Vidiam: Corpus-based Development of a Dialogue Manager for Multimodal Question Answering

Boris van Schooten and Rieks op den Akker

Abstract In this chapter we describe the Vidiam project, which concerns the development of a dialogue management system for multi-modal question answering dialogues as it was carried out in the IMIX project. The approach that was followed is data-driven, that is, corpus-based. Since research in Question Answering Dialog for multi-modal information retrieval is still new, no suitable corpora were available to base a system on. We report on the collection and analysis of three QA dialogue corpora, involving textual followup utterances, multimodal follow-up questions, and speech dialogues. Based on the data, we created a dialogue act typology which helps translate user utterances to practical interactive QA strategies. We then report how we built and evaluated the dialogue manager and its components: dialogue act recognition, interactive QA strategy handling, reference resolution, and multimodal fusion, using off-line analyses of the corpus data.

1 Introduction

The Vidiam project (DIALOGue Management and the VISual channel) aims at combining multimodal unstructured question answering (QA) technology with natural language dialogue. It is well known that, in order to be feasible, dialogue systems can cover only a limited domain, so that typical dialogues within the domain can be modelled explicitly. Most dialogue systems follow a “slot-filling” approach to manage the dialogue, which amounts to having the user fill in predetermined values. There are relatively few dialogue systems which handle dialogue for an underlying QA engine. Most of these systems only have a limited concept of QA dialogue, so we are still far away from a comprehensive coverage of the potential of QA dialogue.

Boris van Schooten
University of Twente, Netherlands e-mail: schooten@ewi.utwente.nl

Rieks op den Akker
University of Twente, Netherlands e-mail: infrieks@ewi.utwente.nl

In the Vidiam project we have concentrated mostly on handling user follow-up utterances (FU), which are any user utterances given after the system produces (or fails to produce) an answer. A major class of FU are follow-up questions (FQ). These are regular QA questions, except that they are incomplete, and can only be understood as part of previous utterances in the dialogue.

We explored the possibilities of FU handling by bringing together two perspectives: the user perspective (what do users want and how do they act?) and the technical perspective (what is technically possible at this moment?). The technical perspective involves a review of current QA dialogue systems, and an overview of the FU handling techniques that are known to work. See section 2. The user perspective involves collecting corpora of QA dialogues and user FU. We then bring together the two perspectives by classifying the utterances in the corpora according to what type of processing and response is required by the system, based on the range of known technical possibilities. This is covered in section 3.

Besides corpus analysis, we also worked on two dialogue managers (DAMs): the Vidiam DAM, which is a module in the IMIX demonstrator, and dialogue management functionality for the Ritel system, which was developed in a 5-week collaboration with the Ritel team [8]. The Vidiam DAM is based on the FU classification, and in section 4 we evaluate how well the classification and subsequent FU handling performs. The Ritel system was chosen because it complements the IMIX system in terms of features (see also section 2). In particular, it is speech-based. Also, in Ritel it was possible to collect a corpus of full dialogues with the real system. Our analysis of Ritel concentrates on speech, and is mostly described in section 3.3. Speech was never implemented in the Vidiam DAM because of the lack of appropriate ASR, and hence, we have no evaluation of speech handling in our Vidiam DAM evaluation.

We described most of our findings in previous work [23, 20, 25, 24]. In this chapter, we summarise our analysis of the different corpora and reflect on the consequences for the design of a QA dialogue system.

1.1 QA dialogue system features

The manner in which FU can or should be handled depends on the features of a particular QA system. We shall characterise QA system features along two dimensions: the available modalities, and the ability to answer different types of questions. Within this framework we shall then place IMIX and Ritel, and the systems from the literature.

1.1.1 Available modalities

Most QA systems only handle text (typed) questions, answered by text answers. A minority of systems, such as Ritel, handle speech input, which is already quite a different kind of game, because of the speech processing problems of large-vocabulary

ASR. Few QA systems, among which are the IMIX and SmartWeb [19] systems, handle multimodal FQ to multimodal answers.

Speech.

ASR output for QA is typically so noisy that one can expect something like half of the output being unuseable. The most serious problem with the current Ritel system lies in the quality of ASR, with a word error rate of about 30%, and the error rate of keywords being around 50% [25]. The IMIX ASR was not even tested with a sufficiently large vocabulary. Speech requires repair dialogue, which is treated as a separate subdialogue in Ritel, and is therefore independent of FQ handling. However, speech may influence user behaviour, and may make certain FQ more or less common or desirable. For example, anaphoric FQ would especially have added value in speech QA, as they reduce the need for repeating the keywords that are so difficult to recognise by an ASR.

Multimodality.

We define multimodality as the combination of the presentation of multimodal answers (text + pictures) with multimodal FQ (text/speech + pointing or gesturing on the screen). Multimodality requires an extra interpretation and/or generation step in the QA process.

Handling of multimodal FQ is rarely found, but there are some multimodal “question answering” dialogue systems which do not use unstructured QA technology, but a more knowledge rich approach. Examples of these are found in [28] (a Chinese storytelling system), [17] (the Andersen storytelling system), and [11] (a navigation dialogue system).

We shall assume that multimodal FQ can be handled in basically the same way as unimodal ones, except that there are specific classes of FQ that only exist in the multimodal case. In [24], we found that almost all multimodal FQ in our corpus were primarily linguistic. The pointing gestures were used only to disambiguate the references made in the linguistic component of the utterances. This is mostly consistent with findings in the Andersen system, which found only 19% gesture-only user turns. These could be interpreted as simple “What is ...?” queries.

1.1.2 Ability to answer different types of questions

In isolated-question QA, we can identify several distinct, rigidly defined, question types. The most common are factoid, list, definition, and relationship questions [26]. Other question types we found in QA dialogue systems are analytic (HITIQA) [21], complex (a set of simpler questions, FERRET) [10], encyclopedic (IMIX) and yes/no (IMIX). In this chapter we will mainly consider factoid and encyclopedic

questions. Factoid questions are answerable by a single word or phrase, while encyclopedic questions are answerable by (slightly modified) text fragments, taken from a document out of a set of documents. The documents are typically a few paragraphs of text on a specific subject. In fact, the IMIX database documents are mostly sections from medical encyclopedias.

This classification concerns isolated questions only. Many or most QA systems assume that an FU is an FQ that can be handled as a regular factoid QA question. An FQ is usually limited to whatever you can feed into an IR engine that is made for isolated questions. Additionally, the FQ hardly ever refers to previous answers, because factoid answers are just single words and phrases. Consequently, FQ handling methods typically concentrate on how to adapt the input to the standard QA or IR process in order to include context from previous questions.

The dialogue tracks of TREC [26], QAC [13], and CLEF [5] have a particularly limited view of what an FQ is, in order to make evaluation easier. However, as a consequence they suffer from other methodological problems, which was the reason why the TREC dialogue track was dropped [27]. All FQ in the dialogue tracks assume that each question always has exactly one answer, consistent with the “factoid” paradigm. The dialogues then assume that the user never needs to react to these answers, but follows a planned path. In reality, answers may have varying degrees of correctness or completeness, and users will respond to this with a continuum from pre-planned FQ to FQ concerning details of the answer, to utterances that indicate the user is unhappy with the answer. For example:

- **Reactions to wrong or unclear answers.** Many answers will be wrong, and many encyclopedic answers will be only partial or unclear. Such bad answers should not cause the dialogue to disintegrate. The DAM should at the least know something about users’ reactions when they are confronted with wrong or unclear answers.
- **Discourse questions.** These are questions that refer to the specific discourse of an answer [22]. So, they cannot be understood without taking into account the answer literally.
- **Other meaningful FU that are not FQ.** Most QA research does not account for user reactions other than domain questions that are readily processable. In real-life dialogues, we see other phenomena: utterances that indicate uncertainty about the correctness of the answer, negative feedback, and acknowledgements.

One goal of the Vidiam project is to explore what answering strategies are best for these kinds of FU. The examples in table 1 illustrate some of the FQ that we believe a QA DAM should handle.

2 Overview of existing systems

In this section we give an overview of existing interactive QA systems that handle FU and are based on unstructured QA. In all systems, we can distinguish a DAM and

Table 1 Example utterances.

Typical TREC-style factoid follow-up question sequence:

Who is the filmmaker of the Titanic?

Who wrote the scenario?

How long did the filming take?

Possible non-factoid follow-up question sequence:

What do ribosomes produce? (factoid initial question)

All types of proteins?

And how does that work?

Some “real life” FQ from the corpora:

So the answer is “no”?

Shorter than what? Another form of schizophrenia?

How often is often?

What are these blue spots? (multimodal FQ)

Is this always true? (a reaction to an explanatory answer)

What does ‘this’ refer to? (reference to a pronoun in the answer)

One minute per what? (reply to a factoid answer “one minute”)

What is meant here by ‘hygienic handling’?

a QA engine. The DAM is the software that determines a question’s context from previous utterances and gives responses other than QA answers. It is imaginable that this software is integrated in such a way that it cannot be practically separated from the QA engine, but in practice it is usually quite clear how context is calculated and passed to a QA/IR system.

In previous work [23] we distinguished between a “black box” and an “open” QA engine, based on what information the DAM can pass to the QA engine. The black box model requires isolated plain-text questions. This gives the fewest possibilities but has the advantage of modularity. The black box model requires the DAM to rewrite any FQ into a self-contained question, which, as we found in the IMIX project, is both difficult and conceptually problematic [23]. Unfortunately, the QA engines in IMIX are all black box. So, we have assessed the possibilities of open QA engines by means of off-line analysis, and collaboration with the Ritel system, which has an open QA engine.

If we have an open QA, the ability to pass context depends on the nature of the QA. Typical QA systems have a separate information retrieval (IR) stage, but there are differences between individual systems. In particular, the internal representation of the IR query varies. Some QA systems translate a question into an IR query by classifying it into a specific question type with appropriate arguments, others build a more complex semantic frame from the question (such as SmartWeb [19] and Ritel [8]). The IR itself may be done in different ways, such as using semantic document tags resulting from a pre-analysis phase (such as IMIX’s Factmine and Rolaquad), or by matching syntactic structures of question and answer (such as IMIX’s QADR). Because of all these variations, we shall assume that a DAM will only be able to give relatively simple and general hints to the QA back-end, such as pointers to the questions that should be considered context for the current question.

Table 2 attempts to give an exhaustive list of QA systems that handle FU from the literature, and compares these systems according to the features we identified.

Table 2 Overview of existing QA systems that handle FU. The **discourse** column indicates that a system handles discourse questions. The **domain** column gives a coarse characterisation of the domain, with *open* being an open-domain system in the classical sense. *GUI* indicates that the system uses graphical user interface style interaction for FQ handling. *Qtype* indicates that the question type (that is, person, date, number, etc) may be passed as dialogue context to the IR. *Keyword* or *kw* indicates that keywords or key phrases may be passed as dialogue context. *Blackbox* indicates that only full NL questions may be passed to the IR.

System	lan- guage	speech	multi- modal	dis- course	question types	QA-DAM interface	domain
IMIX	Dutch	-	yes	yes	encycloped.	blackbox	medical
Ritel [8]	French	yes	-	-	factoid	qtype+kw	open
Hitiqa [21]	English	-	-	GUI	analytic	GUI	news
De Boni et al.'s system [4]	English	-	-	-	factoid	N/A	open
SmartWeb [19]	German	yes	yes	GUI	factoid	GUI	multiple
Rits-QA [7]	Japan.	-	-	-	factoid	blackbox	open
ASKA (Nara institute) [12]	Japan.	-	-	-	factoid	qtype+kw	address
KAIST [18]	English	-	-	-	factoid	qtype+kw	open
NTU [15]	English	-	-	-	factoid	keyword	open
OGI school's system [31]	English	-	-	-	factoid	keyword	multiple
FERRET [10]	English	-	-	GUI	complex	GUI	news

2.1 FQ context completion strategies

Depending on the QA-DAM interface, passing the required context to the QA for answering a regular FQ can be done in several ways. If we look at current QA dialogue literature, we can distinguish three basic approaches to handling FQ:

1. **rewriting a follow-up question to a self-contained question.** This is applicable to black-box QA. The effectiveness of a rewritten sentence still depends on the internals of the QA engine. An advantage of rewriting is that a successfully rewritten question ensures that our interpretation of the FQ is correct and complete. Moreover, this correctness and completeness can be readily evaluated by a human annotator by simply judging whether the question is self-contained and answerable. A sentence can be rewritten in different ways, and according to different criteria, such as:
 - a. all appropriate search terms occur in the rewritten sentence
 - b. sentence is syntactically and semantically correct
 - c. sentence is as simple as possible
 - d. sentence is answerable by a human "QA"
 - e. the QA gives the correct answer

The criteria that we emphasise here are (b), (c), and (d). We believe (a) is insufficient given the number of non-search-term approaches, and is contained in (d). While empirical purists may consider (e) to be the "ultimate proof" of suitability, in reality, it depends on both the quality of the particular QA and the document

database used. The criteria (b)-(d) have the additional advantage that they can be evaluated readily by a human annotator.

Basic forms of rewriting include replacing an anaphor with a description of its referent, and adding missing phrases to elliptical or otherwise incomplete sentences. Such rewriting satisfies criteria (a)-(d). For example, the (Japanese) Rits-QA system [7, 6] uses two kinds of ellipsis expansion, and anaphor resolution. Their scheme managed to rewrite 37% of the FQ in their corpus correctly.

2. **combining the FQ's IR query with that of previous utterances.** We will call this the *IR context* approach. This approach requires an open QA engine. The advantage is that we can take shortcuts, which may enable us to avoid having to fully interpret an FQ when not strictly necessary. As an example of this approach, consider the (Japanese) Nara Institute system [12]. It handles FQ using IR context, based on its "question type/keyword" based IR:

- a. Analyse the question, obtaining keywords, question type, and answer class. Obtain question type and answer class from dialogue history if missing.
- b. Add keywords from dialogue history (from both system and user utterances).
- c. Remove keywords with low weights, or with the same semantic class as answer class. For each semantic class, keep only the keyword with the highest weight.
- d. If no answer is found, relax the current request until an answer is found.

Note that step (b)-(c) are similar to a regular salience-based linguistic reference resolution scheme, although they avoid having to resolve specific referents. The (English) KAIST system [18] uses an approach similar to step (b)-(c), except that their reference resolution does resolve to a specific referent. Similar schemes are found in the (French) Ritel [8] and (German) SmartWeb [19] system, both of which use merging of semantic frame representations. Ritel also uses request relaxation.

3. **Searching within previous search results.** Searching within the top n documents retrieved by a previous question seems to be a successful strategy, as pointed out by De Boni [4]. De Boni found that about 50% of FQ could be answered using this strategy (given a perfect enough QA). Such a facility would even enable users to use a strategy of incremental refinement, that is, using multiple simple questions to assemble a query throughout multiple utterances. A simple version of this method is searching only the document where the previous answer came from. This has some significant advantages in terms of simplicity. In particular it is a method that will work for any QA engine that works by selecting text fragments from documents.

2.1.1 Universal steps in the context completion process

We found that context completion follows the same basic strategy in different systems. Three main steps can be distinguished, each of which can be developed and evaluated separately. We argue that following this three-step approach explicitly will

lead towards a more structured development methodology. QA contests may even design new tracks based on these steps.

1. **Identification of need for context completion.** This is analogous to detecting if the question is self-contained or not. This is the most basic step in any QA that wishes to handle FQ. Some QAs only implement this step, then apply some very basic IR algorithm, with good results. Note that existing TREC/QAC context tracks do not address this step at all, since the TREC/QAC dialogues do not contain topic shifts.

In case a system has support for discourse questions or other FU, they can be detected in this step, as IMIX does.

In the corpora we found many questions where adding context is not required, but not harmful either. Let us define this a little more clearly. At one end we have the *harmfulness* condition, which indicates that adding any context is harmful (such as indicated by the classical “topic shifts”). On the other end we have the *insufficiency* condition, where *not* adding context makes the question insufficiently complete to answer. What is in-between are basically the class of self-contained on-topic FQ.

Features typically used to perform this step are: presence of pronouns and anaphors, ellipsis, general cue words, presence of keywords specific enough for successful ir, semantic distance of keywords with those of previous utterances.

Performance is measured by the percentage of correct (neither harmful nor insufficient) classifications. Performance baseline is choosing the most often occurring FU class (which is typically the non-self-contained FQ).

Ritel uses the notion of *topic shift* which is meant to indicate that it is harmful to use the context completion machinery when the user has changed to a completely different topic. Ritel sets a context completion prohibition flag if topic change is detected. Topic shift is also used by the De Boni and OGI systems. In fact, these two algorithms are based on detecting the boundaries of concatenated sequences of unrelated dialogues or TREC FQ series.

IMIX, on the other hand, tries to make a distinction between questions on the same topic that do and do not require context, even if they are follow-up questions. IMIX is trying to be as lazy as possible as regards context completion, because its particular algorithm is relatively error-prone. It is primarily based on distinguishing between different kinds of FQ in the FQ corpora, in which there are no topic shifts. The paradigm used here is basically the insufficiency condition.

2. **Identification of rewriting strategy and anaphors.** In case we are trying to rewrite the question into a self-contained question, we need to find out how it should be rewritten. Some systems that do not require rewriting to obtain proper input for the IR may still require some structural properties of the question to be passed to the IR, in which case this step must be performed partially. If we are not rewriting the question but just passing “bag of words” information directly to the IR engine, as we do in Ritel, we may skip this step entirely.

This step is explained in detail in [23]. For each FQ, we chose one out of a small set of relatively basic rewriting strategies. We found that the most success-

ful strategy by far for both unimodal and multimodal FQ has been the anaphor substitution strategy. Here, an anaphor in the sentence has to be located and identified as being substitutable. In practice, we found that only a fraction of typical FQ can be rewritten using this method, due to the lack of proper rewriting methods to cover the entire range of FQ satisfactorily.

Performance is measured by assuming that the first step was done correctly, and by counting the percentage of correct (intelligible and syntactically correct) sentences. Any simple baseline is likely to come up with very low performance, so we arguably do not need a baseline to compare with.

3. **Referent selection.** Both rewriting and IR context completion require referents from previous utterances to be identified and selected, while the answer document set strategy doesn't. Each question and answer is scanned for potential referents which are stored in the referent database. For multimodal answers, the referents include the pictures and the visual elements or text labels within the pictures. Unlike the other referents in the database, these are retained only as long as the answer remains on screen. Referent selection then amounts to the selection of suitable referents from referents previously entered into the referent database. For multimodal utterances, picture annotations and gesture recognition is necessary as well to perform this step.

Typical features used in this step are: semantic matching, general importance of keywords, antecedent salience, anaphor-antecedent matching, confidence, presence of deictic gestures. IMIX implements all of these except semantic matching. For multimodal referent selection, we also require gesture referent recognition (recognising what visual element the user is pointing at), which in turn requires pictures shown as part of answers to be annotated with the location and name of the visual elements they contain.

Performance is measured by assuming that previous steps were performed correctly, and counting the number of cases that no harmful neither insufficient antecedents were selected. We argue that a baseline is applicable here. Selecting the most important keywords (such as all named entities) from the previous question has a relatively high chance of success. In fact, the OGI system uses this baseline method with success [31]. Additional evidence is that the IMIX follow-up question corpus also shows that 75% of the anaphoric references refer to a domain-specific noun phrase in the original question. [1] also found that 53% of all FQ in their dialogues refer to a topic introduced in the previous question.

In table 3 we summarise existing systems in terms of these steps, and give performance figures where available. We have to conclude that the performances are hard to compare. Not only do the systems have different languages and domains, they also use different performance criteria and corpora. We find that the same system tested on different corpora can give quite different results. We also find that no distinction was made between harmful and insufficient anywhere.

Table 3 Context completion steps performed by different systems, and some performance figures. “Yes” means the step is performed but no figures are known; Baseline performance scores are shown between brackets. The *need-context* step is the most commonly evaluated one, and performance of different systems could be compared except that the corpora are very different. What is measured in *overall* performance evaluations varies between systems (see the footnotes for what is evaluated), and cannot meaningfully be compared.

system	need-context	rewriting	ref-select	overall
Ritel	yes	-	yes	
IMIX	75%(61%) ⁽¹⁾	yes	yes	14% ⁽²⁾
De Boni	83%(62%) ⁽³⁾ ; 96%(78%) ⁽⁴⁾	-	-	N/A
Rits-QA	-	yes	yes	37% ⁽⁶⁾
Nara	-	-	yes	100% ⁽⁷⁾
KAIST	-	yes	yes	⁽⁸⁾
NTU	-	-	yes	⁽⁸⁾
OGI	93%(62%) ⁽³⁾ ; 74%(64%) ⁽⁵⁾	-	yes	84% ⁽⁹⁾

(1) - unimodal FQ corpus, classification includes discourse question

(2) - unimodal FQ corpus, rewriting and ref-select combined

(3) - sequence of TREC-10 context dialogues

(4) - De Boni dialogue corpus

(5) - HANDQA dialogue corpus

(6) - QAC2 corpus, overall rewriting performance

(7) - overall context completion performance using restricted language dialogue

(8) - TREC-10 context task participants, no results

(9) - retrieval performance in top 50 documents, TREC-10 and TREC 2004

3 The corpora

In this section we describe the three corpora we collected and analysed. The first two are the follow-up question corpus [23], composed of 575 text FQ, and the multimodal follow-up question corpus [24], composed of 202 multimodal (text + gestures) FQ to multimodal (pictures+text) answers. Both are Dutch-language. These are based on presenting users with canned questions and answers, which they can respond to by uttering an FQ. The third corpus is collected from user dialogues with the Ritel system, the Ritel corpus [25]. This is French-language.

We used a special method for collecting the Vidiam corpora, which is low-cost and specifically tailored for FU in QA systems. Instead of having an actual dialogue with the user (using either a dialogue system or a Wizard of Oz setup), we have the user react to a set of canned question/answer pairs. The first user turn consists not of posing a free-form question, but of selecting a question out of an appropriately large set of interesting questions. The second turn of the user consists of posing a FU to the answer then presented. The dialogue simply ends when the user posed his/her FU, and there is no need for the system to reply to the FU, hence there is no dependency of the corpus on any dialogue strategies used. An obvious weakness of the method is that the dialogues are only two turns long. However, we found that such a “second dialogue utterance” corpus can be rapidly collected, and contains many or most of the most important phenomena that a first version of a dialogue

system will need to support. Our first conclusion is that this is a very good low-cost corpus collection method for bootstrapping a QA dialogue system.

3.1 *The follow-up question corpus*

The first corpus we collected is a “second utterance” corpus consisting of text-only dialogue [23]. We first created a collection of 120 hand-picked questions with selected answers. The collection was chosen so as to cover a variety of different question types and answer types, and was also used to evaluate the IMIX QA engines. Answer size ranges from a single phrase to a paragraph of text. The answers had a proportion of fully or mostly correct answers (93 questions, the answers were retrieved manually), a proportion of wrong answers (20 questions, the answers are real output from the QA system), and a proportion of “no answer found” (7 questions). The users participated in the experiment through a Web interface. First, they had to select at least 12 questions which they found particularly interesting. Then the answers were displayed. For each question-answer pair, they had to enter a FU that they thought would help further serve their information need, imagining they were in a real human-computer dialogue. We asked about 100 users to participate, which were mainly people from the computer science department and people working in a medical environment. We collected 575 FU from 40 users. The questions chosen by the users were reasonably evenly distributed. Almost all questions were chosen between 1 and 10 times by users; there was no question that was not chosen by any user.

Examining the corpus, we soon found that the FU could meaningfully be classified into a number of distinct classes. We annotated the corpus with these classes.

We found three main classes of FU (See figure 1):

- **follow-up question (56%).** We consider all domain questions that should be understood in the context of previous utterances, and which can meaningfully be interpreted literally, to be FQ. They illustrate that the user acknowledged the answer at least partially, and indicate a further user information need. Some of the FQ in our corpus contained cue words indicating their “follow-up” nature, but there were none with politeness forms or indirect forms. A significant part of these were effectively self-contained, even though they were clearly on the same topic (25% of all FQ).
- **negative feedback (28%).** This includes negative feedback questions and statements, questions indicating uncertainty about correctness of the answer, and reformulations indicating the user was not happy with the answer. We found several distinct types of these:
 1. **negative questions and statements 20%.** There seemed to be two main forms of these: repetition of the original question, with or without negative cue phrases (with no serious attempts at reformulations made); and a negative remark, usually simple but sometimes containing corrective information. In

general, it appeared that there was relatively little useful information to be obtained by further analysing the negative feedback utterances. In some cases, a negative question and statement were combined in a single utterance.

- q: wat zijn hartkloppingen? *What are heart palpitations?*
 a: De patiënt neemt dit waar als hartkloppingen. *The patient experiences this as heart palpitations*
 fuu:Maar wat zijn hartkloppingen dan? *But what are heart palpitations?*
 Repetition example

2. **verify questions 3.6%.** Questions that indicate that the user is not sure about the meaningfulness or correctness of the answer.

- q: Hoe merk je dat je hoge bloeddruk hebt? *What do you notice when you have high blood pressure?*
 a: Hoge bloeddruk (hypertensie) is meestal een aandoening die over het algemeen geen symptomen veroorzaakt.
High blood pressure (hypertension) is usually an affliction that generally does not cause any symptoms
 fuu:Dus je merkt niet dat je een hoge bloeddruk hebt?
So you don't notice anything when you have high blood pressure?
 Verify question example

3. **reformulations (4.4%).** These usually occurred when the system gave a “no answer” response. They are generally self-contained questions without any special linguistic cues.

- q: Komt RSI in Nederland vaker voor dan in de rest van Europa?
Does RSI occur more often in the Netherlands than in the rest of Europe?
 a: RSI komt niet alleen bij beeldschermwerkers voor maar ook in de industrie en bouwsector.
RSI does not only occur among screen workers but also in industry and construction.
 fuu:hoe vaak komt RSI voor in nederland vergeleken met de rest van europa
how often does RSI occur in the netherlands as compared to the rest of europe?
 Reformulation example

- **acknowledgements (13%).**

Almost all acknowledgements consisted of a one- or two-word acknowledge phrase, such as “ok” or “thanks”.

For the rest of the article, we name the above classes resp. **FQ**, **negative**, **verify-question**, **reformulation**, and **acknowledge**. Everything else was labeled as **other**; this covered only 2.8% of the utterances. About half of these could be classified as “meta” requests, such as asking for a literature source, or requests concerning search strategy or answer form.

In order to get an idea how successful a rewriting strategy would be, we split the FQ portion (56%) into classes, based on how they refer to the dialogue context. In particular, to find out if FQ could be rewritten, we attempted to rewrite each FQ manually into a self-contained question satisfying rewriting criteria (b)-(d). We also identified several special subclasses of rewritable FQ, based on the existence of machine-ready transformations that can be used to rewrite them. The transformations we identify are the following:

Fig. 1 Types of FU in corpus. FQ = followup question. self-contained = followup question that can be understood without context. negative = negative feedback

- **anaphor**: FQ with anaphors which refer to NP antecedents,
- **elliptic**: elliptic FQ (FQ without verb) which could be expanded by including constituents from a previous sentence,
- **other-pp**: FQ which could be expanded by attaching a PP at the end, consisting of a prep and an NP from a previous utterance, or which is a PP from a previous utterance.

The rewritable questions that did not fall into these categories we labeled as **referencing-other**. Our estimation is that most of these will be difficult to rewrite without rich knowledge. Some FQ did not require rewriting, even though almost all of them are clearly within the context of the previous question or answer in terms of information need. We labeled these as **self-contained**.

One more surprising class emerged: namely, a significant number of FQ turned out to be really demands of the user to show a missing part of the text that the text fragment (implicitly or explicitly) referred to. These could not be rewritten except by

actually quoting (parts of) the answer literally. We label these as **missing-referent**. An example:

q: Waar duiden koude voeten op?	<i>What do cold feet indicate?</i>
a: Hierdoor kan een verhoogde bloeddruk ontstaan of een vernauwing van de bloedvaten in het been.	<i>This can cause heightened blood pressure or a constriction of the blood vessels in the leg.</i>
fuu:waardoor?	<i>What can?</i>
- Missing referent example.

Figure 2 shows the breakup of FQ into the different classes. To evaluate the validity of the classification, we also performed an inter-annotator agreement analysis, which resulted in a Kappa [3] of 0.62 over 11 utterance classes total. This is not a very high value, but we considered it sufficient considering the large number of classes.

Fig. 2 Types of FQ in corpus. anaphor = question contains anaphor referring to NP, elliptic = question is elliptic (no verb) and can be expanded by adding constituents from a previous sentence, other-pp = can be rewritten by adding PP with NP from previous sentence, missing-referent = FQ that requested for something missing in the text fragment, referencing-other = all FQ that could be rewritten, but not with any machine-ready technique, self-contained = FQ which can be understood without context

3.2 The multimodal follow-up question corpus

In order to get some data on how users would react multimodally to a multimodal answer, we also collected a corpus of multimodal follow-up questions. It was again collected by means of a set of 20 canned questions with multimodal answers, from which the users select 10. The user is then presented with the answer to each question, to which the user can pose a free-form multimodal follow-up question. We asked the users to pose multimodal follow-up questions, rather than any kind of follow-up question, in order to arrive at an appropriately large number of multimodal utterances for analysis. We defined “multimodal” primarily as referring to pictures in the answer, with or without using pointing gestures. This has the disadvantage of artificiality, which means that the results may be biased in unknown ways, which is not a fatal flaw, since we are primarily interested in the range of phenomena that may occur, rather than their precise relative frequencies.

The users’ output modalities were typed text and pointing/gesturing with the mouse. The users were computer science students and employees, which could access the experiment through a Web page. Before the start of the experiment, the users were presented with one page of instructions, several examples, plus a short introduction to the user interface. To make the multimodal aspect of the answers significant, particularly interesting and complex medical pictures and diagrams were chosen for the 20 question/answer pairs. We collected 202 multimodal “second questions” from 20 users. Figure 3 gives an impression of the collected data we worked with in this research. Note that the text is in Dutch.

Fig. 3 Example interaction from the corpus. At the top is the original question, in the middle is the answer, at the bottom is the users follow-up question. The encirclings in black are the mouse strokes made by the user. The stippled boxes are the words’ bounding boxes. Original question: “What causes vertigo?”, follow-up question: “What do these letters mean?”

3.2.1 Utterance type and possible utterance rewriting

In multimodal QA, we found that “non-QA” questions occur more often than in unimodal QA. The most common type is asking for the identity of a visual element. That is, users say something like “What is this?”, or “Is this the?” while indicating a visual element. Other kinds of visual discourse related questions also occur, for example, “Of what side of the head is this picture?”; “Where in the picture is the medicine?”; “In what direction do these flows go?”. Following this line of thought, we classified the follow-up questions in our corpus into different types.

We found that 19% of the questions were not multimodal (mostly regular questions that did not include mouse pointing and did not refer to any visual referent), or are not follow-up questions (these are mostly remarks). We discard these in the results presented here. Everything else was classifiable into five classes (see also figure 4):

- **Self-contained (18%).** The question does not need any rewriting to make it answerable by a QA.
- **Deictic-anaphor (22%).** A regular QA question, which is rewritable using anaphor substitution to form a self-contained question. A DAM may handle this kind of question by detecting which transformation is applicable and finding the appropriate referents and inserting them.
- **Deictic-other-rewritable (13%).** A question that can be (manually) reformulated so as to form a self-contained QA question. While not rewritable like regular-rewritable, these questions can be handled by a QA in the regular manner.
- **Visual-element-identity (20%).** Not answerable without relating to the answer’s specific discourse, but answerable by just naming the identity of a particular visual element of a picture in the last answer.
- **Visual-property (26%).** Not answerable without relating to the answer’s specific discourse, and has something to do with the content of a particular picture, other than visual-element-identity. This is a difficult type of question to handle properly, but might simply be answered by producing a suitable figure caption paraphrasing the entire figure.

3.2.2 Visual referents

In our corpus, users never asked a follow-up question by just pointing. There was always some text. In fact, almost all follow-up questions can be considered primarily linguistic, and the meaning of pointing within the interaction can generally be understood as just hints (though sometimes essential ones) to disambiguate the anaphors and other references in the question text. Referents in the utterances were usually visual elements, but some of the utterances referring to visual elements did not include pointing actions. Instead, the visual elements were typically referred to by means of their colour, shape, or name. Often, a redundant combination of these were used; for example, one user asked “What function do these blue spots have?” while encircling several blue circles, thus combining colour, shape, and pointing action as hints to disambiguate the utterance’s referents. Overall, our findings indicate that traditional (anaphor) reference resolution is a meaningful and important first step in the interpretation of our multimodal utterances. In the rest of this section, we shall try to find out in what ways users refer to visual referents.

Fig. 4 Pie chart showing the percentage distribution of the follow-up question classes in the corpus.

How do users indicate using the mouse? We encouraged the use of encircling by producing several encircling examples before the users started the experiment. We consider encircling the most important type of indication, because it allows indication of both location and size of a visual element. However, as is usual in “natural” dialogue systems, we allowed users to use any other kind of pointing action, and users did commonly use several other types of pointing action.

In order to provide a more systematic analysis, we looked at three aspects of the user utterances: the pointing gestures, the anaphors, and the possible relations between these two.

The first aspect involves segmenting the series of mouse strokes of one utterance into a set of pointing gestures. We define a mouse stroke as a continuous line or curve drawn with the mouse, and a pointing gesture as a set of mouse strokes that has the goal of indicating a visual element. We found that most pointing gestures consist of only one mouse stroke, but some, like arrows, typically consist of two or three mouse strokes. We found that almost all mouse strokes were clearly identifiable as being part of pointing gestures. We found that 81% of the utterances contained at least one pointing gesture, and 23% of these (19% of all utterances) contained multiple ones.

We assigned a type to each pointing gesture. We found that 4 distinct types were sufficient to cover almost all cases: encircle, tap (that is, just a mouse click), underline (as in, underlining a word), and arrow. What was left were a small amount of mouse strokes that appear to be erroneous (labeled as error), and a small miscellany category called labeled other. Figure 5 shows the relative frequencies of the different classes.

Fig. 5 Pie chart showing the percentage distribution of pointing gesture types.

The second aspect we look at is the relation between the pointing gestures and the linguistic components of the questions. This was done by labeling the anaphors that clearly referred to visual elements, and the anaphors that clearly correspond with pointing gestures. We found that there was only a small minority of pointing gestures (9%) for which no anaphor could be pointed out, indicating that anaphors and pointing gestures are closely related. We found no gestures that correspond to multiple anaphors, but some anaphors do refer to multiple gestures. We found that these anaphors were significantly more often in plural form, explicitly indicating a set of referents. Both singular and plural forms were found in all cases, however (see figure 6).

The third aspect we annotated is the ways in which the identified anaphors provide hints towards resolving their referents, beside the pointing gestures. We classify

Fig. 6 Bar chart showing the anaphor-gesture relationship: the number of gestures that each anaphor corresponded with (either none, one, or multiple), and the plurality of the anaphor.

hints according to the aforementioned types: colour, shape, and name. We found that 49% of the anaphors provide no hints, they were just indications like “this” or “this area”. Of the remaining 51%, name occurred the most often by far. Colour and shape were relatively often used simultaneously. As one might expect, name was almost never used simultaneously with colour and shape. Our findings are summarised in figure 7.

Fig. 7 A pie chart showing the relative distribution of hints used in anaphors.

3.3 *The Ritel corpus*

The Ritel corpus is the only corpus we collected of a fully functional dialogue system. The Ritel system [8, 25] is a factoid open-domain QA dialogue systems that works over the phone. Users can call Ritel and ask questions using free-form speech utterances. The system answers with a factoid type answer, which is usually just a single word. Dialogue management functionality was added, which mainly concerned confirmation whether the recognised keywords, question type, and answer were correct, and signalling incomplete or non-understanding.

Users were invited to call Ritel over the phone. We collected 15 dialogues in this manner. These comprise a total of 244 user utterances. We annotated these with dialogue act type, presence of topic shifts, and self-containedness, mostly following [23], see table 4. 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 FQ. Particularly ill recognised were reformulations, self-contained FQ, 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 5. 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 6.

So, we found 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 4 *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

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.

Table 5 *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 6 *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)

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 The dialogue manager

In this section, we describe how the corpus results were used to design and evaluate effective dialogue strategies, based on the collection of strategies explained in section 2. We describe what we implemented in the Vidiam DAM, and how well the DAM performed. Our overall approach is to implement simple, knowledge-poor techniques and depend as little as possible on a specific QA architecture. We describe three aspects of dialogue performance: FU classification, according to our classification scheme, context completion performance, and overall answering performance.

4.1 FU classification performance

We first implemented the FU classifier by manually selecting words, phrases, and POS tags, that would be cues for specific classes according to intuitive language theory, and which would give the best performance when tested on our corpus. We then compared this with machine learning classifiers, using all unigrams, bigrams, and trigrams of words and POS tags in the sentence as input features.

The manual approach has two advantages over machine learning: prevention of non-generalisable results or “overfitting” (since the corpus is really a bit small for machine learning), and obtaining insight into the language phenomena involved. If the manual classifier has a performance similar to an optimal machine learning algorithm that uses a wide choice of features, this gives us an indication that our intuitive algorithm didn’t miss something important.

With our manual algorithm, we obtained a performance of 55% over all utterances in the corpus. We used the Weka toolkit [30] to try different machine learning algorithms. The best performance we found was in the 55%-60% range, very close to our manual algorithm. The different algorithms and feature selection filters also tended to select mostly the same features as our manually chosen ones. The binary tree classifier even came up with binary decision trees similar to our own. We managed to get a performance of 62%-63% by using support vector machines, and adding an extra feature which denoted whether the semantic classifier found a sufficiently specific medical domain term in the sentence.

This is a reasonable result for such a knowledge-poor approach. It is enough to start with, especially since the manual algorithm concentrates on avoiding false positives with respect to non-default behaviour, rather than optimising overall classification performance per se. For example, misclassifying an “anaphor” FU as “negative” would be a disaster, while misclassifying “anaphor” as “referencing-other” only does limited damage. This ensures that most misclassifications are relatively safe, rather than costly, in terms of dialogue performance. In particular, referencing-other and self-contained were chosen as “sensible defaults” in case of uncertainty. If we consider all matches of self-contained and referencing-other as safe defaults for the utterances classified as FQ, our manual algorithm arrives at an accuracy of 73%.

In the rest of the section, we will describe the most important rules we used for classification. These rules are very simple, and most of them seem transferable to other languages.

The words *niet* (*not*) and *geen* (*none*), together with the utterance not being a question, detected some 80% of negative statements. Negative questions could be detected using *maar* (*but*) at the beginning of a sentence or phrase, or the occurrence of *of niet* (*isn’t it*).

Some 75% of acknowledgements could be detected by the words *dank* (*thank*) and variants, *ok*, and *oh*, *jammer* (*a pity*), *duidelijk* (*that’s clear*), *mooi* (*nice*); and *dan* (*so*) at the beginning.

A variety of special cue words could be used to detect some subclasses of difficult-to-rewrite sentences (which should be classified as referencing-other). In our corpus we found *zo’n* (*such a*) (indicating reference based on similarity, which we can’t handle), *zoveel* (*that many*) (referring to quantities), *andere* (*other*) (referring to set operations).

We found that questions starting with the word *dus* (*so*) are almost always verify-questions.

The occurrence of certain wh-words at the end of a sentence (indicating a question written in statement form) was a good indicator of some of the cases of missing-

referent. In particular ... *wat* ? (... *what?*), or ... *waarvan* ? (... *of what?*), ... *waardoor* ? (... *by what?*).

For analysing anaphors, we looked at the presence of regular determiners, like *de* (*the*) and *deze* (*this*), and PP-type determiners, which are a specifically Dutch phenomenon, for example, *erdoor* (*by it*), *hiermee* (*with this*), *daarvan* (*of that*). This detects most of the instances of the anaphor class along with the positions of the anaphors.

4.1.1 Multimodal FU classification performance

In the VidiAM DAM, we added multimodal support by extending the utterance type classifier with the utterance types found in the multimodal dialogue corpus. In particular, the visual-element-identity and visual-property types proposed in section 2 are completely new types requiring special handling.

We found, however, that distinguishing between all unimodal and multimodal FU types is a difficult task. We fed all follow-up questions of the corpus, including the non-multimodal ones, into a machine learning tool, using the number of mouse strokes, and the number of occurrences of specific words and part of speech tags in the sentences as features. The tool had to classify the combined set of classes we identified for both unimodal and multimodal FU. Up till now, we have not been able to obtain classification performance above 40%. We did find that particularly important features are the number of gestures in the utterance (with no gestures indicating the question is non-multimodal, and multiple gestures that the question concerns visual discourse), occurrence of the determiner “this” (indicating a regular follow-up question with a deictic anaphor), and occurrence of the word “picture” (indicating visual discourse). We implemented these simple rules into the VidiAM DAM. To obtain a better performance, we will likely need high-level integrated knowledge, such as dialogue context.

We also developed a gesture recogniser specifically for multimodal FU. It can handle taps, encircles and underlines. Besides taps, encircle gestures were the most important gesture class. We found that a simple bounding box algorithm, comparing the magnitude of the gesture’s bounding box, the visual element’s bounding box, and their intersection, could correctly identify 66% of all encircling gestures. A significant part of the failures concerned visual referents which were not annotated and are likely not naturally annotatable (16% of the gestures), and gestures encompassing multiple visual referents in one gesture (6%). Of the remaining referents, our algorithm managed to identify 88% of the referents correctly. We consider this a very good result for such a simple algorithm, showing that resolving gestures’ referents is a relatively easy task. We also found that taps, underlines, and arrows need different handling. Full support requires stroke segmentation and gesture type recognition, such as that found in [29].

4.2 Rewriting and context completion performance

In this section we estimate what our corpus results mean for potential QA performance using different context completion approaches with help of the (unimodal) follow-up utterance corpus. We will discuss regular FQ here; other FU such as discourse questions will be addressed in section 4.3.2.

Let's assume that the 13.2% of "missing referent" questions may be answered by just displaying more of the document the answer came from. We can say that, of the 61.9% of FQ that are not already self-contained nor of type missing-referent, some 62.7% are potentially rewritable using relatively basic techniques (that is, they belong to the classes anaphor, elliptic, and other-pp). This is an *upper bound* for the rewriting approach, assuming that we can correctly classify and rewrite the user utterances.

The remaining ones may alternatively be resolved using the IR context approach. How many of these will be resolvable in this way cannot easily be determined, and depends on the inner workings of the IR engine and database. In contrast to the rewriting approach, we cannot say with certainty if a certain IR context operation is "correct" and effective, given a specific QA engine.

4.2.1 Rewriting performance

Attempts were made to rewrite the anaphor, elliptic, and PP attachment classes of the unimodal FU corpus. We also looked at whether antecedents were found in the initial question or the answer. We found that antecedents were often found in *both* the question and the answer. We obtained the following results:

Anaphor. Anaphor proved to be the least difficult to process, as the majority of these can be found using cue words and POS tags, while the antecedents can be found using a Lappin and Leass [14] type salience algorithm. Some simplifications could be made. In particular we found that the antecedent could be found in the question in some 75% of the cases, so we optimised our salience algorithm a bit by increasing the salience of antecedents found in the user utterance. Still, our achievements were limited. Of all FQ labeled by the system as "anaphor", only 42% were rewritten properly. This was in part due to a large number of false positives and in part due to errors in the reference resolution.

Elliptic. Elliptic proved to be more difficult. The classification performance was reasonable. We detected 86% of all FQ of class elliptic by just looking at the absence of a verb, with 44% false positives. Rewriting was done by finding the sentence that the elliptic expression referred to, and then using constituents from that sentence to form a self-contained sentence. Performance of finding these antecedent sentences and rewriting were unusably low, however. Just a small minority of the antecedent sentences can be found by matching the elliptic sentence with words from previous sentences. There were no easy shortcuts available either. We found that only 55% of the elliptic FQ referred exclusively to the original user question, and 14% referred to candidate sentences in both the user question and the system answer. Building a

correct sentence from these proved difficult as well. Syntactic transformation using the Alpino dependency tree parser [2] did not work in most cases. Our first attempts showed that the transformed sentences were unsyntactic or did not make sense. We also tried building sentences from relation-argument type semantic frames using the Rolaquad semantic tagger, but the tagger did not seem reliable enough to get useable results.

Other-PP. The other-pp class of rewritable questions are relatively difficult to recognise, as FQ which require a PP to be attached are not readily distinguishable from self-contained questions. An obvious approach is to use semantic knowledge in the form of verb-argument relations, as is used in PP attachment disambiguation [9]. Again, however, our available semantic tagger did not prove reliable enough for this. What we did find is that 62% of other-pps referred to an NP occurring in both user and system utterance, and a total of 89% referred to an NP occurring in the user utterance. This suggests that other-pp has at least potential for use in the IR context approach.

Our first conclusion is that rewriting is not easy, and we may need accurate low-noise domain-specific semantic knowledge to do it. We did not focus on such domain-specific techniques here, so our current system only fully handles the anaphor class. Alternatively, anaphor and other-pp can be used as an IR context indicator that the user utterance should be used.

4.2.2 Potential performance of the IR context approach

One of the most popular IR context approaches is to search only through the previous n documents retrieved by the previous question. De Boni [4] reported a success rate of 50% for his own corpus (which is a combination of real-life and TREC dialogues). Why does this simple approach work so well? Intuitively, it is likely that most documents are coherent with respect to their topic, and that a single document is made so as to answer a number of common questions on that topic. In fact, this is underwritten by results from research by [16]. They studied answer length in relation to user preference and behaviour. They found that, within a specific information need scenario, using longer text fragments as answers produced dialogues with less questions.

To gain more insight into this most basic IR context approach, we evaluated the simplest version of it, namely only looking at the document the answer came from (be it correct or incorrect). This way, the result depends only on the nature of these specific documents, and not on the way in which the IR matches documents. The result we obtain may be considered a lower bound for the performance of the IR context approach with respect to document selection performance, and an upper bound with respect to the expected document fragment selection performance.

We checked manually whether the answer to each FQ in the unimodal FU corpus could be found in the document where the original answer came from. We only did this for the *correct* answers, since considering incorrect answers here introduces noise related to the performance of the specific IR used. That is, you will also be

measuring the tendency of the actual IR to either select the wrong document, or the wrong sentence from the right document.

We checked all FQ, including self-contained FQ. For about 1/3 of the answers, the source document could not be retrieved because of errors or incompleteness in their source references. The documents were nearly all sections from encyclopedias, and ranged from 50-500 words in size, with an average of about 150. A total of 196 FQ were checked this way. As a way of indicating the existence of vagueness in the documents' answering potentials, each FQ was annotated with a 3-point scale, thus including a "partial match" option:

- no match: the document could not answer the question in any way.
- partial match: the document gave only a partial answer or an implicit answer.
- full match: a satisfactory answer was explicitly stated in the document.

We found that relatively few cases needed to be annotated as "partial match", so we considered only the distinction "full match" versus "no full match". We found that some 39% of the FQ could be answered by the document the answer came from. For the subclass missing-referent, this percentage was 73%. This high figure is consistent with the concept of missing-referent as directly referring to the document the answer came from. This means that FQ of this class can be effectively dealt with by directly showing more of the answer's source document.

For the remaining FQ classes, the percentage was 35% on average. Percentages for each class varied somewhat, ranging from about 20% to about 43%. The differences were not very significant, and in particular we did *not* find that self-contained FQ have a lower percentage of matches, in fact it was 43%. The lowest was elliptic, with 20% (3 out of 15).

This rather high figure may have some interesting implications. In fact, it is consistent with our stated intuition of document coherence being the real reason behind the success of this approach. Implementing a simple "use last document" strategy is likely to be worthwhile in any QA dialogue system, as long as there is a proper way to detect when the answer could not be found in the last document, and other strategies are available to complement it.

Another implication is that perhaps we should reconsider the ways in which to select a text fragment from a document. Our results suggest that including more text in the answer text fragments may lead to better satisfaction of the users' information needs. On the other hand, our users seemed to prefer relatively small text fragments (as some started making comments on the text size when these were more than 4-5 sentences long).

4.3 Answering performance

In this section we consider the potential overall performance of a QA dialogue system, based on our corpora.

4.3.1 Responding to non-FQ

In our unimodal FU corpus, we found that 44% of the utterances are not FQ. Recognising and dealing with these classes of FU will already improve the system significantly. Even just reacting with appropriate prompts will help. In some cases, showing (more of) the document where the answer came from is a meaningful reaction. More sophisticated techniques can be imagined, involving system clarification questions for example.

Now, let’s look at the distributions for the different types of answer correct/incorrect/no-answer (see figure 1). As we might expect, correct answers are replied mostly by FQ, and incorrect answers by negative feedback and verify-questions, although a significant minority were FQ. Almost half of the no-answers were spontaneously reacted to by reformulations, though the users were not prompted to do so. A quarter were acknowledgements, indicating the users accepted the absence of an answer. The “other” category was significant here. In fact, we found that almost all “other” utterances amounted to explicit requests to search again. A dialogue system could easily react to this by an appropriate reformulation prompt.

The FU class seems to be an indication of whether an answer is satisfactory and/or correct. It is interesting to find out to what extent we can discover the quality of the answer by just looking at the FU class. Significant is that no strong conclusions can be drawn when the user poses a FQ. But if we look at the other FU classes’ potential of classifying between correct and incorrect answers, we can identify the following patterns:

- acknowledge almost always means correct.
- verify-question almost always means incorrect.
- reformulation almost always means incorrect.
- negative usually means incorrect. There is a small but significant percentage of negative feedback to answers labeled as correct. A look at these answers indicated that the quality of these was less than that of most answers labeled as correct. This appears to be in part due to the fact that the answers were limited to selected text fragments from the corpus only. In these cases, a negative reaction is understandable. In fact, the user reactions we found can be used to reconsider the correctness of these answers in some cases.

These simple rules would allow us to determine answer correctness to some degree. In particular, we can predict a large part of the incorrect answers (72% to be precise, see table 7).

Table 7 Prediction of correctness of answers by looking at FU type.

<i>actual</i> ↓ <i>predicted</i> →	correct	incorrect	unknown
correct	15.3%	10.3%	74.4%
incorrect	1.2%	71.8%	27.0%

4.3.2 Overall dialogue performance of the current system

In this section, we describe a technique to estimate the overall performance impact of the dialogue handling of our dialogue system offline. We use the unimodal FU corpus, and consider whether we have access to the simplest “search within previous results” method, namely, showing more of the document that the answer came from. We distinguish the following two tasks:

1. Identifying whether a FU is a FQ or not, and whether it is an acknowledge, negative feedback, or verify-question.
2. In case the FU is a FQ, passing the rewritten question and IR context hints to the QA, or producing the document in case the FQ is a missing-referent. For this task, we consider two cases:
 - a. QAs without any IR context abilities. We assume that the DAM cannot pass IR context hints. The baseline is always passing the question as a self-contained question.
 - b. QAs with IR context abilities. We assume the DAM may pass IR context hints. The baseline is to consider all FU to be FQ, with only the context flag set.

With respect to these tasks, we consider the cases in which our DAM would provide better, equal, or worse results than the baseline, using the following criteria:

- **Better.** FQ is correctly rewritten; IR context is correctly specified; FU is correctly classified as negative, acknowledge, or verify-question; FQ is correctly classified as missing-referent.
- **Same.** Behaves same as baseline.
- **Worse.** FQ is wrongly rewritten while the original FQ would be a better query to the QA, or the wrong IR context hints are passed; wrongly identifies missing-referent; wrongly identifies negative, acknowledge, or verify-question.

Our DAM has the following general strategy:

- If FU is negative-feedback, acknowledge, or verify-question, we prompt accordingly (we might use any system clarification or answer selection strategies if present, currently there are none).
- If FU is missing-referent, we show the document that the answer came from.
- If FU is another type of FQ, we pass to the QA: the IR context (either the last question or the last answer), and the rewritten question. In particular:
 - anaphor: we pass the IR context according to the predicted antecedent, and/or the rewritten question.
 - other: we pass no specific hints, just that the question is a FQ.

For task 1 we found the following results:

Better	negative, acknowledge, verify-question identified	138 (24%)
Same	all FQ identified as FQ	405 (70%)
Worse	wrongly identified a non-FQ class	32 (5.6%)

For task 2 we considered only the 405 utterances classified as FQ. We found the following for case (a) (QA without context abilities):

Better	missing-referent correctly identified	8 (2.0%)
	question correctly rewritten	37 (9.1%)
Same	original question passed to QA	405 (70%)
	question wrongly rewritten, original not self-contained	52 (13%)
Worse	question wrongly rewritten, original was self-contained	1 (0.2%)
	incorrectly identified missing-referent	1 (0.2%)

The system does marginally better (11%) by grace of the missing-referent handling, the fact that the QA has no alternative to the anaphor rewriting, and the fact that there are hardly any false positives where it produces worse dialogue behaviour.

For case (b), the QA can handle context on its own, but we are enabled to use IR contexts shortcuts where possible. We found that, in the anaphors we detected correctly, 89% referred to a concept in the user utterance. As a strategy, we can just pass a pointer to the user utterance instead of rewriting the sentence. This only gives us a dubious improvement of 1%, however. We would get:

Better	missing-referent correctly identified	8 (2.0%)
	IR context correctly specified	40 (10%)
Same	original question passed to QA	405 (70%)
Worse	IR context wrongly specified	49 (12%)

In other words, we do worse in as many cases as we do better. Finally, let's see how much better the QA in case (b) handles FQ, if we assume the IR context strategy we discussed in section 5.3. We found there that QAs are likely to find at least 35% of FQ by a default IR context approach, providing that the original answer is correct. For the non-self-contained FQ (75% of all FQ) this may give us an improvement of 35%, which amounts to an improvement of up to 26%, depending on how many of the answers are correct.

5 Conclusions

In this chapter we examined dialogue and QA strategies in existing QA dialogue systems, and summarised our analysis of the different corpora we collected. With help of the corpora we assessed the potential performance of a number of simple, knowledge-poor techniques for handling FU.

Although existing QA dialogue systems have different approaches to FQ handling and operate in different domains, we have shown that they have certain tasks and a certain general architecture in common. In particular we identified the following tasks: identification of the need for context completion, question rewriting, and referent selection. We've tried to give an account of the general issues to consider in these tasks. An interesting future direction is to develop a new set of standardised evaluations of these tasks, for example in the form of competition tracks.

In our corpus analyses, we found that there is a broad range of ways in which users can express information need during a QA session, besides regular FQ. We

also conclude that a large part of these FU can be handled by a set of simple knowledge-poor techniques. However, there is still a large part of FU that cannot be handled this way, and mistakes made by the natural language processing system also result in a significant number of errors. The most difficult tasks are utterance type classification, and determining the exact context of an FQ. We obtained only limited results using the range of techniques we tried, and it is likely we will need more knowledge intensive techniques.

Multimodal FQ are still a new research area. We found that user pointing can be understood by interpreting pointing actions as hints for disambiguating linguistic anaphors. We have shown how pointing can be handled using annotated pictures. The implementation remains infeasible, however, as long as we rely on labour intensive picture annotation techniques.

References

1. Núria Bertomeu, Hans Uszkoreit, Anette Frank, Hans-Ulrich Krieger, and Brigitte Jörg. Contextual phenomena and thematic relations in database QA dialogues: results from a Wizard-of-Oz experiment. In *Workshop on Interactive Question Answering, HLT-NAACL 06*, pages 1–8, 2006.
2. Gosse Bouma, Jori Mur, Gertjan van Noord, Lonneke van der Plas, and Jörg Tiedemann. Question answering for dutch using dependency relations. In *Proceedings of the CLEF2005 workshop*, 2006.
3. J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37–46, 1960.
4. M. De Boni and S. Manandhar. Implementing clarification dialogues in open domain question answering. *Journal of Natural Language Engineering*, 2004.
5. Pamela Forner, Anselmo Pe nas, Eneko Agirre, naki Alegria, I Corina Forăscu, Nicolas Moreau, Petya Osenova, Prokopis Prokopidis, Paulo Rocha, Bogdan Sacaleanu, Richard Sutcliffe, and Erik Tjong Kim Sang. Overview of the clef 2008 multilingual question answering track. In *CLEF'08: Proceedings of the 9th Cross-language evaluation forum conference on Evaluating systems for multilingual and multimodal information access*, pages 262–295, Berlin, Heidelberg, 2009. Springer-Verlag.
6. Junichi Fukumoto. Answering questions of information access dialogue (iad) task using ellipsis handling of follow-up questions. In *Workshop on Interactive Question Answering, HLT-NAACL 06*, 2006.
7. Junichi Fukumoto, Tatsuhiko Niwa, Makoto Itoigawa, and Megumi Matsuda. RitsQA: List answer detection and context task with ellipses handling. In *Working notes of the Fourth NTCIR Workshop Meeting*, pages 310–314, 2004.
8. Olivier Galibert, Gabriel Illouz, and Sophie Rosset. Ritel: an open-domain, human-computer dialog system. In *Interspeech 2005*, pages 909–912, 2005.
9. Daniel Gildea and Martha Palmer. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, Annual Meeting of the ACL, Philadelphia, 2001.
10. Andrew Hickl, Patrick Wang, John Lehmann, and Sanda M. Harabagiu. FERRET: Interactive question-answering for real-world environments. In *ACL 2006*, pages 25–28, 2006.
11. Dennis Hofs, Mariët Theune, and Riëks op den Akker. Natural interaction with a virtual guide in a virtual environment: A multimodal dialogue system. *Journal on Multimodal User Interfaces*, 3 (1-2):141–153, 2010.

12. Kentaro Inui, Akiko Yamashita, and Yuji Matsumoto. Dialogue management for language-based information seeking. In *Proc. First International Workshop on Language Understanding and Agents for Real World Interaction*, pages 32–38, 2003.
13. Tsuneaki Kato, Junichi Fukumoto, and Fumito Masui. Question answering challenge for information access dialogue – overview of NTCIR4 QAC2 subtask 3. In *Working notes of the Fourth NTCIR Workshop Meeting*, 2004.
14. Shalom Lappin and Herbert J. Leass. An algorithm for pronominal anaphora resolution. *Computational Linguistics*, 20(4):535–561, 1994.
15. Chuan-Jie Lin and Hsin-Hsi Chen. Description of NTU system at TREC-10 QA track. In *TREC 10*, 2001.
16. J. Lin, D. Quan, V. Sinha, K. Bakshi, D. Huynh, B. Katz, and D. R. Karger. What makes a good answer? the role of context in question answering. In *Proceedings of the Ninth IFIP TC13 International Conference on Human-Computer Interaction (INTERACT-2003)*, 2003.
17. J.-C. Martin, S. Buisine, G. Pitel, and N. O. Bernsen. Fusion of children’s speech and 2D gestures when conversing with 3D characters. *Special issue on multimodal interfaces of the Signal Processing journal*, 86(12):3596–3624, 2006.
18. Jong-Hoon Oh, Kyung-Soon Lee, Du-Seong Chang, Chung Won Seo, and Key-Sun Choi. Trec-10 experiments at kaist: Batch filtering and question answering. In *TREC*, 2001.
19. Norbert Reithinger, Simon Bergweiler, Ralf Engel, Gerd Herzog, Norbert Pfleger, Massimo Romanelli, and Daniel Sonntag. A look under the hood: design and development of the first smartweb system demonstrator. In *ICMI ’05: Proceedings of the 7th international conference on Multimodal interfaces*, pages 159–166, New York, NY, USA, 2005. ACM Press.
20. B. W. van Schooten, Akker, R. op den, S. Rosset, O. Galibert, A. Max, , and G. Illouz. Follow-up question handling in the IMIX and Ritel systems: a comparative study. *JNLE*, 15(1):97–118, 2009.
21. Sharon Small, Ting Liu, Nobuyuki Shimizu, and Tomek Strzalkowski. HITIQA: an interactive question answering system: A preliminary report. In *Proceedings of the ACL 2003 Workshop on Multilingual Summarization and Question Answering*, 2003.
22. Mariet Theune, Emiel Krahmer, Boris van Schooten, Rieks op den Akker, Charlotte van Hooijdonk, Erwin Marsi, Wauter Bosma, Dennis Hofs, and Anton Nijholt. Questions, pictures, answers: Introducing pictures in question-answering systems. In *Tenth international symposium on social communication*, pages 450–463, Universidad de Oriente Santiago de Cuba, 2007.
23. Boris van Schooten and Rieks op den Akker. Follow-up utterances in QA dialogue. *Traitement Automatique des Langues*, 46(3):181–206, 2005.
24. Boris van Schooten and Rieks op den Akker. Multimodal follow-up questions to multimodal answers in a QA system. In *Tenth international symposium on social communication*, pages 469–474, Universidad de Oriente Santiago de Cuba, 2007.
25. Boris van Schooten, Sophie Rosset, Olivier Galibert, Aurélien Max, Rieks op den Akker, and Gabriel Illouz. Handling speech input in the Ritel QA dialogue system. In *Interspeech 2007*, 2007.
26. Ellen Voorhees. Overview of the TREC 2005 question answering track. Technical report, NIST, 2005.
27. Ellen M. Voorhees. Overview of TREC 2001. In *TREC*, 2001.
28. Danli Wang, Jie Zhang, and Guozhong Dai. A multimodal fusion framework for children’s storytelling systems. In *Edutainment*, pages 585–588, 2006.
29. D. J. M. Willems, S. Y. P. Rossignol, and L. G. Vuurpijl. Features for mode detection in natural online pen input. In *BIGS 2005: Proceedings of the 12th Biennial Conference of the International Graphonomics Society*, pages 113–117, 2005.
30. Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques, 2nd Edition*. Morgan Kaufmann, 2005.
31. Fan Yang, Junlan Feng, and Giuseppe Di Fabbrizio. A data driven approach to relevancy recognition for contextual question answering. In *Workshop on Interactive Question Answering, HLT-NAACL 06*, pages 33–40, 2006.