Natural Language Engineering **15** (1): 73–95. © 2008 Cambridge University Press
doi:10.1017/S1351324908004919 Printed in the United Kingdom

Designing an interactive open-domain question answering system

S. QUARTERONI and S. MANANDHAR
The University of York, York, YO10 5DD, UK
e-mail: silvia.quarteroni@gmail.com, suresh@cs.york.ac.uk

(*Received 20 September 2007; first published online 28 October 2008*)

Abstract

Interactive question answering (QA), where a dialogue interface enables follow-up and clarification questions, is a recent although long-advocated field of research. We report on the design and implementation of YourQA, our open-domain, interactive QA system. YourQA relies on a Web search engine to obtain answers to both fact-based and complex questions, such as descriptions and definitions. We describe the dialogue moves and management model making YourQA interactive, and discuss the architecture, implementation and evaluation of its chat-based dialogue interface. Our Wizard-of-Oz study and final evaluation results show how the designed architecture can effectively achieve open-domain, interactive QA.

1 Introduction

Question answering (QA) systems can be seen as information retrieval systems that aim to respond to queries in natural language by returning concise answers rather than informative documents. State-of-the-art QA systems often compete in the annual TREC-QA evaluation campaigns, where participating systems must find concise answers to a benchmark of test questions within a document collection compiled by NIST (http://trec.nist.gov).

A commonly observed behaviour is that users of information retrieval systems often issue queries not as stand-alone questions 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 for entering successive related queries have been advocated to avoid users having to enter contextually independent queries (Hobbs, 2002). Efforts have been carried out in recent TREC-QA in order to approach the issue of context management by the introduction of 'targets' in the question sets from TREC 2004. Here, questions are grouped according to a common topic, upon which different queries (that require factoid, list or 'other' answer types) are formulated.

Since TREC-QA 2004, queries can contain references (such as pronominal anaphora) to their targets without such targets being explicitly mentioned in the query texts. However, the current TREC requirements only address one aspect of the complex issue of context management: the problem of detecting that one query

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

is related to a topic introduced by a previous one is artificially solved by the presence of an explicit target, which would not be specified in a real interaction context.

It has been argued that providing a QA system with a dialogue interface would encourage and accommodate the submission of multiple related questions and handle the user's requests for clarification: the 2006 interactive QA workshop aimed to set a roadmap for information-seeking dialogue applications of QA (Webb and Strzalkowski, 2006). Indeed, interactive QA systems are often reported at an early stage, such as Wizard-of-Oz studies, or applied to closed domains (Jönsson and Merkel, 2003; Bertomeu *et al*., 2006; Kato *et al*., 2006).

In this paper, we report on the design, implementation and evaluation of the dialogue interface for our open-domain, personalized QA system, YourQA (Quarteroni and Manandhar, 2007). The core QA component in YourQA is organized according to the three-tier partition underlying most state-of-the-art QA systems (Kwok, Etzioni and Weld, 2001): question processing, document retrieval and answer extraction. An additional component in YourQA is the User Modelling component, introduced to overcome the traditional inability of standard QA systems to accommodate the users' individual needs (Voorhees, 2003).

This article is structured as follows: Sections 2 and 3 focus on the two main components of the system, i.e., a User Modelling component to provide personalized answers and the core QA module which is able to provide both factoid and complex answers. Sections 4–7 discuss a dialogue model and dialogue manager suitable for interactive QA and Section 8 describes an exploratory study conducted to confirm our design assumptions. The implementation and evaluation of the dialogue model are reported in Sections 9 and 10. Section 11 briefly concludes on our experience with open-domain QA dialogue.

2 User Modelling component

A distinguishing feature of our model of QA is the presence of a User Modelling component. User Modelling consists of creating a model of some of the target users' characteristics (e.g. preferences or level of expertise in a subject), and is commonly deployed in information retrieval applications to adapt the presentation of results to the user characteristics (Teevan, Dumais and Horvitz, 2005).

It seemed natural to adapt User Modelling within QA, with the purpose of filtering the documents from which to search for answers and for re-ranking candidate answers based on the degree of match with the user's profile. Since the current application scenario of YourQA is a system to help students find information on the Web, we designed the following User Model (UM) parameters:

- Age range: $a \in \{7\text{–}11,\ 11\text{–}16,\ adult\}$; this matches the partition between primary school, secondary school and higher education age in Britain;
- Reading level: $r \in \{basic,\ medium,\ good\}$; its values ideally (but not necessarily) correspond to the three age ranges and may be further refined;
- Interests: i; a set of topic key-phrases extracted from webpages, bookmarks and text documents of interest to the user.

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

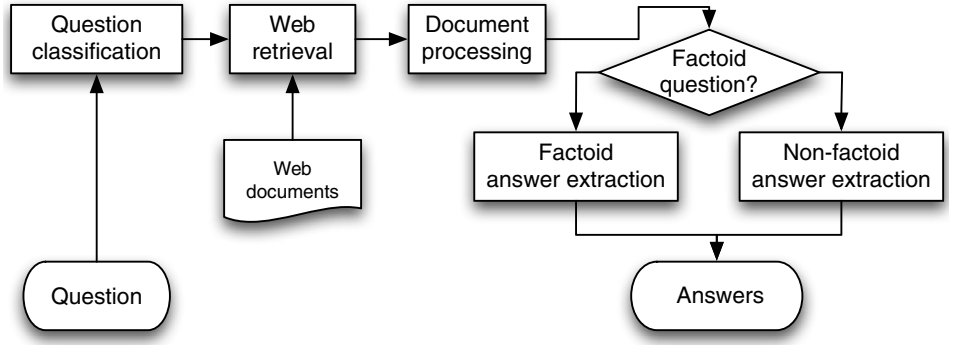


Fig. 1. Core QA architecture: question processing, retrieval, answer extraction.

A detailed account on how the UM parameters are applied during answer presentation has been reported in Quarteroni and Manandhar (2006, 2007). As the focus of this paper is on the dialogue management component of YourQA, the contribution of the UM to the core QA component is only briefly mentioned in this paper. Within this paper, we assume an adult user able to read any document and an empty set of interests, hence no UM-based answer filtering or re-ranking is performed in the experiments reported in this paper.

3 Core question answering component

The core QA component, illustrated in Figure 1, carries on three-QA phases: question processing, document retrieval and answer extraction.

3.1 Question processing and document retrieval

Question processing is centred on question classification (QC), the task that maps a question into one of the k expected answer classes in order to constrain the search space of possible answers and contribute towards selecting specific answer extraction strategies for each answer class.

Answer classes generally belong to two types: factoid ones – seeking short fact-based answers (e.g., names, dates), and non-factoid, seeking descriptions or definitions. An *ad hoc* question taxonomy has been constructed for YourQA with a particular attention to questions that require non-factoid answers, such as lists, descriptions and explanations. To compile it, we studied the questions in the TREC-8 to TREC-12 testsets.[1] Based on these, we designed a coarse-grained question taxonomy, which consists of the eleven question types described in Table 1. While the six classes in Column 1 can be considered of the factoid type, the five in Column 2 are non-factoid; depending on such type, the answer extraction process is different as described in Section 3.2.

[1] Publicly available at `http://trec.nist.gov`

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

Table 1. *YourQA's eleven class expected answer taxonomy*

Question class	Expected answer	Question class	Expected answer
PERS	Human	LIST	List of items
LOC	Geographical expression	DEF	Definition, description
ORG	Collective, group	HOW	Procedure, manner
QTY	Numerical expression	WHY	Cause
TIME	Temporal expression	WHY-F	Salient facts
OBJ	Generic entity		(e.g., 'famous for . . . ')

Most QC systems apply supervised machine learning, e.g. Support Vector Machines (SVMs) (Zhang and Lee, 2003) or the Sparse Network of Winnows (SNoW) model (Li and Roth, 2005), where questions are represented using lexical, syntactic and semantic features.

Moschitti, Quarteroni, Basili and Manandhar (2007) extensively studied a QC model based on SVMs: the learning algorithm combined tree kernel functions to compute the number of common subtrees between two syntactic parse trees. As benchmark data, the question training and test set available at `l2r.cs.uiuc.edu/~cogcomp/Data/QA/QC/` were used, where the test set are the TREC 2001 test questions.

Based on such experiments, which reached a state-of-the-art accuracy (i.e., 86.1% in ten-fold cross-validation) using the the question's bag-of-words and parse tree, we applied the same features to learn multiclassifiers for the eleven-class YourQA taxonomy. The overall SVM accuracy obtained using the dataset of 3,204 TREC 8–TREC 12 test questions and five-fold cross-validation was 82.9%.

We also tested the SNoW algorithm for YourQA, following Li and Roth (2005). We found the most effective question features to be: (1) bag-of-words, bigrams and trigrams; (2) bag-of-named Entities[2]; (3) part-of-speech unigrams, bigrams and trigrams. In the YourQA task, we achieved an accuracy of 79.3%.[3]

The second phase carried out by the core QA module is document retrieval, where relevant documents to the query are obtained via an information retrieval engine, then downloaded and analysed. YourQA uses Google (`http://www.google.com`) to retrieve the top twenty Web documents for the query.

3.2 Answer extraction

Answer extraction takes as input the expected answer type, estimated during question classification and the set of documents retrieved for the question during document retrieval. In this process, the similarity between the question and the document passages is computed in order to return the best passages in a ranked list. Each remaining retrieved document D is then split into sentences, which are compared

[2] Extracted using Lingpipe, `http://www.alias-i.com/lingpipe/`

[3] The lower result when compared to SVMs is confirmed by the experiment with the same features and the previous task, where the accuracy reached 84.1%.

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

one by one to the question; the most similar sentence to the question is selected as the most likely sentence from document D to answer the question.

For factoid answers – i.e., PERS, ORG, LOC, QTY, TIME, MONEY – the required factoid can be pinpointed down to the phrase/word level in each candidate answer sentence. For non-factoids, other criteria are adopted to compute the similarity between the original question and each candidate answer, as explained later.

In both cases, the bag-of-word similarity *bw(q,a)* is computed between the question q and a candidate answer a. This is the number of matches between the keywords in q and a divided by $|q|$; the number of keywords in q:
$bw(q,a) = \frac{\sum_{i<|q|,j<|a|}^{|q|} match(q_i,a_j)}{|q|}$, where $match(q_i,a_j) = 1$ if $q_i = a_j$, 0 otherwise.

3.2.1 Factoid answers

Our primary focus is on non-factoid QA and the criteria we apply for factoids are simple. We distinguish between two cases: (a) the expected type is a person (PERS), organization (ORG) or location (LOC), which corresponds to the types of Named Entities (NEs) recognized by the NE recognizer (Lingpipe in our case); (b) the expected answer type is QTY, TIME, MONEY.

PERS, ORG, LOC – In this case, NE recognition is performed on each candidate answer a. If a phrase p labelled with the required NE class is found w_k^p, i.e., the minimal distance d (in terms of number of words) between p and each of the question keywords k found in a, is computed: $w_k^p = \min_{k \in a} d(p,k)$. In turn, $ne(a) = \min_{p \in a} w_k^p$, i.e., the minimal w_k^p among all the NE phrases of the required class found in a, is used as a secondary ranking criterion for a (after the bag-of-words criterion).

QTY, TIME, MONEY – In this case, class-specific rules are applied to find factoids of the required class in each candidate answer a. These are manually written based on regular expressions and the candidate answer's POS tags (e.g., the ordinal number tag). The presence of a substring of a matching such rules is the second similarity criterion between the question q and a after the bag-of-words criterion.

3.2.2 Non-factoid answers

We assign to the non-factoid group the WHY, HOW, WHY-F, DEF and LIST types, as well as the OBJ type which is too generic to be seized using a factoid answer approach. In these cases, we aim at more sophisticated sentence similarity metrics than the one applied for factoids. We compute a number of normalized similarity measures each measuring the degree of match between sentences and the question. The final similarity is a weighted sum of all such measures. In addition to the bag-of-word similarity, we compute the metrics below.

Bigram similarity – N-gram similarity is a function of the number of common keyword n-grams between q and a: $ng(q,a) = \frac{commonN(q,a)}{|ngrams(q)|}$, where *commonN* is the number of shared n-grams between q and a and *ngrams(q)* is the set of question n-grams. We adopt bigrams ($n = 2$) as Web data is very noisy and allows for

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

different formulations using the same words, making it unlikely that matches of longer keyword sequences are found.

Chunk similarity – Sentence chunks can be defined as groups of consecutive, semantically connected words in one sentence, which can be obtained using a shallow parser (in our case, the OpenNLP chunker). Compared to bigrams, chunks encode a more semantic type of information. Chunk similarity, *ck(q,a)*, is defined as the number of common chunks between q and a divided by the total number of chunks in q: $ck(q,a) = \frac{commonC(q,a)}{|chunks(q)|}$, where *commonC* is the number of shared chunks between q and a and *chunks(q)* is the set of question chunks.

Head NP-VP-PP similarity – The idea behind this metric is to find matching groups consisting of a noun phrase (NP), verb phrase (VP) and prepositional phrase (PP) chunk in q and a. Head NP-VP-PP similarity is defined as
$hd(q,a) = \mu \times HNPmatch(q,a) + \nu \times VPmatch(q,a) + \xi \times PPmatch(q,a)$.
For generalization, VPs are lemmatized and the semantically most important word in the NP (called "head NP") is used instead of the NP. In case q contains several VPs, we choose the VP for which *hd(q,a)* is maximal. Based on empirical observation of YourQA's results, we are currently using $\mu = \nu = 0.4$, $\xi = 0.2$.

WordNet similarity – This semantic metric is based on the WordNet lexical database (`http://wordnet.princeton.edu`) and the Jiang–Conrath word-level distance (Jiang and Conrath, 1997). WordNet similarity is $wn(q,a) = 1 - \frac{\sum_{i<|q|,j<|a|} jc(q_i,a_j)}{|q|}$, $jc(q_i,a_j)$ being the J.-C. distance between q_i and a_j.

Combined similarity –The similarity formula combining the above metrics is

$$sim(q,a) = \alpha \times bw(q,a) + \beta \times ng(q,a) + \gamma \times ck(q,a) + \delta \times hd(q,a) + \epsilon \times wn(q,a).$$

For efficiency reasons, we do not compute *wn(q,a)* at the moment. We have estimated $\alpha = 0.6, \beta = 0.2, \gamma = \delta = 0.1$ as suitable coefficients. The bag-of-word criterion has a higher impact than metrics which rely on word structures (i.e., bigrams or chunks) because of the noisy Web data we are processing.

Moschitti *et al.* (2007) took the above criteria for non-factoid QA as a baseline and applied various combinations of features to learn SVM answer re-rankers. The experiments on 1,309 YourQA answers to the TREC 2001 non-factoid questions, showed that the baseline MRR of 56.21 ± 3.18 was greatly improved by adding a combination of lexical, deep syntactic and shallow semantic features, reaching 81.12 ± 2.12.

3.3 Answer presentation

From the preceding steps, YourQA obtains a list of answer sentences ranked by decreasing distance to the query. Windows of up to five sentences centred around these sentences are then produced to be returned as answer passages. To present answers, we fix a threshold *th* for the maximal number of passages to be returned (currently $th = 5$); these are ordered following the ranking exposed earlier. In case of a tie between two candidate answers, the Google ranks of their respective documents are compared and the answer with the highest Google rank index obtains a higher position in the list.

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

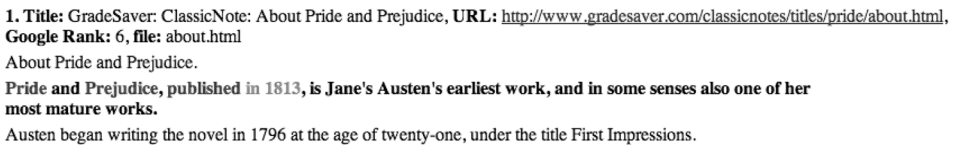


Fig. 2. YourQA: sample result (from `http://www.cs.york.ac.uk/aig/aqua/`).

The answer passages are listed in an HTML page where each list item consists of a document title and result passage obtained as described earlier. In the passages, the sentence which best answers the query according to the similarity metric described earlier is highlighted. In case the expected answer is a factoid, the recognized factoids are highlighted in different colours based on their type. A link to the URL of the original document is also available if the user wants to read more (see Figure 2).

Section 4 discusses the issues and design of a dialogue interface for YourQA to achieve interactive QA.

4 Modelling interactive question answering

Interactive QA dialogue can be considered as a form of information-seeking dialogue where two roles are modelled: inquirer (the user), looking for information on a given topic, and expert (the system), interpreting the inquirer's needs and providing the required information.

We agree with Dahlbaeck, Jonsson and Ahrenberg (1993) that attempting to perfectly emulate human dialogue using a machine is an unrealistic and perhaps unimportant goal. On the other hand, we believe that an understanding of human dialogues can greatly facilitate building human–machine information-seeking dialogue systems. Hence, the design of task-oriented dialogue systems cannot happen without an accurate analysis of the conversational phenomena observed in human–human dialogue.

4.1 Salient features of human information-seeking dialogue

For the purpose of describing information-seeking dialogue, we focussed on the following aspects:

- *Overall structure*: As observed by Sinclair and 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 computational approaches to dialogue to the present day, utterances are therefore considered as dialogue acts as they aim at achieving an effect (obtaining information, planning a trip, 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 and Van

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

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*: Dialogue participants often contribute more information than required by their interlocutors (Churcher, Atwell and Souter, 1997). This usually enables dialogue to be more pleasant and time-efficient as participants do not need to explicitly ask for all the desired information.
- *Contextual interpretation*: Human interaction relies on the conversation participants sharing common notion of context and topic (Grosz and Sidner, 1986). Such common context is used by participants to issue and correctly interpret rhetorical phenomena such as ellipsis, anaphora and more complex phenomena such as reprise and sluice (see Section 4.2).
- *Grounding*: It has been observed that to prevent or recover from possible misunderstandings, speakers engage in a collaborative, coordinated series of exchanges, instantiating new mutual beliefs and making contributions to the common ground of a conversation. This process is known as grounding (Cahn and Brennan, 1999).

Section 4.2 underlines the fundamental issues implied by accounting for such phenomena when modelling information-seeking human–computer dialogue.

4.2 Issues in modelling information-seeking dialogue

Based on the observed features of human information-seeking dialogue, we summarize the main issues in modelling task-oriented human–computer dialogue, with an eye on the relevance of such issues to interactive QA.

Ellipsis. Ellipsis is an omission of part of the sentence, resulting in a sentence with no verbal phrase. Consider the exchange: *User*: 'When was Shakespeare born?', *System*: 'In 1564.', *User*: 'Where?'. The interpretation and resolution of ellipsis requires an efficient modelling of the conversational context to complete the information missing from the text.

Anaphoric references. An anaphor is a linguistic form whose full meaning can only be recovered by reference to the context; 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 anaphor and 'Shakespeare' is the antecedent. A common form of anaphora is third person pronoun/adjective anaphora, where pronouns such as 'he/she/it/they' or possessive adjectives such as 'his/her/its/their' are used in place of the entities they refer to: the latter can be single or compound nouns (such as William Shakespeare), or even phrases (*The Taming of the Shrew*). Resolving anaphora, i.e., finding its most likely referent, is a critical problem in QA

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

as it directly affects the creation of a meaningful query. However, in information-seeking dialogue, resolution is simpler than in tasks such as document summarization (Steinberger *et al*., 2005) as the exchanged utterances are generally brief and contain fewer cases of anaphora.

Grounding and Clarification. While in formal theories of dialogue complete and flawless understanding between speakers is assumed, there exists a practical need for grounding (Cahn and Brennan, 1999). A typical QA scenario where requests for confirmation should be modelled anaphora resolution: *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 NEs of type 'person': hence, *he* 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.

Turn-taking. According to conversation analysis, a conversation is modelled as turns or pairs of utterances, often called *adjacency pairs* (Schegloff and Sacks, 1973). Our dialogue management system encodes adjacency pairs, where participants speak in turns so that dialogue can be modelled as a sequence of ⟨request, response⟩ pairs.

In natural dialogue, there is very little overlap between when one participant stops speaking and when the other starts speaking resulting in a fluid discourse. 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 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, where system replies are instantaneous and system/user overlap is not possible.

4.3 Summary of desiderata for interactive question answering

Based on the phenomena and issues observed in Section 4.2, we summarize the desiderata for interactive QA in the following list:

- *context maintenance*: maintaining the conversation context and topic to allow the correct interpretation of the user's utterances (in particular, requests for clarification);
- *utterance understanding:* in the context of the previous dialogue; this includes follow-up/clarification detection and the resolution of ellipsis and anaphoric expressions;
- *mixed initiative*: users should be able to take the initiative during the conversation, for example to issue clarification requests and to quit the conversation when they desire to do so;
- *follow-up proposal*: an interactive QA system should be able to encourage the user to provide feedback about satisfaction with the answers received and

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

also to keep the conversation with the user active until they have fulfilled their information needs;
- *natural interaction:* wide coverage of the user utterances to enable smooth conversation; generation of a wide range of utterances to encourage users to keep the conversation active.

5 A dialogue model for interactive question answering

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 (or speech acts) performed when a participant speaks. Based on speech act theory, several annotation schemes of speech acts – also called *dialogue moves* have been developed for task-oriented dialogues.

While the level of granularity of such schemes as well as the range of moves of most of the schemes were determined by the application of the dialogue system, as pointed out in Larsson (1998) there are a number of generic common dialogue moves, which include

- Core speech acts (e.g., TRAINS, Traum, 1996) such as 'inform'/'request';
- Conventional (e.g., DAMSL, Core and Allen, 1997) or discourse management (e.g., LINLIN, N. Dahlbaech and A. Jonsson, 1998, A coding manual for the Linkping dialogue model, unpublished manuscript) moves: opening, continuation, closing, apologizing;
- Feedback (e.g., VERBMOBIL, Alexandersson, Reithinger and Maier, 1997) or grounding (e.g., TRAINS) moves: to elicit and provide feedback;
- Turn-taking moves (e.g., TRAINS), relating to subutterance level (e.g., 'take-turn', 'release-turn').

Taking into account such general observations, we developed the set of user and system dialogue moves given in Table 2. In our annotation, the core speech acts are represented by the *ask* and *answer* moves. Among discourse management moves, we find *greet*, *quit* in both the user and system moves, and *followup* proposal from the system. The user feedback move is *usrReqClarif*, mirrored by the system's *sysReqClarif* move. A common feedback move to both user and system is *ack*, while the *ground* and *clarify* moves are only in the system's range. We do not annotate the scenario described earlier with turn-taking moves as these are at a subutterance level.

These moves are used in the following dialogue management algorithm:

(1) An initial greeting (*greet* move), or a direct question q from the user (*ask*(q) move);
(2) q is analysed to detect whether it is related to previous questions (*clarify*(q) move) or not;
(3) (a) If q is unrelated to the preceding questions, it is submitted to the QA component;

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

Table 2. *User and system dialogue moves*

User move	Description	System move	Description
greet	Conversation opening	*greet*	Conversation opening
ack	Acknowledge system	*ack*	Acknowledge user
ask(q)	Ask (q: question)	*answer(a)*	Answer (a: answer)
usrReqClarif	Clarification request	*sysReqClarif*	Clarification request
quit	Conversation closing	*quit*	Conversation closing
		followup	Proposal to continue
		clarify(q)	Clarify (q: question)
		ground(q)	Ground (q: question)

(b) If q is related to the preceding questions (i.e., follow-up question), and is elliptic (e.g., *'Why?'*), the system uses the previous questions to complete q with the missing keywords and submits a revised question q' to the QA component (notice that no dialogue move occurs here as the system does not produce any utterance);

(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;

(4) Once the QA component results are available, an answer a is provided (*answer(a)* move);

(5) 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*);

(6) 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. We now discuss the choice of a dialogue management model to implement such moves.

6 Previous work on dialogue management

Broadly speaking, dialogue management models fall into two categories: pattern-based approaches or plan-based approaches (Cohen, 1996; Xu *et al.*, 2002). The following sections provide a brief critical overview of these, underlining their issues and advantages when addressing interactive QA.

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

6.1 Pattern-based approaches: grammars and finite-state

When designing information-seeking dialogue managers, Finite-State (FS) approaches provide the simplest methods for implementing dialogue management. Here, the dialogue manager is represented as an FS 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 state machines are used, the system first recognizes the user's speech act from the utterance, makes the appropriate transition, and then 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 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.

Moreover, the model typically assumes that only one state results from a transition; however, in some cases utterances are multifunctional, e.g., both rejection and assertion, and a speaker may expect the response to address more than one interpretation.

6.2 Information state and plan-based approaches

Plan-based theories of communicative action and dialogue (Traum, 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.

Within plan-based approaches, one approach to dialogue management is the Information State (IS) approach (Larsson and Traum, 2000). Here, the conversation is centred on the notion of IS, which comprises the topics under discussion and common ground in the conversation and is continually queried and updated by rules triggered by participants' dialogue moves. The IS theory has been applied to a range of closed-domain dialogue systems, such as travel information and route planning (Bos *et al.*, 2003).

6.3 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.

Also, as pointed out in Allen (2000), there are a number of problems in using plan-based approaches in actual systems, including knowledge representation and engineering, computational complexity and noisy input. Moreover, 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

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

shortcoming of pattern-based dialogue models does not appear to greatly impact on the type of dialogue we are addressing.

The ideal dialogue management module for interactive QA seems to lie somewhere in between the FS and IS models. This is what we propose here.

7 Chatbot-based interactive question answering

As an alternative to the FS and IS models, we studied conversational agents based on AIML (Artificial Intelligence Markup Language). AIML was designed for the creation of conversational robots (chatbots) such as ALICE.[4] It is based on pattern matching, which consists in matching the last user utterance against a range of dialogue patterns known to the system and producing a coherent answer following a range of 'template' responses associated to such patterns. Pattern/template pairs form 'categories', an example of which is the following greeting category:

```
<pattern>WHO ARE YOU</pattern>
<template>I am ALICE, nice to meet you!</template>
```

Designed for chatting, 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.

Chatbot dialogue seems particularly well suited to handle the dialogue phenomena introduced in Section 4.1; in particular, the way in which such phenomena can be handled by a chatbot dialogue management model is discussed in detail as follows:

- *Mixed initiative*: As mentioned earlier, the system must be able to take control in order to confirm given information, clarify the situation or constrain user responses. In our system, the *ground* move is used to confirm that the system has correctly interpreted elliptic or anaphoric requests, while the *sysReqClarif* move is used to verify that the current user's utterance is an information request in ambiguous cases (see Section 9).
  The patterns used by the system are oriented towards QA conversation so that the user is encouraged to formulate information requests rather than engage in smalltalk, for instance, the pattern: `<pattern>HELLO *</pattern>` triggers: `<template>Hello, what is your question?</template>`.
  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 (e.g., *User*: 'What is a thermometer?', *System*: 'The answers are ... Are you happy with these answers?', *User*: 'How does it measure the temperature?'), triggering a new ask/answer adjacency pair with a new conversation focus. Moreover, the user can formulate a request for clarification at any time during the interaction.
- *Over-informativeness*: Providing more information than required is useful both from the system's viewpoint and from the user's viewpoint: this usually

[4] http://www.alicebot.org/

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

enables dialogue to be more pleasant as there is no need to ask for all desired pieces of information.

In the current approach, the user can respond to the system by providing more than a simple acknowledgement. For instance, the following exchange is possible: *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?'.

- *Contextual interpretation*: Contextual interpretation of user utterances are handled by a clarification resolution module designed to take care of ellipsis and anaphoric references, as described in Section 5.
- *Error recovery*: The management of misunderstandings is possible due 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, the pattern:
  `<pattern>I NEED *</pattern>`
  (e.g., 'I need information about Shakespeare') would trigger the template:
  `<template>Is that a question you want me to look up? </template>`.
  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, the user can enter a request for clarification of the system's latest utterance at any time should they find the latter unclear.

It must be pointed out that chatbots have rarely been used for task-oriented dialogue in the literature. An example is Ritel (Galibert, Illouz and Rousset, 2005), a spoken chat-based dialogue system integrated with an open-domain QA system. However, the project seems at an early stage and no thorough description is available about its dialogue management model.

8 A Wizard-of-Oz experiment for the dialogue component

To assess the feasibility of chatbot-based QA dialogue, we conducted an exploratory Wizard of Oz experiment Wizard-of-Oz (WOz) experiment, a procedure 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). A human operator (or 'Wizard') emulates the behaviour of the computer system by carrying on a conversation with the user; the latter believes to be interacting with a fully automated prototype.

Design. We designed six tasks, to be issued in pairs 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, e.g., *'Find out who painted Guernica and ask the system for more information about the artist', 'Find out when Jane Austen was born', 'Ask about the price of the iPod Shuffle and then about the PowerBook G4'*.

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.

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

Table 3. *Questionnaire results for the Wizard-of-Oz experiment (WOz), the initial experiment (Init.) and the final experiment (standard and interactive version).*

	Question	WOz	Init.	Stand.	Inter.
Q_1	Did you get all the information you wanted using the system?	4.3 ± 0.5	3.8 ± 0.8	4.1 ± 1	4.3 ± 0.7
Q_2	Do you think the system understood what you asked?	4	3.8 ± 0.4	3.4 ± 1.3	3.8 ± 1.1
Q_3	How easy was it to obtain the information you wanted?	4 ± 0.8	3.7 ± 0.8	3.9 ± 1.1	3.7 ± 1
Q_4	Was it easy to reformulate your questions when you were invited to?	3.8 ± 0.5	3.8 ± 0.8	N/A	3.9 ± 0.6
Q_5	Overall, are you satisfied with the system?	4.5 ± 0.5	4.3 ± 0.5	3.7 ± 1.2	3.8 ± 1.2
Q_6	Do you think you would use this system again?	4.1 ± 0.6	4 ± 0.9	3.3 ± 1.6	3.1 ± 1.4
Q_7	Was the pace of interaction with the system appropriate?	N/A	3.5 ± 0.5	3.2 ± 1.2	3.3 ± 1.2
Q_8	How often was the system slow in replying? *(1: 'always' to 5: 'never')*	N/A	2.3 ± 1.2	2.7 ± 1.1	2.5 ± 1.2

Result format: average $\pm$ standard deviation.

Since our hypothesis was that a conversational agent is sufficient to handle QA, a set of AIML categories was created to represent the range of utterances and conversational situations handled by a chatbot. The role of the Wizard was to choose the appropriate category and utterance within the available set, and type it into the chat interface; if none of these appeared appropriate to handle the situation at hand, he would create one to keep the conversation alive. The Wizard would ask if the user had any follow-up questions after each answer (e.g., *'Can I help you further?'*).

To collect user feedback, we used two sources: *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 chatbot 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. Inspired by the WOz experiment in Munteanu and Boldea (2000), it consists of the questions numbered Q_1 to Q_6 in Table 3. Questions Q_1 and Q_2 assess the performance of the system and were ranked on a scale from 1: 'Not at all' to 5: 'Yes, Absolutely'. 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.

The WOz experiment was run over 1 week and involved one Wizard and seven users. These were three women and four men of different ages who came from different backgrounds and occupations and were regular users of search engines.

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

The users interacted with the Wizard via a popular, free chat interface 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 (two cases) and a minimum of 5 (one case).

From the *chat logs*, we observed that users preferred not to 'play' with the system's chat abilities but rather to issue information-seeking questions. Users often asked two things at the same time (e.g., *'Who was Jane Austen and when was she born?'*): to account for this in the final prototype, we decided to handle double questions, as described in Section 9.

The *sysReqClarif* dialogue move proved very useful, with 'system' clarification requests such as *'Can you please reformulate your question?'*. Users seemed to enjoy 'testing' the system and accepted the invitation to produce a follow-up question *'Can I help you further?',* around 50% of the time.

The values obtained for the user satisfaction *questionnaire* show that users were generally satisfied with the system's performances (see Table 3, column WOz). None of them had difficulties in reformulating their questions when this was requested and for the remaining questions, satisfaction levels were high. Users seemed to receive system grounding and clarification requests well, e.g., *' ... on references to "him/it", pretty natural clarifying questions were asked'*.

9 Resulting dialogue component architecture

The dialogue manager and interface were implemented based on the scenario discussed in Section 4 and the outcome of the WOz experiment.

9.1 Dialogue manager

Chatbot dialogue follows a pattern-matching approach, and is therefore 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 double and follow-up questions, and tasks involving the QA component.

Handling double 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 double question. Since the core QA component in YourQA is not able to handle multiple questions, these need to be broken into simple questions.

For this, the system uses the OpenNLP chunker[5] to look for the presence of 'and' which does not occur within a noun phrase. For instance, while in the sentence: *'When was Barnes and Noble founded?'* the full noun phrase *Barnes and Noble* is recognized as a chunk, in *'When and where was Jane Austen born?'* the conjunction 'and' forms a stand-alone chunk. If a stand-alone 'and' is found, the system splits the

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

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

double question in order to obtain the single questions composing it, then proposes the user to begin answering the on containing more words (as this is more likely to be fully specified).

Handling follow-up questions. In handling QA dialogue, it is vital to apply an effective algorithm for the recognition of follow-up requests (De Boni and Manandhar, 2005; Yang, Feng and Di Fabbrizio, 2006). Hence, the following task accomplished by the Dialogue Manager is the detection of follow-up questions.

The types of follow-up questions which the system is able to handle are elliptic questions, questions containing third person pronoun/possessive adjective anaphora or questions containing noun phrase anaphora (e.g., 'the river' instead of 'the word's longest river').

For the *detection* of follow-up questions, the algorithm in De Boni and Manandhar (2005) is used, which achieved an 81% accuracy on TREC-10 data. The algorithm is based on the following features: presence of pronouns, absence of verbs, word repetitions and similarity between the current and the eight preceding questions.[6]

If no follow-up is detected in the question q, it is submitted to the QA component; otherwise the following reference *resolution* strategy is applied:

(1) If q is *elliptic*, 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 an NP compatible in number with the referent.
(3) If q contains *NP anaphora*, the first NP in the stack of preceding questions which contains all of the words in the referent is used in place of the latter.

In cases 2 and 3, 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.

Finally, when the QA process is terminated, a message directing the user to the HTML answer page is returned and the follow-up proposal is issued (see Figure 3).

9.2 Implementation

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) and enriched with the WOz experience (Section 8).

We used the Java-based AIML interpreter Chatterbean[7] and extended its original set of AIML tags (e.g., `<pattern>`, `<template>`) with two new ones:

(1) `<query>`, which invokes the core QA module;
(2) `<clarify>`, which invokes the follow-up detection and resolution module (this follows the strategy in Section 5).

[6] At the moment the condition on semantic distance is not included for the sake of speed.
[7] `http://chatterbean.bitoflife.cjb.net/`

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

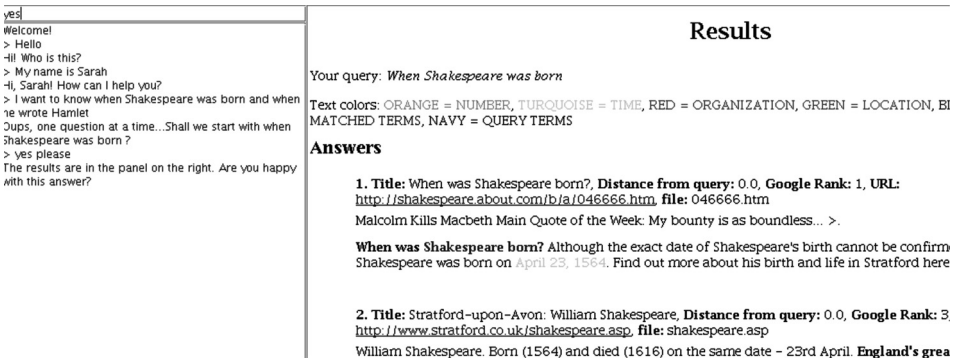


Fig. 3. YourQA's chat interface.

The conversation context is represented as a stack of recent user questions, which is used to perform follow-up resolution as explained in Section 9.1. Moreover, the Chatterbean framework allows to instantiate and update a set of variables, visible during the whole dialogue session. We defined several variables, including:

- the conversation *topic*, i.e., the keywords of the last question in the session which received an answer. These keywords are added to the set of keywords extracted from elliptic questions to clarify them.
- the current user's *name*, used as a key to access the corresponding user model in case the personalization module is activated. This allows the integration of personalized and interactive QA.

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':

```
<pattern>DO YOU KNOW *</pattern>
<template><srai>CLARIFY *</srai></template>
```

The pattern `CLARIFY *` triggers a template calling the AIML tag `<clarify>`, which invokes the follow-up detection and resolution module on the text matching the '*' expression. The module 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`.

```
<pattern>CLARIFY *</pattern>
<template> <set name='clarif'><clarify></star><clarify></set>
<condition name='clarif' value='ELLIPTIC'>...</condition>
<condition name='clarif' value='DOUBLE'> ...</condition> </template>
```

Dialogue Interface. The layout of the YourQA interactive interface consists of a left panel where the chat takes place and a right panel where results are visualized (see Figure 3). As in a normal chat application, users write in a text field and the current session history and the interlocutor replies are visualized in an adjacent text area.

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

10 Evaluation

While the accuracy of standard QA systems can be evaluated and compared using quantitative information retrieval metrics (Voorhees, 2003), dialogue interfaces pose complex evaluation challenges as they differ in appearance, intended application and target users. Indeed, these are often evaluated using qualitative metrics such as user satisfaction and perceived time of usage (Walker, Kamm and Litman, 2000). Similarly, user satisfaction questionnaires and interaction logs appear to be effective tools to evaluate interactive QA systems (Kelly *et al*., 2006).

10.1 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). 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 3 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 slow in replying to you?
Q_9 Which version of the system did you prefer and why?

Questions Q_7 and Q_8 could be answered using a scale from 1 to 5 and were taken from the PARADISE evaluation questions (Walker *et al*., 2000). Q_9 was chosen to assess if and in what terms users perceived a difference between the two prototypes.

Results. From the initial evaluation, which involved eight volunteers, we gathered the following salient results. In the *chat logs*, we observed that the system was able to correctly 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 causing queries to be erroneously completed with previous query keywords. Users tended not to reply explicitly to the chatbot offers to carry on the interaction, 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 successively added new categories to account for this shortcoming.

From the *questionnaire* (Table 3, column Init.), user satisfaction levels (Q_1 to Q_6) are sightly lower than in the WOz experiment (column WOz). Users felt the system slow in replying to the questions, 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%) said they preferred the chat interface of the system (Q_9), because of its liveliness and ability to understand when questions were related.

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

10.2 Final evaluation

Building on the results of the initial evaluation and after drawing additional AIML patterns from the analysis of over hundred chat logs collected since then, we designed a further experiment. For this purpose, we chose nine question series from the TREC-QA 2007 campaign (i.e., series 224, 239, 242, 244, 264, 266, 270, 272 and 278) so that the first question in each series could be understood by a QA system without the need of explicitly mentioning the series target. Moreover, questions should contain anaphoric and/or elliptic references and three questions were retained per series to make each evaluation balanced. For instance, the following questions from series 266 were used to form one task:

```
266.1: 'When was Rafik Hariri born?'
266.2: 'To what religion did he belong (including sect)?'
266.4: '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, so 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, the 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.

Results. The second 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.

The results obtained from the questionnaire for the standard and interactive versions are reported in columns 'Stand.' and 'Inter.' of Table 3, respectively. Although the paired t-test conducted on such results did not register statistical significance, we believe that the evidence we collected is quite interesting. This suggests a good overall satisfaction with both versions of the system (Q_8), with a slight difference in favour of the interactive version.

The standard and interactive versions of the system seem to offer different advantages: while the ease of use of the standard version was rated higher (Q_5), users felt that they obtained more information using the interactive version (Q_1). Concerning interaction comfort, users felt that the interactive version understood their requests better than the standard one (Q_2); they also found it easy to reformulate questions when the former asked to (Q_6). These findings show that even a simple chat interface like ours is very useful in terms of user satisfaction.

However, while the pace of interaction was judged slightly more appropriate in the interactive case (Q_3), interaction was considered faster with the standard version (Q_4); unfortunately, in both cases the interaction speed rarely appears adequate, as also registered from user comments. This probably motivated the fact that users seemed more ready to use again the standard version of the system (Q_7).

An interesting difference with respect to the first evaluation was the preference question Q_9: seven out of twelve users (58.3%) said that they preferred the interactive

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

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 smalltalk 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.

10.3 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 fail to appreciate the advantages of interactivity. We believe that this is due partly to cultural reasons (the search-engine-like non-interactive model of IR biasing users), partly to the fact that the follow-up resolution mechanism of the interactive version is not always 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) this lowers user satisfaction.

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 the potential of which must still be fully grasped by the larger public.

11 Conclusions

We describe YourQA, an open-domain, interactive QA system with a chatbot-based dialogue interface. First, we identify the dialogue model and dialogue manager required to implement open-domain, interactive QA. We verify our model via a WOz study and discuss the practical implementation of such model using a chatbot interface. Our user-centred evaluation shows that users tend to be generally more satisfied with the interactive version of YourQA than with the baseline version. In the future, we will focus on exploring data-driven answer clarification strategies suitable for the open domain such as answer clustering.

References

Alexandersson, J., Reithinger, N., and Maier, E. 1997. Insights into the dialogue processing of VERBMOBIL. Technical Report 191, DFKI GmbH, Saarbrücken, Germany.

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

Allen, J., Ferguson, G., Ringger, E., Zollo, T. S., and Miller, B. 2000. Dialogue systems: from theory to practice in TRAINS-96. In R. Dale, H. Moisl, and H. Somers (eds.), *Handbook on Natural Language Processing*, Chapter 14, pp. 347–376.

Austin, J. L. 1962. *How to do Things with Words*. Oxford, UK: Oxford University Press.

Bertomeu, N., Uszkoreit, H., Frank, A., Krieger, H., and Joerg, B. 2006. Contextual phenomena and thematic relations in database QA dialogues: results from a Wizard-of-Oz experiment. In *Proceedings of the Interactive Question Answering Workshop at HLT-NAACL 2006 (IQA'06)*, New York, USA.

Bos, J., Klein, E., Lemon, O., and Oka, T. 2003. Dipper: description and formalisation of an information-state update dialogue system architecture. In *4th SIGdial Workshop on Discourse and Dialogue*, Sapporo, Japan.

Cahn, J. E., and Brennan, S. E. 1999. A psychological model of grounding and repair in dialog. In *Working Notes: AAAI Fall Symposium on Psychological Models of Communication in Collaborative Systems*, Seacliff, MI, pp. 25–33. Menlo Park, CA: AAAI press.

Churcher, G. E., Atwell, E. S., and Souter, C. 1997. Dialogue management systems: a survey and overview. Technical Report 97.06, School of Computer Studies, University of Leeds.

Cohen, P. 1996. Dialogue modeling. In *Survey of the State of the Art in Human Language Technology*, Chapter 6.3, pp. 192–197. Cambridge, UK: Cambridge University Press.

Core, M. G., and Allen, J. F. 1997. Coding dialogs with the DAMSL annotation scheme. In *Working Notes: AAAI Fall Symposium on Communicative Action in Humans and Machines,* Boston, MA, pp. 28–35. Menlo Park, CA: AAAI press.

Dahlbaeck, N., Jonsson, A., and Ahrenberg, L. 1993. Wizard of Oz studies: why and how. In *Workshop on Intelligent User Interfaces*, Orlando, FL, pp. 193–200, ACM Press.

De Boni, M., and Manandhar, S. 2005. Implementing clarification dialogue in open-domain question answering. *Journal of Natural Language Engineering* **11**(4): 343–361.

Galibert, O., Illouz, G., and Rousset, S. 2005. Ritel: an open-domain, human-computer dialogue system. In *INTERSPEECH'05*.

Grosz, B. J., and Sidner, C. L. 1986. Attention, intentions, and the structure of discourse. *Journal of Computational Linguistics* **12**(3): 175–204.

Hearst, M., Allen, J., Horvitz, E., and Guinn, C. 1999. Mixed-initiative interaction. *IEEE Intelligent Systems* **14**(5): 14–23.

Hobbs, J. R. 2002. From question-answering to information-seeking dialogs. Project presentation. Available at http://www.ai.sri.com/aquaint/.

Jiang, J. J., and Conrath, D. W. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of ROCLING*, Taiwan.

Jönsson, A. 1993. A method for development of dialogue managers for natural language interfaces. In *Proceedings of AAAI'93*, Washington, DC.

Jönsson, A., and Merkel, M. 2003. Some issues in dialogue-based question-answering. In *Working Notes from AAAI Spring Symposium on New Directions in Question Answering*, Stanford, pp. 45–48.

Kato, T., Fukumoto, J., Masui, F., and Kando, N. 2006. Woz simulation of interactive question answering. In *Proceedings of the Interactive Question Answering Workshop at HLT-NAACL 2006 (IQA'06)*, New York, USA.

Kelly, D., Kantor, P., Morse, E., Scholtz, J., and Sun, Y. 2006. User-centered evaluation of interactive question answering systems. In *Proceedings of the Interactive Question Answering Workshop at HLT-NAACL 2006 (IQA'06)*, New York, USA.

Kitano, H., and Van Ess-Dykema, C. 1991. Toward a plan-based understanding model for mixed-initiative dialogues. In *Proceedings of ACL'91*, East Stroudsburgh, MA, pp. 25–32.

Kwok, C. T., Etzioni, O., and Weld, D. S. 2001. Scaling question answering to the web. In *Proceedings of the 10th International Conference on WWW*, pp. 150–161.

Larsson, S. 1998. Using a type hierarchy to characterize reliability of coding schemas for dialogue moves. Gothenburg Papers in Computational Linguistics. New York: ACM.

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press

Larsson, S., and Traum, D. R. 2000. Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Natural Language Engineering* **6**(3–4): 323–340.

Li, X., and Roth, D. 2005. Learning question classifiers: the role of semantic information. *Journal of Natural Language Engineering* **12**(3): 229–249.

Moschitti, A., Quarteroni, S., Basili, R., and Manandhar, S. 2007. Exploiting syntactic and shallow semantic kernels for question/answer classification. In *Proceedings of ACL'07*, Prague, Czech Republic.

Munteanu, C., and Boldea, M. 2000. MDWOZ: a wizard of Oz environment for dialog systems development. In *Proceedings of LREC'00*, Athens, Greece.

Quarteroni, S., and Manandhar, S. 2006. User modelling for adaptive question answering and information retrieval. In *Proceedings of FLAIRS'06*, Melbourne Beach, Florida.

Quarteroni, S., and Manandhar, S. 2007. User modelling for personalized question answering. In *Proceedings of AI*IA'07*, Frascati, Italy.

Schegloff, E., and Sacks, H. 1973. Opening up closing. *Semiotica* **7**: 289–327.

Sinclair, J. M., and Coulthard, R. M. 1975. *Towards an Analysis of Discourse: The English Used by Teachers and Pupils*. Oxford: Oxford University Press.

Steinberger, J., Kabadjov, M. A., Poesio, M., and Sanchez-Graillet, O. 2005. Improving LSA-based summarization with anaphora resolution. In *Proceedings of HLT '05*, East Stroudsburgh, MA, pp. 1–8.

Sutton, S. 1998. Universal speech tools: the cslu toolkit. In *Proceedings of ICSLP'98*, Sydney, Australia.

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

Traum, D. 1996. Dialogue management in conversational agency: the TRAINS-93 dialogue manager. In *Proceedings of the Twente Workshop on Langauge Technology: Dialogue Management in Natural Language Systems (TWLT 11)*, Netherlands, pp. 1–11.

Voorhees, E. M. 2003. Overview of the TREC 2003 question answering track. In *Proceedings of TREC'03*, Gaithersburg, MD.

Walker, M. A., Kamm, C., and Litman, D. 2000. Towards developing general models of usability with PARADISE. *Journal of Natural Language Engineering*, **6**(3–4): 363–377 (Special Issue on Best Practice in Spoken Dialogue Systems).

Webb, N., and Strzalkowski, T. (eds.) 2006. *Proceedings of the Interactive Question Answering Workshop at HLT-NAACL 2006 (IQA'06)*, New York, USA.

Xu, W., Xu, B., Huang, T., and Xia, H. 2002. Bridging the gap between dialogue management and dialogue models. In *Proceedings of SIGDIAL 3*, Philadelphia, PA, pp. 201–210.

Yang, F., Feng, Z., and Di Fabbrizio, G. 2006. A data driven approach to relevancy recognition for contextual question answering. In *Proceedings of the Interactive Question Answering Workshop at HLT-NAACL 2006 (IQA'06)*, New York, USA.

Zhang, D., and Lee, W. 2003. Question classification using support vector machines. In *Proceedings of SIGIR'03*, Toronto, Canada. New York: ACM.

https://doi.org/10.1017/S1351324908004919 Published online by Cambridge University Press