

# Handling speech input in the Ritel QA dialogue system

Boris van Schooten<sup>1</sup>, Sophie Rosset<sup>2</sup>, Olivier Galibert<sup>2</sup>,  
Aurélien Max<sup>3</sup>, Rieks op den Akker<sup>1</sup>, Gabriel Illouz<sup>3</sup>

<sup>1</sup>Human Media Interaction, University of Twente, Netherlands

<sup>2</sup>Spoken Language Processing Group, LIMSI-CNRS, France

<sup>3</sup> Language, Information and Representations, LIMSI-CNRS and Université Paris-Sud 11, France

{schooten, infrieks}@cs.utwente.nl, {rosset, galibert, amax, gabrieli}@limsi.fr

## Abstract

The Ritel system aims to provide open-domain question answering to casual users by means of a telephone dialogue. This implies some unique challenges, such as very fast overall performance and large-vocabulary speech recognition. Speech QA is an error-prone process, but errors or problems that occur may be resolved with help of user feedback, as part of a natural dialogue. This paper reports on a pilot study conducted with the latest version of Ritel, which includes search term confirmation and improved follow-up question handling, as well as various improvements on the other system components. We collected a small dialogue corpus with the new system, which we use to evaluate and discuss it.

**Index Terms:** speech QA, dialogue systems, confirmation.

## 1. The Ritel platform

The overall objective of the RITEL project is to develop a research system to investigate new methods in natural Human-Computer Interaction. While there are few interactive speech QA systems in existence, Ritel is especially unique because it emphasises practical use. We require that the system should react in no more time than a human would, and be perceived as natural as possible by human users. This would both help users perform their tasks more efficiently, and make them want to use such systems. To achieve this, overall system speed should be very high, and vocabulary should be possibly unlimited.

The system architecture (illustrated in figure 1) is highly distributed and based on servers and specialized modules that can exchange messages. The advantages of this type of architecture are twofold. First, computation-intensive processing can be performed on dedicated machines, assigning all available memory and CPU to a specific task, which will allow us to investigate smooth asynchronous operating in future versions of the system. Second, the modular approach invites external contributions to be integrated into the system, thus offering a research framework for investigation in Human-Computer Interaction with competing strategies but also different application tasks. For instance, a project based on the presented system will include building a personal phone assistant.<sup>1</sup>

Particularly unique in this architecture are: a unique analysis module is used for document indexing and user query analysis; user inputs are handled by specific registered modules according to their type; dialogue management is not centralised as in other approaches [1], but rather, distributed over the various components of the platform.

<sup>1</sup>The funding of a post-doctoral researcher for this particular project starts from April 2007.

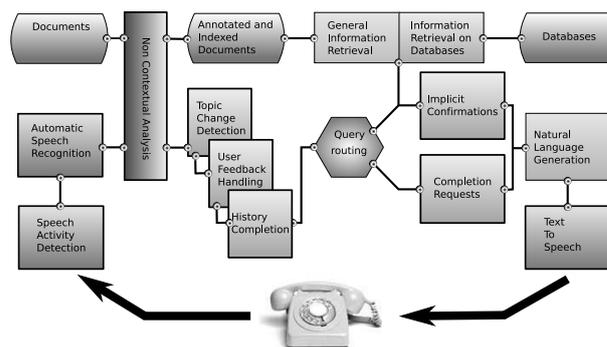


Figure 1: System architecture

### 1.1. Speech Activity Detection (SAD) and Automatic Speech Recognition (ASR)

The SAD system is identical to the one in [2], and the ASR system is similar with new specialized acoustic models which have been built from audio data comprised of 5 hours from the RITEL corpus and 70 hours from the ARISE corpus [3]. The language models have been created by interpolating various sources on the RITEL development corpus (1 hour). These sources are: RITEL training corpus, newspapers text from 1988 to today, manual transcriptions of radio and TV news shows, and 1,100 questions containing 13,000 words from quiz questions. We also extracted from the training corpus about 130 patterns of user utterances representing 5 different topics and generated 39,000 utterances from them using the HTK toolkit [4]. The vocabulary is composed of about 65K words. The out-of-vocabulary rate is 0.8% on the development corpus and 0.4% on the test corpus, which is rather good for an open-domain system for French. Performance is 0.85 real time streamed for a word-error-rate of 27.8% on the development corpus and 28.0% on the test corpus. The major difference between this system and others (for example [5, 6]) is that the ASR component has to be true real time and has to handle spontaneous users utterances instead of well formed, often read, questions. At that level, the ASR terminates when the user stops speaking, thus enabling extremely snappy and natural interaction.

### 1.2. Non contextual analysis and Question-answering

Analysis of both indexed documents and user utterances are handled by the same module which is called *Non Context-*

tual Analysis (NCA) because no information from the dialog or previous utterances is used. The general objective of NCA (see [2] for details) is to find the bits of information that can be of use for search and extraction, which we call *pertinent information chunks*. These can be of different categories: named entities, linguistic entities (e.g. verbs, prepositions), or specific entities (e.g. scores). All words that do not fall into such chunks are also annotated by following a longest-match strategy to find chunks with coherent meanings. Specific types of entities were added to improve communication management (e.g. *<Dopening>hello</Dopening>*; *<Qdial>I'm looking for</Qdial>*), and specific linguistic phenomena such as negation (e.g. *<Qnegdial>I'm not looking for</Qnegdial>* an *<Qneg\_info>animal</Qneg\_info>* but for a type of car ; *<Qneg\_sys>I did't ask for that</Qneg\_sys>*), and topic markers extracted from our corpus (e.g. *literature, geography, history, politics*) are used for topic detection. The types that should be detected correspond to two levels of analysis: named-entity recognition and chunk-based shallow parsing. The analysis uses an inhouse word-based regular expression engine and handcrafted rules which have access to various lists for names, countries, cities, etc. The full analysis comprises some 50 steps and takes roughly 4 ms on a typical user utterance, which proves fast enough for the analysis of short text but yet too slow for allowing smooth reaction time if analyzing dynamically large document sets from the Web. Figure 2 shows a result of the annotation of a user utterance.

```
<_Qneg_dial> je ne veux pas d' informations </>
<_neg_info> <_prep> sur </> <_pers> Benedetti </>
</> <_Qdial> je voudrais <1> une </> information
sur </> <_det> le </> <_range_objet> dernier </>
<_prix> <_Prix> prix Nobel </> <_type_prix> de la
paix </> </>
```

Figure 2: Annotation of a user utterance: *je ne veux pas d' informations sur Benedetti je voudrais une information sur le dernier prix Nobel de la paix (I am not looking for information about Benedetti, I want information on the last recipient of the Nobel Peace Prize)*

The Question-answering system is described in [2]. The indexing server main role is to retrieve snippets, i.e. lines of documents corresponding to a given query. Queries take the form of a list of type/value pairs and lone types. If no results can be found that match the query, backoff rules are applied.

In a QA dialogue, we expect users to ask questions in the context of previous utterances, i.e. follow-up questions (FQ) [7]. RITEL handles these by adding elements coming from the dialogue history (context completion) by means of semantic matching. This form of completion relies heavily on the defined entities and matching rules. We rely on confirmation and user feedback to repair errors in this process.

## 2. Handling user feedback

Both speech and QA systems are known for their high error rates. RITEL is no exception. We have different sources of noise in our system, in particular ASR, context completion, and the actual information retrieval (IR). Additionally, the ASR and context completion modules do not yield any confidence ratings, which means it is hard to make out when or where errors occur. So, we consider feedback from the user an important

mechanism for handling these sources of error. We identify several ways to do this:

- Confirm the material that we decide to feed into the IR (confirming both ASR and context completion in one go)
- Confirm the answer (confirming IR)
- Confirm topic or let the user express topic (confirming context completion and answer type)
- Let the user repeat him/herself, with or without varying formulations (confirming in particular ASR and possibly IR and context completion)

In order to examine the meaningfulness for improving recognition by means of these techniques, we first examined the ASR performance in the original RITEL corpus for relevant cases. We observed the following:

- While named entities and other specific keywords were recognised with a rate of less than 55% (less than the average WER), we found that if the user repeats him/herself several times, keywords are more likely to be recognised. In a series of dialogues where the user repeats him/herself between 4-6 times, we found that 80% of the keywords that were recognised multiple times were correct.
- General topics were more often recognised correctly. We found that approximately 90% of the general topics (cinema, sports, etc.) that are mentioned by users are recognised by the ASR.
- Positive/negative feedback words and phrases like "yes", "no", etc., were usually recognised correctly.

This indicates that the strategies we mentioned are good strategies for confirmation as far as the ASR is concerned. So, we decided to allow for all of them in our confirmation strategy. Note that confirmation using speech also implies that users will not always understand the system's speech. However, we found that users almost never signalled non-understanding or asked the system to repeat itself, neither did the dialogues indicate that users regularly misunderstood the system. So, we decided to leave this issue as a possible subject for future research.

### 2.1. Confirmation strategy

While we wish to confirm items, we also wish the system to be able to directly answer a series of questions without requiring extra dialogue turns for confirmation. Since we have no confidence information to enable us to choose when to confirm explicitly, we chose to always use implicit confirmations. The system gives an answer whenever IR can be performed successfully. Even if the user does not react to confirmations, this strategy should still let the user judge whether the system has understood correctly, and repeat him/herself when appropriate. The user may choose to either give negative feedback and repeat (part of) the question (i.e. "No, I meant ..."), explicitly disconfirm anything mentioned by the system (i.e. "No, I didn't mean ..."), or implicitly or explicitly confirm anything mentioned by the system, and provide additional information as necessary.

It seems most natural to confirm IR material by producing a coherent sentence. However, this has some major problems: the ASR's high WER, and the nature of context completion and IR, which is based on keywords and answer type detection, usually makes it impossible to produce coherent sentences. So, we decided to confirm IR material directly, assuming that users will understand what is meant. That is, we confirm key phrases and the recognised question type directly (see figure 3).

## 2.2. Implementation

In our confirmation model, we separate the dialogue context used for confirmation of the current question from that used for FQ history completion, so that we obtain two separate contexts. Confirmation occurs in a cycle, which is completed only after the system has obtained enough material to perform IR and give an answer, and the user (implicitly or explicitly) confirms satisfaction with the system's answer. So, as long as the user gives negative feedback or tries to correct the system's confirmations, s/he is able to adapt the query until it is deemed satisfactory. When the cycle ends, the confirmation context is copied to the FQ context, and a new cycle is started (see also figure 3).

U1: <i>On what continent lies Botswana ?</i> (ASR: <i>on what continent lies {fw} the</i> ) <b>conf.context:</b> kw = {continent, lies} qtype = location <b>FQ context:</b> (empty) S1: You are looking for a location, but what precisely do you want to know? U2: <i>The continent of Botswana.</i> (ASR: <i>the continent of Botswana</i> ) <b>conf.context:</b> kw = {continent <sub>confirmed</sub> , lies, Botswana} <b>qtype=location<sub>confirmed</sub></b> <b>FQ context:</b> (empty) S2: You are looking for a location related to continent and Botswana. I think it's South Africa. Your next question? U3: <i>What is its land area</i> (ASR: <i>what is its land area</i> ) <b>conf.context:</b> kw = {land area} qtype = number <b>FQ context:</b> kw = {continent, lies, Botswana, S.Africa}
---

Figure 3: Example dialogue illustrating confirmation strategy and history mechanism (translated from French); kw = keywords, qtype=answer type, conf.context=confirmation context.

Dialogue management partakes in the QA process at two specific points: handling the user's confirmations after NCA but before FQ context completion, and generating an utterance after QA has finished. It makes use of the following NLP: dialogue act recognition by means of the special dialogue NCA tags; topic change detection, which is performed as part of FQ history completion; and importance ranking of keywords based mostly on their types. Context is modelled by adding salience levels to each keyword, based on their ranking and frequency of mention. In particular, higher ranked keywords which have not yet been confirmed have priority in confirmation, and a keyword is considered confirmed when it is mentioned twice by the user during one confirmation cycle.

## 3. User evaluation

For the user evaluation, a database of 60K documents from the Web has been collected, totaling 8.5Gb after annotation. It contains some 9 million different (*type, value*) pairs, only 2 million of which have 3 or more entries in the documents. We obtain the following figures: the in-memory part is 220Mb; on-disk document references are around 3.3Gb; lists of values are around 300Mb. The indexing server (see [2] for more details) allows to achieve good speed for answering.

Six different people has been asked to call the system. Each subject had received a list of 10 possible topics (as examples). They were told to feel free to ask the system whatever they want however they want. The users did not have any familiarity

with QA technology, although four of them have interacted with an older version of the system. They thought the system "did progress" and it understood their demands better. Since answering performance wasn't really better, this is apparently the result of the feedback given through confirmation. This scenario was used to collect 20 dialogs comprising 280 user utterances for 1,852 words (364 distinct).

### 3.1. ASR and NCA evaluation

The 15 minutes of user speech are a little small for hard numbers, but still we obtained a WER of 25.3% on general vocabulary, 48.4% on proper nouns and 12.6% on topic markers. We observed that the overall behavior of the system tends to make the users speak more naturally (i.e., avoiding over-articulation, lessening manifestations of annoyance, etc.), which in turn may help speech recognition.

In order to evaluate the NCA component, we manually annotated the collected corpus (280 user utterances) with the tags defined in our ontology. Evaluation was carried out on the results of the analysis produced by the system during the dialogs. The results obtained for the 118 different types of tags found in this corpus were 88.5% Precision, 88.4% Recall and 88.5% F-measure. Specific error rates on tags were also measured:

1. wrong type and boundary errors (**Sft**): 124 tags (10.0%). 51 of these were on tags related to communication management and negation, which play an important role in the confirmation process and history management. Errors whereby a title question marker was confused for a citation question marker occurred 3 times, which is quite problematic for subsequent QA.
2. wrong type errors (**St**): 33 tags (2.6%). Of these, 6 lead to topic detection errors (e.g. confusion between an organisation and a substantive), and 6 other were simply confusion between titles and proper nouns which are less problematic as the backoffs implemented in the QA system can correct them. Other confusions (e.g. verb or substantive for adjective) had less impact on QA.
3. wrong boundary errors (**Sf**): 23 tags (1.8%). Of these, 18 involve question tags or types and can therefore have a negative impact on confirmation and possibly QA.

Although this evaluation was carried out on manual transcriptions, it provides realistic measures of actual and upper-bound performance of the system.

### 3.2. Dialogue and user feedback

For evaluating dialogue, only 15 of the 20 dialogues were considered appropriate, since the first 5 were found to contain FQ only, which led to an adaptation of the instructions given to the users. These comprise a total of 244 user utterances. We annotated these with dialogue act type, presence of topic shifts, and self-containedness for the sake of QA, mostly following [7], see table 1. A vast majority of the utterances were questions. The non-questions were mostly explicit topic and type mentions (such as "This is a new question" or "I have a question about ...", and answers to system questions. There were only a few explicit disconfirms, all of them by one user.

How well did the ASR manage to recognise the relevant information in the different types of utterance? To measure this, we subdivided the ASR results according to whether the essential information was recognised correctly. We found that 131 utterances (54%) were sufficiently well recognised, that is, all

relevant keywords and answer type cues were recognised, as well as any relevant communication management cue phrases. Some 76 (31%) were partially recognised, that is, at least part of the IR material and dialogue cues. This leaves 37 utterances (15%) which were not recognised to any useful degree.

We found some user act types where the ASR performance distribution deviates significantly from the overall one. Relatively well recognised were topic announcements, negative feedback, and non-self-contained FQs, as we predicted. Particularly ill recognised were reformulations, self-contained FQs, and repetitions. This seems to be related to the presence of domain-specific keywords such as named entities, which were the toughest for the ASR. Interesting here is that non-self-contained FQ were better recognised than self-contained ones because, typically, the named entities were left out. This suggests that context completion can be useful if we have already established the most difficult keywords earlier in the dialogue.

To further examine the dialogue interactions between user and system, we look at the subdivision of the different system utterance types, the user's feedback utterances, and the relationships between them. There were 229 system responses in total, subdivided as in table 2. Most user feedback is implicit, consisting of informs (users giving partial information in a non-question form, mostly responding to a system question), and repetitions and reformulations. A minority were direct negative responses or explicit announcements of a new topic, see table 3.

So, we find that almost all corrections are repetitions or informs. As far as our confirmation strategy is concerned, it appears that confirmation was not picked up in the sense that users confirmed or disconfirmed anything explicitly, but users did use it to determine whether they were unsatisfied with the response. What users mostly did was essentially repeat when they found the system reply unsatisfactory, which means that the "repeat at least twice" kind of confirmation will work well.

Table 1: *User dialogue act types found in the corpus.*

29	(12%)	new questions (that is, in-dialogue topic shifts)
74	(30%)	FQ (27 non-self-contained, 47 self-contained)
87	(36%)	reformulations, repetitions and informs
18	(7%)	negative feedback or topic announcements
7	(3%)	byes
12	(5%)	miscellaneous utterances

Table 2: *Types of system responses in the corpus.*

115	(50%)	give an answer and confirm IR material
55	(24%)	confirm answer type and ask for more info
43	(19%)	confirm keywords and ask for more info
7	(3%)	ask the user to repeat
9	(4%)	indicate non-understanding

Table 3: *Types of user feedback in the corpus.*

47	repetitions (of which 35, or 74% were self-contained; 60% were after system asked for more info)
23	reformulations (almost all were self-contained; 52% were after system asked for more info)
17	informs (almost all were partial, most (75%) were after system asked for more info)
12	topic change announcements
6	explicit topic announcements
3	disconfirms (all by one user)

If only we can detect the difference between when the user repeats or when the user poses a new question, we can use this to handle confirmation properly. However, it is not clear how to do this. Most repetitions and reformulations have no or few surface cues to help detect them, although informs could be detected by looking at the absence of a question form.

The system was quite successful at detecting topic change announcements. This was less so for explicit topic/type mentions. While the system tags phrases that can be considered cues for topics, we found no significant correlations with topic announcements, topic shifts, or self-contained versus non-self-contained questions.

## 4. Conclusion

The main differences/improvements of the RITEL system concern the ASR which offer now a WER under 28% and the NCA component which better take into account specific communication management types. This version of the system fully integrates the general open-domain QA system.

This version includes search term confirmation and improved follow-up question handling. We found that users seldomly reacted directly to the implicit confirmations, but tended to react by repeating and rephrasing. This appears consistent with findings on other, non-QA dialogue systems [8]. Possibly, explicit confirmation may be useful, if we can determine in what cases to confirm explicitly. We did find that spotting repeated keywords, as well as recognising topic shifts and topic announcements appears to be a successful strategy for confirmation and handling dialogue context.

## 5. Acknowledgements

This work was partially funded by the Netherlands Organisation for Scientific Research (NWO) IMIX programme, the European Commission under the FP6 Integrated Project IP 506909 CHIL and the LIMSI AI/ASP Ritel grant.

## 6. References

- [1] V. Zue and J. Glass, "Conversational interfaces: Advances and challenges," *Proceedings of the IEEE, Special Issue on Spoken Language Processing*, vol. 88, 2000.
- [2] S. Rosset, O. Galibert, G. Illouz, and A. Max, "Integrating spoken dialog and question answering: the ritel project," in *InterSpeech'06*, Pittsburgh, USA, 2006.
- [3] L. Lamel, S. Rosset, J. Gauvain, S. Bennacef, M. Garnier-Rizet, and B. Prouts, "The LIMSI ARISE system," *Speech Communication*, vol. 31, no. 4, pp. 339–354, 2000.
- [4] C. U. E. Department, "Htk speech recognition toolkit," <http://htk.eng.cam.ac.uk/>, 2006.
- [5] S. Harabagiu, D. Moldovan, and J. Picone, "Open-domain voice-activated question answering," in *Proceedings of COLING*, 2002, pp. 1–7.
- [6] C. Hori, T. Hori, H. Isozaki, E. Maeda, S. Katagiri, and S. Furui, "Study on spoken interactive open domain question answering," in *Proceedings of SSPR 2003*, 2003.
- [7] B. W. van Schooten and R. op den Akker, "Follow-up utterances in QA dialogue," *Traitement Automatique des Langues*, vol. 46, no. 3, 2005.
- [8] J. Shin, S. Narayanan, L. Gerber, A. Kazemzadeh, and D. Byrd, "Analysis of user behaviour under error conditions in spoken dialogs," in *Proceedings of ICSLP*, 2002.