategy is to look for a pattern matching it and fire the corresponding template response. Our main focus of attention in terms of dialogue manager design was therefore directed to the dialogue tasks invoking external resources, such as handling multiple and follow-up questions, and tasks involving the QA component.

Handling multiple questions

As soon as the dialogue manager identifies a user utterance as a question (using the question recognition categories), it tests whether it is a multiple question. Indeed, since the core QA component in YourQA is not able to handle multiple questions, these need to be detected and broken into simple questions.

For this, the system uses the OpenNLP chunker[4] to look for the presence of "and" which does not occur within a noun phrase.

As illustrated in Algorithm 11, if a standalone "and" is found, the system "splits" the multiple question in order to obtain the single questions composing it. It then proposes to the user to begin answering the single question containing more words.

Algorithm 11 Multiple question handling algorithm

```
if q contains "and" then
    set chunksQ[] = chunk(q);
    set i = indexIndependentAnd(chunksQ);

    if  i!=-1 then
        set splitQ[] = split (q,i);
        return mostWords(splitQ);
    else return q;

else return q;
```

Handling follow-up questions

After detecting and handling multiple questions, the next task accomplished by the DM is the detection and resolution of follow-up questions. As a matter of fact, there is evidence that it is vital in handling QA dialogue to apply an effective algorithm for the recognition of follow-up requests (De Boni & Manandhar, 2005; Yang *et al.*, 2006).

The types of follow-up questions that the system is able to handle are:

1. elliptic questions,

[4] http://opennlp.sourceforge.net/

Algorithm 12 Follow-up detection algorithm (De Boni & Manandhar, 2005)

Followup_question (q_i, $\{q_i..q_{i-n}\}$) **is true**

if q_i has pronoun/possessive adjective references to $\{q_i..q_{i-n}\}$

else if q_i contains no verbs

else if q_i has repetition of common or proper nouns in $\{q_i..q_{i-n}\}$ **or** q_i has a strong semantic similarity to some $q_j \in \{q_i..q_{i-n}\}$

2. questions containing third person pronoun/possessive adjective anaphora,
3. questions containing noun phrase (NP) anaphora[5].

These are detected and subsequently solved as follows.

Follow-up detection For the *detection* of follow-up questions, the algorithm in De Boni & Manandhar (2005) is used, reported in Algorithm 12.

Given the current question q_i and the list of previous questions $\{q_i..q_{i-n}\}$, the algorithm uses the following features: presence of pronouns or absence of verbs in q_i, the presence of word repetitions between q_i and the n preceding questions or a high semantic similarity between q_i and one of the preceding questions as elements to determine whether q_i is a follow-up question with respect to the current context. We apply the algorithm by using $n = 8$, following De Boni & Manandhar (2005); at the moment the condition on semantic distance is not included for the sake of processing speed.

Follow-up resolution If the question q is not identified as a follow-up question, it is submitted to the QA component. Otherwise, the *reference resolution* strategy in Algorithm 13 is applied on q. This distinguishes between three types of follow-up: ellipsis, pronominal/adjective anaphora, and NP anaphora. In the second and third case, when no antecedent can be found, a clarification request is issued by the system until a resolved query can be submitted to the QA component.

[5]An example of use of NP anaphora could be the following:

- U_1 "What is the world's longest river?"
- S_1 "The answer is the Nile river."
- U_2 "How long is *the river*?"

Here, the NP "the river" in U_2 is used to signify "the word's longest river".

Algorithm 13 Follow-up resolution algorithm

1. If q is *elliptic* (i.e. contains no verbs), its keywords are completed with the keywords extracted by the QA component from the previous question for which there exists an answer. The completed query is submitted to the QA component.

2. If q contains *pronoun/adjective anaphora*, the chunker is used to find the first compatible antecedent in the previous questions in order of recency. The latter must be a NP compatible in number with the referent.

3. If q contains *NP anaphora*, the first NP in the stack of preceding questions that contains all of the words in the referent is used to replace the latter in the query.

Finally, when the QA process is terminated, a message directing the user to the HTML answer page (see Figure 5.6) is returned and a follow-up proposal or an enquiry about user satisfaction is optionally issued.

Figure 5.2 reports an illustrative transcript of an actual conversation between the interactive interface of YourQA and a user. In utterance $System_2$, the system proposes to the user to issue a follow-up question (*followup* move). In utterance $System_3$, the systems issues a *sysClarif* move to confirm that the user's follow-up request has been correctly interpreted. Finally, in utterance $System_4$, the system tries to keep the conversation going by encouraging the user to provide feedback about the results (this is another *followup* move).

5.5.2 AIML Framework

Following the typical design of an AIML-based conversational agent, we created a set of categories to fit the dialogue scenarios elaborated during dialogue design (Section 5.2.1) and enriched with the WOz experience (Section 5.4). We grouped the categories in different .aiml files, each corresponding to one of the dialogue moves in Table 5.2; some of these are illustrated below.

AIML interpreter and context

We used the Java-based AIML interpreter Chatterbean[6], which allows to define custom AIML tags and allows a seamless integration between the QA module and the chat interface.

First, we augmented the Chatterbean tag set with two AIML tags:

[6] `http://chatterbean.bitoflife.cjb.net/`

$User_1$: hi

$System_1$: *Hi, dear! How can I help you?*

$User_2$: when was the wwe founded

$System_2$: *The answers are on the right (you can click the links to know more!). Can I help you further?*

$User_3$: who is its chairman

$System_3$: *Do you mean to ask who is THE WWE chairman ?*

$User_4$: yes

$System_4$: *I found the answers in the panel on the right. Are you happy with these answers?*

$User_5$: yes thank you

$System_5$: *Great!*

Figure 5.2: A conversation between YourQA and user U_1

- **<query>**, to invoke the YourQA Question Answering module;
- **<clarify>**, to support the tasks of follow-up detection and reference resolution.

Moreover, the Chatterbean context implementation (in a dedicated Java `Context` class) allows to instantiate and update a set of variables, represented as context properties. We defined several of these, including:

- the user's *ID*, which is matched against a list of known user IDs to select a profile for answer extraction;
- the current *query*, which is used to dynamically update the stack of recent user questions. The stack is used by the clarification request detection module to perform reference resolution, following the algorithm exposed above;
- the *resolved* question, i.e. the current query as resolved during follow-up handling (or the current query if no follow-up is detected);
- the *topic* of conversation, i.e. the keywords of the last question issued by the user which received an answer. The topic variable is used to clarify elliptic questions, by augmenting the current query keywords with those in the topic when ellipsis is detected.

Category 14 The YourQA DO YOU KNOW category

```
<category>
<pattern>DO YOU KNOW *</pattern>
<template>
<srai>CLARIFY *</srai>
</template>
</category>
```

Example

To illustrate the dynamics of AIML and the use of tags and variables we take the category used by the system to clarify the nature of requests introduced by the cue words "Do you know", represented in Category 14. This invokes the pattern:

```
<pattern>CLARIFY *</pattern>
```

in the CLARIFY category (reported in full in Category 15). This pattern triggers a template calling the AIML tag **<clarify>**. The latter invokes the follow-up question detection and resolution module on the text matching the "*" expression. The results of such call are not visible to the user (as would normally happen in AIML) thanks to the **<think>** tag.

```
<template>
<think>
<set name="clarif">
<clarify></star><clarify>
</set>
</think>
```

The resolution module follows the strategy exposed in Section 5.2.3 and returns a judgment (e.g. "ELLIPTIC"), which is assigned to the context variable clarif (using the **<set>** tag).

Finally, a conditional branch invoked by the **<condition>** tag on the clarif variable determines the appropriate QA routine based on the value of clarif:

```
<condition name="clarif" value="TRUE"> ...</condition>
...
<condition name="clarif" value="ELLIPTIC"> ...</condition>
```

Category 15 The YourQA CLARIFY category

```
<category>
<pattern>CLARIFY *</pattern>
<template>
<think>
<set name="clarif">
<clarify><star/></clarify>
</set>
</think>
<condition name="clarif" value="TRUE">
Do you mean to ask <get name="resolved"/> ?
</condition>
<condition name="clarif" value="FALSE">
<srai>QUERY <get name="resolved"/></srai>
</condition>
<condition name="clarif" value="MULTIPLE">
Oups, one question at a time...Shall we start with <get
name="resolved"/> ?
</condition>
<condition name="clarif" value="NULL">
<srai>REFORMULATE</srai>
</condition>
<condition name="clarif" value="ELLIPTIC">
<srai>QUERY <get name="resolved"/></srai>
</condition>
</template>
</category>
```

Based on the value of clarif, the conditional branches invoke the QUERY, ANSWER and FOLLOWUP categories, reported in Categories 16, 18, and 17, respectively.

While the QUERY category is in charge of interfacing with the standard QA system, the ANSWER category's templates direct the user's attention to the area of the system interface where answers are located (with expressions such as: "The answers are on the right (you can click the links to know more!)"). Finally, the FOLLOWUP category is in charge of suggesting a follow-up session to the user.

Figure 5.3 reports an extract from a real interaction with the system, implemented using the above AIML categories and others, highlighting how the system supports follow-up (utterance $System_3$).

An example of how the system tries to recover from the user's unsatisfactory reaction

Category 16 The YourQA QUERY category, invoking the standard QA component

```
<category>
<pattern>QUERY *</pattern>
<template>
<query><star/></query>
<srai>ANSWER</srai>
<srai>FOLLOWUP</srai>
</template>
</category>
```

Category 17 The YourQA FOLLOWUP category, encoding follow-up proposal

```
<category>
<pattern>FOLLOWUP</pattern>
<template>
<random>
<li>Are you happy with these answers? </li>
<li>Can I help you further?</li>
<li>Do you have another question?</li>
<li></li> // here, no follow-up proposal occurs
</random>
</template>
</category>
```

Category 18 The YourQA ANSWER category, returning the answer to the user

```
<category>
<pattern>ANSWER</pattern>
<template>
<random>
<li>I found the answers in the panel on the right. </li>
<li>The answers are on the right (you can click the links
to know more!). </li>
<li>Possible answers are at the panel on the right. </li>
</random>
</template>
</category>
```

$User_1$: Hi

$System_1$: *Hi! How can I help you?*

$User_2$: what is a thermometer?

$System_2$: *The results are in the panel on the right. Can I help you further?*

$User_3$: How does it measure the temperature?

$System_3$: *Do you mean how does A THERMOMETER measure the temperature?*

$User_4$: yes

$System_4$: *The results are in the panel on the right. Are you happy with this answer?*

$User_5$: ...

Figure 5.3: Extract from a conversation between YourQA and user U_2

to the results is visible in Figure 5.4. Here, in two occasions the system enquires about the user's satisfaction after returning results (utterances $System_2$ and $System_7$), and the user replies "no" (utterances $User_3$ and $User_8$); the system thus tries to amend by asking for different question formulations ($System_3$: *Can you write your question in a different way?* , $System_9$: *In other words, what are you looking for?*).

Also notice how, in utterance $System_4$, YourQA attempts to clarify the potentially ambiguous utterance $User_4$ ("I want ... ") by asking whether such utterance is an actual question. This uses a category similar to the "`I NEED *`" category mentioned in Section 5.3.4.

An overview of the implemented architecture for the interactive version of YourQA is illustrated in Figure 5.5. Here, the dialogue component is represented with its three main sub-components: these are the multiple/follow-up question recognition module, the query management module and the common AIML chat module. Interaction between the dialogue component and the standard QA system takes place through the dialogue query management module, which submits a resolved question to the standard QA system's question processing module.

Once results are available, these are returned by the QA system using the format presented in Figure 2.7; then, the dialogue component replies to the user with the *answer* and *follow-up* patterns described earlier.

$User_1$: Who invented the Rubik's Cube?

$System_1$: *The answers are on the right (you can click the links to know more!). Can I help you further?*

$User_2$: What was the world record time set in the 2006 competition?

$System_2$: *Possible answers are at the panel on the right. Are you happy with these answers?*

$User_3$: no

$System_3$: *Can you write your question in a different way?*

$User_4$: I want results for Rubik's Cube competition

$System_4$: *If I get it right, your question is results for Rubik's Cube competition.*

$User_5$: yes

$System_5$: *The answers are on the right (you can click the links to know more!). Do you have another question?*

$User_6$: What was the world record time set in the Rubik's Cube 2006 competition?

$System_6$: *The answers are on the right (you can click the links to know more!). Are you happy with these answers?*

$User_7$: Who founded the International Rubik's Cube Competition in the United States?

$System_7$: *The answers are on the right (you can click the links to know more!). Are you happy with these answers?*

$User_8$: no

$System_8$: *In other words, what are you looking for?*

Figure 5.4: Extract from a conversation between YourQA and user U_3

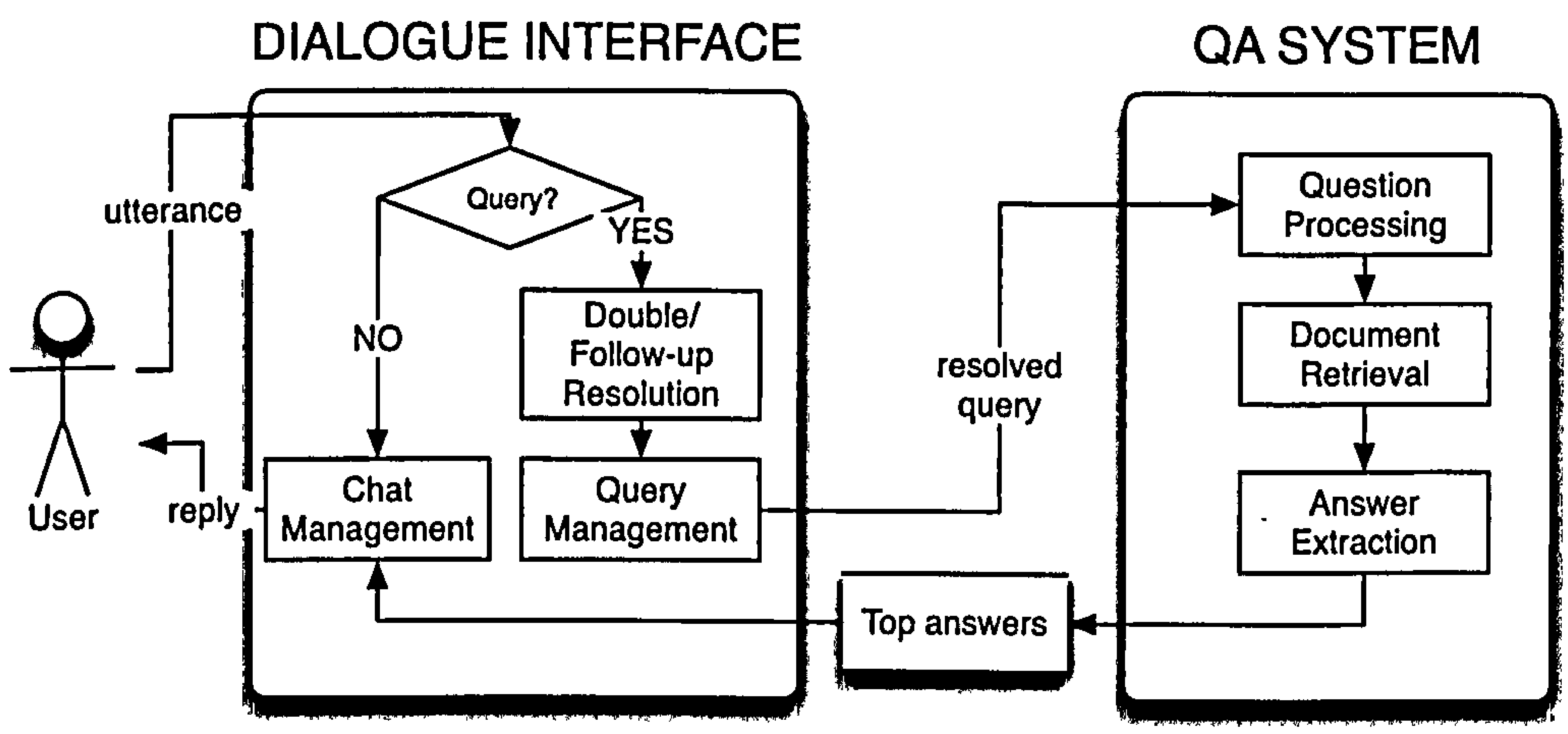

Figure 5.5: High-level architecture of the interactive version of YourQA

5.5.3 Dialogue Interface

The YourQA interactive interface has been used to evaluate the interactivity aspect of Question Answering described in this chapter. Two versions of such interface have been created: Figure 5.6 shows a desktop version of the application (Java applet), while Figure 5.7 illustrates a version of the system accessible from a Web service (Java servlet).

In both cases, the interactive interface consists in a main window with a left panel where the chat takes place and a right panel where results are visualized. As in a normal chat application, users write in a text field and the current session history as well as the interlocutor replies are visualized in an adjacent text area.

5.6 Evaluation

While the accuracy of standard QA systems can be evaluated and compared using quantitative information retrieval metrics such as F1-measure and MRR (Voorhees, 2003), dialogue interfaces pose complex evaluation challenges, as the latter differ in appearance, intended application and target users.

First of all, there is not always a clear metric to determine the success of a dialogue session; success and usefulness depend greatly on the users' subjective impressions. Moreover, it is difficult to find baseline dialogue systems to make comparisons, as dialogue applications vary greatly. Finally, computing the similarity between two dialogue systems would be, in itself, a very difficult task.

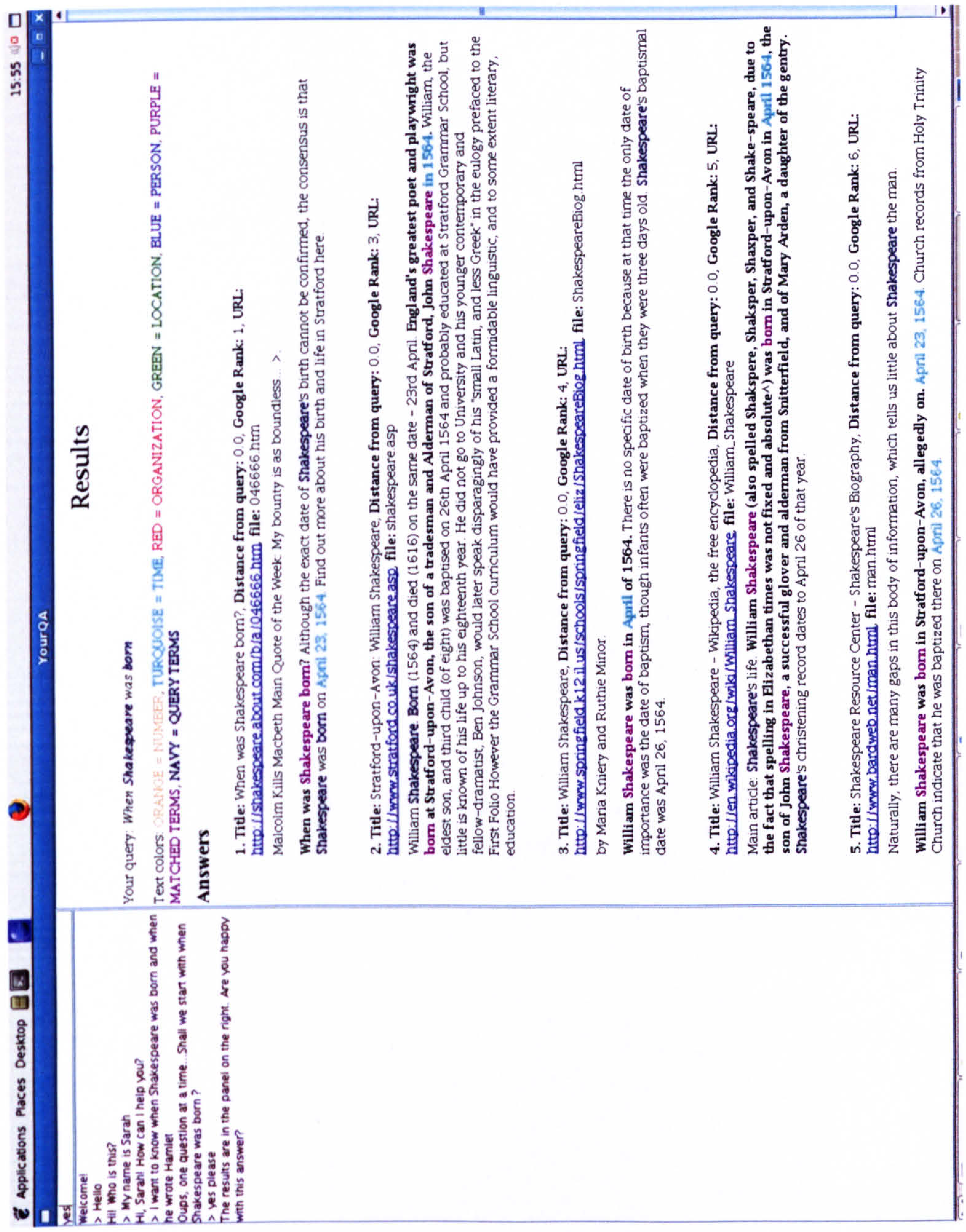

Figure 5.6: YourQA's interactive interface: desktop (applet) version

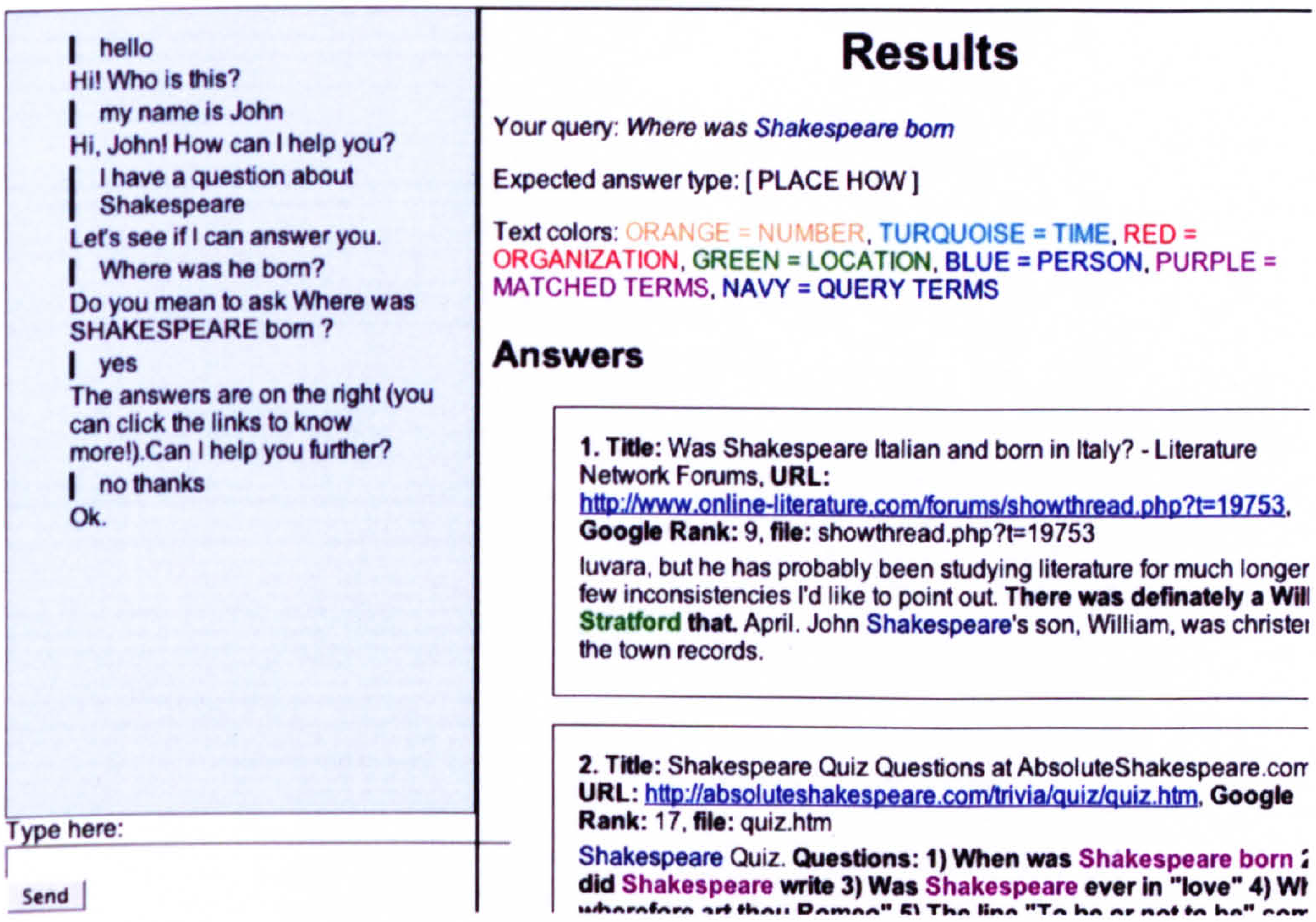

Figure 5.7: YourQA's interactive interface: Web service (servlet) version

Indeed, dialogue systems are often evaluated using qualitative metrics such as user satisfaction and perceived time of usage (Walker *et al.*, 2000). A typical way of evaluating a dialogue system is the definition of a task which the user has to carry out interactively, and upon which several measurements can be made. For instance, TRAINS-95 and TRAINS-96 (Allen *et al.*, 2000) had a task (i.e. find as short a route as possible) which allowed multiple possible solutions but which was clear enough to allow a special metric (called solution quality) to be used effectively.

A variety of means of *collecting data* for analyzing info-seeking dialogue are found, including *interaction logs*, i.e. the collection of all data that can be logged automatically. It is relatively easy to obtain a log corpus for a given system: for example, by offering the system as a free service to the people it was designed for.

Another useful tool are *user surveys*, or post-hoc questionnaires. These may ask for both qualitative information ("Why did you do that?") or quantitative information ("Did you find this particular aspect of the system good or bad?").

User satisfaction questionnaires and interaction logs have been found to be among the most effective tools also in the context of interactive QA system evaluation, when compared to other sources of information such as one-to-one interviews and cross-user evaluations (Kelly *et al.*, 2006).

Driven by these guidelines, we conducted first a brief initial evaluation and then a final, more extensive one, as described below.

5.6.1 Initial Evaluation

To conduct a preliminary evaluation of our prototype, we designed three scenarios where users had to look for two different items of information relating to the same topic (e.g. Shakespeare's date of birth and when he wrote Hamlet). These topics are illustrated in Figure 5.8.

TASK A You're looking for info about: Shakespeare.
Find out:

- When he was born
- When he wrote Hamlet

TASK B You're looking for info about: thermometers.
Find out:

- What a thermometer is
- How it measures the temperature

TASK C You're looking for info about: Barnes and Noble.
Find out:

- What Barnes and Noble is
- Where is its headquarters

Figure 5.8: First evaluation: tasks

Users had to choose one or more topics and use first the non-interactive Web interface of the QA prototype (handling questions in a similar way to a search engine) and then the interactive version depicted in Figure 5.6 to find answers.

After using both versions of the prototype, users filled in a questionnaire about their experience with the *chat version* which comprised the same questions as the WOz questionnaire and the following additional questions:

Q_7 Was the pace of interaction with the system appropriate?

Q_8 How often was the system sluggish in replying to you?

Q_9 Did you prefer the chat or the Web interface and why?

Questions Q_7 and Q_8 could be answered using the usual Likert scale from 1 to 5 and were taken from the PARADISE evaluation framework (Walker *et al.*, 2000). Q_9 was particularly interesting to assess if and in what terms users perceived a difference between the two prototypes. All the interactions were logged.

Results

From the initial evaluation, which involved eight volunteers, we gathered the following salient results.

Chat logs In the *chat logs*, we observed that the system was able to resolve pronominal anaphora in nine out of the eleven cases when it occurred. No elliptic queries were issued, although in two cases verbs were not spotted by the system[7] causing queries to be erroneously completed with previous query keywords.

Users tended not to reply to the chatbot offers to carry on the interaction explicitly, directly entering a follow-up question instead.

Due to the limited amount of AIML categories, the system's requests for reformulation occurred more frequently than expected: we subsequently added new categories to account for this shortcoming.

Questionnaire From the *questionnaire* (Tab. 5.4), user satisfaction levels (Q_1 to Q_6) are slightly lower than in the WOz experiment (Section 5.4), ranging from 3.4$\pm$.5 for Q_3 to 4.4$\pm$.5 for Q_6.

Users felt the system slow in replying to the questions: Q_7 and Q_8 achieved 3.8$\pm$.5 and 2.4$\pm$1.3, respectively. This is mainly because the system performs document retrieval in real time, hence heavily depends on the network download speed.

However, all but one user (87.5%) answered Q_9 by saying that they preferred the chat interface of the system, because of its liveliness and ability to understand when questions were related (i.e. anaphora).

5.6.2 Final Evaluation

Building on the results of the initial evaluation and after drawing additional patterns from the analysis of over 100 chat logs collected since then, we designed a further evaluation.

[7]This was due to the POS tagger, which incorrectly annotated as nouns verbs for which nouns with identical spelling existed (such as "turns").

	Question	Results
Q_1	Did you get all the information you wanted using the system?	3.8±.8
Q_2	Do you think the system understood what you asked?	3.8±.4
Q_3	How easy was it to obtain the information you wanted?	3.7±.8
Q_4	Was it difficult to reformulate your questions when you were invited to?	3.8±.8
Q_5	Do you think you would use this system again?	4.0±.9
Q_6	Overall, are you satisfied with the system?	4.3±.5
Q_7	Was the pace of interaction with the system appropriate?	3.5±.5
Q_8	How often was the system sluggish in replying to you?	2.3±1.2
Q_9	Did you prefer the chat interface to the Web interface?	83.3%

Table 5.4: First evaluation questionnaire results: average ± standard deviation

Design

For the final evaluation, we chose 9 question series from the TREC-QA 2007 campaign with the following criteria:

1. the first question in each series could be understood by a QA system without the need of explicitly mentioning the series target;

2. at least one half of the total number of questions contained anaphoric and/or elliptic references,

3. three questions were retained per series to make each evaluation balanced.

For instance, the three questions from series 266 reported in Table 5.5 were used to form one task.

Table 5.5: Example of TREC 2007 question series used for the final evaluation

Series ID: 266		*Target:* Rafik Hariri
Question ID	*Type*	*Text*
266.1	FACTOID	When was Rafik Hariri born?
266.2	FACTOID	To what religion did he belong (including sect)?
266.4	FACTOID	At what time in the day was he assassinated?

Twelve users were invited to find answers to the questions in two different series from the nine collected, in such a way that the first series was to be addressed using the standard version of YourQA, the second one using the interactive version. Each series was evaluated at least once using both versions of the system.

At the end of the experiment, users had to fill the same user satisfaction questionnaire as in the first evaluation, but this time they had to give feedback about *both* versions of the system.

Given its design, the final evaluation was more accurate and challenging than the first one in two respects: first, comparative feedback was collected from the standard and interactive versions of the system; second, question series contained more questions and came from TREC-QA, making them hard to answer using the Web.

Results

The results obtained from the questionnaire for the standard and interactive versions are reported in columns "Standard" and "Interactive" of Table 5.6, respectively.

	Question	Standard	Interactive
Q_1	Did you get all the information you wanted using the system?	4.1±1	4.3±.7
Q_2	Do you think the system understood what you asked?	3.4±1.3	3.8±1.1
Q_3	How easy was it to obtain the information you wanted?	3.9±1.1	3.7±1
Q_4	Was it difficult to reformulate your questions when you were invited to?	N/A	3.9±.6
Q_5	Do you think you would use this system again?	3.3±1.6	3.1±1.4
Q_6	Overall, are you satisfied with the system?	3.7±1.2	3.8±1.2
Q_7	Was the pace of interaction with the system appropriate?	3.2±1.2	3.3±1.2
Q_8	How often was the system sluggish in replying to you?	2.7±1.1	2.5±1.2
Q_9	Did you prefer the standard or the interactive interface and why?	41.7%	58.3%

Table 5.6: Second evaluation questionnaire results: average ± standard deviation

As a first remark about such results it must be specified that the paired t-test conducted to compare the questionnaire replies to the standard and interactive versions of YourQA did not register statistical significance. Nonetheless, we believe that the evidence we collected from the experiment, both quantitative (the questionnaire replies) and qualitative (user comments), suggests a few interesting interpretations which we report below.

First, a good overall satisfaction appears with both versions of the system (Q_8, Figure 5.12), with a slight difference in favor of the interactive version.

The standard and interactive versions of the system seem to offer different advantages (Figure 5.9): while the ease of use of the standard version was rated higher (Q_5) –probably because the system's requests for reformulation added a challenge to users used to a search engine-style interaction– users felt that they obtained more information using the interactive version (Q_1).

Concerning interaction comfort (Figure 5.11), users seemed to feel that the interactive version understood better their requests than the standard one (Q_2); they also found it easy to reformulate questions when the former asked to (Q_6). These findings suggest that even a simple chat interface like YourQA's can be very useful in terms of user satisfaction.

However, while the pace of interaction was judged slightly more appropriate in the interactive case (Q_3, Figure 5.10), interaction was considered faster when using the standard version (Q_4). Unfortunately, in both cases the interaction speed rarely appears adequate, as also registered from user comments. This partly explains the fact that users seemed more ready to use again the standard version of the system (Q_7, Figure 5.12); some users clearly stated so, saying e.g. "I would be very interested in using the system again once it reaches industrial speed".

An interesting difference with respect to the first evaluation was the "preference" question, Q_9: 7 out of 12 users (58.3%) said that they preferred the interactive version, hence a smaller ratio of the users than in the first evaluation. The reasons given by users in their comments were mixed: while some of them were enthusiastic about the chatbot's small talk features and felt that the interface interacted very naturally, others clearly said that they felt more comfortable with a search engine-like interface and that the design of the interactive prototype was inadequate.

Discussion

From these results, we gather the following remarks: first, the major weakness of our system remains speed, which must be greatly optimized. As supporting the interactive features of YourQA requires more processing time, we believe that this is one of the main reasons for which in our second evaluation, where tasks required an intensive use of follow-up detection and resolution, the interactive model was penalized with respect to the standard version.

Moreover, although the interactive version of the system was well received, some users seem to prefer more traditional information retrieval paradigms and value the advantages of interactivity at a lesser extent. We believe that this is due partly to cultural reasons (the search engine-like non-interactive model of IR biasing users), and partly to the fact that the follow-up resolution mechanism of the interactive version is not always

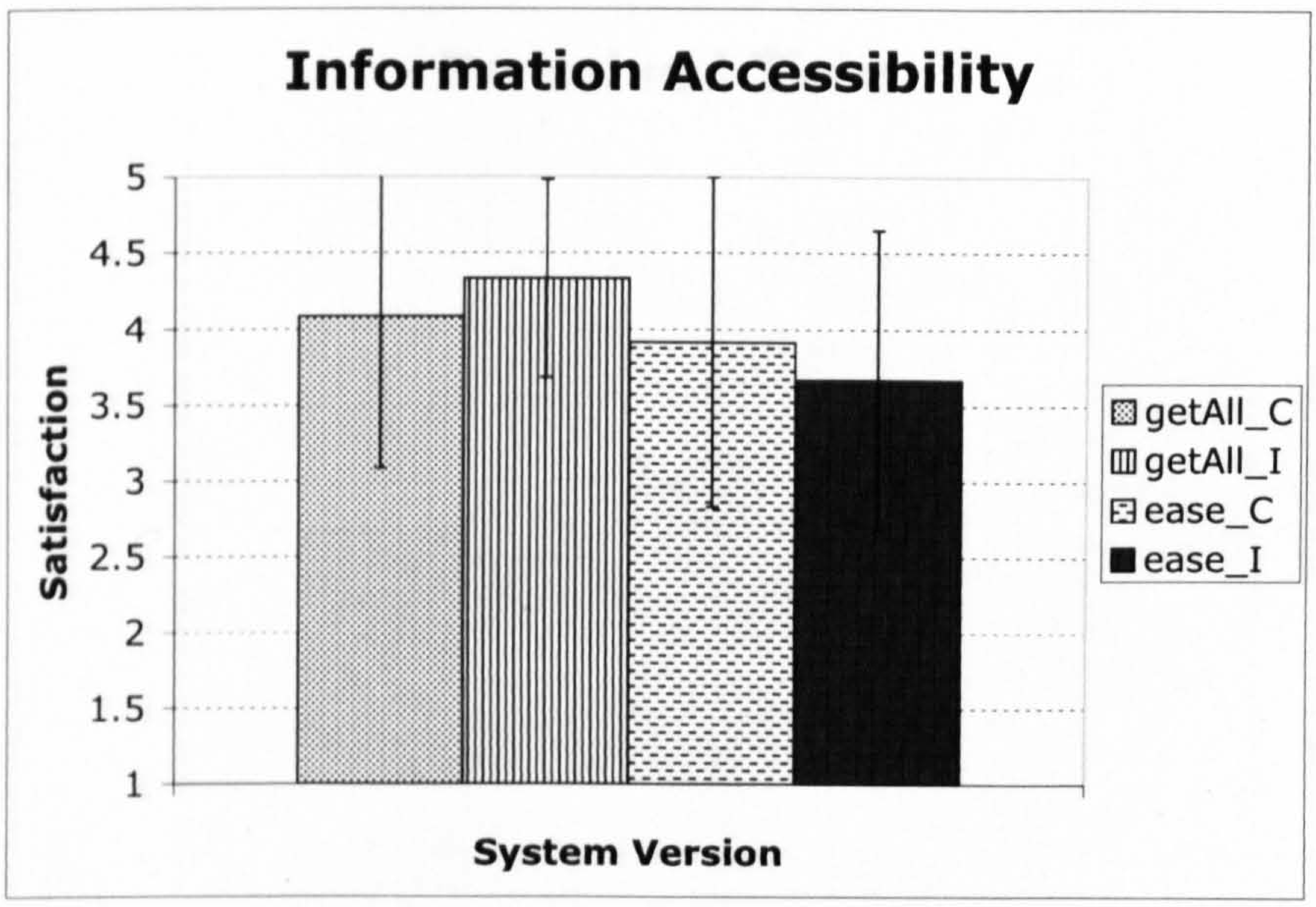

Figure 5.9: Final evaluation: perceived accessibility of information ("C" = standard version, "I"=interactive version)

accurate, generating errors and delaying the delivery of results.

Finally, the chat interface raises expectations concerning what the system can understand; when these are not met (i.e. the system misunderstands or asks for reformulation at a frustrating frequency), this lowers user satisfaction. An examples of this is Figure 5.13, which shows that it is crucial to improve the range of patterns matched by the chatbot in order to cover more user utterances than the ones currently understood.

However, most of the critical aspects emerging from our overall satisfactory evaluation depend on the specific system we have tested rather than on the nature of interactive QA, to which none of such results appear to be detrimental.

We believe that the search-engine-style use and interpretation of QA systems are due to the fact that QA is still a very little known technology. It is a challenge for both developers and the larger public to cooperate in designing and discovering applications that take advantage of the potentials of interactivity.

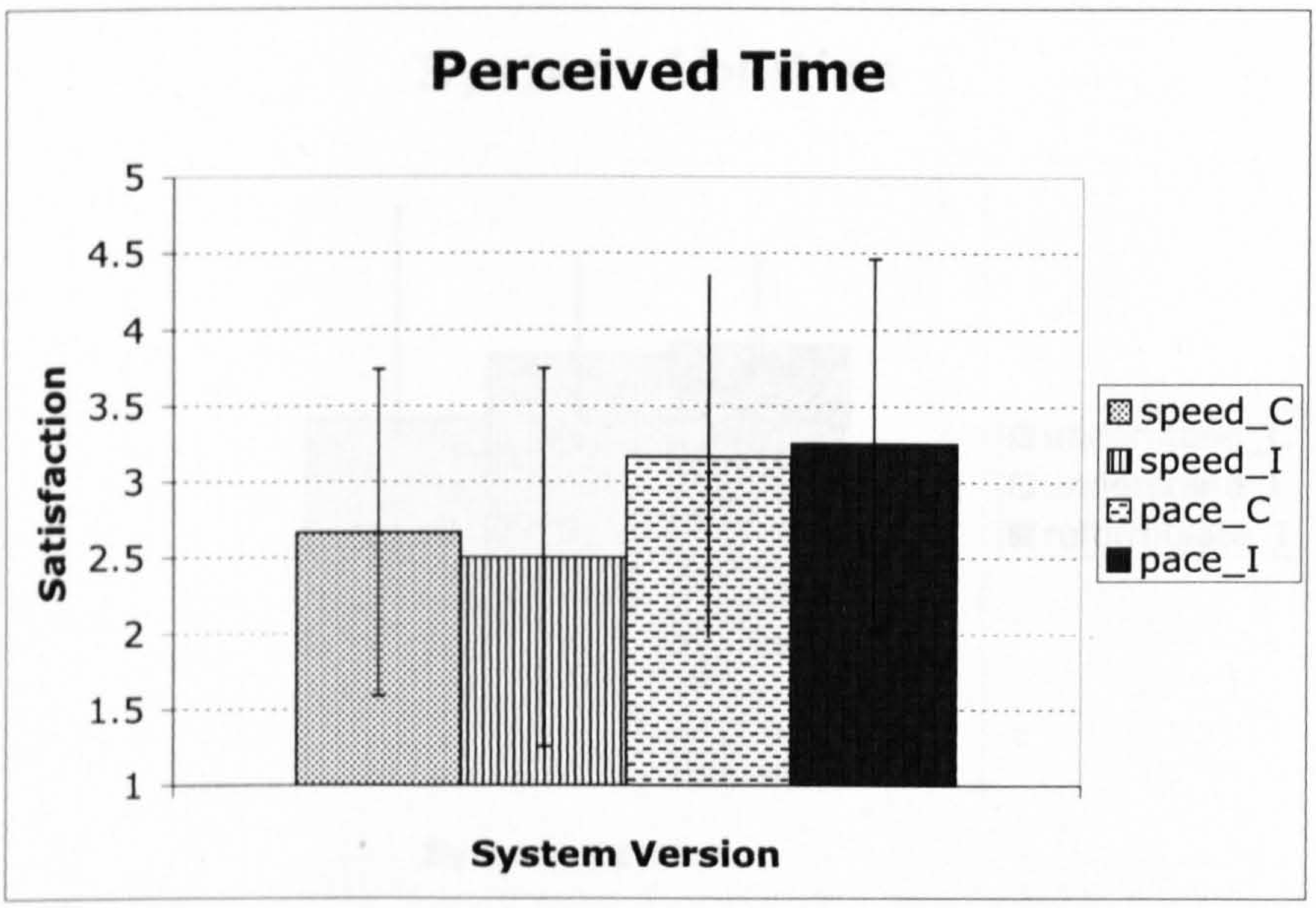

Figure 5.10: Final evaluation: perceived pace and speed of interaction ("C" = standard version, "I"=interactive version)

5.7 Conclusions

This chapter reports a study on the requirements of Interactive Question Answering in terms of dialogue modelling and dialogue management design. A possible dialogue scenario is outlined, and a proposal of chat-based IQA dialogue management is subsequently fomulated. A Wizard-of-Oz experiment confirms the chatbot dialogue management options and details are provided on how to implement such dialogue management approach. Finally, the evaluation of the interactive version of the YourQA prototype suggests optimistic conclusions on the feasibility of chatbot-based interactive QA.

In the future, it would be interesting to study more advanced strategies for anaphora resolution in questions, taking inspiration from statistical approaches (see Poesio *et al.*, 2001), which meet the time efficiency requirements of chat-based dialogue.

Moreover, a widely unexplored research topic of QA research is the study of data-driven answer clarification strategies suitable for the open domain: YourQA currently

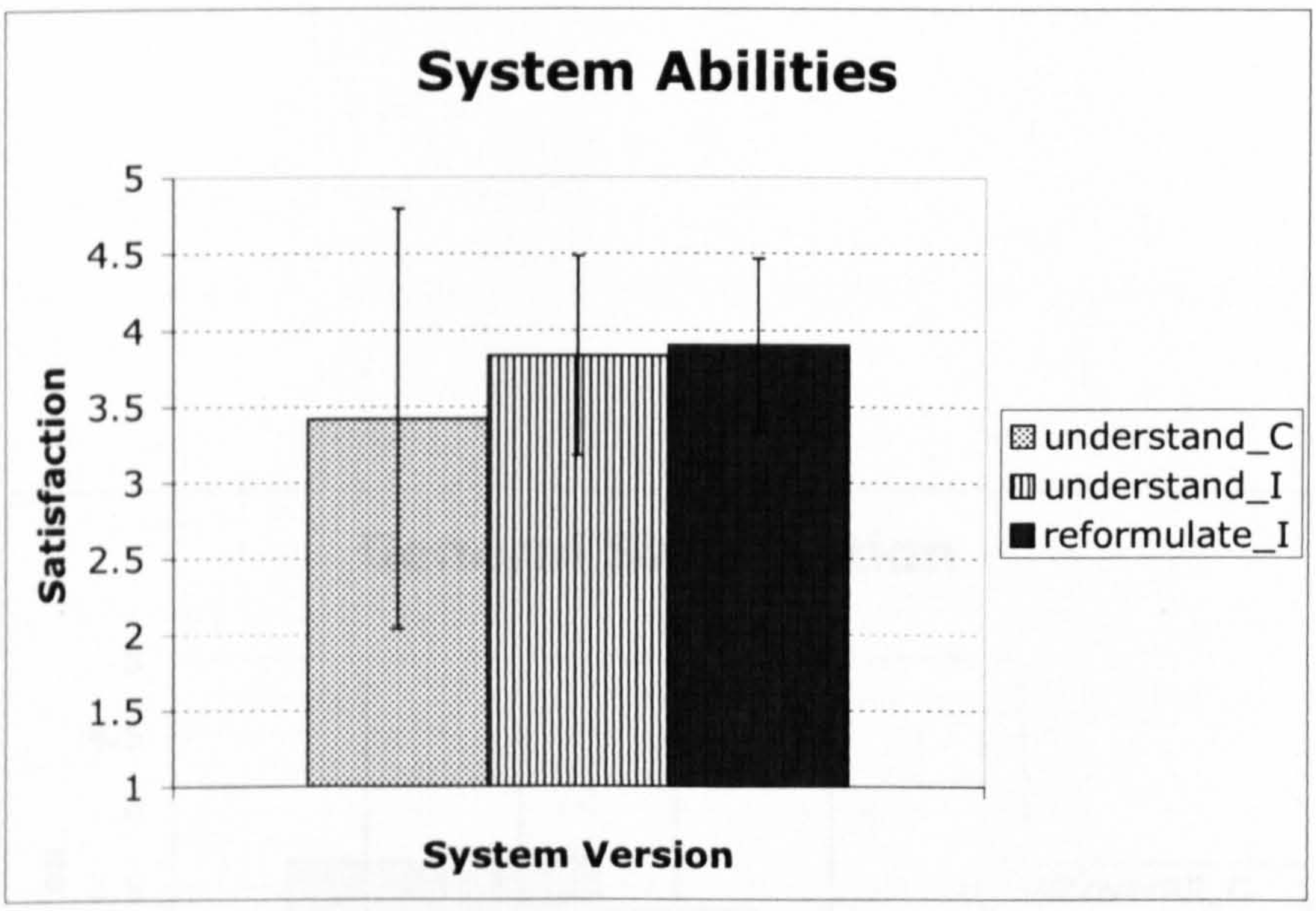

Figure 5.11: Final evaluation: perceived system understanding of the context and ease of reformulation ("C" = standard version, "I"=interactive version)

only handles follow-up in questions, and there do not seem to be many dialogue applications able to cope with this aspect. In this respect, techniques such as answer clustering (Rasmussen, 1992) may prove useful, by providing summarized views of the available information and enabling to propose meaningful suggestions to the user.

Interesting future work can also involve the integration of the Interactive QA with personalized QA abilities, as introduced in Chapter 4 and further discussed in Chapter 6.

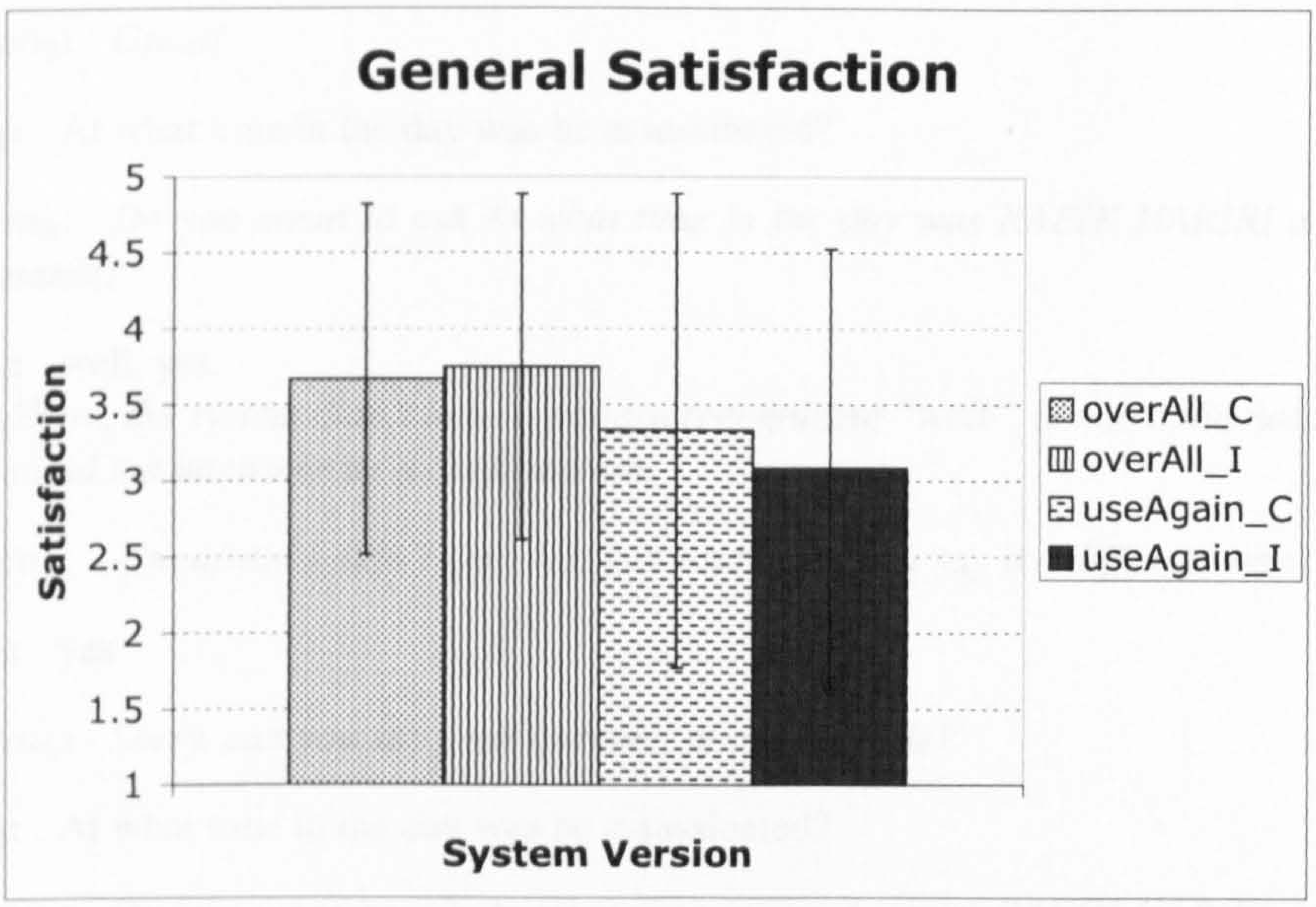

Figure 5.12: Final evaluation: overall satisfaction with the system ("C" = standard version, "I"=interactive version)

$User_1$: When was Rafik Hariri born?

$System_1$: *I found the answers in the panel on the right. Are you happy with these answers?*

$User_2$: yes

$System_2$: *Great!*

$User_3$: At what time in the day was he assassinated?

$System_3$: *Do you mean to ask At what time in the day was RAFIK HARIRI assassinated?*

$User_4$: well, yes.
//here, the system didn't have a pattern recognizing "well", hence it did not understand the utterance as a confirmation.

$System_4$: *I'm afraid I didn't get what you said...Can you say it a different way?*

$User_5$: yes

$System_5$: *Sorry, can you say your question in other words?*

$User_6$: At what time in the day was he assassinated?

$System_6$: *Do you mean to ask At what time in the day was RAFIK HARIRI assassinated?*

$User_7$: yes

$System_8$: *The answers are on the right (you can click the links to know more!). Can I help you further?*

Figure 5.13: Extract from a conversation between YourQA and user U_4, collected during the second evaluation

Chapter 6

Future Research in Question Answering

So far, the previous chapters of this thesis covered several aspects of Question Answering: advanced methods for addressing complex questions, techniques to achieve personalization and finally the issue of interactivity.

However, there still remains a large number of research areas that need to be explored. In this chapter, we propose a series of models to extend the research topics developed so far.

6.1 Future Directions for Question Answering

In this section, future lines of research are developed in the fields of standard QA (with a particular focus on techniques for complex answers), personalization and interactivity. These are addressed in Sections 6.1.1, 6.1.2 and 6.1.3, respectively.

6.1.1 Standard Question Answering

In the field of standard Question Answering, several issues still need to be approached and optimizations can be made in order to improve the baseline performance of YourQA, in particular during the answer extraction phase.

Firstly, the metrics we use to compute the similarity between questions and candidate answer sentences for the factoid case are quite simple, basically relying on the bag-of-words approach and on the performance of our off-the-shelf Named Entity recognizers, Part-Of-Speech tagger and of some hand-written rules.

While it is certainly difficult to tune the performance of a NE recognizer on Web data, which is by nature heterogeneous, further work could focus on the improvement of the rules applied for questions invoking time and quantity expressions; machine learning methods (see Mani *et al.*, 2006) could also be studied to improve our baseline performance with factoids.

Regarding non-factoid answers such as definitions, the algorithms we have reported are based on a combination of similarity metrics tuned by hand based on empirical data. Sounder optimization strategies should be involved in order to improve our baseline similarity metrics also in the case of non-factoids.

An interesting method we have been experimenting with in the initial phase of this research (Quarteroni & Manandhar, 2005, 2006) involves clustering, a technique often used for information extraction and retrieval (Rasmussen, 1992). In a previous version of YourQA, during the document retrieval phase, the initial set of documents was hierarchically clustered. The Cobweb algorithm was applied for this purpose, using a data representation composed by the document key-phrases (extracted using Kea) and optionally their reading levels (estimated via language modelling).

Clustering can be an efficient method for answer visualization (see Wu & Wilkinson, 1998) and can also provide additional criteria for answer extraction. In this phase, answers could be returned to the user by taking into account the document clusters they were extracted from, thus being ordered by topic further to relevance. An interesting example is the search engine SNAKET (Ferragina & Gulli, 2005), where text snippets resulting from a search are clustered and labelled according to their topics.

Moreover, in personalized QA, documents belonging to a cluster with common key-phrases with the User Model profile could be given preference in the final answer ranking.

However, the weakness of clustering for open domain (Web) documents is finding effective criteria for evaluation: for this reason, we have not explored clustering in this thesis.

Interesting ongoing work is being carried on at the moment in the field of complex Question Answering, following the successful outcome of the application of tree kernel learning and shallow semantic features (Kaisser & Webber, 2007). We are currently conducting experiments with answer classification and re-ranking by using the AQUAINT document collection for the document retrieval phase.

Assessing the performance of definition answer classifiers using the same data used for TREC will further align our results with those encountered in the literature. However, such alignment is difficult as the characteristics of TREC evaluation (i.e. the nugget approach and the multiple-assessor evaluation) make it complex to assess results unless when participating to the annual evaluation campaigns.

Another planned future work involves the use of semantic roles automatically extracted from FrameNet (Baker *et al.*, 1998) as shallow semantic features to construct Predicate Argument Structures. With respect to PropBank, where data is semantically annotated using the generic roles of "predicates" and "arguments" (which closely reflect its syntactic features), events in FrameNet are represented as "frames", for which specific semantic roles are defined. For instance, the frame "theft" has roles such as "victim", "perpetrator", and "goods".

The advantage of using FrameNet annotation would therefore reside in an increased amount of semantic knowledge and the availability of more specific information than in PropBank.

6.1.2 Personalized Question Answering

In the area of personalization, several extensions may be made to the currently used technologies. As a first general remark, it must be pointed out that although generic, the User Models designed for YourQA reflected usage in a particular information retrieval framework. New UM attributes may be defined in the future to suit different or more specific information needs.

Secondly, the technique of unigram language modelling for reading level estimation can be used to model any category of user, provided that sufficient representative samples can be collected. The work by Collins-Thompson & Callan (2004) applied the technique first to a three-level class, then to a more fine-grained taxonomy representing the twelve grades of primary and secondary school in the USA.

Similarly, it would be interesting to study more fine-grained models of reading levels as well as different taxonomies of readability, taking into account the user's hastiness (see Komatani *et al.*, 2003) (measured on a scale defined by the authors) or defining novice/expert profiles in more restrained domains (see Hickl & Harabagiu, 2006).

A third area of future research involves the evaluation aspect. As explained in Section 4.6, the evaluation of the reading level and profile component have been carried out independently to avoid biases. It would be interesting to carry on an additional evaluation regarding complete User Models, measuring for instance the amount of profile information that is lost when filtering based on a specific reading level.

A further evaluation on reading level estimation should be carried out by addressing end-users with low reading skills, and notably children. Indeed, there has not yet been the opportunity to study this aspect of adaptivity in detail.

Finally, an important aspect of future work involves achieving a dynamic framework for User Modelling. As anticipated in Chapter 4, the presence of a dialogue component interacting with the standard QA module would allow the interaction history to be used as a source of information about the user's evolving reading abilities and needs. Periodically analyzing the dialogue history would allow for instance to estimate the reading level of users based not only on what they read but also on how they chat with the interface and to extract key-phrases from their requests.

6.1.3 Interactive Question Answering

Interactive Question Answering is still at an early stage and so is our research in this field. A first point of research is the representation of the conversation context. Improved strategies for reference resolution, going beyond the simple algorithm in Chapter 5, would prevent the system from relying on confirmation from the user about its hypotheses and speed up the conversation, resulting in increased user satisfaction.

Moreover, at the moment our follow-up resolution strategy only applies to previous user questions. However, the system's results, currently visualized separately from the chatbot conversation, should also be taken into account when performing reference resolution. Answer-driven clarification strategies, allowing user's clarification and follow-up requests referring to the contents of results, should be implemented to allow users to fully grasp the potential of dialogue interfaces for information retrieval.

Finally, one of the main points which should be addressed in the future is a thorough study of the potentials and limitations of a chatbot-based dialogue management model to address open-domain QA. Indeed, the chatbot solution presented in this thesis appears to be very powerful as the dialogue interface is directly connected to two lightweight Java-based modules: one for reference resolution and one for interfacing with the underlying search engine. It is clear that the chatbot itself is mainly an interface for the underlying modules, and can thus suffer from scalability issues. Indeed, while the architecture presented in this thesis, connecting such interface to a small resolution module and to a bridge to the QA system, remains easy to handle, the addition of heavier modules and their interactions may prove to be problematic.

For instance, the presence of a more advanced reference resolution module, performing resolution not only on previous questions but also on the answer content, would imply a representation of the conversational context that goes beyond the current one, based on a simple stack, and additionally would cause the chatbot to respond more slowly. Moreover, the conversational topic, currently represented as a set of keywords, might require a more advanced form of representation in an extended version of the work that may be difficult to represent as a simple variable as currently done.

However, it is encouraging to observe that chat-based interfaces are used in demanding applications, such as persuasive dialogue (see Andrews *et al.*, 2006). Persuasion requires the integration of a planning component, which must decide the following system utterance according to a model of argumentation and to the current context of the conversation, and of a lively conversational interface to achieve user comfort. These requirements are addressed in Andrews *et al.* (2006) by constructing a multi-layer framework where a chat-based reactive component is integrated with an underlying Prolog-based planning component to achieve both aims of persuasiveness and naturalness.

Drawing from the experience collected during the research on personalization and interactivity, the final contribution of this thesis is the creation of a comprehensive model of personalized, interactive Question Answering. Section 6.2 describes such unified framework, which has been implemented with the creation of the three prototype systems described in the following section.

6.2 A Unified Model of Personalized, Interactive Question Answering

The research reported so far has demonstrated the utility of the techniques of personalization and interactivity to the Question Answering task. By observing their positive outcome, it is inevitable to regard the formulation of a unified model of personalized, interactive QA as a valuable by-product of these two technologies.

Indeed, designing the interaction between a User Modelling component and a dialogue manager is not a straight-forward task, which has rarely been approached in the literature, especially in Question Answering research.

Section 6.2.1 discusses the main modelling issues behind such integrated model. Moreover, Section 6.2.2 introduces previous work on the integration between dialogue and User Modelling in adaptive systems.

Section 6.2.3 presents a possible personalized, interactive Question Answering scenario performed by YourQA.

Section 6.2.4 provides a high-level overview of the architecture implementing such scenario and some issues that are still unsolved by the current strategy. Finally, Section 6.2.5 proposes future challenges to make the proposed unified architecture more powerful and effective.

6.2.1 Modelling Issues

Three general modelling issues can be drawn in modelling personalized, interactive Question Answering: these concern directionality, aspect and evaluation.

Directionality The aspect of directionality involves the following research questions:

1. Should the dialogue component affect the User Modelling component?
   This question inevitably depends on what aspects of the user are modelled by the UM component; for instance, if the user's reading level is the only attribute of the UM, it may not be necessary to use information deriving from the user's interaction with the chat interface to update such model, while other sources of information (e.g. documents read by the user, user's age, etc.) could prove to be more useful.

   Another related question is: if indeed the dialogue component affects the User Model, should the information from the dialogue history be exploited to update the User Model and how exactly?
   Assuming that the dialogue history carries any types of more or less latent information that are relevant to the User Model, such as the user's evolving interests or the frequency of occurrence of his/her clarification requests, these must be extracted and exploited efficiently. This involves on the one hand making such "latent" information explicit from the dialogue logs, and on the other deciding how frequently and when (on login, offline, etc.) to access it.

   Finally, a further question concerns what other aspects of the dialogue management strategy should affect the User Model, i.e. whether there are other ways through which the dialogue component can be useful to the UM which are not inherent to past interactions.

2. Should the User Modelling component influence the dialogue interface?
   The UM component could indeed have an impact not only on the behavior of the core QA component but also on the dialogue component's interface. One obvious way to achieve this would be to modify the format of user utterances based on a representation of the current user's preferences in terms of conciseness, or alternatively to simplify utterances based on the user's reading level.

   A directly related question is how should the format of the dialogue interface's utterances change to accommodate the individual users.
   We could imagine a Natural Language Generation component in charge of rendering the system's replies in different ways; if the dialogue interface is implemented

as a chatbot, a simpler alternative would be to pre-compile different templates for different reading levels and to select them using a conditional branch based on the value of the reading level.

Aspects The aspect issue concerns what attributes of the User Model should have a direct relation with the dialogue management strategy. As a matter of fact, both the reading level and the profile attributes of YourQA's current User Model only affect the behavior of YourQA's Question Answering component. The following research questions should be addressed concerning the UM attributes:

1. If and how each of the UM attributes –and of the further attributes which may be defined to model users of personalized QA systems in the future– should impact on the dialogue management strategy.

   Analogously, a second question is whether and how the UM attributes should affect the surface generation of text in the dialogue interface.

   Both questions are partly anticipated in the directionality issues, where the user's reading abilities and preferences in terms of system utterance length are seen as potentially useful to the dialogue management strategy and the dialogue interface.

2. If and in what way the dialogue management strategy and the dialogue interface should affect each User Model attribute.
   An aspect of the dialogue management strategy which could be represented in the User Model is for instance of how frequently the system is supposed to produce explicit clarification requests. In terms of dialogue interface, the influence on the User Model of the length or complexity of the system's utterances are potentially useful aspects to be implemented especially when modelling a QA application for children or people with reading disabilities.

Evaluation Thirdly, while methodologies for the evaluation of personalized QA and interactive QA have been proposed and carried out independently in Chapters 4 and 5, contributing to the research objectives of this thesis, a combined methodology for evaluating a Question Answering system resulting from the interaction between the two above technologies has never been proposed in the literature and still needs to be defined.

Assessing the *combined* contributions of personalization and interactivity is not an easy task: as mentioned previously, even a separate evaluation of the two components has barely been approached in the past. In terms of evaluation, interactivity and personalization could be seen as two separate ways to achieve the same goal; on the one hand, a perfectly matching personalization would spare the system from the need of interactivity, while on the other hand if the level of interaction was perfect, there would be no need to personalize results or rather personalization would already be achieved by the progressive clarification of the user's need through dialogue.

The above research questions involve an investigation that goes well beyond the scope of this thesis. However, as a first attempt to address them, this section proposes a roadmap for the creation of a unified model of Question Answering joining the aspects of personalization and interactivity.

Based on existing work on User Modelling for dialogue applications and on our study of personalization and dialogue in the context of QA, we describe a version of YourQA that is already capable of providing personalized results through a dialogue interface: this is the basis upon which further research is advocated to improve the proposed integration between User Modelling and dialogue.

The underlying model of such system is centered on the standard Question Answering module described in Chapter 2 which is responsible of efficiently extracting answers to both factoid and non-factoid questions using the Web. The QA module interacts with two satellite components: the User Modelling component described in Chapter 4, and the dialogue component described in Chapter 5.

The UM component is in charge of the personalization aspect and provides criteria to filter candidate answer documents and re-rank candidate answer passages based on their appropriateness to an individual model of the user's information needs; the dialogue component is responsible of correctly interpreting the user's information needs, of maintaining a notion of conversational context and of delivering answers efficiently through the dialogue interface.

6.2.2 Previous Approaches to User Modelling for Dialogue Applications

We have already mentioned that dialogue systems have been among the first fields of application in the history of User Models. Although not as popular as in e-commerce applications, there are several reasons which motivated the deployment of User Modelling approaches in dialogue systems and natural language applications.

Kass & Finin (1988) and Kobsa (1990) have summarized the positive effects of User Models in dialogue applications as follows:

1. Supporting the task of recognizing and interpreting the user's plans and goals;

2. Taking into account what the user probably already knows or does not know about a situation, and thus avoiding redundancy and incomprehensibility in its answers

and explanations.

3. Providing the user with tailored help and advice, such as handling misconceptions, and knowing when to volunteer information;

4. Eliciting information, getting input, and resolving ambiguity;

5. Providing output, i.e. deciding what to say and how to say it.

As it can be observed, this view puts the accent on the beneficial effects of UMs for natural language applications, without in fact mentioning the effects of dialogue interfaces on natural language applications. Indeed, natural language applications that perform User Modelling as described in the literature seem to regard the UM component as an accessory to planning and generation.

Nevertheless, it can be interesting to report one of the few approaches in this sense, that is among the few examples of dialogue based on User Modelling found in the literature: the Embodied Conversational Agent described in Cavalluzzi *et al.* (2003).

The Embodied Conversational Agent (ECA) interacts with the user to provide advice in a system providing healthy eating advice, a domain influenced by affective factors. The agent tries to show some form of emotional intelligence through a model of emotion activation in the ECA. This is represented with a Dynamic Belief Network, used as a goal monitoring system. The model generates a probabilistic model of the agent's mind at time T_{i+1}, based on the recent behavior of the agent and the model built at time T_i.

The dialogue manager in the prototype includes three main modules: a deliberative layer which selects the goal with the highest priority and the plan to achieve it; a reactive layer deciding whether the goal priority should be revised, by applying reacting rules; finally, an I/O communicative layer which executes the next action in the agenda.

Among the knowledge sources employed by these modules is a User Model. This model includes both long-term settings that are stable during the dialogue and influence the initial plan and behavior of the agent goals (the agent's personality, its role, its relationship with the user), and short-term settings, that evolve during the dialog, such as the emotional state of agent.

The User Model affects the planning strategy deployed by the system in several ways: changing the priorities of the subsequent utterances based on the user's current feelings about the topics under discussion, starting insertion sequences concerning a topic that seems of particular interest to the user or discarding the current plan for future utterances if these are perceived as no longer compatible with the user's current attitude towards them.

With respect to the approach described above, the requirements for a model of interaction between User Model and dialogue component in an open-domain QA setting are very different. In fact, as the domain of interaction is open, there is no attempt to implement a planning architecture in either component of the system as this would be a too complex task.

Moreover, in the above cases, the interaction between the UM component and the dialogue manager are uni-directional: the UM component is simply an accessory to the dialogue component and User Models do not draw any benefit from the dialogue process.

In the following section, we propose an algorithm for personalized interactive Question Answering which may be considered as an extension of the chat-based dialogue scenario proposed in Chapter 5 and of the personalized QA scenario in Chapter 4.

6.2.3 A Personalized, Interactive QA Scenario

In order to fulfill the aims for personalized, interactive QA indicated in Section 1.3, we propose the dialogue scenario illustrated in Algorithm 19.

In this scenario, the user starts interacting with the Question Answering system's dialogue interface that, once a user utterance is recognized as a question, submits the question to the core Question Answering module. The latter is charged of processing the question and retrieving a list of relevant documents by accessing a Web search engine.

As soon as relevant documents are available, the standard Question Answering module exchanges information with the User Modelling component based on which the QA component outputs personalized answers. These are returned to the user via the dialogue interface.

Finally, information from the dialogue history is periodically used to update the user model based on the current conversation and on the topics he/she has researched.

Such scenario integrates the standard Question Answering component with the User Modelling component and the dialogue component. The benefits of such integration are a complete model of QA in which the advantages of dialogue and personalization complete each other. Without the availability of a User Model representing the user's reading level and loaded at the beginning of the interaction, an only interactive system might need several clarification exchanges to eventually return results that are appropriate for the user. On the other hand, without a dialogue interface, even the most a personalized system would not be able to carry out the simple task of reference resolution.

The integrated architecture resulting from the combination of the UM and dialogue component is discussed in the following section.

Algorithm 19 Personalized, interactive Question Answering

1. The user interacts with the dialogue interface formulating an utterance q;
2. If q is recognized as a question q, it is analyzed by the dialogue manager (DM) which attempts to detect and resolve multiple and follow-up questions;
3. As soon as a clarified version of q is available, q', the DM passes q' to the QA module;
4. The QA module processes q' and retrieves a set $Retr(q')$ of relevant documents by accessing the underlying Web search engine;
5. As soon as $Retr(q')$ is available, the QA module exchanges information with the User Modelling component which is responsible of maintaining and updating the User Model, u;
6. Based on u, the QA module outputs a list $L(q', u)$ of personalized results;
7. The DM produces a reply r, which is returned along with $L(q', u)$ to the user via the dialogue interface;
8. The dialogue interface enquires about the user's satisfaction and/or proposes to carry on the conversation;
9. The current QA session is logged into the dialogue history $H(u)$;
10. $H(u)$ is used to update u;

6.2.4 High-Level Architecture

The first step towards a unified model of QA was the design of the standard QA prototype described in Chapter 2. A black-box view of the system is illustrated in Figure 6.1 from the user's viewpoint; here, the user submits a question to the system, which after the three phases of question processing, document retrieval and answer extraction returns a list of top answers.

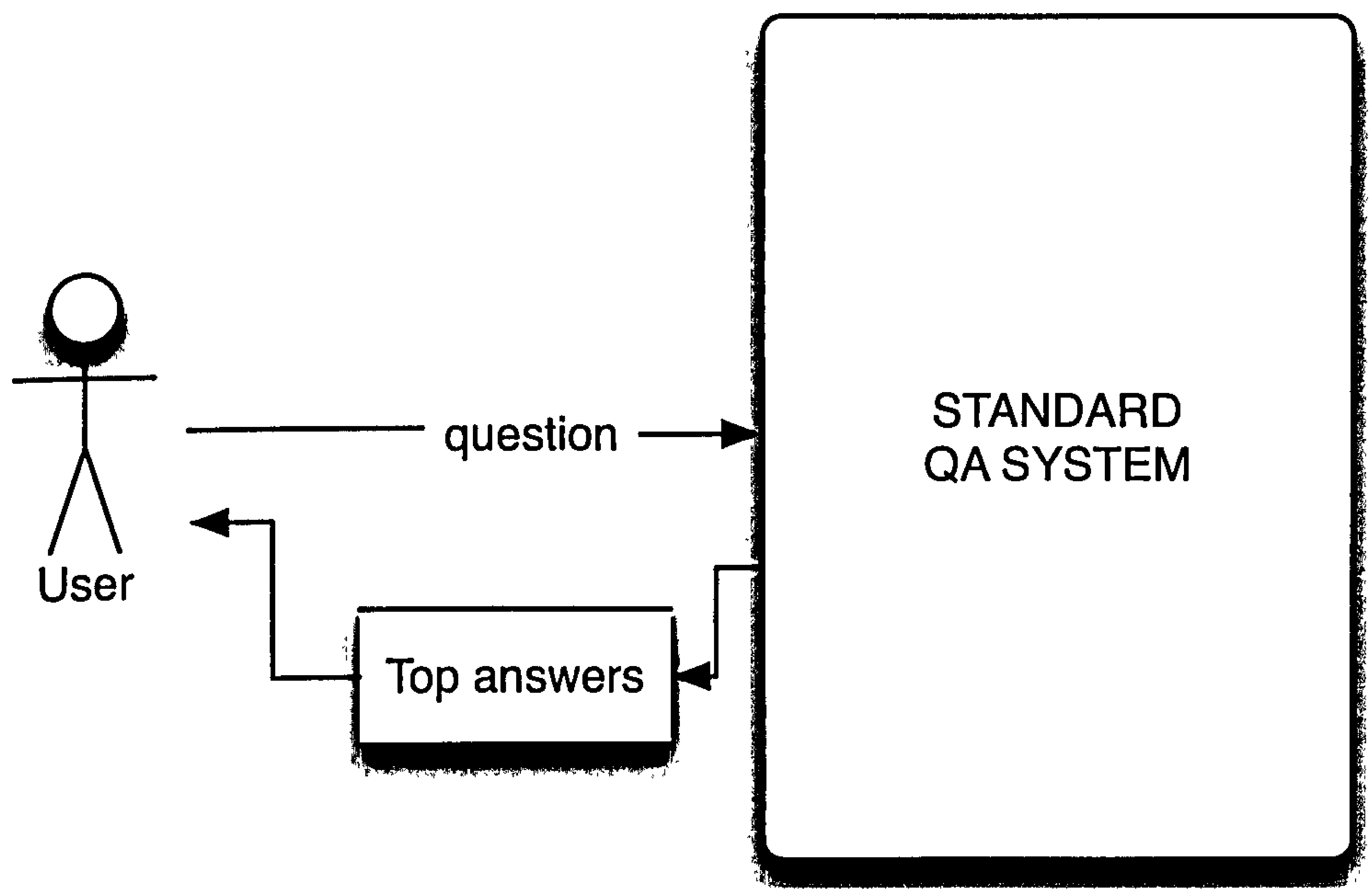

Figure 6.1: Black-box view of the standard version of YourQA

In a second stage, the User Modelling component was designed to interact with the baseline Question Answering system and provide personalized results, as described in Chapter 4 and illustrated in Figure 6.2.

The third, parallel step consisted in creating the dialogue component to add a layer of interactivity to the system. This resulted in the architecture defined in Chapter 5 and illustrated in Figure 6.3.

Final step: personalized, interactive architecture

The final step to implement the scenario in Section 6.2.3 involves the integration of the User Modelling component and dialogue component. A black-box view of the resulting system architecture is illustrated in Figure 6.4.

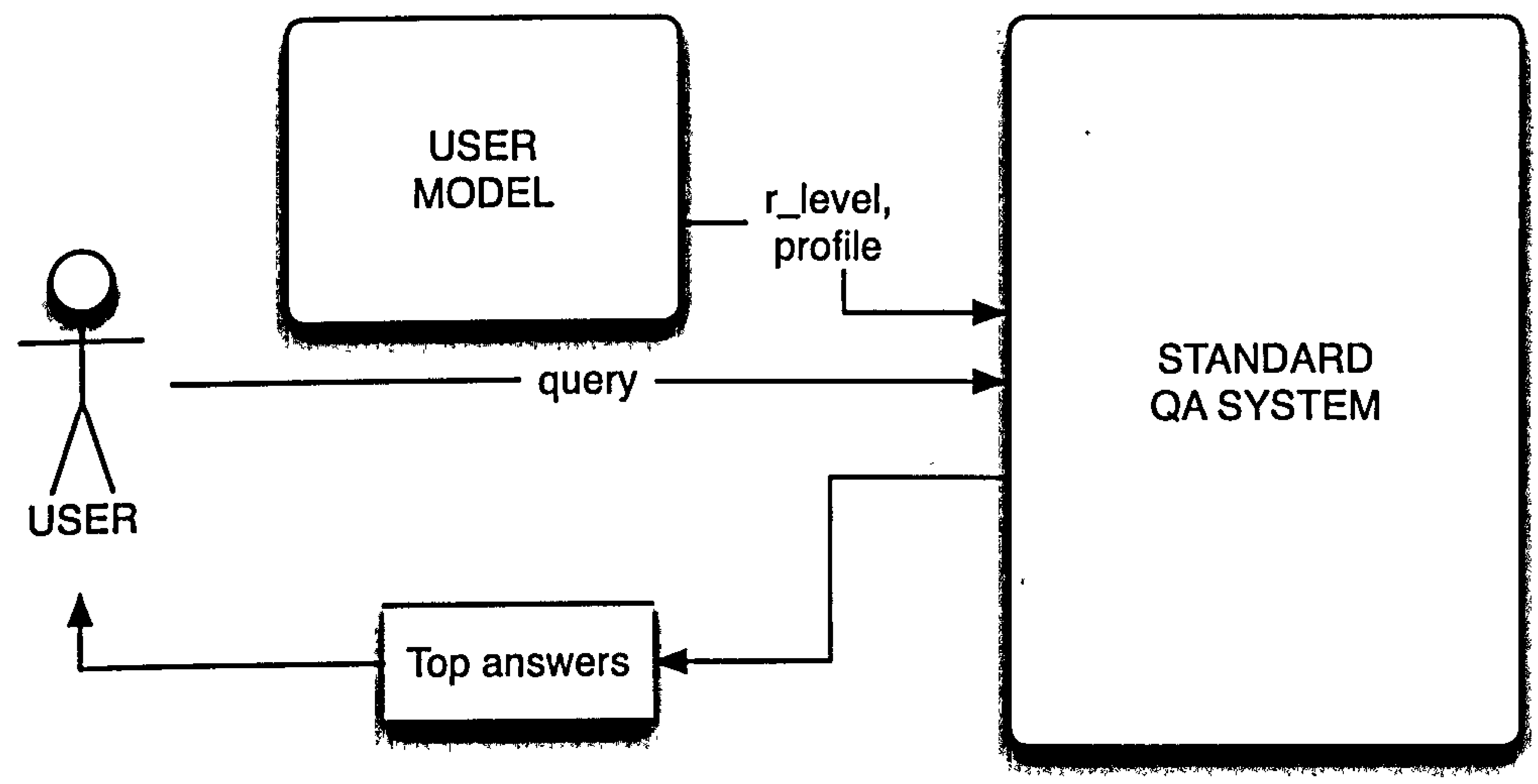

Figure 6.2: Black-box view of the personalized version of YourQA

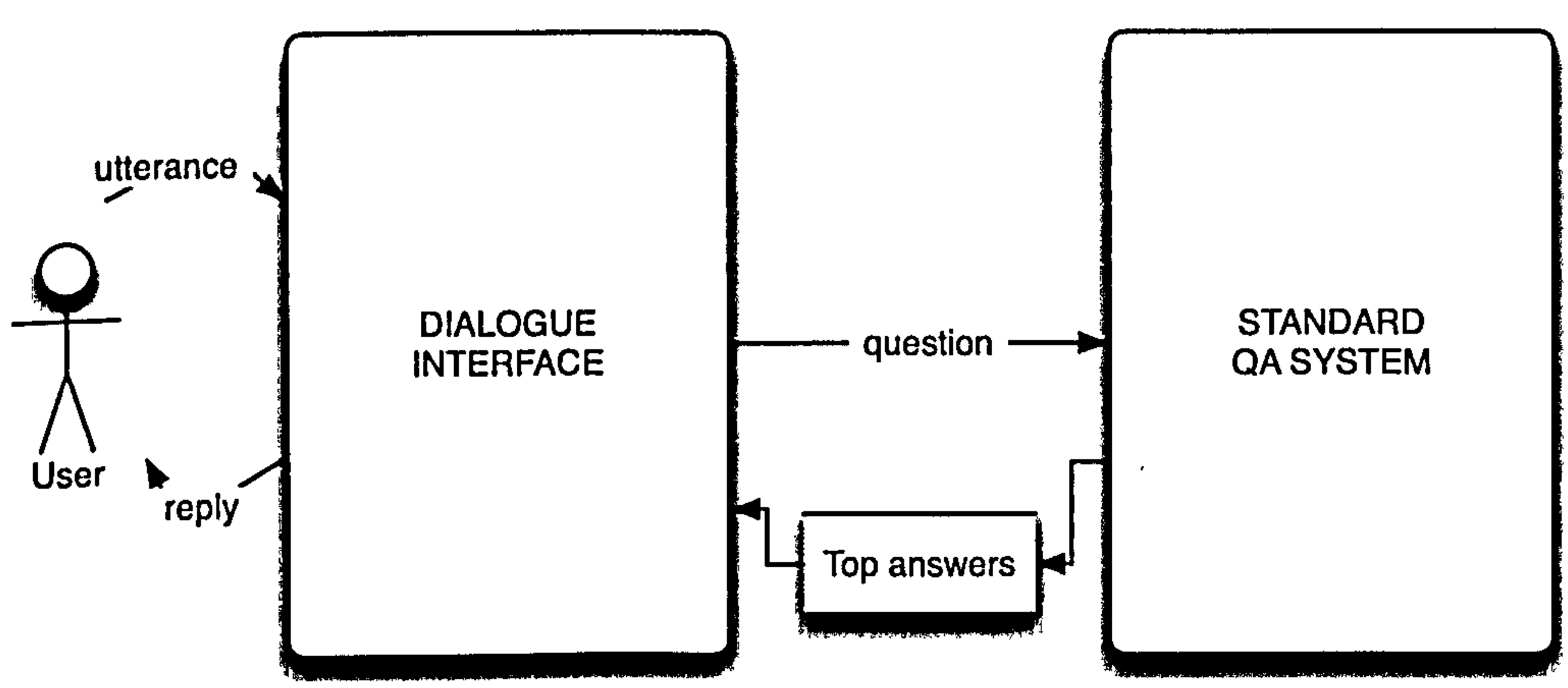

Figure 6.3: Black-box view of the interactive version of YourQA

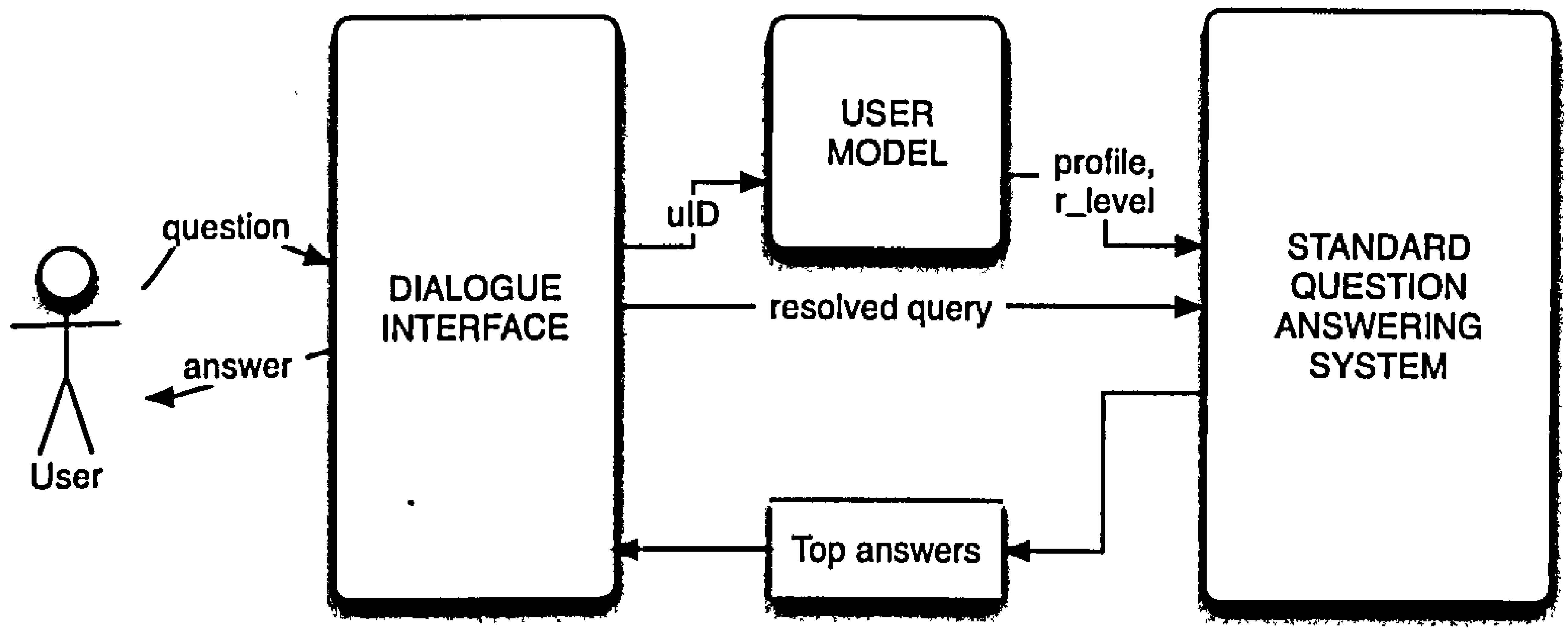

Figure 6.4: Black-box view of the personalized, interactive version of YourQA

In this architecture, the user interacts with the dialogue interface, providing questions. The dialogue interface detects among the user's utterances those that are actual questions, thanks to the QA patterns defined in AIML. Such questions are resolved according to the double and follow-up question resolution algorithm and a resolved query is created; this is ready to be submitted to the question processing module of the standard QA system.

A strategy to obtain the User Model which will be applied during document retrieval and answer extraction (hence to complete the Question Answering phase) is not yet specified at this stage. The current strategy used in YourQA for eliciting the User Model from the user relies on the definition of a context variable `userID` in the dialogue manager, which at the moment corresponds to the user's name. The strategy is explained in detail below.

Eliciting a User Model Categories 20 and 21 show how a value is obtained for the `userID` context variable. As it can be seen from Category 20, if the system does not know the user's name (i.e. this is a fresh QA session), the name is asked from the user during the initial greetings.

Then, such name is assigned to the `userID` variable, using categories such as Category 21. The `userID` variable will then be passed to the User Modelling component as a parameter of the user's query.

The User Modelling component accesses the database of currently defined User Mod-

Category 20 YourQA greeting category prompting for the user's name. If the `userID` is uninitialized, the system will refer to the user as "dear" and will try to obtain the user's name. Otherwise, Category `HELLO1` will be invoked.

```
<category>
<pattern>HELLO</pattern>
<template>
<condition>
<li name="userID" value="dear">Hi!  Who is this?</li>
<li name="userID" value="dear">Hi there!  What's your
name?</li>
<li><srai>HELLO1</srai></li>
</condition>
</template>
</category>
```

Category 21 YourQA category assigning the user's name to `userID`

```
<category>
<pattern>*</pattern>
<that>* WHO IS THIS</that>
<template>
<think>
<set name="userID"><star/></set>
</think>
<srai>HELLO1</srai>
</template>
</category>
```

els by using the `userID` as a key, hence obtaining the UM to be applied during document retrieval and answer extraction. Once results are available from the standard QA system, following the interactive scenario described in Chapter 5, the dialogue interface outputs an answer to the user and visualizes the HTML page containing the results.

Figure 6.5 illustrates an example of a personalized, QA session in YourQA. Here, the user's name is associated with a UM with a basic reading level. This affects the document retrieval phase, where only documents with simple wordings are retained for answer extraction.

Discussion

The passage of information between the User Modelling component and the dialogue manager remains uni-directional in this first example. By this, we mean that there is no actual exchange between the two components, as the only link between the dialogue

Category 22 The YourQA `HELLO1` category, greeting the user with his/her name. This category is invoked by Categories 20 and 21 once the name of the current user is known.

```
<category>
<pattern>HELLO1</pattern>
<template>
<random>
<li>Hi, <get name="userID"/>!  How can I help you?</li>
<li>Hi there!  What's your question?</li>
<li>Hi there.  I was just waiting for your question, <get
name="userID"/>.</li>
<li>Hello <get name="userID"/>!  What would you like to
ask?</li>
</random>
</template>
</category>
```

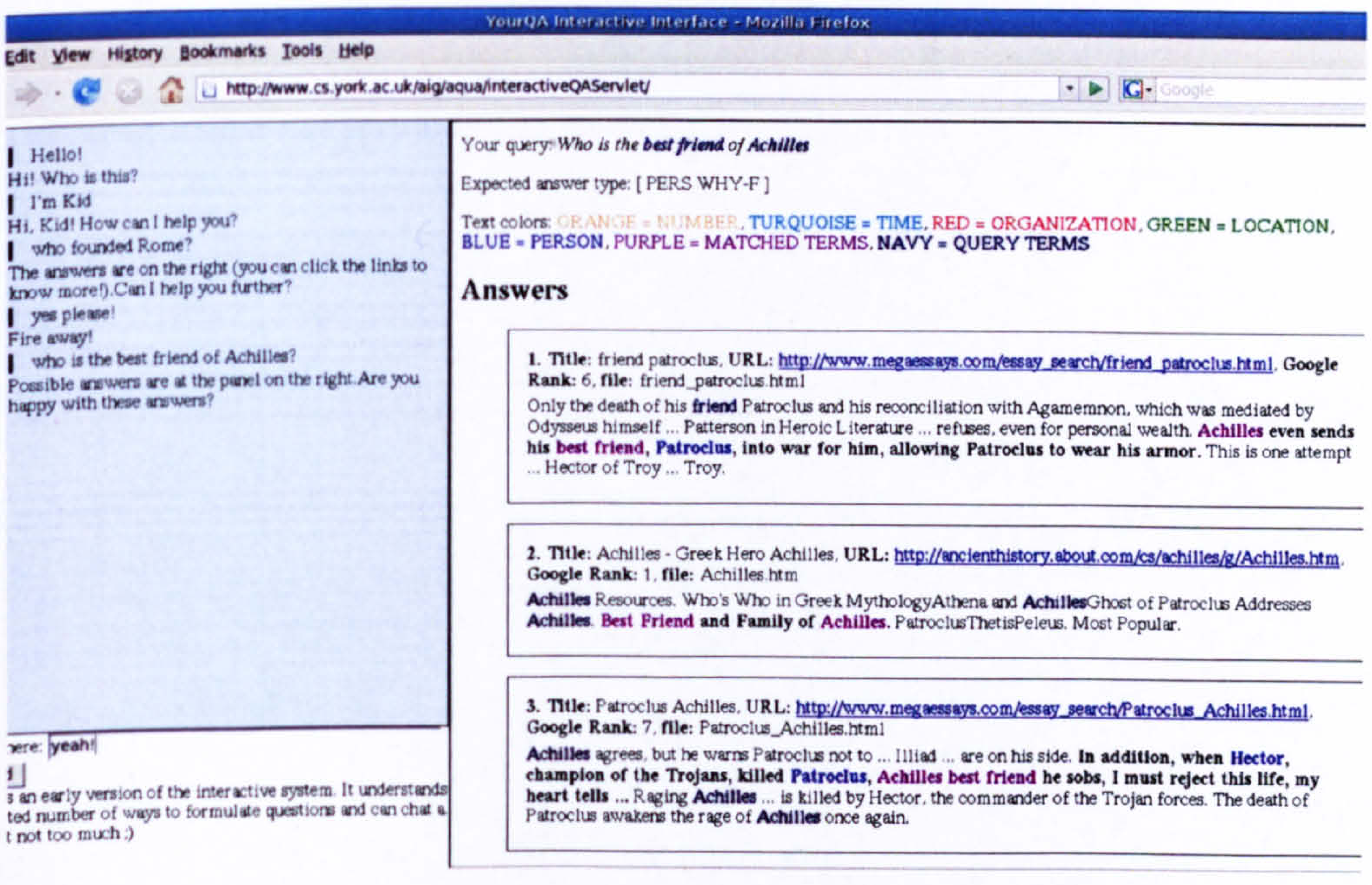

Figure 6.5: Screenshot from a personalized, interactive QA session. Here, the user's name ("Kid") is associated with a User Model requiring a basic reading level, hence the candidate answer documents are filtered accordingly.

component and the User Modelling component is the passing of the `userID` parameter from the dialogue interface to the UM component. The UM component provides no information to the dialogue component aiming at modifying its behavior, and the impact of the User Model is perceived only by the QA component which creates personalized results.

In the future, more complex models of interaction should be devised, for instance by tailoring not only the content of answers to the users, but also the system's expressivity, or by adding affect as in the system by Cavalluzzi *et al.* (2003).

Interesting work in natural language understanding and User Modelling could happen at discourse level, involving for instance the adaptation of the dialogue to user preferences such as short replies or lesser occurrences of requests for clarification as the user tends to respond positively to all of them. Section 6.2.5 proposes future challenges for personalized, interactive QA in this and other respects.

6.2.5 Future Challenges

Many extensions can be made to complete the personalized, interactive framework proposed above, and to respond to the research questions formulated in this Chapter.

First, it must be pointed out that the advanced techniques developed in Chapter 3 for addressing complex answers have not yet been integrated in the framework. This is because the SVM-based classifier and re-ranker are not yet real-time efficient and further studies must be completed concerning their abilities.

Other than the above-mentioned aspect, a first extension would be a study of efficient strategies for the creation of User Models based on current and past conversations with the user in question. Indeed, while the current strategy – mapping user names (used as IDs) to User Models – assumes that the latter are "hand-coded", a study of efficient strategies for the creation of UMs during the conversation must still be carried on.

Moreover, the problem of updating user interests and reading levels based on the dialogue history, previously mentioned in Chapter 4, is yet to be solved.

Obviously, another important issue in natural language applications is to generate coherent and relevant discourse. As noted in Moore & Paris (1989), the amount of detail in an explanation as well as the kind of information given should depend on the user's expertise in a domain, which could be represented in the User Model for a number of subjects. The user's personal preference in interaction style (utterance length, follow-up proposals, etc.), and knowledge about terminology could affect the actual words used at surface generation.

Finally, users may not always want their own profile to be active during information seeking and prefer standard results. Hence, the AIML patterns defined by the system must accommodate on the one hand system proposals to apply a User Model, and on the other hand ways for the users to specify when they require information filtering according to a different reading level (or none at all) and to select or reject the use of their profile.

Chapter 7

Conclusions

This thesis contributes to the field of open-domain Question Answering (QA) through the design and deployment of a model of a Web-based QA system offering several advanced features with respect to the state-of-the-art.

The first contribution of this thesis is the development of core Question Answering techniques, leading to the implementation of a Web-based system, YourQA, to serve as a baseline for the study on complex QA and for the extensions achieving personalization and interactivity. The core QA module designed for this purpose is structured according to the three-layer architecture of current QA systems, i.e. question processing, document retrieval and answer extraction (see Chapter 2).

A second contribution of this thesis is the efficient approach of complex questions, such as definitions. The approach consists in an investigation of the impact of syntactic and shallow semantic information (such as semantic role annotations) in vital tasks in the QA process: question classification, answer classification and answer re-ranking. This is conducted via the study of tree kernel functions implemented in Support Vector Machines using such complex textual features.

The outcome of this research, reported in Chapter 3, is applied first to improve the question classification performance of the question processing module in the YourQA architecture. Then, we drastically improve the performance of a baseline answer extractor by automatically re-ranking answers to complex questions based on the newly introduced data representations and machine learning algorithm, thus contributing to the solution of the problem of addressing complex questions.

Thirdly, this thesis reports one of the first full-fledged applications of personalized open-domain Question Answering. Personalization is achieved through the implementation of a User Modelling component interacting with the standard Question Answering system: the former's tasks are mainly to filter answers based on the user's reading abilities and and to re-rank them based on the latter's interests.

In Chapter 4, personalization is demonstrated to be a useful approach for Question Answering by defining an evaluation methodology comparing the "standard" and the "personalized" versions of the QA system on the grounds of user satisfaction. These experiments reveal an important positive contribution of User Modelling to QA.

Moreover, this thesis presents the design of an information-seeking dialogue management strategy suitable for Interactive Question Answering. First, the requirements for modelling Interactive Question Answering are discussed, and a dialogue model based on such requirements is studied (see Chapter 5). Furthermore, a chatbot-based interface, which is able to maintain a notion of the recent conversational context and to resolve the most common types of anaphoric and elliptic expressions, is implemented.

Since chatbot-based dialogue is a new application for Question Answering, the theoretical and design assumptions are first validated in the course of an exploratory Wizard-of-Oz study. The encouraging results of such experiment lead to the implementation of an interactive version of YourQA interactive interface. A methodology for the evaluation of such interactive QA prototype is discussed and performed with remarkable results.

A further contribution of this thesis is the construction and deployment of a proof-of-concept of the proposed approaches to Question Answering. Three different versions of such prototype system have been implemented:

- A standard version, i.e. a basic Web-based open-domain QA system;
- A personalized version, which constructs and applies User Models to adjust to the reading level and interests of individual users;
- An interactive version, able to interact with the user through a chatbot interface, maintain the interaction context and resolve anaphoric and elliptic utterances.

Drawing from the experience collected during the research on personalization and interactivity, the final contribution of this thesis is the creation of a comprehensive model of personalized, interactive Question Answering. The model integrates the User Modelling techniques and the model of dialogue management developed so far in a new unified concept of Question Answering, as described in Chapter 6.

A Look into the Future

A few issues still remain unsolved in current Question Answering, and deserve to be pointed out for future research: first, the problem of determining the criteria to assess the quality of answers to non-factoid questions such as definitions is still an open issue. In this perspective, it is in our opinion mandatory to rely on semantic representations of text, such as those relying on FrameNet and PropBank, to identify the complex semantic relationships between entities in text.

Moreover, research on personalized Question Answering is yet at a pioneering stage and much remains to be done at the modelling level (i.e. deciding how to design suitable User Model attributes to represent users of QA systems) and as far as the interaction between the User Modelling component and the other components of a QA system is concerned. In particular, the relationship between User Modelling and dialogue is a notable area of study.

Interactive Question Answering is also at an early age and many research issues remain open, in particular when the QA task is deployed in an open-domain setting. Further to the problem of assessing the most suitable dialogue management strategy for this type of task, the problem of evaluation appears in this case as a most urgent one. Indeed, this aspect has traditionally been a weakness of modelling dialogue systems.

Finally, the model of personalized, interactive Question Answering we have proposed sets the road for a new unified concept of QA. Much remains to be investigated as far as the interaction between personalization and interactivity is concerned; for this reason, we believe that this is a very exciting area for future QA research.

Appendix A

Publications

The following is a chronologically ordered list of publications which have contributed to this thesis.

2007

- S. Quarteroni and S. Manandhar, *Designing an Open-Domain Interactive Question Answering System.* To appear in: Journal of Natural Language Engineering, Special issue on Interactive Question Answering. N. Webb, B. Webber (Eds.), Cambridge University Press, 2007.
- S. Quarteroni and S. Manandhar, *User Modelling for Personalized Question Answering.* In: AI*IA 2007: Artificial Intelligence and Human-Oriented Computing, Proceedings of the 10th Conference of the Italian Association for Artificial Intelligence (AI*IA '07). R. Basili, M.T. Pazienza (Eds.), Springer LNAI Vol. 4733, Heidelberg, Germany, October 2007.
- S. Quarteroni and S. Manandhar, *A Chatbot-based Interactive Question Answering System.* In: Proceedings of DECALOG (SEMDIAL '07). R. Artstein, L. Vieu (Eds.), pp. 83-90, Rovereto, Italy, May 2007.
- A. Moschitti, S. Quarteroni, R. Basili and S. Manandhar, *Exploiting Syntactic and Shallow Semantic Kernels for Question/Answer Classification.* In: Proceedings of the 45th Conference of the Association for Computational Linguistics (ACL). A. Zaenen, A. van den Bosch (Eds.).ACL press, Prague, Czech Republic, June 2007.
- S. Quarteroni, A. Moschitti, S. Manandhar and R. Basili, *Advanced Structural Representations for Question Classification and Answer Re-ranking.* In: Advances in Information Retrieval, 29th European Conference on IR Re-

search, ECIR 2007, Rome, Italy, April 2-5, 2007, Proceedings. G. Amati, C. Carpineto, G. Romano (Eds.), Lecture Notes in Computer Science Vol. 4425 Springer, Heidelberg, 2007.

2006

- S. Quarteroni and S. Manandhar, *Incorporating User Models in Question Answering to Improve Readability.* In: Proceedings of KRAQ'06. P. St-Dizier, F. Benamara (Eds.). Trento, Italy, April 2006.
- S. Quarteroni and S. Manandhar, *Adaptivity in Question Answering with User Modelling and a Dialogue Interface* (short paper). In: Proceedings of EACL'06. D. McCarthy, S. Wintner (Eds.). Trento, Italy, April 2006.
- S. Quarteroni and S. Manandhar, *User Modelling for Adaptive Question Answering and Information Retrieval.* In: Proceedings of FLAIRS-19. G. Sutcliffe, R. Goebel (Eds.). Melbourne Beach, FL, USA, May 2006.

2005

- S. Quarteroni and S. Manandhar, *Adaptivity in Question Answering Using Dialogue Interfaces.* In: Proceedings of the Workshop on Cultural Heritage - 9th Conference of the Italian Association for Artificial Intelligence (AI*IA 2005). S. Bandini (Ed.). Milan, Italy, September 2005.

Bibliography

Abbattista, F., Degemmis, M., Fanizzi, N., Licchelli, O., Lops, P., Semeraro, G., & Zambetta, F. 2002. *Learning User Profiles for Content-Based Filtering in e-Commerce.*

Adams, D. 1979. *The Hitchhiker's Guide to the Galaxy.* London, UK: Pan Books.

Adams, D. 1985. *The Original Hitchhiker Radio Scripts.* London, UK: Pan Books.

Ahn, D., Jijkoun, V., Mishne, G., Mueller, K., de Rijke, M., & Schlobach, S. 2004. Using Wikipedia at the TREC QA Track. *In: Proceedings of TREC 2004.*

Alexandersson, J., & Becker, T. 2001. Overlay as the Basic Operation for Discourse Processing in a Multimodal Dialogue System. *In: Proceedings of 2nd IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems.*

Alexandersson, J., Reithinger, N., & Maier, E. 1997. *Insights into the dialogue processing of VERBMOBIL.* Tech. rept. 191. Saarbrücken, Germany.

Alfonseca, E., & Rodriguez, P. 2003. Modelling Users' Interests and Needs for an Adaptive Online Information System. *In: User Modelling: Proceedings of the 9th International Conference.* LNAI/LNCS, vol. 2702. Springer.

Allan, J., Aslam, J., Belkin, N., & Buckley, C. 2002. Challenges in IR and language modeling. *In: Report of a Workshop at the University of Amherst.*

Allen, J., Ferguson, G., Ringger, E., Zollo, T. Sikorski, & Miller, B. 2000. Dialogue systems : from theory to practice in TRAINS-96. *Pages 347–376 of:* Dale, R., Moisl, H., & Somers, H. (eds), *Handbook on Natural Language Processing.* R. Dale and H. Moisl and H. Somers.

Allen, J. F., & Perrault, C. R. 1980. Analyzing intention in utterances. *Journal of Artificial Intelligence.*

Anderson, A. H., & Bader, M. 1991. The HCRC Map Task Corpus. *Language and Speech*, 34, 351–366.

Andrews, P., Manandhar, S., & De Boni, M. 2006. Integrating Emotions in Persuasive Dialogue: A Multi-Layer Reasoning Framework. *In: Proceedings of Flairs 19, Special Track: Trends in Natural Language Processing.*

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

Austin, J.L. 1962. *How to do things with words.* Oxford, UK: Oxford University Press.

Baker, C. F., Fillmore, C. J., & Lowe, J. B. 1998. The Berkeley FrameNet Project. *Pages 86–90 of: Proceedings of the 17th international conference on Computational linguistics.* Morristown, NJ, USA: Association for Computational Linguistics.

Barrett, R., Maglio, P. P., & Kellem, D. C. 1997. How to Personalize the Web. *In: Proceedings of SIGCHI'97.*

Basili, R., Cao, D. De, Giannone, C., & Marocco, P. 2007. Data-Driven Dialogue for Interactive Question Answering. *In: Proceedings of AI*IA'07.*

Belkin, N. J., & Croft, W.B. 1992. Information Filtering and Information Retrieval: Two Sides of the Same Coin? *Communications of the ACM*, 35(12), 29–38.

Bell, T. C., Moffat, A., Witten, I. H., & Zobel, J. 1995. The MG retrieval system: compressing for space and speed. *Communications of the ACM*, 38(4), 41–42.

Bertomeu, N., Uszkoreit, H., Frank, A., Krieger, H., & Joerg, B. 2006. Contextual phenomena and thematic relations in database QA dialogues: results from a Wizard-of-Oz experiment. *In: Proceedings of IQA'06.*

Blair-Goldensohn, S., McKeown, K. R., & Schlaikjer, A. H. 2003. DefScriber: a hybrid system for definitional QA. *Pages 462–462 of: SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval.* New York, NY, USA: ACM Press.

Bos, J., Klein, E., Lemon, O., & Oka, T. 2003. DIPPER: Description and Formalisation of an Information-State Update Dialogue System Architecture. *In: Proceedings of SIGdial'03.*

Brusilovsky, P., & Tasso, C. 2004. User modeling for Web information retrieval. *Special issue of User Modeling and User-Adapted Interaction*, 14(2-3).

Buchholz, S., & Daelemans, W. 2001. Complex Answers: A Case Study using a WWW Question Answering System. *Natural Language Engineering, Special Issue on Question Answering*, 14(3), 130–137.

Bueno, D., & David, A. A. 2001. METIORE: A Personalized Information Retrieval System. *In: User Modelling: Proceedings of the 8th International Conference*. LNCS, vol. 2109. Springer.

Bui, T., & Rajman, M. 2004. *Rapid Dialogue Prototyping Methodology*. Tech. rept. 2004-01-05. EPFL, Switzerland.

Cahn, J. E., & Brennan, S. E. 1999. A Psychological Model of Grounding and Repair in Dialog. *Pages 25–33 of:* Brennan, Susan E., Giboin, Alain, & Traum, David (eds), *Working Papers of the AAAI Fall Symposium on Psychological Models of Communication in Collaborative Systems*. Menlo Park, California: American Association for Artificial Intelligence.

Carlson, R. 1996. The dialog component in the Waxholm system. *Pages 209–218 of:* S., LuperFoy, A., Nijholt, Veldhuijzen, & van Zanten G. (eds), *Proceedings of Twente Workshop on Language Technology. Dialogue Management in Natural Language Systems (TWLT 11)*.

Carreras, X., & Màrquez, L. 2005. Introduction to the CoNLL-2005 Shared Task: SRL. *In: Proceedings of CoNLL-2005*.

Carroll, J., Minnen, G., Pearce, D., Canning, Y., Devlin, S., & Tait, J. 1999. Simplifying text for language-impaired readers. *Pages 269–270 of: Proceedings of EACL'99*.

Cavalluzzi, A., Carolis, B. De, Carofiglio, V., & Grassano, G. 2003. Emotional Dialogs with an Embodied Agent. *In: Proceedings of UM'03*.

Charniak, E. 2000. A Maximum-Entropy-Inspired Parser. *In: Proceedings of NAACL'00*.

Chen, Y., Zhou, M., & Wang, S. 2006. Reranking answers from definitional QA using language models. *In: ACL'06*.

Chin, D. N. 1994. KNOME: Modeling What the User Knows in UC. *In:* Kobsa, A., & Wahlster, W. (eds), *User Models in Dialog Systems*. Springer Verlag, Symbolic Computation Series.

Chin, D. N. 2001. Empirical Evaluation of User Models and User-Adapted Systems. *User Modeling and User-Adapted Interaction*, 11(1-2), 181–194.

Chu-Carroll, J., Prager, J., Welty, C., Czuba, K., & Ferrucci, D. 2002. A Multi-Strategy and Multi-Source Approach to Question Answering. *In: Proceedings of TREC'02.*

Churcher, G. E., Atwell, E. S., & Souter, C. 1997. *Dialogue Management Systems: a Survey and Overview*. Tech. rept. 97.06.

Cohen, P. 1996. *Dialogue Modeling*. Cambridge University Press, Cambridge. Chap. 6.3, pages 192–197.

Cohen, P., & Perrault, C. R. 1979. Elements of a plan-based theory of speech acts. 423–440.

Collins, M. 1999. *Head-driven statistical models for natural language parsing*. Ph.D. thesis, University of Pennsylvania.

Collins, M., & Duffy, N. 2002. New Ranking Algorithms for Parsing and Tagging: Kernels over Discrete Structures, and the Voted Perceptron. *In: Proceedings of ACL'02.*

Collins-Thompson, K., & Callan, J. P. 2004. A language modeling approach to predicting reading difficulty. *In: Proceedings of HLT/NAACL'04.*

Core, M. G., & Allen, J. F. 1997. Coding Dialogs with the DAMSL Annotation Scheme. *In: Proceedings of the 1997 AAAI Fall Symposium on Communicative Actions in Humans and Machines.*

Cui, H., Kan, M., & Chua, T. 2005. Generic soft pattern models for definitional QA. *In: Proceedings of SIGIR'05*. Salvador, Brazil: ACM.

Dahlbaeck, N., & Jonsson, A. 1998. *A coding manual for the linkoping dialogue model.*

Dahlbaeck, N., Jonsson, A., & Ahrenberg, L. 1993. Wizard of Oz studies: why and how. *Pages 193–200 of: Proceedings of IUI '93*. New York, NY, USA: ACM Press.

D'Avanzo, E., Magnini, B., & Vallin, A. 2004. Keyphrase Extraction for Summarization Purposes: The LAKE System at DUC-2004. *In: Proceedings of the Document Understanding Conference (DUC'04).*

De Boni, M. 2004. *Relevance in the open domain question answering: theoretical framework and application*. Ph.D. thesis.

De Boni, M., & Manandhar, S. 2003. An Analysis of Clarification Dialogue for Question Answering. *In: Proceedings of HLT-NAACL'03.*

De Boni, M., & Manandhar, S. 2005. Implementing Clarification Dialogue in Open-Domain Question Answering. *Journal of Natural Language Engineering*, **11**.

Dimitrova, V. 2003. Using dialogue games to maintain diagnostic interactions. *Pages 117–121 of: User Modelling: Proceedings of the 9th International Conference.* LNAI/LNCS. Springer.

Dumais, S., Banko, M., Brill, E., Lin, J., & Ng, A. 2002. Web question answering: is more always better? *Pages 291–298 of: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR'02).*

Dyer, M. G. 1983. *In-depth understanding.* MIT Press.

Fellbaum, C. 1998. A semantic network of English: the mother of all WordNets. 137–148.

Ferragina, P., & Gulli, A. 2005. A personalized search engine based on web-snippet hierarchical clustering. *Pages 801–810 of: WWW '05: Special interest tracks and posters.* New York, NY, USA: ACM Press.

Fikes, R., & Nilsson, N. 1971. Strips: A new approach to the application of theorem proving to problem solving. *Pages 189–205 of: Proceedings of IJCAI'71.*

Finin, T. 1986. A General User Modeling System. *Journal of User Modeling.*

Frank, E., Paynter, G. W., Witten, I. H., Gutwin, C., & Nevill-Manning, C. G. 1999. Domain-Specific Keyphrase Extraction. *Pages 668–673 of: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI '99).* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Fry, E. 1969. A readability formula that saves time. *Journal of Reading*, **11**(7), 265–71.

Fujii, A., & Ishikawa, T. 2002 (Feb. 27). *Question Answering Using Encyclopedic Knowledge.*

Galibert, O., Illouz, G., & Rousset, S. 2005. Ritel: an open-domain, human-computer dialogue system. *In: Proceedings of INTERSPEECH'05.*

Gaudioso, Elena, & Boticario, Jesus G. 2002. User Data Management and Usage Model Acquisition in an Adaptive Educational Collaborative Environment. *In: Lecture Notes in Computer Science*. LNCS, vol. 2347. Springer.

Ge, N., Hale, J., & Charniak, E. 1998. A statistical approach to anaphora resolution. *Pages 161–171 of: Proceedings of the Sixth Workshop on Very Large Corpora.*

Ginzburg, J. 1996. *Interrogatives: Questions, Facts and Dialogue*. Blackwell, Oxford.

Ginzburg, J., & Gregory, H. 2001. SHARDS: Fragment Resolution in Dialogue. *Pages 156–172 of:* Bunt H., van der Slius, Thijsse (ed), *Proc. of the 4th International conference on Computational Semantics.*

Ginzburg, J., & Sag, I. A. 2004. *A semantics for interaction in dialogue*. CSLI Publications.

Goddeau, D., Meng, H., Polifroni, J., Seneff, S., & Busayapongchai, S. 1996. A Form-Based Dialogue Manager for Spoken Language Applications. *Pages 701–704 of: Proc. ICSLP '96*, vol. 2.

Graziola, I., Pianesi, P., Zancanaro, M., & Goren-Bar, D. 2005 (January). Dimensions of Adaptivity in Mobile Systems: Personality and People's Attitudes. *In: Proceedings of Intelligent User Interfaces (IUI'05).*

Green, B. F., Wolf, A. K., Chomsky, C., & Laughery, K. 1961. BASEBALL: An automatic question answerer. *Pages 219–224 of: Proceedings of the Western Joint Computer Conference*, vol. 19.

Grosz, B. J., & Sidner, C. L. 1986. Attention, Intentions, and the Structure of Discourse. *Computational Linguistics*, 12(3), 175–204.

Habieb-Mammar, H., Tarpin-Bernard, F., & Prevut, P. 2003. Adaptive Presentation of Multimedia Interface Case Study: "Brain Story" Course. *In: User Modelling: Proceedings of the 9th International Conference*. LNAI/LNCS, vol. 2702. Springer.

Harabagiu, S., Pasca, M., & Maiorano, S. 2000. Experiments with open-domain textual question answering. *In:* Kaufmann, Morgan (ed), *Proceedings of COLING-2000.*

Hearst, M., Allen, J., Horvitz, E., & Guinn, C. 1999 (Sept/Oct). *Mixed Initiative Interaction*. IEEE Intelligent Systems.

HERMJAKOB, U., Hovy, E. H., Lin, C., & Ravichandran, D. 2002. Using Knowledge to Facilitate Pinpointing of Factoid Answers. *In: Proceedings of the COLING-2002 Conference.*

Hickl, A., & Harabagiu, S. 2006. Enhanced interactive question answering with conditional random fields. *In: Proceedings of IQA'06.*

Hobbs, J. R. 2002. From Question-Answering to Information-Seeking Dialogs.

Hori, C., Hori, T., Tsukada, H., Isozaki, H., Sasaki, Y., & Maeda, E. 2003. Spoken interactive ODQA system: SPIQA. *Pages 153–156 of: ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics.* Morristown, NJ, USA: Association for Computational Linguistics.

Horvitz, E., & Paek, T. 2001. Harnessing Models of Users' Goals to Mediate Clarification Dialog in Spoken Language Systems. *In: User Modelling: Proceedings of the 8th International Conference.* LNCS, vol. 2109. Springer.

Hovy, E., Gerber, L., Hermjakob, U., Junk, M., & Lin, C. 2000. *Question Answering in Webclopedia.*

Hulstijn, J. 1996. Topics in SCHISMA Dialogues. *Pages 88–89 of: Proceedings of the 11th Twente Workshop on Language Technology.*

Inui, K., Fujita, A., Takahashi, T., Iida, R., & Iwakura, T. 2003. Text simplification for reading assistance: a project note. *Pages 9–16 of: Proceedings of the ACL 2003 Workshop on Paraphrasing: Paraphrase Acquisition and Applications.*

Jiang, J. J., & 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).*

Joachims, T. 1999. Making large-Scale SVM Learning Practical. *In:* Schölkopf, B., Burges, C., & Smola, A. (eds), *Advances in Kernel Methods - Support Vector Learning.*

Jönsson, A. 1993. *A Method for Development of Dialogue Managers for Natural Language Interfaces.*

Jönsson, A., & Merkel, M. 2003. Some Issues in Dialogue-Based Question-Answering. *In: Working Notes from the AAAI Spring Symposium '03.*

Kaisser, M., & Webber, B. 2007. Question Answering based on Semantic Roles. *In: Proceedings of the ACL 2007 Deep Linguistic Processing Workshop, ACL-DLP 2007.*

Kaplan, S. J. 1979. *Cooperative responses from a portable natural language data base query system.* Ph.D. thesis, University of Pennsylvania.

Karis, D., & Dobroth, K. M. 1991. Automating Services with Speech Recognition over the Public Switched Telephone Network: Human Factors Considerations. *IEEE Journal on Selected Areas in Communications*, 9(4)(4), 574–585.

Kass, R., & Finin, T. 1988. Modeling the User in Natural Language Systems. *Computational Linguistics*, 14(3), 5–22.

Kato, T., Fukumoto, J., F.Masui, & Kando, N. 2006. WoZ simulation of interactive question answering. *In: Proceedings of IQA'06.*

Katz, B., Bilotti, M., Felshin, S., Fernandes, A., Hildebrandt, W., Katzir, R., Lin, J., Loreto, D., Marton, G., Mora, F., & Uzuner, O. 2004. Answering multiple questions on a topic from heterogeneous resources. *In: Proceedings of the 13th Annual Text REtrieval Conference (TREC2004).*

Katz, B., Marton, G., Borchardt, G., Brownell, A., Felshin, S., Loreto, D., Louis-Rosenberg, J., Lu, B., Mora, F., Stiller, S., Uzuner, O., & Wilcox, A. 2005. External Knowledge Sources for Question Answering. *In: Proceedings of TREC2005.*

Kazawa, H., Isozaki, H., & Maeda, E. 2001. NTT Question Answering system in TREC 2001. *In: Proceedings of TREC'01.*

Kelly, D., & Lin, J. 2007. Overview of the TREC 2006 ciQA Task. *SIGIR Forum*, 41(1), 107–116.

Kelly, D., Kantor, P., Morse, E., Scholtz, J., & Sun, Y. 2006. User-centered evaluation of Interactive question answering systems. *In: Proceedings of IQA'06.*

Kincaid, J., Fishburne, R., Rodgers, R., & Chissom, B. 1975. *Derivation of new readability formulas for navy enlisted personnel.* Tech. rept. 8-75. Chief of Naval Training.

Kingsbury, P., & Palmer, M. 2002. From Treebank to PropBank. *In: Proceedings of LREC'02.*

Kitano, H., & Ess-Dykema, C. Van. 1991. Toward a Plan-Based Understanding Modèl for Mixed-Initiative Dialogues. *Pages 25–32 of: Proceedings of ACL'91.*

Kobsa, A. 1990. Modeling the user's conceptual knowledge in BGP-MS, a user modeling shell system. *Computational Intelligence*, **6**(4), 193–208.

Kobsa, A. 2001. Generic User Modeling Systems. *User Modeling and User-Adapted Interaction*, **11**, 49–63.

Komatani, K., Ueno, S., Kawahara, T., & Okuno, H. G. 2003. User Modeling in Spoken Dialogue Systems for Flexible Guidance Generation. *In: Proceedings of Eurospeech'03*.

Kowtko, J., & Isard, S. 1993. *Conversational Games Within Dialogue*. Tech. rept. 31.

Kupiec, J. M. 1999. *MURAX: Finding and Organizing Answers from Text Search*. Dordrecht, Netherlands: Kluwer Academic. Pages 311–332.

Kwok, C. T., Etzioni, O., & Weld, D. S. 2001. Scaling question answering to the Web. *In: Proceedings of WWW'01*.

Langville, A. N., & Meyer, C. D. 2006. *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton University Press.

Lappin, S., & Leass, H. J. 1994. An Algorithm for Pronominal Anaphora Resolution. *Computational Linguistics*, **20**(4), 535–561.

Larsson, S. 2002. *Issue-based dialogue management*. Ph.D. thesis, Goteborg, Sweden.

Larsson, S., & Traum, D. R. 2000. Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Journal of Natural Language Engineering*, **6**(3-4), 323–340.

Larsson, S., Ljunglöf, P., Cooper, R., Engdahl, E., & Ericsson, S. 2000. GoDiS—An Accommodating Dialogue System. *Pages 7–10 of:* Sidner, C. (ed), *ANLP/NAACL Workshop on Conversational Systems*. Somerset, New Jersey: ACL.

Larsson, Staffan. 1998. *Using a Type Hierarchy to Characterize Reliability of Coding Schemas for Dialogue Moves*.

Lashkari, Y., Metral, M., & Maes, P. 1994. Collaborative Interface Agents. *In: Proceedings of AAAI'94*.

Lemon, O., Bracy, A., Gruenstein, A., & Peters, S. 2001. The WITAS Mult-Modal Dialogue System I. *In: Proceedings of EuroSpeech 2001*.

Levy, D., Catizone, R., Battacharia, B., A.Krotov, & Wilks, Y. 1997. CONVERSE: A Conversational Companion. *In: Proceedings of the 1st International Workshop on Human-Computer Conversation.*

Lewin, I., Rupp, C.J., Hieronymus, J., Milward, D., Larsson, S., & Berman, A. 2000. *Siridus system architecture and interface report.* Tech. rept. D6.1.

Li, X., & Roth, D. 2002. Learning Question Classifiers. *In: Proceedings of ACL'02.*

Li, X., & Roth, D. 2005. Learning Question Classifiers: The Role of Semantic Information. *Journal of Natural Language Engineering.*

Lin, J., & Demner-Fushman, D. 2006. Will pyramids built of nuggets topple over? *Pages 383–390 of: Proceedings of HLT-NAACL'06.* Morristown, NJ, USA: ACL press.

Lin, J., Quan, D., Sinha, V., & Bakshi, K. 2003. What makes a good answer? The role of context in Question Answering. *In: Proceedings of INTERACT'03.*

Linton, Fr., Goodman, B., Gaimar, R., Zarrella, J., & Ross, H. 2003. Student Modeling for an Intelligent Agent in a Collaborative Learning Environment. *In: Proceedings of UM'03.*

Litman, D., & Allen, J. 1990. Discourse processing and commonsense plans. *In:* Cohen, P., Morgan, J., & Pollack, M. (eds), *Intentions in communication.* Cambridge, Mass.: MIT Press.

Magnini, B., & Strapparava, C. 2001. Improving User Modelling with Content-Based Techniques. *In: User Modelling: Proceedings of the 8th International Conference.* LNCS, vol. 2109. Springer.

Mani, I., Verhagen, M., Wellner, B., Lee, C. M., & Pustejovsky, J. 2006. Machine learning of temporal relations. *Pages 753–760 of: ACL '06: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL.* Morristown, NJ, USA: Association for Computational Linguistics.

Maybury, M. T. 2002. *Towards a Question Answering Roadmap.* Tech. rept. MITRE Corporation.

Mays, E., Joshi, A. K., & Webber, B. L. 1982. Taking the Initiative in Natural Language Data Base Interactions: Monitoring as Response. *In: Proceeding of IJCAI'82.*

McCoy, K. F. 1983. Correcting misconceptions: What to say when the user is mistaken. *Pages 197–201 of: CHI '83: Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. New York, NY, USA: ACM.

McGlashan, S., Fraser, N., Gilbert, N., Bilange, E., Heisterkamp, P., & Youd, N. 1992. Dialogue management for telephone information systems. *In: Proceedings of the International Conference on Applied Language Processing (ICSLP'92)*.

McKeown, K. R. 1985. *Text generation: using discourse strategies and focus constraints to generate natural language text*. New York, NY, USA: Cambridge University Press.

McLaughlin, G. 1969. SMOG grading: A new readability formula. *Journal of Reading*, 12(8), 693–46.

Miliaraki, S., & Androutsopoulos, I. 2004. Learning to identify single-snippet answers to definitional questions. *In: Proceedings of Coling '04*.

Miller, B., Albert, I., Lam, S.K., Konstan, J., & Riedl, J. 2003. MovieLens Unplugged: Experiences with a Recommender System on Four Mobile Devices. *In: Proceedings of the 17th Annual Human-Computer Interaction Conference (HCI 2003)*.

Miller, G. A. 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11), 39–41.

Mitkov, R. 2002. *Anaphora Resolution*. Pearson.

Moldovan, D., Pasca, M., Harabagiu, S., & Surdeanu, M. 2003. Performance issues and error analysis in an open-domain question answering system. *ACM Trans. Inf. Syst.*, 21(2), 133–154.

Moore, J. D., & Paris, C. 1989. Planning Text for Advisory Dialogues. *Pages 203–211 of: Meeting of the Association for Computational Linguistics (ACL'89)*.

Moschitti, A. 2006. Efficient Convolution Kernels for Dependency and Constituent Syntactic Trees. *In: Proceedings of ECML'06*.

Moschitti, A., Coppola, B., Giuglea, A., & Basili, R. 2005. Hierarchical Semantic Role Labeling. *In: Proceedings of the CoNLL 2005 shared task*.

Moschitti, A., Quarteroni, S., Basili, R., & Manandhar, S. 2007. Exploiting Syntactic and Shallow Semantic Kernels for Question/Answer Classification. *In: Proceedings of ACL'07*.

Moschitti, A., Pighin, D., & Basili, R. 2008, to appear. Tree Kernels for Semantic Role Labeling. *Computational Linguistics, Special Issue on Semantic Role Labeling.*

Munteanu, C., & Boldea, M. 2000. MDWOZ: A Wizard of Oz Environment for Dialog Systems Development. *In: Proceedings of LREC'00.*

Noriko, S. L. 2000. The Use of WordNet Sense Tagging in FAQFinder. *In: Proceedings of AAAI'00.*

Nyberg, E., Mitamura, T., Carbonnell, J., Callan, J., Collins-Thompson, K., Czuba, K., Duggan, M., Hiyakumoto, L., Hu, N., Huang, Y., Ko, J., Lita, L.V., Murtagh, S., Pedro, V., & Svoboda, D. 2002. The JAVELIN Question-Answering System at TREC 2002. *In: Proceedings of TREC 2002.*

Oppenheimer, F., Zhang, D., & Nebel, W. 2001. Modelling Communication Interfaces with COMIX. *Pages 337–348 of: Proceedings of the 6th Ade-Europe International Conference Leuven on Reliable Software Technologies.* Springer-Verlag.

Page, L., Brin, S., Motwani, R., & Winograd, T. 1998. *The PageRank Citation Ranking: Bringing Order to the Web.* Tech. rept. Stanford Digital Library Technologies Project.

Perrault, C. R., Allen, J. F., & Cohen, P. R. 1978. Speech acts as a basis for understanding dialogue coherence. *Pages 125–132 of: Proceedings of the theoretical issues in natural language processing-2.*

Person, N. K., Craig, S., Price, P., Hu, X., Gholson, B., & Graesser, A. C. 2000. Incorporating Human-like Conversational Behaviors into AutoTutor. *In: Agents 2000 Proceedings of the Workshop on Achieving Human-like Behavior in the Interactive Animated Agents.*

Pitkow, J., Schuetze, H., Cass, T., Cooley, R., Turnbull, D., Edmonds, A., Adar, E., & Breuel, T. 2002. Personalized search. *Communications of the ACM*, **45**(9), 50–55.

Poesio, M., Reyle, U., & Stevenson, R. 2001. *Justified sloppiness in anaphoric reference – Computing meaning.* Kluwer. Chap. 3.

Porter, M.F. 1980. An algorithm for suffix stripping. *Program*, **14**(3), 130–137.

Prager, J., Radev, D., Brown, E., Coden, A., & Samn, V. 1999. The use of predictive annotation for question answering in TREC. *In: Proceedings of TREC-8.*

Purver, M., Ginzburg, J., & Healey, P. 2002. On the means for clarification in dialogue. *Current and New Directions in Discourse and Dialogue.*

Quarteroni, S., & Manandhar, S. 2005. Adaptivity in Question Answering Using Dialogue Interfaces. *In:* Bandini, S. (ed), *Proceedings of the AI*IA '05 Workshop on Cultural Heritage.*

Quarteroni, S., & Manandhar, S. 2006. User Modelling for Adaptive Question Answering and Information Retrieval. *In:* Sutcliffe, G., & Goebel, R. (eds), *Proceedings of FLAIRS'06.* AAAI Press.

Quarteroni, S., Moschitti, A., Manandhar, S., & Basili, R. 2007. Advanced Structural Representation for Question Classification and Answer Reranking. *In: Proceedings of ECIR'07.*

Rajman, M., Bui, T. H., Rajman, A., Seydoux, F., Trutnev, A., & Quarteroni, S. 2004. Assessing the usability of a dialogue management system designed in the framework of a rapid dialogue prototyping methodology. *Acta Acustica united with Acustica,* 90(6).

Rasmussen, E. 1992. Clustering algorithms. *Information retrieval: data structures and algorithms,* 419–442.

Resnick, P., Iacovou, N., Suchak, M., Bergstorm, P., & Riedl, J. 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. *Pages 175–186 of: Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work.* Chapel Hill, North Carolina: ACM.

Rich, E. 1979a. User modeling via stereotypes. *Readings in intelligent user interfaces,* 329–342.

Rich, E. A. 1979b. *Building and exploiting user models.* Ph.D. thesis.

Romero, C., Ventura, S., de Bra, P., & de Castro, C. 2003. Discovering Prediction Rules in AHA! Courses. *In: User Modelling: Proceedings of the 9th International Conference.* LNAI/LNCS, vol. 2702. Springer.

Sacks, H., Schegloff, E. A., & Jefferson, G. 1974. A Simplest Systematics for the Organization of Turn-Taking for Conversation. *Language,* 50(4), 696–735.

Schank, R. C., & Abelson, R. P. 1977. *Scripts, Plans, Goals and Understanding: an Inquiry into Human Knowledge Structures.* Hillsdale, NJ: L. Erlbaum.

Schegloff, E., & Sacks, H. 1973. Opening up closing. *Semiotica*, August, 289–327.

Seneff, S., Hurley, E., Lau, R., Pao, C., Schmid, P., & Zue, V. 1998. Galaxy-II: A reference architecture for conversational system development. *Pages 931–934 of: Proc. ICSLP-98*, vol. 3.

Shawe-Taylor, J., & Cristianini, N. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Shen, X., Tan, B., & Zhai, C. 2005. Implicit user modeling for personalized search. *Pages 824–831 of: Proceedings of CIKM '05*. New York, NY, USA: ACM Press.

Simmons, R. F. 1965. Answering English questions by computer: a survey. *Communications of the ACM*, 8(1), 53–70.

Simmons, R. F. 1973. *Semantic networks Their computation and Use for Understanding English Semantics*. Freeman and Company. Chap. 1, pages 1–113.

Sinclair, J. M., & Coulthard, R. M. 1975. *Towards an Analysis of Discourse: The English Used by Teachers and Pupils*. Oxford University Press.

Small, S., Liu, T., Shimizu, N., & Strzalkowski, T. 2003. HITIQA: an interactive question answering system- a preliminary report. *Pages 46–53 of: Proceedings of the ACL 2003 workshop on Multilingual summarization and QA*. Morristown, NJ, USA: ACL.

Srihari, R., & Li, W. 1999. Information extraction supported question answering. *In: Proceedings of TREC-8*.

Srihari, R., & Li, W. 2000. A Question Answering System Supported by Information Extraction. *Pages 166–172 of: Proceedings of the 1 st Meeting of the North American Chapter of the ACL (ANLP-NAACL-00)*.

Stede, M., & Schlangen, D. 2004. Information-Seeking Chat: Dialogue Management By Topic Structure. *In: Proceedings of CATALOG'04*.

Steinberger, J., Kabadjov, M. A., Poesio, M., & Sanchez-Graillet, O. 2005. Improving LSA-based summarization with anaphora resolution. *Pages 1–8 of: HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*. Morristown, NJ, USA: Association for Computational Linguistics.

Stock, O., & AlFresco Project Team. 1993. ALFRESCO: Enjoying The Combination of NLP and Hypermedia for Information Exploration. *Intelligent Multimodal Interfaces.*

Sturm, J., den Os, E. A., & Boves, L. 1999. Issues in Spoken Dialogue Systems: Experiences from the Dutch Arise Train Timetable Information System. *In: Proceedings of the ESCA Workshop on Interactive Dialogue in Multi-Modal Systems.*

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

Sutton, S. 1998. Universal speech tools: the CSLU toolkit. *In: Proceedings of the International Conference on Spoken Language Processing (ICSLP'98).*

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

Thai, V., O'Riain, S., Davis, B., & O'Sullivan, D. 2006. *Personalized Question Answering: A Use Case for Business Analysis.*

Traum, D. 1996. Dialogue Management in Conversational agency: The TRAINS-93 dialogue manager. *Pages 1–11 of: Proceedings of the Twente Workshop on Langauge Technology: Natural Language Systems (TWLT 11).*

Verberne, S., Boves, L., Oostdijk, N., & Coppen, P. 2007. Evaluating Discourse-based Answer Extraction for Why-Question Answering. *Pages 735–737 of: SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in informaion retrieval.* New York, NY, USA: ACM Press.

Vicedo, J.L., Ferrandez, A., & Llopis, F. 2001. University of Alicante at TREC-10. *In: Proceedings of TREC'01.*

Virvou, M., & Moundridou, M. 2001. Student and Instructor Models: Two Kinds of User Model and Their Interaction in an ITS Authoring Tool. *In: User Modelling: Proceedings of the 8th International Conference.* LNCS, vol. 2109. Springer.

Voorhees, E. M. 1999. The TREC-8 Question Answering Track report. *In: Proceedings of TREC-8.*

Voorhees, E. M. 2000. Overview of the TREC-9 Question Answering Track. *In: Proceedings of TREC-9.*

Voorhees, E. M. 2001. Overview of the TREC 2001 Question Answering Track. *In: Proceedings of TREC'01.*

Voorhees, E. M. 2002. Overview of the TREC 2002 Question Answering Track. *In: Proceeding of TREC'02.*

Voorhees, E. M. 2003. Overview of the TREC 2003 Question Answering Track. *In: Proceedings of TREC'03.*

Voorhees, E. M. 2004. Overview of the TREC 2004 Question Answering Track. *In: Proceedings of TREC'04.*

Voorhees, E. M., & Dang, H. T. 2005. Overview of the TREC 2005 Question Answering Track. *In: Proceedings of TREC'05.*

Wahlster, W., Marburger, H., Jameson, A., & Busemann, S. 1983. Over-Answering Yes-No Questions: Extended Responses in a NL Interface to a Vision System. *In: Proceedings of IJCAI.*

Walker, M. A., Kamm, C., & Litman, D. 2000. Towards Developing General Models of Usability with PARADISE. *Nat. Lang. Eng. Special Issue on Best Practice in Spoken Dialogue Systems.*

Waltz, D. L. 1978. An English Language Question Answering System for a Large Relational Database. **21**(7).

Webb, N., & Strzalkowski, T. (eds). 2006. *Proceedings of the HLT-NAACL Workshop on Interactive Question Answering.*

Weizenbaum, J. 1966. ELIZA - a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, **9**(1), 36–45.

Wilensky, R., Mayfield, J., Albert, A., Chin, D.N., & Cox, C. 1987. *UC (UNIX Consultant). A Progress Report.* Tech. rept. 122. UC Berkeley Computer Science Division.

Williams, S., & Reiter, E. 2005. Generating Readable Texts for Readers with Low Basic Skills. *Pages 140–147 of: Proceedings of ENLG-2005.*

Winograd, T. 1972. *Understanding Natural Language.* Academic Press.

Witten, I. H., Paynter, G. W., Frank, E., Gutwin, C., & Nevill-Manning, C. G. 1999. KEA: Practical Automatic Keyphrase Extraction. *Pages 254–255 of: ACM DL.*

Woods, W. A., Kaplan, R. M., & Nash-Webber, B. 1972. *The lunar sciences natural language information system: Final report.* Tech. rept. 2378. MIT.

Wu, L., Faloutsos, C., Sycara, K. P., & Payne, T. R. 2000. FALCON: Feedback Adaptive Loop for Content-Based Retrieval. *Pages 297–306 of: VLDB '00: Proceedings of the 26th International Conference on Very Large Data Bases.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Wu, M., & Wilkinson, R. 1998. Using Document Relationships for Better Answers. *Pages 43–52 of: PODDP '98: Proceedings of the 4th International Workshop on Principles of Digital Document Processing.* Springer-Verlag.

Xu, J., & Croft, W. B. 1996. Query expansion using local and global document analysis. *Pages 4–11 of: Proceedings of SIGIR '96.* New York, NY, USA: ACM Press.

Xu, J., Cao, Y., Li, H., & Zhao, M. 2005. Ranking definitions with supervised learning methods. *In: Special interest tracks and posters of WWW'05.* New York, NY, USA: ACM.

Xu, W., Xu, B., Huang, T., & Xia, H. 2002. Bridging the gap between dialogue management and dialogue models. *Pages 201–210 of: Proceedings of the 3rd SIGdial workshop on Discourse and dialogue.* Morristown, NJ, USA: Association for Computational Linguistics.

Yang, F., Feng, Z., & Di Fabbrizio, G. 2006. A data driven approach to relevancy recognition for contextual question answering. *In: Proceedings of IQA'06.*

Yin, L. 2006. A Two-Stage Approach to Retrieving Answers for How-To Questions. *In: Proceedings of EACL '06 (Student Session).*

Zelenko, D., Aone, C., & Richardella, A. 2003. Kernel methods for relation extraction. *Journal of Machine Learning Resources.*

Zhang, D., & Lee, W. S. 2003. Question classification using Support Vector Machines. *Pages 26–32 of: SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval.* ACM Press.