

The AMITIÉS System: Data-Driven Techniques for Automated Dialogue¹

Hilda Hardy^a, Alan Biermann^b, R. Bryce Inouye^b, Ashley McKenzie^b, Tomek Strzalkowski^a,
Cristian Ursu^c, Nick Webb^c and Min Wu^a

^aILS Institute
University at Albany, SUNY
1400 Washington Ave., SS262
Albany, NY 12222 USA
hhardy|tomek|minwu@cs.albany.edu

^bDepartment of Computer Science
Duke University
P.O. Box 90129, Levine Science Research Center, D101
Durham, NC 27708 USA
awblrbilarmckenz@cs.duke.edu

^cDepartment of Computer Science
University of Sheffield
Regent Court, 211 Portobello St.
Sheffield S1 4DP UK
c.ursu|n.webb@dcs.shef.ac.uk

Corresponding author: Hilda Hardy
Office: (518) 442-3135
Fax: (518) 442-2606

Abstract

We present a natural-language customer service application for a telephone banking call center, developed as part of the AMITIÉS dialogue project (Automated Multilingual Interaction with Information and Services). Our dialogue system, based on empirical data gathered from real call-center conversations, features data-driven techniques that allow for spoken language understanding despite speech recognition errors, as well as mixed system/customer initiative and spontaneous conversation. These techniques include robust named-entity extraction, slot-filling Frame Agents, vector-based task identification and dialogue act classification, a Bayesian database record selection algorithm, and a natural language generator designed with templates created from real agents' expressions. Preliminary evaluation results indicate efficient dialogues and high user satisfaction, with performance comparable to or better than that of current conversational information systems.

Keywords: human-computer dialogue, spoken dialogue systems, language understanding, language generation

¹ Portions of this work were presented at the 42nd Annual Meeting of the Association for Computational Linguistics (ACL '04), Barcelona, Spain (Hardy et al., 2004).

1 Introduction

Spoken language dialogue systems (SLDS) have been in evidence for some time. Early systems were usually confined to highly structured domains, where a restricted, regularized language set could be expected, such as train timetabling scenarios. Examples of these are SUNDIAL (Peckham 1993) and ARISE (Lamel et al., 1999). Later dialogue systems, like those in the US Defense Advanced Research Projects Agency (DARPA) Communicator program (Walker et al., 2001, 2002), allowed more complete travel planning but still relied heavily on the regular nature of information interchange in these scenarios. Although by now competent at performing this role, with reasonable task completion rates (an average of 56% across Communicator), the systems are not conversationally adept but somewhat inflexible and abrupt in approach. Coupled with these stylistic issues is the construction effort for building a new dialogue system for a given domain. For example, most SLDS comprise a finite-state automaton, often hand coded, for the representation of the dialogue management component. Although a data-driven approach requires a good deal of time-consuming annotation, we can speculate that, once core technologies are in place, moving to a new domain requires less time and effort than would be necessary for a traditional system. In fact, we found that once we had several hundred dialogues from a new domain annotated (helpdesk support), we were able to build a new task identification module, which was originally based on banking data, in short order. (Results are shown in Table 1.)

This paper outlines work performed as part of the AMITIÉS project to discover to what extent we can leverage recorded human-human dialogue data to build a data-driven SLDS. One side effect of this work is the use of human-human data to characterize elements of an information-seeking dialogue which existing human-computer interfaces fail to represent. There are arguments which claim specifically that human-human data is not useful as a basis for building human-computer systems, because of the level of dysfluencies, ellipses and anaphora (Dahlbäck and Jönsson, 1992). Others point out that the usability of natural language systems, especially spoken language dialogue systems in the telecommunications setting, could profit from techniques that allow users to engage in more natural dialogues (Karis and Dobroth, 1991).

We ultimately want to show that there are aspects of human-human interaction which should be modeled to create a system which is better able to handle a natural interaction style—to be more *conversationally plausible*. The results of our preliminary evaluation (Sections 5 and 6), particularly the measures of call duration, length of user utterances, and levels of user satisfaction, suggest that data-driven methods are preferred to traditional methods for building dialogue systems that users find more natural and pleasant to use.

We observe that for interactions with structured data—whether these data consist of flight information, spare parts, or customer account information—domain knowledge need not be built ahead of time. Rather, methods for handling the data can arise from the way the data are organized. Once we know the basic data structures, the transactions, and the protocol to be followed (e.g., establish caller’s identity before exchanging sensitive information); we need only build dialogue models for handling various conversational situations, in order to implement a dialogue system.

Finally, we recognize that in recent years, the performance of key components in SLDS has increased dramatically. In particular, the level of word error rate (WER) for automatic speech recognition (ASR) engines has fallen to a level where perhaps we can forgo strategies of dialogue which minimize the effect of poor speech recognition, and can experiment with established language processing technologies from other related fields of research, such as information retrieval (IR) and information extraction (IE).

The AMITIÉS project seeks to develop novel technologies for building empirically induced dialogue processors to support multilingual human-computer interaction, and to integrate these technologies into systems for accessing information and services (<http://www.dcs.shef.ac.uk/nlp/amities>). Sponsored jointly by the European Commission and the US DARPA, the AMITIÉS Consortium includes partners in both the EU and the US, as well as financial call centers in the UK and France. The AMITIÉS banking demonstrator is currently available to take calls in English at sites in the UK and the US (http://www.dcs.shef.ac.uk/nlp/amities/amities_demos.htm).

2 Related Work

Approaches to designing dialogue systems range from 1) finite state models, which are suitable for only the simplest of tasks, with a limited number of possible states and transitions; to 2) form-filling models, in which the user can answer system prompts with additional or out-of-order information to fill slots in a frame; to 3) more complex plan-based models, such as the classic TRAINS project and its more complex successor, TRIPS. These systems are required to reason about an explicit model of the domain, so that they can conduct collaborative problem-solving between system and user (Allen et al., 1995, 2001).

There has been a proliferation of dialogue systems in the last decade, although the similarity of application domains is no accident. Early systems, such as SUNDIAL (Peckham 1993) and PEGASUS (Zue et al., 1994), dealt with travel reservations. Such domains are highly structured, and the language used is easy to predict and model. SUNDIAL, for example, had a vocabulary of no more than 2000 words.

Later systems tried new domains, such as the JUPITER conversational interface for weather information (Zue et al., 2000), or added new modalities to the spoken-language application, like a touch-screen for both the MASK kiosk, which provides train information and reservations (Lamel et al., 2002) and the MATCH system for restaurant and subway information (Johnston et al., 2002).

The US DARPA Communicator program has been instrumental in bringing about practical implementations of spoken dialogue systems. The goal was to support rapid development of speech-enabled dialogue systems, in the domain of travel planning. To achieve this, all sites involved in the program were encouraged to use a single dialogue platform, the Galaxy Communicator Software Infrastructure (GCSI) (Seneff et al., 1998).

Systems developed under this program include CMU's script-based dialogue manager, in which the travel itinerary is a hierarchical composition of frames (Xu and Rudnicky, 2000). The AT&T mixed-initiative system uses a sequential decision process model, based on concepts of dialogue state and dialogue actions (Levin et al., 2000). MIT's Mercury flight reservation system uses a dialogue control strategy based on a set of ordered rules as a mechanism to manage complex interactions (Seneff and Polifroni, 2000). CU's dialogue

manager is event-driven, using a set of hierarchical forms with prompts associated with fields in the forms. Decisions are based not on scripts but on current context (Ward and Pellom, 1999).

3 AMITIÉS Corpus

A large corpus of recorded, transcribed telephone conversations between real agents and customers gives us a unique opportunity to analyze and incorporate features of human-human dialogues into our automated system. The data we have collected spans two different application areas: software support and customer banking. The financial domain corpus currently stands at 1,000 dialogues, comprising some 30,000 utterances and a vocabulary size of around 8,000 words. This corpus forms the basis of our initial multilingual triaging application, implemented for English, French and German (Hardy et al., 2003a); as well as our prototype automatic financial services system, presented in this paper, which completes a variety of tasks in English (Hardy et al., 2004). The much larger software support corpus (10,000 calls in English and French) is still being collected and processed and will be used to develop the next AMITIÉS prototype.

All conversations were recorded in call centers in Europe, and are anonymized following strict guidelines whereby all names, account numbers and personal information are removed from the audio data and replaced by generic substitutes in the word transcripts.

For annotation, we have used a modified DAMSL tag set (Allen and Core, 1997) to capture the functional layer of the dialogues, and a frame-based semantic scheme to record the semantic layer (Hardy et al., 2003b).

By adopting the general DAMSL markup scheme, we hope in a later stage to be able to acquire domain-independent models of dialogue, simply by changing the set of domain-specific semantic frames. Models of dialogue derived from tags which combine functional and semantic information within the same tag, for example, *give-departure-airport* in the travel domain, cannot be moved to a new domain, such as financial banking.

4 System Architecture and Components

The AMITIÉS system uses the Galaxy Communicator Software Infrastructure (Seneff et al., 1998). Galaxy is a distributed, message-based, hub-and-spoke infrastructure, optimized for spoken dialogue systems. The hub is programmable, using a scripting language to control the flow through each dialogue.

Components in the AMITIÉS system (Figure 1) include a telephony server, automatic speech recognizer,

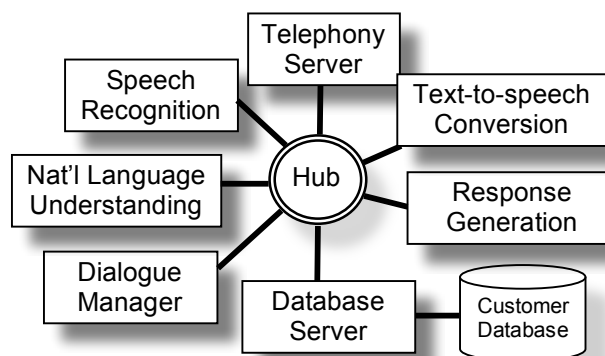


Figure 1. AMITIÉS System Architecture

natural language understanding unit, dialogue manager, database interface server, response generator, and text-to-speech conversion.

4.1 Audio Components

Audio components for the AMITIÉS system are provided by LIMSI. Because acoustic models have not yet been trained, the current demonstrator system uses the Nuance “Say Anything” statistical recognition engine, version 8.

To enhance ASR performance, we integrated static GSL (Grammar Specification Language) classes provided by Nuance for recognizing several high-frequency items: numbers, dates, money amounts, names and yes-no statements.

Training data for the recognizer were collected both from our corpus of human-human dialogues and from dialogues gathered using a text-based version of the human-computer system. Using this version we created around 100 dialogues and annotated important domain-specific information, as in this example: “Hi my name is [fname ; David] [lname ; Oconnor] and my account number is [account ; 278 one nine five].”

Next we replaced these annotated entities with grammar classes. We also utilized utterances from the AMITIÉS banking corpus (Hardy et al., 2002) in which the customer specifies his/her desired task, as well as utterances which constitute common, domain-independent speech acts such as acceptances, rejections, and indications of non-understanding. These were also used for training the task identifier and the dialogue act classifier (Sections 4.3.1 and 4.3.2). The training corpus for the recognizer consists of 1744 utterances totaling about 14,377 words. There are 652 distinct words in the AMITIÉS vocabulary (not including words from our sample customer database). This vocabulary is much smaller than that found in the corpus of human-human dialogues, largely because we have initially automated only a small subset of the tasks found in that corpus.

Using tools supplied by Nuance for building recognition packages, we created two speech recognition components: a British model in the UK and an American model at two US sites.

For the text to speech synthesizer we used Nuance’s Vocalizer 3.0, which supports multiple languages and accents. We integrated the Vocalizer and the ASR using Nuance’s speech and telephony API into a Galaxy-compliant server accessible over a telephone line.

4.2 Natural Language Understanding

The goal of the natural language understanding component (NLU) is to take the word string output of the ASR module, and identify key semantic concepts relating to the target domain. Increasingly, recognition engines can perform direct acoustic wave to semantic concept recognition with some success, although this tends to be limited to highly specific domain-based applications (Young, 2002). An alternative approach would be simple keyword spotting, although again this is restricted to straightforward slot-filler dialogues, where utterance length and context are such that ambiguities—both across utterances and within the utterance (such as multiple keywords)—are likely to be a rare phenomenon.

Some speech providers, Nuance among them, offer an ASR-NLU integrated component with the possibility of directly filling slots with information from the recognized text. This has the advantage of an

easy implementation, but at the cost of being heavily dependent on the ASR engine. Reliance on features implemented by a specific off-the-shelf product would have serious consequences for the system's flexibility. Within AMITIÉS, our goal is to create a multilingual system with the possibility of working with different recognizers. This led us to the decision of creating a multilingual, stand-alone NLU engine.

The recognition of key concepts in the utterance can be seen as a specialized kind of information extraction (IE) application, and we have adapted existing IE technology to this task. IE has made significant advances in recent years, in no small part thanks to the successful Message Understanding Conferences, also known as MUC (SAIC, 1998). Although composed of many tasks, central to IE is the idea of named entity (NE) recognition. The NEs used in MUC tasks were specific to each application, but there are core entities used across domains, including person, location, organization, date, time, money and percent.

The domain of AMITIÉS, that of financial banking, is such that we need to be able to recognize some additional NEs in order to perform the required tasks. Specifically, we are interested in account numbers, debit card numbers, person names, dates, amounts of money, addresses and telephone numbers.

For the AMITIÉS NLU component we have used a modified version of the ANNIE engine (A Nearly-New IE system; Cunningham et al., 2002; Maynard, 2003a). ANNIE is distributed as the default built-in IE component of the GATE framework (Cunningham et al., 2002). GATE is a pure Java-based architecture developed over the past eight years at the University of Sheffield Natural Language Processing group. ANNIE has been used for many language processing applications, in a number of languages both European and non-European. This versatility makes it an attractive proposition for use in a multilingual speech processing project.

ANNIE includes customizable components necessary to complete the IE task—tokenizer, gazetteer, sentence splitter, part of speech tagger and a named entity recognizer based on a powerful engine named JAPE (Java Annotation Pattern Engine; Cunningham et al., 2000).

If gazetteer information for each target language is available, the nature of ANNIE and the patterns needed to recognize entities mean that the process is to some extent language-independent, if appropriate processing resources are available. For example, some patterns used for entity recognition rely on part-of-speech tags, produced in the English version using a modified version of the Brill POS tagger (1992).

Given an utterance from the user, the NLU unit produces both a list of tokens for detecting dialogue acts, an important research goal inside this project, and a frame with the possible named entities specified by our application. By using a named entity recognizer based on a fast pattern-matching engine, we could use patterns to spot likely occurrences of entities despite the presence of recognition errors.

In order to extract the entities used in AMITIÉS, we have updated the gazetteer, which works by explicit look-up tables of potential candidates, and modified the rules of the transducer engine, which attempts to match new instances of named entities based on local grammatical context. There are some significant differences between the kind of prose text more typically associated with information extraction, and the kind of text we are expecting to encounter. Current models of IE rely heavily on certain orthographic information, such as capitalized words indicating the presence of a name, company or location, as well as

punctuation. We have access to neither of these in the output of the ASR engine, and so it was necessary to retune our processors to data which reflected that.

In addition, we created new processing resources in the modified ANNIE, such as those required to spot number units and translate them into textual representations of numerical values; for example, to take *twenty thousand one hundred and fourteen pounds*, and produce £20,114. The ability to do this is of course vital for the performance of the system.

If none of the main entities can be identified from the token string, we create a list of possible fallback entities, in the hope that partial matching would help narrow the search space. For instance, if a six-digit account number is not identified, then the incomplete number recognized in the utterance is used as a fallback entity and sent to the database server for partial matching.

The main strategy used by the NLU module in AMITIÉS was to present as much information as possible to the dialogue manager, where an interpretation could be made considering the context. For instance, the utterance *twenty three four nineteen five oh* (23 4 19 5 0) could be interpreted both as an account number (2341950) and as a date (23/4/1950). The NLU offers both formats to the dialogue manager, which can then choose based on context (e.g., account number has been supplied, but birthdate is missing).

The ANNIE system has undergone some significant evaluation on written prose. The NE recognition of ANNIE over a news corpus (looking for the standard MUC NE classes) returned figures of 89% precision and 91.8% recall, for a combined f-measure score of 90.4% (Maynard et al., 2003b). We fully expect the AMITIÉS ANNIE system to perform at a lower accuracy rate, due in part to errors in speech recognition output, and no orthographic information to leverage; however, an indicative evaluation shows promising results. Ten users (five at our UK site and five in the US) interacted with the system nine times each, creating a total of 90 interactions. Each user was asked to attempt nine scenarios, the performance of which required the system to recognize on average nine *concepts* each. *Concepts* in this context are not only semantic entities, but also task identification and positive or negative responses, such as whether the customer wishes to receive a new card. For example, the system may ask *What can I do for you?* and the user might respond, *I'm John Smith and I need to know my account balance*. In this reply three concepts are represented: first name, last name and task. Concept accuracy was computed as follows: the number of concepts accurately recognized by the system divided by the number of concepts spoken by the user. Thus, if the system misrecognizes a concept the first time and recognizes it accurately after re-prompting, the accuracy rate for those user utterances would be 50%. Over the 90 calls, the system's average recognition accuracy was 80%, which is acceptable, if only indicative at this stage. (Task identification alone also averaged 80%.) The concept recognition performance varied depending on the user, from a high of 89% to a low of 70% over all concepts. With good error recovery techniques the task success rate can be quite high, despite relatively low concept accuracy rates (see section 5, Preliminary Evaluation).

Our robust IE techniques have proved invaluable to the efficiency and spontaneity of our data-driven dialogue system. In a single utterance the user is free to supply several values for attributes, prompted or unprompted, allowing tasks to be completed with fewer dialogue turns.

4.3 Dialogue Manager

Systems that handle travel information or financial services, such as ours, can operate with a frame-filling approach, but they also require the ability to recognize different tasks, order them, and move from one task to the next. DARPA Communicator systems, for example, carry this level of complexity.

Our data-driven strategy is similar in spirit to that of the CU Communicator System (Ward and Pellom, 1999). We take a statistical approach, in which a large body of transcribed, annotated conversations forms the basis for task identification, dialogue act recognition, and form filling for task completion. A small set of rules governs navigation within and between tasks.

The AMITIÉS dialogue manager evaluates the caller's input, as filtered through the NLU component, and controls the progression of the dialogue from beginning to end. Our current prototype demonstrator handles some common tasks that are usually performed by a human agent in a financial services call center, such as verifying the customer's identity against a database of sample customers, identifying the customer's desired transaction, and executing that transaction. These transactions range from a simple balance inquiry or a report of a lost or stolen card, to the more complex debit-card payment and change of a customer's address. Several Frame Agent (FA) modules, implemented inside the dialogue manager, handle these tasks. (In this context *frame*, *task* and *transaction* are synonymous.) The structure of the dialogue manager is illustrated in Figure 2.

Completing a task means identifying that task and collecting information from the customer, such as last name and payment amount, in order to fill values for attributes in the task frame. Observing that order does not make a difference in the frame-filling process, we use a data-driven approach rather than a script to guide the progression of the dialogue. As slots are filled, the need for dialogue decreases. This way the caller is free to say the necessary details in a variety of ways: prompted or volunteered, one piece of information at a time or several during the same turn, and in any order. The task-specific FAs update and consult a table for the details needed. The dialogue manager makes a response decision based on a set of rules that take into account the dialogue history as well as the requests from the FAs.

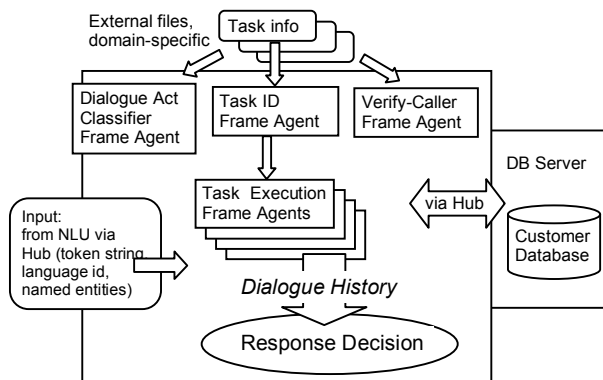


Figure 2. AMITIÉS Dialogue Manager

The Dialogue Manager maintains two histories: task history and attribute history. The task history records the topology of tasks, tasks that have been executed successfully or unsuccessfully, the task currently under

control, etc. The attribute history records the list of attributes needed by the current task, attributes requested by the system, attributes provided by user, etc.

For instance, if the customer says his name and account number before requesting a transaction, or even if he or she provides several pieces of information in one turn—transaction, name, account number, payment amount—the dialogue manager is flexible enough to move ahead after these variations. Prompts for attributes, if needed, are not restricted to one at a time, but they are usually combined in the way human agents request them. Some examples of these prompts are: 1) *Can I have your birthdate and phone number, please?* 2) *Can I have your new city and county, Miss Browning?* 3) *What's the debit card number, Ms. Rock?* ... *May I have the expiration date and issue number, Ms. Rock?* In the excerpt below, the dialogue manager works with the data the customer provides: house number and street for the old address; and house number, street, city and county for the new address.

System: Hello, this is Am-ih-tee-eh! How can I help you today?

Customer: I lost my store card

System: Can you just confirm your account number?

Customer: eight six one nine seven nine

System: Can you just confirm your address, please?

Customer: one one five three oh Bascom Avenue

System: All right, we will block that account. Can I send you a new card?

Customer: yes

System: So we'll get a new card out, with a brand new account number. That should reach you in another seven to ten days time.

Customer: ok but but I've also moved so I'd like to give you my new address

System: What's your new address, Mr. Thompson?

Customer: one five three Gregory Avenue Cheshire Cheshire or sorry scuse me Chester Cheshire

System: Can you just confirm your home phone number, please?

Customer: one five nine seven eight five nine one one eight

System: All right, I have your new information as 153 Gregory Avenue, Chester, Cheshire, (159)785-9118.

Is that correct?

Customer: that's perfect yes

...

If the system fails to recognize or gather the necessary values from the user, re-prompts are used, but no more than once for any single attribute. For the customer verification task, different attributes may be requested. If the system cannot get the required information even after re-prompts, it will gracefully give up with an explanation such as, *I'm sorry, we have not been able to obtain the information necessary to update your address in our records. Please hold while I transfer you to a customer service representative.*

4.3.1 Task Identification

The goal of the Task Identification Module, or Frame Agent, is to choose the task the customer wants the system to perform, given an utterance and a list of possible tasks. To build this module the AMITIÉS team

collected data from over 500 conversations recorded at a British call center. These were then transcribed and annotated. Annotations were made on several levels: dialogic, stylistic and semantic (Hardy et al., 2002). For task identification we focused on utterances, usually questions or action-directives, that indicate some semantic topic or task the caller wants the agent to perform. Examples of such utterances are:

Oh I see okay so what is the balance on the account just now? [Info-Balance]

Erm I'd like to cancel the account cover premium that's on my, appeared on my statement [CancelInsurance]

Erm just to report a lost card please [Lost/StolenCard]

Err, I need to pay, erm I had a erm, washing machine, and a fridge freezer and I'd like to pay for them please today by VISA [MakePayment]

Hi, could I change my address please, for my Debenhams card? [ChangeAddress]

Often these utterances are made in response to the agent's open-ended offer, *How can I help?* but sometimes they appear elsewhere in a conversation. Our annotators discovered 14 distinct tasks or transactions in the British banking corpus. Of these, four common customer-service tasks were chosen to be implemented in the current application, though the Task ID FA returns scores for all.

For our Task ID Frame Agent, we adapted a vector-based similarity method reported by Chu-Carroll and Carpenter (1999), to handle this classification problem. The FA uses a term-vector approach common in information retrieval. It is domain-independent and automatically trained. *Terms* are stemmed content words found in task-specific utterances. *Documents* are vectors of weighted term frequencies derived from the corpus (one document for each task).

The purpose of the training process is to create document vectors that will be used in task identification, where queries are compared with documents to determine the most likely task. We chose a random 80% of the corpus data to use for training, and the remaining 20% for testing. Training proceeds as follows:

1. Begin with corpus of transcribed, annotated calls.
2. Document creation: For each transaction, collect raw text of callers' queries. Remove documents that have fewer than 3 queries. Yield: one "document" for each transaction (14 of these in our telephone banking corpus).
3. Text processing: Remove stopwords, stem content words, weight terms by frequency.
 - The list of stopwords includes standard text stops such as *also, here, and please*. To these we added speech dysfluencies and other content-free words such as *ah, ahem, err, hello*. We also added to the stoplist proper names, number words, and words containing digits.
 - Stemming is done with the widely-used Porter suffix-stripping algorithm.
 - Our term weighting scheme uses average term frequency in a document as the normalization factor. In the function $1 + \log(tf) / 1 + \log(\text{average}(tf))$, $tf(t)$ is the actual frequency of term t in document D . Combining this factor with the pivoted normalization used in the SMART information retrieval system, we arrive at the weighting strategy: $[1 + \log(tf) / 1 + \log(\text{average}(tf))] / (1 - c)\text{average}(L) + c \times L$, where L is the number of unique terms in document D and c is a constant between 0 and 1 (Singhal et al., 1996). This weighting scheme is related to the BM25 formula used in the Okapi IR

system (Robertson et al., 1995), which has been reported to perform consistently better than standard cosine normalization in document retrieval applications.

Yield: one “document vector” for each task.

4. Measure the similarity between queries and documents in the training set: Create “query vectors,” and obtain a cosine similarity score for each query/document pair.

- Query terms are each given a standard weight.
- We experimented with two n -gram weighting methods in an effort to improve the accuracy of our similarity scores. Because it would be cumbersome to assign a weight to every n -gram we find, we weight individual terms instead, dividing the n -gram weight among all the terms in the n -gram (we look only at the n -grams that are found in matches between queries and documents). The weight of term t in document D is the weight of t in document D (as described above) plus the n -gram weight. The n -gram weight is given as $(n_j)/n^2$, where n_j is the length of the n -gram of which term t is an element, and n is the length of the maximum n -gram. When n is 6, for example, a term that is part of a bigram receives a weight premium of 0.056 in addition to its document term weight. Every term in a matching 6-gram gets an additional 0.167 weight.

- The first n -gram weighting method is: $wt(t) = \frac{\sum_{j=1}^t \frac{n_j}{n^2}}{tf(q,t) \times tf(D,t)}$, where $tf(q,t)$ is the frequency of term t in query q , and $tf(D,t)$ is the frequency of term t in document D .

- The second method is: $wt(t) = \frac{\sum_{j=1}^t \frac{n_j}{n^2}}{(1-c)avg(L) + c \times L}$.

Both methods tend to favor short documents, method 1 more so than method 2.

- The final query-document similarity function is as follows:

$$sim(Q_i, D_i) = \frac{\sum_{j=1}^t q_{ij} \times d_{ij}}{\sqrt{\sum_{j=1}^t (q_{ij}) \times \sum_{j=1}^t (d_{ij})}}$$

where q_{ij} is the weight of term t_j in query Q_i and d_{ij} is the weight of term t_j in document D_i .

Yield: cosine scores/classification values for each query/document pair in the training set. The classification value is 1 if the query was labeled with that particular document (task) and 0 if it was not. We obtained similarity scores for three methods: single terms only, n -gram method 1, and n -gram method 2.

5. Obtain coefficients for creating confidence scores: Use binary logistic regression. We observed that sometimes the raw cosine scores are insufficient to classify a query according to the correct task, and so we used a technique that allows us to predict the probability that a query should be classified a certain way, by fitting a logistic curve to the data. The formula to obtain confidence scores by means of this

predictive analysis is $P = \frac{\exp(b_0 + b_1 \times x)}{1 + \exp(b_0 + b_1 \times x)}$, where P is the proportion at each value of the explanatory variable x (in our case, the raw cosine score), b_0 and b_1 are numerical constants to be estimated, and \exp is the exponential function with base e . Yield: a set of coefficients for each document (task).

Next, the Task ID Frame Agent is tested on unseen utterances or queries (20% of our corpus data), which proceeds as follows:

1. Begin with one or more user queries.
2. Text processing: Remove stopwords, stem content words, weight terms. Each term in the query is assigned a constant weight. Yield: “query vectors”.
3. Compare each query with each document, using single-term scoring, n -gram method 1 or n -gram method 2. Yield: cosine similarity scores for the 3 methods.
4. Compute confidence scores (use training coefficients). Yield: confidence scores, representing the system’s confidence that the queries indicate the user’s choice of a particular transaction.

Tests performed over the corpus resulted in a classification accuracy rate of 84.7% based on confidence scores (correct task is one of the system’s top 2 choices). Results for the different weighting schemes and different corpora are displayed in Table 1.

The accuracy rate rises to 93.0% (cosine similarity with n -gram weighting, method 1) when we eliminate four tasks that have confusing or lengthy utterances, such as requests for information about payments, statements, and general questions about a customer’s account. These can be difficult even for human annotators to classify. Similar experiments performed over part of our computer helpdesk corpus, in which there were 12 defined tasks, showed the best results for cosine similarity, n -gram weighting, method 2 (86.9%).

| | Banking corpus (14 tasks) | Banking corpus (10 best tasks) | Helpdesk corpus (12 tasks) |
|--|------------------------------|-----------------------------------|-------------------------------|
| Cosine similarity, single-term weighting | 83.6% | 92.8% | 82.5% |
| Cosine similarity, n -gram weighting, method 1 | 78.1% | 93.0% | 82.0% |
| Cosine similarity, n -gram weighting, method 2 | 78.9% | 90.6% | 86.9% |
| Confidence scores (based on best cosine scores) | 84.7% | 89.6% | 85.4% |

Table 1. Classification accuracy rates with different corpora and weighting schemes (correct task is one of the system’s top 2 choices). Confidence scores were calculated based on the highest-scoring cosine similarity scores for queries and tasks in each corpus, and the classification values for those queries.

An alternative classification technique called *boosting* combines many simple and moderately inaccurate prediction rules into a single, highly accurate rule. Using the BoosTexter implementation, in which each base rule makes its predictions based on the presence or absence of a word or short phrase in the utterance (Schapire and Singer, 2000), the AT&T Help Desk system reports a semantic classification accuracy rate of

84% (De Fabrizio et al., 2002). Cortes et al. (2003) demonstrate improved results using full word lattices output by the speech recognizer, for the AT&T *How May I Help You* spoken dialog system, which contains a difficult call-classification task with 38 categories. They report a 15% reduction in classification error rate at a 30% rejection level, by implementing the Support Vector Machine (SVM) framework with lattice kernels. Methods such as these have been shown to be more robust to noisy speech data.

4.3.2 Dialogue Act Classifier

Dialogue act (DA) recognition is an important component of most spoken language systems. In 1962 Austin introduced *speech acts* (the terms *speech acts*, *dialogue acts* and *dialogue moves* are often used interchangeably) as a fundamental concept of linguistic pragmatics, analyzing, for example, what it means to ask a question or make a statement. Although major dialogue theories treat DAs as a central notion (see, for example, Grosz and Sidner, 1986; and Allen et al., 1996), the conceptual granularity of the DA labels used varies considerably among alternative analyses, depending on the application or domain.

Two key approaches use machine learning over annotated corpora to recognize dialogue acts. First, there are *n-gram* or HMM language modeling approaches, exploiting techniques from speech recognition. Reithinger and Klesen (1997), for example, apply such an approach to the VERBMOBIL corpus, which provides rather limited training data, and report a 74.7% tagging accuracy. Stolcke et al. (2000) apply a more complicated *n-gram* method to the SWITCHBOARD corpus (Jurafsky et al., 1998), using HMM models of both individual utterances and DA sequences. They report 71% accuracy, using 198,000 utterances for training and 4,000 utterances for testing.

A second approach, based on *transformation-based learning* (Samuel et al., 1998), achieves a tagging accuracy of 75.1% on VERBMOBIL, using features such as utterance length, speaker turn and the DA tags of adjacent utterances. Machine learning has been applied to other aspects of dialogue; for example, Fernández et al. (2004) apply both rule-based and memory-based learning in distinguishing subclasses of bare *sluice* (*wh*-phrases that exhibit a sentential meaning) within British National Corpus dialogues. Such a disambiguation of role may also contribute to DA recognition.

The purpose of our DA Classifier Frame Agent is to identify a caller's utterance as one or more domain-independent dialogue acts. These include Accept, Reject, Non-understanding, Opening, Closing, Backchannel, and Expression. Clearly, it is useful for a dialogue system to be able to identify accurately the various ways a person may say *yes*, *no*, or *what did you say?* As with the task identifier, we have trained the DA classifier on our corpus of transcribed, labeled human-human calls, and we have used vector-based classification techniques. Two differences from the task identifier are 1) an utterance may have multiple correct classifications, and 2) a different stoplist is necessary. Here we can filter out the usual stops, including speech dysfluencies, proper names, number words, and words with digits; but we need to **include** words such as *yeah*, *uh-huh*, *hi*, *ok*, *thanks*, *pardon* and *sorry*.

Some examples of DA classification results are shown in Figure 3. For *sure*, *ok*, the classifier returns the categories Backchannel, Expression and Accept. If the dialogue manager is looking for either Accept or Reject, it can ignore Backchannel and Expression in order to detect the correct classification. In the case of *certainly not*, the first word has a strong tendency toward Accept, though both together constitute a Reject

act. The current prototype system uses a set of rules to narrow the possibilities for interpreting the user's DA. For example, after the system issues an Offer such as *What can I do for you*, it looks for a Statement of a task, or a Non-understanding DA. After a Confirmation-request, it looks for Accept, Reject, or Non-understanding. Given enough tagged examples, it would be possible to create a probabilistic model to help the system determine the likelihood of various user DAs following various system DAs.

Our classifier performs well if the utterance is short and falls into one of the selected categories (86% accuracy on the British data); and it has the advantages of automatic training, domain independence, and the ability to capture a great variety of expressions. However, it can be inaccurate when applied to longer utterances, and it is not yet equipped to handle domain-specific assertions, questions, or queries about a transaction.

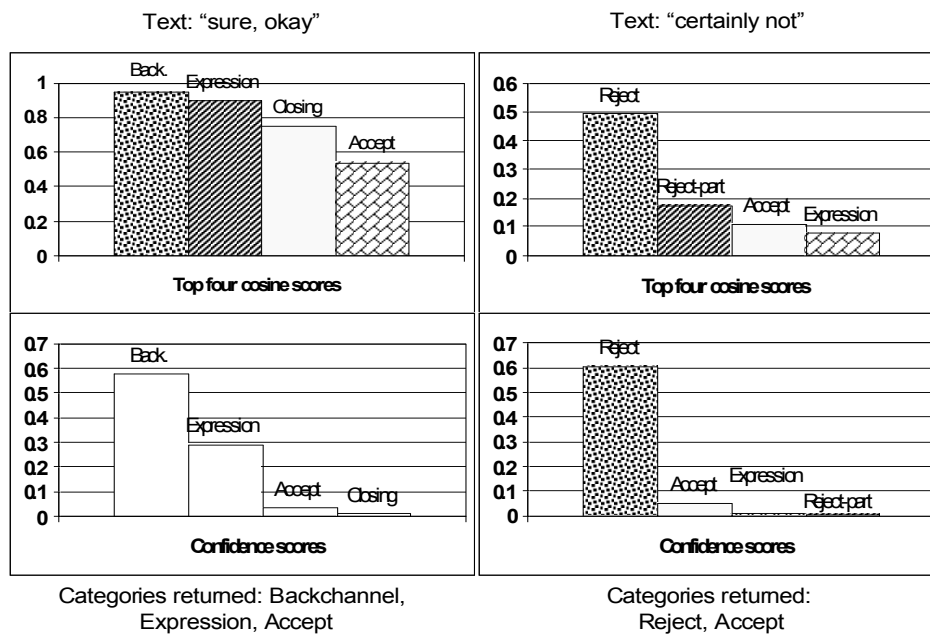


Figure 3. DA Classification examples

4.4 Database Manager

An important problem for our domain is the identification of the caller, and we use a database of card holders with their associated information, name, address, account number, and so forth, to support this computation. We implemented a version of the Bayesian record selector algorithm reported in Biermann et al. (2004) to calculate, for each person listed in the database, the probability that this person is the caller given the currently received information. The input to the algorithm is a sequence of tokens t_1, t_2, \dots, t_n that correspond to values of fields in a database record, or elements of those values. The output of the algorithm is generated after each token t_i comes in. It is the assignment to each database record the probability that that record identifies the caller.

Each record r_i in the database is assigned an initial probability $P(r_i)$ that it identifies the caller. In the current implementation, all records are assigned an equal initial probability. After the system receives the first token, the probabilities are updated according to Bayes' rule:

$$P(r_i|t_j) = P(t_j|r_i)P(r_i) / P(t_j)$$

$P(r_i)$ is the prior probability that the record r_i is the target record, according to the evidence seen so far. $P(t_j|r_i)$ is obtained from a table of probabilities. The method for generating these tables is discussed later in this section. As is common in calculations involving Bayes' rule, the denominator is calculated implicitly in the course of normalizing the probabilities at the end of the update.

The recognition process is as follows: the dialogue manager provides the database record selection module with a sequence of field-value pairs. The database record selector then processes the sequence of pairs one at a time by updating the probability that each record in the database is the target record given the new observation. At the end of the update, the database record probabilities are normalized and compared against an acceptance threshold. After processing the field-value pair, if any record has a probability greater than the threshold (0.95 in our experiments), the record is returned immediately to the dialogue manager and the database record selector terminates. The caller has been identified with high probability.

If no record reaches the acceptance threshold after processing all of the field-value pairs, we return the most probable record along with a list (the *re-prompt* list) of the fields in that record that differ greatly from the recognized values for those fields. The re-prompt list is obtained by computing the Levenshtein edit distance² between each field in the most probable record and the recognized value for that field. If the distance between the two values is greater than a predetermined threshold for that field (See Table 2 for threshold values for various attributes) the field is added to a re-prompt list. The dialogue manager can use the re-prompt list to determine what information to request from the caller again.

The re-prompt threshold values were assigned based on several factors. Having low expected ASR error rates, a low average number of characters, and high value in record identification caused fields to receive lower threshold values.

Table 2 lists the fields in the records in the database that the dialogue manager could request. Some fields, such as the first and last names, were simply spoken by the user, resulting in a single recognized token. Other fields, such as the account number, were requested character-by-character, so that a ten-digit phone number results in ten separate recognition tokens passed to the record selection module.

The system gets the value for probability $P(t_i|r_j)$ by consulting a table. For fields recognized as a whole word, the values in the table were estimated from experimental evidence. In fields recognized character-by-character, the probability $P(t_i|r_j)$ corresponds to the probability that the character t_i was recognized given that the speaker actually spoke the i th character of the value for the appropriate field in r_j . The confusion matrices for digits and characters were obtained empirically by recording the output of the Nuance speech recognizer from users speaking sequences of letters and digits.

Previous work on the database record identification problem has shown that one common cause of error in recognizing field values obtained character-by-character is the misalignment of character sequences that results when the speech recognizer fails to detect the beginning of the sequence, causing the entire sequence

² The Levenshtein edit distance is defined as the minimum number of character insertions, deletions, and substitutions needed to transform one string into another.

to be shifted in relation to the target sequence. To counter this problem, we modified the code to calculate probability updates by moving in both directions along the character sequence and using the update with the greater value.

| Field Name | Type of recognition | Re-prompt threshold value |
|-------------------|------------------------|---------------------------|
| FirstName | Whole word | 3 |
| LastName | Whole word | 3 |
| MothersMaidenName | Whole word | 3 |
| PostalCode | Character-by-character | 3 |
| PhoneNumber | Character-by-character | 2 |
| Street | Whole word | 3 |
| House | Whole word | 1 |
| City | Whole word | 3 |
| County | Whole word | 3 |
| Title | Whole word | 3 |
| LastPayment | Whole word | 2 |
| Balance | Whole word | 0 |
| ExpirationDate | Whole word | 0 |
| BirthDate | Whole word | 0 |
| AccountNumber | Character-by-character | 0 |

Table 2. Database fields for records in the accounts database. The field name describes the contents of the field. The “type of recognition” column indicates whether users are asked to say the value or to spell out the value character-by-character. The re-prompt threshold value is the number of characters (measured as a Levenshtein distance) by which the field value can differ from the recognized value before it is added to the list of candidates of fields to be re-requested of the user.

Figure 4 shows an example of the two alignment patterns. For example, the target attribute value might be *012345*, but the speech recognizer returns *345*. If the alignment proceeds from the beginning of both sequences, the resulting record probability will be quite low. However, if the sequences are aligned from their right ends, the recognized characters match perfectly with their counterparts in the record.

The method reported in our previous work on the database record selection algorithm included techniques for preselecting a subset of records out of the database to avoid having to update probabilities for every record in the database. The AMITIÉS system has an accounts database with 100 records, so the subsetting algorithm was not employed, as probabilities for every record could be computed without incurring a performance penalty.

```

Target:      012345
Recognized:  345___

Target:      012345
Recognized:  ___345

```

Figure 4. Two methods of aligning sequences of characters.

In tests reported in Biermann et al. (2004), callers were correctly identified out of a database with one million records 96.3% of the time (289 identifications in 300 trials), with no false identifications when the two alignment strategies of Figure 4 were used. This performance was achieved even though the word

misrecognition rates for this vocabulary ranged between 17% and 40%. Records included fields for first name, last name, city, postal code, phone number, and account number. The names in the records were generated using combinations of 4,375 female first names, 1,129 male first names, and 88,799 last names which comprised the top 90% in cumulative frequency of all names in the 1990 US Census (obtained at <http://www.census.gov/genealogy/names/>). The test consisted of thirty subjects placing ten telephone calls each to the system in which they verbally presented the requested identifying information for randomly chosen records in the database. There was no a priori preparation of the database, so recognition of spoken names was not attempted. Instead, all inputs were spelled via spoken input. Our results, obtained using the subset selection methodology, indicate that our system can identify individuals with high probability even in conditions of poor speech recognition, and that the entire record selection module is capable of scaling well to large databases.

There have been many approaches to caller identification in recent years. One method organizes the names to be recognized into a tree with a letter at each node. Cole et al. (1991) and Hild and Waibel (1995) used this technique and obtained mid-90's recognition rates on populations of size 50,000 and 14 million. Meyer and Hild (1997) collected two fields, a spoken name and a spelled name, and used the combined information to obtain a correct recognition rate of 97.7% on a database of 1337 names. Our system has the advantage that the database is used directly without any preparatory computation and it works well with very poor recognition rates.

4.5 Response Generation

For our initial tri-lingual triaging application (Hardy et al., 2003a), response generator servers were implemented in English, French and German within the Galaxy Communicator Software Infrastructure. Our automatic telephone-banking demonstrator expands the functionality of the English response generator in order to handle selected tasks in full. Both versions accept a request from the Dialogue Manager expressed in an abstract, language-independent format; and they create appropriate responses to send to the text-to-speech component of the system.

A multilingual dialogue system can work much more efficiently when the dialogue manager is language- and domain-independent. To that end, we have devised an abstract, extensible template-based scheme for representing system utterances. We use an initial set of five keys: 1) type, indicating a speech act such as a general prompt, a request for data or for confirmation, backchannel, acceptance, rejection, or an informative statement for the caller; 2) repetition, indicating the number of times this particular response has been made (currently 0, 1 or 2); 3) attributes, a list of items needed in the request, or an indication of the topic or the kind of response to deliver; 4) modifiers, optional descriptors to accompany attributes, such as “previous” or “new” for the attribute “city”; and 5) (optionally) person, usually the title and surname of the caller, retrieved from the database. All attributes are separated from their language-specific renderings and given meta-names such as `fname`, `lname` (first name, last name), `house_number`, `street`, `debit_card_number`, and `debit_card_expiry`. Our telephone-banking demonstrator requires only 34 response templates, 22 attributes and 3 modifiers, to be able to handle 5 tasks.

Utterances were collected from and modelled after the natural variations in human agents' expressions, and the generation module selects segments at random. These features help the system sound more realistic and human-like. For example, the system's opening prompt takes one of nine possible forms:

Prompt (greeting), part 1 of 2:

Hello, this is Am-ih-tee-eh! | Hello, this is the Am-ih-tee-eh system. | This is the Am-ih-tee-eh system.

Prompt (greeting), part 2 of 2:

How may I help you? | How can I help you today? | What can I do for you?

Similarly, if the dialogue manager were to send a request for the customer's expiration date and issue number (of a debit card), the response can take a variety of forms:

Request, part 1 of 2:

Can you just confirm | Can I have | Can I take | What is | What's | May I have

Request, part 2 of 2:

[list of attributes = "the expiration date and issue number"], [person name = "Ms. Estrada"]? | [list of attributes = "the expiration date and issue number"], please?

Responses can be piggy-backed when necessary. For example, the system may make a confirmation request for a payment amount, *So you want to pay 50 pounds on your account, Mrs. Estrada. Is that right?* The user may accept the amount and initiate a new task: *Yes, that's right. So what's my balance now?* In this case, the dialogue manager must first identify the "Accept" speech act, update the customer's record with the payment and new balance, correctly identify the "Inquire_balance" task, and retrieve the balance amount from the database manager. Next, the dialogue manager must direct the response generator to produce 3 speech acts: **inform** the user that the payment went through, **inform** the user of the new balance on the account, and **prompt** for a new task. The complete turn may take this form (one of over 400 possible combinations): *Ok, your payment has been authorized and will reflect on your next statement. Your balance is 650 pounds. Is there anything else I can help you with today?*

This template-based system, with the language-specific and domain-specific responses and attributes stored separately from the actual code yet linked by means of our extended data model, captures the lingo of a real agent, is easy to modify and can easily be extended to new domains and new languages.

5 Preliminary Evaluation

Ten native speakers of English, six female and four male, were asked to participate in a preliminary in-lab system evaluation (half in the UK and half in the US). The AMITIÉS system developers were not among these volunteers. Each made nine phone calls to the system from behind a closed door, according to scenarios designed to test various customer identities as well as single or multiple tasks (scenarios are listed in the Appendix). After each call, participants filled out a questionnaire to register their degree of satisfaction with aspects of the interaction.

Overall call success was 70%, with 98% successful completions for the VerifyId and 96% for the CheckBalance subtasks (Figure 5). "Failures" were not system crashes but simulated transfers to a human agent. There were five user terminations.

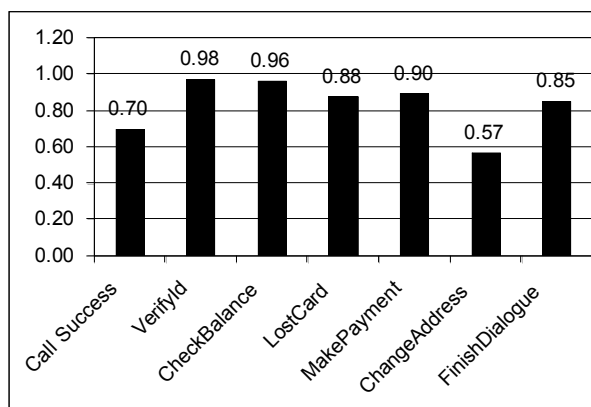


Figure 5. Task Completion Rates

Average word error rates were 17% for calls that were successfully completed, and 22% for failed calls. Word error rate by user ranged from 11% to 26%.

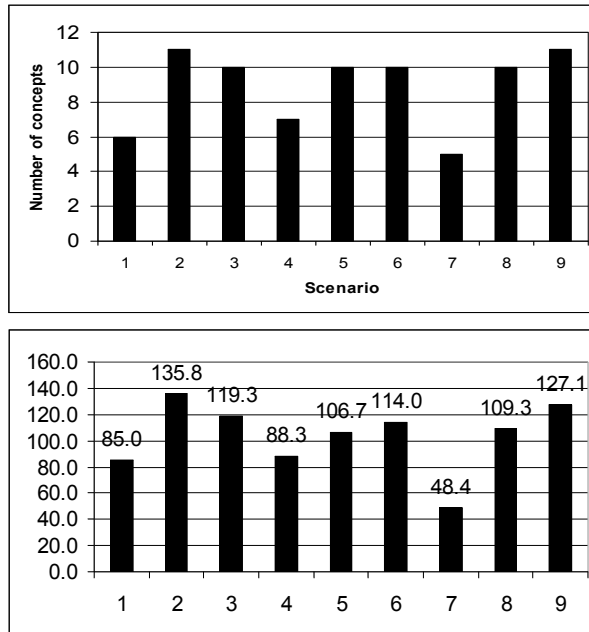
Call duration was found to reflect the complexity of each scenario, where complexity is defined as the number of “concepts” needed to complete each task. The following items are judged to be concepts: task identification; values such as first name, last name, house number, street and phone number; and positive or negative responses such as whether a new card is desired. Figures 6 and 7 illustrate the relationship between length of call and task complexity. It should be noted that customer verification, a task performed in every dialogue, requires a minimum of 3 personal details to be verified against a database record, but may require more in the case of recognition errors.

The overall average number of turns per dialogue was 18.28. Users spoke an average of 6.89 words per turn and the system spoke an average of 11.42 words.

User satisfaction for each call was assessed by way of a questionnaire containing five statements. These covered the clarity of the instructions, ease of doing the task, how well the system understands the caller, how well the system works, and the caller’s enjoyment of the system. Participants rated each on a five-point Likert scale. Summed results showed an average score of 20.45 over all users (range 5–25; higher = stronger agreement).

Although user satisfaction was high, we were more interested in identifying the major problems for the callers. Users were often frustrated by recognition failures and/or unsuccessful attempts to capture values such as a new street address, county, or phone number. Sometimes the system failed to determine that the user was finished. Because the system is designed to be non-restrictive in accepting users’ input, misidentifications were common. One user spoke for many by saying, *There were times when the system would ask me to repeat information I had not yet given to it.* We plan to revise our strategy so that we are a little more cautious in our reprompts. Occasionally, the system misidentified the user’s desired task and had difficulty backing off gracefully and starting the correct task. We are working on improving our recovery strategies for these cases.

The following transcription of an interaction with our system illustrates the efficiency made possible when the data supplied by the caller drives the conversation:



Figures 6 and 7. Scenario Complexity (top) and Average Call Duration by Scenario (in seconds)

System: Hello, this is Am-ih-tee-eh! How can I help you today?

Customer: Yes this is Jeanette Calvert my account number is seven one seven six seven three. Can you give me the balance on my account?

System: Please repeat the county.

Customer: uh it's Dorchester Dorset

System: All right, your balance is 1191 pounds and 90 pence. Anything else I can do for you today?

Customer: yes I'd like to pay twenty five pounds with my debit card

System: What is the debit card number, Mrs. Calvert?

Customer: it's five eight four one three three zero nine one eight one nine four four six two. It expires January two thousand seven and the issue number is eight.

System: What's the issue number, please?

Customer: eight

System: And it's 25 pounds you'd like to pay. Did I get that right?

Customer: yes

System: Ok, that's gone through. Your payment will be credited overnight, Mrs. Calvert, and it will appear on your next statement. Is there anything else at all I can help you with?

Customer: Can you tell me my new balance please?

System: All right, your balance is 1166 pounds and 90 pence. Is there anything else at all I can do for you?

Customer: I have a new address.

System: Can I have your new address, Mrs. Calvert?

Customer: one fifty three Ramsey Street, Sheffield Yorkshire, and the phone number is eight six eight two four eight six five nine eight

System: All right, I have your new information as 153 Ramsey Street, Sheffield, Yorkshire, (868)248-6598. Is that right?

Customer: yes that's right

System: That's been updated for you. All future correspondence will go to your new address. Anything else I can do for you?

Customer: no thanks that's all

System: Thank you, goodbye.

6 Discussion, Future Work

The preliminary evaluation reported here indicates promise for an automated dialogue system such as ours, which incorporates robust, data-driven techniques for information extraction, record matching, task identification and dialogue act classification. Task duration and number of turns per dialogue both appear to indicate greater efficiency and corresponding user satisfaction than many other similar systems. In the DARPA Communicator evaluation, for example, between 60 and 79 calls were made to each of eight participating sites (Walker, et al., 2001, 2002). A sample scenario for a domestic round-trip flight contained eight concepts (airline, departure city, state, date, etc.). The average duration for such a call was over 300 seconds; whereas our overall average was 104 seconds for a similar average number of concepts (around 8.8). In part this can be attributed to the ability of our system to prompt for, and receive, multiple variables in a single utterance.

This flexibility is made possible by the significant improvement in ASR accuracy rates. In 2001, the Communicator ASR word error rates were about 40% for airline itineraries not completed and 25% for those completed; and task completion rates were 56%. By comparison, our WER of 17% for successful calls is a significant improvement for a system which handles tasks of comparable complexity, a difference that may be partly attributed to the larger vocabulary size of many Communicator systems. Our average number of user words per turn, 6.89, is also higher than that reported for Communicator systems. This number seems to reflect lengthier responses to open prompts, responses to system requests for multiple attributes, and greater user initiative.

Exactly how much of this system improvement is attributable to improved ASR performance is an interesting question. In one evaluation metric devised for SLDS, the PARADISE framework (Walker, 2000), dialogue management strategies can be optimized using machine learning, to provide higher task completion (and so better user satisfaction). The resultant dialogue strategy minimizes the impact of the worst performing component, the speech recognizer. Short, system-initiative utterances which heavily constrain user input are favoured, as might be expected. However, if the ASR does not place such a heavy restriction on the operation of the system, it appears that we can make the interaction more natural and users more comfortable. The system can achieve the goal of increased conversational competency.

We are currently working on transferring the system to a new domain: from telephone banking to computer helpdesk support. As part of this effort we are again collecting and analyzing data from real human-human calls. Experiments on task identification performed over the new helpdesk corpus yielded

similarly positive results (Section 4.3.1). Taking into account our previous experiments on caller identification, we would also expect those methods to scale well to a larger customer database (Section 4.4). For advanced speech recognition, we hope to train our ASR on new acoustic data. We also plan to expand our dialogue act classification so that the system can recognize more types of acts, and to improve our classification reliability.

Appendix: Scenarios for System Evaluation

Each user was asked to make nine calls to the system according to the following scenarios. Sample identities included first and last name, account number, street address, post code, city and county, home telephone number and birthdate. These changed after every 3 calls. The evaluation team provided users with debit-card number, expiration date and issue number for payment scenarios; and a new street address, city and county, and telephone number for change-of-address scenarios.

1. You lost your card (you can't find it anywhere), and you would like the company to send you a new one.
2. First you need to know the balance on your account; then you want to make a payment of 35 pounds with your VISA debit card.
3. You have just moved to a new house; you want the company to update your address in their records.
4. Your card was stolen last night; you want it replaced. Next, you would like to know how much money you owe on your account (your current balance).
5. You want to pay 60 pounds on your account, with your debit card.
6. You want to inform the company of your change of address.
7. You need to know the balance on your account.
8. You want to make a payment on your account over the telephone. You have a debit card with the numbers given, and you'll pay 25 pounds.
9. You have lost your store card and you want a new one. Second, you have moved to a different house and you need to let the company know what your new address is.

Acknowledgements

This paper is based on work supported in part by the European Commission under the 5th Framework IST/HLT Programme, and by the US Defense Advanced Research Projects Agency. Audio components for speech recognition and text-to-speech synthesis were provided by Nuance Communications. The authors wish to thank the three anonymous reviewers for their helpful comments on an earlier draft of this paper.

References

- Allen, J., Byron, D., Dzikovska, M., Ferguson, G., Galescu, L., and Stent, A., 2001. Towards conversational human-computer interaction. *AI Magazine*.
- Allen, J., and Core, M., 1997. Draft of DAMSL: dialog act markup in several layers. <http://www.cs.rochester.edu/research/cisd/resources/damsl/>.

- Allen, J., Miller, B., Ringger, E., and Sikorski, T., 1996. A robust system for natural spoken dialogue. Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL '96), Santa Cruz, California.
- Allen, J., Schubert, L.K., Ferguson, G., Heeman, P., Hwang, Ch.L., Kato, T., Light, M., Martin, N.G., Miller, B.W., Poesio, M., and Traum, D.R., 1995. The TRAINS project: a case study in building a conversational planning agent. *Journal of Experimental and Theoretical AI*, 7 (1995), 7–48.
- AMITIÉS, <http://www.dcs.shef.ac.uk/nlp/amities>.
- Austin, J.L., 1962. *How to do Things with Words*. Clarendon Press, Oxford.
- Biermann, A., Inouye, R.B., McKenzie, A., and Wu, S., 2004. Caller identification from spelled-out personal data using a database for error correction. Submitted for publication.
- Brill, E., 1992. A simple rule-based part-of-speech tagger. Proceedings of the 3rd Conference on Applied Natural Language Processing.
- Chu-Carroll, J., and Carpenter, B., 1999. Vector-based natural language call routing. *Computational Linguistics*, 25 (3): 361–388.
- Cole, R., Fanty, M., Gopalakrishnan, M., and Janssen, R.D.T., 1991. Speaker-independent name retrieval from spellings using a database of 50,000 names. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Toronto, Ontario, Canada.
- Cortes, C., Haffner, P., and Mohri, M., 2003. Lattice kernels for spoken-dialogue classification. Proceedings of ICASSP '03, Hong Kong.
- Cunningham, H., Maynard, D., Bontcheva, K., and Tablan, V., 2002. GATE: a framework and graphical development environment for robust NLP tools and applications. Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL '02), Philadelphia, Pennsylvania.
- Cunningham, H., Maynard, D., and Tablan, V., 2000. JAPE: a Java annotation patterns engine (second ed.). Technical report CS--00--10, University of Sheffield, Department of Computer Science.
- Dahlbäck, N., and Jönsson, A., 1992. An empirically based computationally tractable dialogue model. Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society.
- DARPA, <http://www.darpa.mil/iao/Communicator.htm> .
- De Fabbrizio, G., Dutton, D., Gupta, N., Hollister, B., Rahim, M., Riccardi, G., Schapire, R., and Schroeter, J., 2002. AT&T Help Desk. Proceedings of the 7th International Conference on Spoken Language Processing, Denver, Colorado.
- Fernández, R., Ginzburg, J., and Lappin, S., 2004. Clarifying ellipsis in dialogue: a machine learning approach. Proceedings of the 20th International Conference on Computational Linguistics, Geneva, Switzerland.
- Hardy, H., Biermann, A., Inouye, R.B., McKenzie, A., Strzalkowski, T., Ursu, C., Webb, N., and Wu, M., 2004. Data-driven strategies for an automated dialogue system. Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL '04), Barcelona, Spain.

- Hardy, H., Baker, K., Bonneau-Maynard, H., Devillers, L., Rosset, S., and Strzalkowski, T., 2003b. Semantic and dialogic annotation for automated multilingual customer service. Eurospeech 2003, Geneva, Switzerland.
- Hardy, H., Baker, K., Devillers, L., Lamel, L., Rosset, S., Strzalkowski, T., Ursu, C., and Webb, N., 2002. Multi-layer dialogue annotation for automated multilingual customer service. Proceedings of the ISLE Workshop on Dialogue Tagging for Multi-Modal Human Computer Interaction, Edinburgh, Scotland.
- Hardy, H., Strzalkowski, T., and Wu, M., 2003a. Dialogue management for an automated multilingual call center. Research Directions in Dialogue Processing, Proceedings of the HLT-NAACL 2003 Workshop, Edmonton, Alberta, Canada.
- Hild, H., and Waibel, A., 1995. Integrating spelling into spoken dialogue recognition. In Proceedings of Eurospeech, Vol. 2, pp. 1977-1980.
- Johnston, M., Bangalore, S., Vasireddy, G., Stent, A., Ehlen, P., Walker, M., Whittaker, S., and Maloor, P., 2002. MATCH: an architecture for multimodal dialogue systems. Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL '02), Philadelphia, Pennsylvania.
- Jurafsky, D., Bates, R., Coccaro, N., Martin, R., Meteer, M., Ries, K., Shriberg, E., Stolcke, A., Taylor, P., and Van Ess-Dykema, C., 1998. Switchboard discourse language modeling project final report. Research Note 30, Center for Language and Speech Processing, Johns Hopkins University, Baltimore.
- Karis, D., and Dobroth, K.M., 1991. Automating services with speech recognition over the public switched telephone network: Human factors considerations. IEEE Journal of Selected Areas in Communications, 9(4), 574–585.
- Lamel, L., Bennacef, S., Gauvain, J.L., Dartigues, H., and Temem, J.N., 2002. User evaluation of the MASK kiosk. Speech Communication, 38 (1-2), 131–139.
- Lamel, L., Rosset, S., Gauvain, J.L., and Bennacef, S., 1999. The LIMSI ARISE system for train travel information. Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 501–504.
- Levin, E., Narayanan, S., Pieraccini, R., Biatov, K., Bocchieri, E., Di Fabrizio, G., Eckert, W., Lee, S., Pokrovsky, A., Rahim, M., Ruscitti, P., and Walker, M., 2000. The AT&T-DARPA Communicator mixed-initiative spoken dialog system. ICSLP 2000.
- Maynard, D., 2003a. Multi-source and multilingual information extraction. Expert Update.
- Maynard, D., Bontcheva, K., and Cunningham, H., 2003b. Towards a semantic extraction of named entities. Recent Advances in Natural Language Processing, Bulgaria.
- Meyer, M., and Hild, H., 1997. Recognition of spoken and spelled proper names. In Proceedings of Eurospeech, pp. 1579-1582.
- Peckham, J., 1993. A new generation of spoken dialogue systems: results and lessons from the SUNDIAL project. Proceedings of the Third European Conference on Speech Communication and Technology, 33-40.

- Reithinger, N., and Klesen, M., 1997. Dialogue act classification using language models. Proceedings of Eurospeech 1997, pp. 2235–2238, Linköping University Natural Language Processing Laboratory.
- Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M.M., and Gatford, M., 1995. Okapi at TREC-3, In: Harman, D. (Ed.), The Third Text Retrieval Conference (TREC-3). National Institute of Standards and Technology Special Publication 500–225, 219–230.
- SAIC, 1998. Proceedings of the Seventh Message Understanding Conference (MUC-7). http://www.itl.nist.gov/iaui/894.02/related_projects/muc/.
- Samuel, K., Carberry, S., and Vijay-Shanker, K., 1998. Dialogue act tagging with transformation-based learning. Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Montreal.
- Schapire, R., and Singer, Y., 2000. BoosTexter: A boosting-based system for text categorization. *Machine Learning* 39(2-3), 135-168.
- Searle, J., 1969. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press, Cambridge.
- Seneff, S., Hurley, E., Lau, R., Pao, C., Schmid, P., and Zue, V., 1998. Galaxy-II: a reference architecture for conversational system development. ICSLP 1998, Sydney, Australia.
- Seneff, S., and Polifroni, J., 2000. Dialogue management in the Mercury flight reservation system. Satellite Dialogue Workshop, ANLP-NAACL, Seattle, Washington.
- Singhal, A., Buckley, C., and Mitra, M., 1996. Pivoted document length normalization. SIGIR 1996, 21–29.
- Stolcke, A., Ries, K., Coccaro, N., Shriberg, E., Bates, R., Jurafsky, D., Taylor, P., Martin, R., Van Ess-Dykema, C., and Meteer, M., 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics* 26(3), 339–373.
- Walker, M., 2000. An application of reinforcement learning to dialogue strategy selection in a spoken dialogue system for email. *Journal of Artificial Intelligence Research* 12, 387–416.
- Walker, M., Aberdeen, J., Boland, J., Bratt, E., Garofolo, J., Hirschman, L., Le, A., Lee, S., Narayanan, S., Papineni, K., Pellom, B., Polifroni, J., Potamianos, A., Prabhu, P., Rudnicky, A., Sanders, G., Seneff, S., Stallard, D., and Whittaker, S., 2001. DARPA Communicator dialog travel planning systems: the June 2000 data collection. Eurospeech 2001.
- Walker, M., Rudnicky, A., Aberdeen, J., Bratt, E., Garofolo, J., Hastie, H., Le, A., Pellom, B., Potamianos, A., Passonneau, R., Prasad, R., Roukos, S., Sanders, G., Seneff, S., and Stallard, D., 2002. DARPA Communicator evaluation: progress from 2000 to 2001. ICSLP 2002.
- Ward, W., and Pellom, B., 1999. The CU Communicator system. IEEE ASRU, pp. 341–344.
- Young, S., 2002. Talking to machines (statistically speaking). Int'l Conference Spoken Language Processing, Denver, Colorado.
- Xu, W., and Rudnicky, A., 2000. Task-based dialog management using an agenda. ANLP/NAACL Workshop on Conversational Systems, pp. 42–47.

Zue, V., Glass, J., Goodine, D., Leung, H., Phillips, M., Polifroni, J., and Seneff, S., 1994. PEGASUS: a spoken dialogue interface for online air travel planning. *Speech Communication*, 15, 331–340.

Zue, V., Seneff, S., Glass, J., Polifroni, J., Pao, C., Hazen, T., and Hetherington, L., 2000. JUPITER: a telephone-based conversational interface for weather information. *IEEE Transactions on Speech and Audio Processing*, 8 (1).