

Data-Driven Strategies for an Automated Dialogue System

Hilda HARDY, Tomek STRZALKOWSKI, Min WU
ILS Institute
University at Albany, SUNY
1400 Washington Ave., SS262
Albany, NY 12222 USA
hhardy|tomek|minwu@
cs.albany.edu

Cristian URSU, Nick WEBB
Department of Computer Science
University of Sheffield
Regent Court, 211 Portobello St.
Sheffield S1 4DP UK
c.ursu@sheffield.ac.uk,
n.webb@dcs.shef.ac.uk

Alan BIERMANN, R. Bryce INOUE, Ashley MCKENZIE
Department of Computer Science
Duke University
P.O. Box 90129, Levine Science
Research Center, D101
Durham, NC 27708 USA
awb|rbi|armckenz@cs.duke.edu

Abstract

We present a prototype natural-language problem-solving application for a financial services call center, developed as part of the Amitiés multilingual human-computer dialogue project. Our automated dialogue system, based on empirical evidence from real call-center conversations, features a data-driven approach that allows for mixed system/customer initiative and spontaneous conversation. Preliminary evaluation results indicate efficient dialogues and high user satisfaction, with performance comparable to or better than that of current conversational travel information systems.

1 Introduction

Recently there has been a great deal of interest in improving natural-language human-computer conversation. Automatic speech recognition continues to improve, and dialogue management techniques have progressed beyond menu-driven prompts and restricted customer responses. Yet few researchers have made use of a large body of human-human telephone calls, on which to form the basis of a data-driven automated system.

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 Defense Advanced Research Projects Agency, the Amitiés Consortium includes partners in both the EU and the US, as well as financial call centers in the UK and France.

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. (Generic

names and numbers were substituted for all personal details in the transcriptions.) This corpus spans two different application areas: software support and (a much smaller size) customer banking. The banking corpus of several hundred calls has been collected first and it 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. 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.

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. For our corpus, 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). The “frames” or transactions in our domain are common customer-service tasks: *VerifyId*, *ChangeAddress*, *InquireBalance*, *Lost/StolenCard* and *Make Payment*. (In this context “task” and “transaction” are synonymous.) Each frame is associated with attributes or slots that must be filled with values in no particular order during the course of the dialogue; for example, account number, name, payment amount, etc.

2 Related Work

Relevant human-computer dialogue research efforts include the TRAINS project and the DARPA Communicator program.

The classic TRAINS natural-language dialogue project (Allen et al., 1995) is a plan-based system which requires a detailed model of the domain and therefore cannot be used for a wide-ranging application such as financial services.

The US DARPA Communicator program has been instrumental in bringing about practical implementations of spoken dialogue systems. 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 dialog state and dialog 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).

Our data-driven strategy is similar in spirit to that of CU. 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.

3 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.

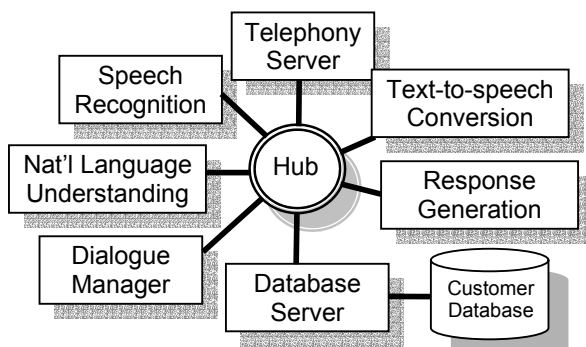


Figure 1. Amitiés System Architecture

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

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

3.1 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 a Nuance ASR engine and TTS Vocalizer.

To enhance ASR performance, we integrated static GSL (Grammar Specification Language) grammar 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 collected 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 (Section 3.3.2). The training corpus for the recognizer consists of 1744 utterances totaling around 10,000 words.

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.

3.2 Natural Language Understanding

The goal of the language understanding component is to take the word string output of the ASR module, and identify key semantic concepts relating to the target domain. This is a specialized kind of information extraction application, and as such, we have adapted existing IE technology to this task.

We have used a modified version of the ANNIE engine (A Nearly-New IE system; Cunningham et al., 2002; Maynard, 2003). 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 in 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).

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. We are interested particularly in account numbers, credit card numbers, person names, dates, amounts of money, locations, addresses and telephone numbers.

In order to recognize these, 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 punctuation as well as certain orthographic information, such as capitalized words indicating the presence of a name, company or location. We have access to neither of these in the output of the ASR engine, and so had to retune our processors to data which reflected that.

In addition, we created new processing resources, 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.

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.

3.3 Dialogue Manager

The dialogue manager identifies the goals of the conversation and performs interactions to achieve those goals. Several “Frame Agents”, implemented within the dialogue manager, handle tasks such as verifying the customer’s identity, identifying the customer’s desired transaction, and executing those transactions. These range from a simple balance inquiry to the more complex change of address and debit-card payment. The structure of the dialogue manager is illustrated in Figure 2.

Rather than depending on a script for the progression of the dialogue, the dialogue manager takes a data-driven approach, allowing the caller to take the initiative. Completing a task depends on identifying that task and filling values in frames, but this may be done in a variety of ways: one at a time, or several at once, and in any order.

For example, if the customer identifies himself or herself before stating the transaction, or even if he or she provides several pieces of information in one utterance—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; for example, city and county, expiration date and issue number, birthdate and telephone number.

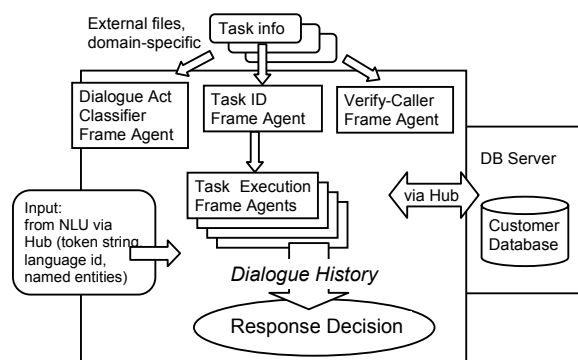


Figure 2. Amitiés Dialogue Manager

If the system fails to obtain the necessary values from the user, reprompts are used, but no more than once for any single attribute. For the customer verification task, different attributes may be

requested. If the system fails 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.”

3.3.1 Task ID Frame Agent

For task identification, the Amitiés team has made use of the data collected in over 500 conversations from a British call center, recorded, transcribed, and annotated. Adapting a vector-based approach reported by Chu-Carroll and Carpenter (1999), the Task ID Frame Agent is domain-independent and automatically trained. Tasks are represented as vectors of terms, built from the utterances requesting them. Some examples of labeled utterances are: “Erm I’d like to cancel the account cover premium that’s on my, appeared on my statement” [CancelInsurance] and “Erm just to report a lost card please” [Lost/StolenCard].

The training process proceeds as follows:

1. Begin with corpus of transcribed, annotated calls.
2. Document creation: For each transaction, collect raw text of callers’ queries. Yield: one “document” for each transaction (about 14 of these in our corpus).
3. Text processing: Remove stopwords, stem content words, weight terms by frequency. Yield: one “document vector” for each task.
4. Compare queries and documents: Create “query vectors.” Obtain a cosine similarity score for each query/document pair. Yield: cosine scores/routing values for each query/document pair.
5. Obtain coefficients for scoring: Use binary logistic regression. Yield: a set of coefficients for each task.

Next, the Task ID Frame Agent is tested on unseen utterances or queries:

1. Begin with one or more user queries.
2. Text processing: Remove stopwords, stem content words, weight terms (constant weights). Yield: “query vectors”.
3. Compare each query with each document. Yield: cosine similarity scores.
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 entire corpus, 80% of which was used for training and 20% for testing,

resulted in a classification accuracy rate of 85% (correct task is one of the system’s top 2 choices). The accuracy rate rises to 93% when we eliminate 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.

3.3.2 Dialogue Act Classifier

The purpose of the 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, okay*, 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.

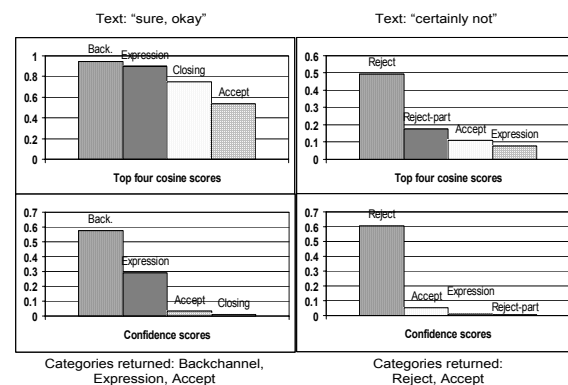


Figure 3. DA Classification examples

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.

3.4 Database Manager

Our system identifies users by matching information provided by the caller against a database of user information. It assumes that the speech recognizer will make errors when the caller attempts to identify himself. Therefore perfect matches with the database entries will be rare. Consequently, for each record in the database, we attach a measure of the probability that the record is the target record. Initially, these measures are estimates of the probability that this individual will call. When additional identifying information arrives, the system updates these probabilities using Bayes' rule.

Thus, the system might begin with a uniform probability estimate across all database records. If the user identifies herself with a name recognized by the machine as "Smith", the system will appropriately increment the probabilities of all entries with the name "Smith" and all entries that are known to be confused with "Smith" in proportion to their observed rate of substitution. Of course, all records not observed to be so confusable would similarly have their probabilities decreased by Bayes' rule. When enough information has come in to raise the probability for some record above a threshold (in our system 0.99 probability), the system assumes that the caller has been correctly identified. The designer may choose to include a verification dialog, but our decision was to minimize such interactions to shorten the calls.

Our error-correcting database system receives tokens with an identification of what field each token should represent. The system processes the tokens serially. Each represents an observation made by the speech recognizer. To process a token, the system examines each record in the database and updates the probability that the record is the target record using Bayes' rule:

$$P(rec|obs) = \frac{P(obs|rec) \times P(rec)}{P(obs)}$$

where rec is the event where the record under consideration is the target record.

As is common in Bayes' rule calculations, the denominator $P(obs)$ is treated as a scaling factor, and is not calculated explicitly. All probabilities are renormalized at the end of the update of all of the records. $P(rec)$ is the previous estimate of the

probability that the record is the target record. $P(obs|rec)$ is the probability that the recognizer returned the observation that it did given that the target record is the current record under examination. For some of the fields, such as the account number and telephone number, the user responses consist of digits. We collected data on the probability that the speech recognition system we are using mistook one digit for another and calculated the values for $P(obs|rec)$ from the data. For fields involving place names and personal names, the probabilities were estimated.

Once a record has been selected (by virtue of its probability being greater than the threshold) the system compares the individual fields of the record with values obtained by the speech recognizer. If the values differ greatly, as measured by their Levenshtein distance, the system returns the field name to the dialogue manager as a candidate for additional verification. If no record meets the threshold probability criterion, the system returns the most probable record to the dialogue manager, along with the fields which have the greatest Levenshtein distance between the recognized and actual values, as candidates for reprompting.

Our database contains 100 entries for the system tests described in this paper. We describe the system in a more demanding environment with one million records in Inouye et al. (2004). In that project, we required all information to be entered by spelling the items out so that the vocabulary was limited to the alphabet plus the ten digits. In the current project, with fewer names to deal with, we allowed the complete vocabulary of the domain: names, streets, counties, and so forth.

3.5 Response Generator

Our current English-only system preserves the language-independent features of our original trilingual generator, storing all language- and domain-specific information in separate text files. It is a template-based system, easily modified and extended. The generator constructs utterances according to the dialogue manager's specification of one or more speech acts (prompt, request, confirm, respond, inform, backchannel, accept, reject), repetition numbers, and optional lists of attributes, values, and/or the person's name. As far as possible, we modeled utterances after the human-human dialogues.

For a more natural-sounding system, we collected variations of the utterances, which the generator selects at random. Requests, for example, may take one of twelve possible 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], [person name]? | [list of attributes], please?

Offers to close or continue the dialogue are similarly varied:

Closing offer, part 1 of 2:

Is there anything else | Anything else | Is there anything else at all

Closing offer, part 2 of 2:

I can do for you today? | I can help you with today? | I can do for you? | I can help you with? | you need today? | you need?

4 Preliminary Evaluation

Ten native speakers of English, 6 female and 4 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 9 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. 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 4). “Failures” were not system crashes but simulated transfers to a human agent. There were 5 user terminations.

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%.

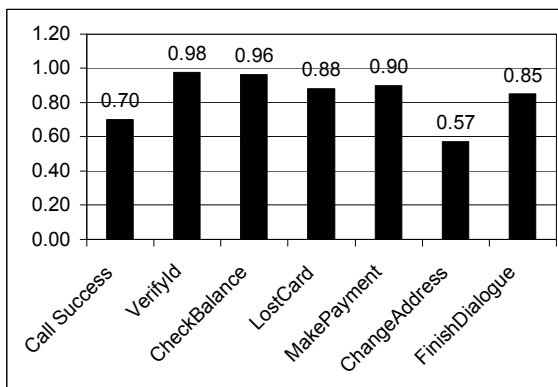


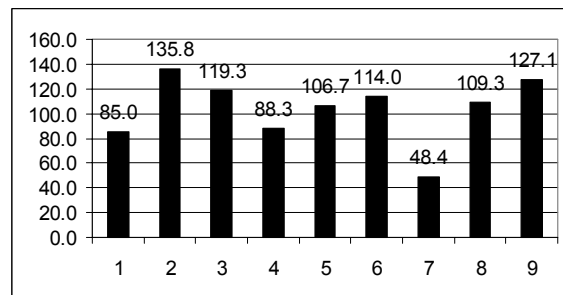
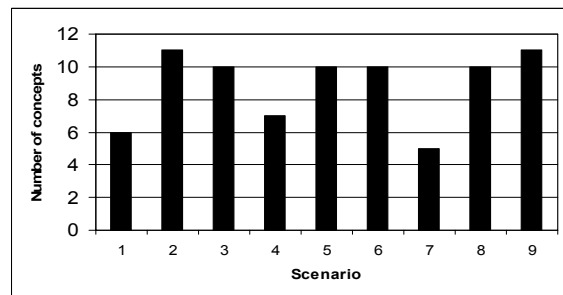
Figure 4. Task Completion Rates

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 5 and 6 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. The user spoke an average of 6.89 words per turn and the system 11.42.

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).



Figures 5 and 6. Scenario Complexity (top) and Average Call Duration by Scenario (in seconds)

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:

AMITIÉS: Hello, this is Am-ih-tee-eh! What can I do for you?

USER: yes this is Stella Lang. My card number is three one six seven one four. Can you tell me the balance on my account please?

AMITIÉS: Can you just confirm the city and county?

USER: uh yes it's London, Middlesex

AMITIÉS: I'm sorry, I didn't understand what you said. Can you repeat, please?

USER: London, Middlesex

AMITIÉS: All right, your balance is 1753 pounds and 20 pence. Anything else I can do for you?

USER: yes I'd like to make a payment of fifty three pounds with my debit card please

AMITIÉS: Can I have the debit card number, Miss Lang?

USER: sure, it's two eight seven four six six four nine five eight nine five three eight nine one, and it expires October two thousand six

AMITIÉS: What's the issue number, please?

USER: one

AMITIÉS: Ok, Miss Lang, I'll send in the payment amount of 53 pounds. Did I get that right?

USER: yes, what's my new balance please?

AMITIÉS: Ok, that's gone through. Your payment will be credited overnight and will reflect on your next statement. Your balance is 1700 pounds and 20 pence. Anything else I can do for you today?

USER: no, that's it thanks

AMITIÉS: Thanks very much for your call, goodbye.

USER: bye

5 Discussion, Future Work

The preliminary evaluation reported here indicates promise for an automated dialogue system such as ours, which incorporates robust techniques for information extraction, record matching, task identification, dialogue act classification, and an overall data-driven strategy. 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 8 participating sites (Walker, et al., 2001, 2002). A sample scenario for a domestic round-trip flight contained 8 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. ASR accuracy rates in 2001 were about 60% and 75%, for airline itineraries not completed and completed; and task completion rates were 56%. 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.

We plan to port the system to a new domain: from telephone banking to information-technology support. As part of this effort we are again collecting data from real human-human calls. 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.

6 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.

References

- J. Allen and M. Core. 1997. Draft of DAMSL: Dialog Act Markup in Several Layers. <http://www.cs.rochester.edu/research/cisd/resources/damsl/>.
- J. Allen, L. K. Schubert, G. Ferguson, P. Heeman, Ch. L. Hwang, T. Kato, M. Light, N. G. Martin, B. W. Miller, M. Poesio, and D. R. Traum. 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>.
- J. Chu-Carroll and B. Carpenter. 1999. Vector-Based Natural Language Call Routing. *Computational Linguistics*, 25 (3): 361–388.
- H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. *Proceedings of the 40th Anniversary Meeting of the Association for*

- Computational Linguistics (ACL'02)*, Philadelphia, Pennsylvania.
- H. Cunningham and D. Maynard and V. Tablan. 2000. JAPE: a Java Annotation Patterns Engine (Second Edition). *Technical report CS--00--10*, University of Sheffield, Department of Computer Science.
- DARPA,
<http://www.darpa.mil/iao/Communicator.htm>.
- H. Hardy, K. Baker, L. Devillers, L. Lamel, S. Rosset, T. Strzalkowski, C. Ursu and N. Webb. 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.
- H. Hardy, T. Strzalkowski and M. Wu. 2003a. Dialogue Management for an Automated Multilingual Call Center. *Research Directions in Dialogue Processing, Proceedings of the HLT-NAACL 2003 Workshop*, Edmonton, Alberta, Canada.
- H. Hardy, K. Baker, H. Bonneau-Maynard, L. Devillers, S. Rosset and T. Strzalkowski. 2003b. Semantic and Dialogic Annotation for Automated Multilingual Customer Service. *Eurospeech 2003*, Geneva, Switzerland.
- R. B. Inouye, A. Biermann and A. Mckenzie. 2004. Caller Identification from Spelled-Out Personal Data Using a Database for Error Correction. Duke University Internal Report.
- E. Levin, S. Narayanan, R. Pieraccini, K. Biatov, E. Bocchieri, G. Di Fabbrizio, W. Eckert, S. Lee, A. Pokrovsky, M. Rahim, P. Ruscitti, and M. Walker. 2000. The AT&T-DARPA Communicator Mixed-Initiative Spoken Dialog System. *ICSLP 2000*.
- D. Maynard. 2003. Multi-Source and Multilingual Information Extraction. Expert Update.
- S. Seneff, E. Hurley, R. Lau, C. Pao, P. Schmid, and V. Zue. 1998. Galaxy-II: A Reference Architecture for Conversational System Development. *ICSLP 98*, Sydney, Australia.
- S. Seneff and J. Polifroni. 2000. Dialogue Management in the Mercury Flight Reservation System. *Satellite Dialogue Workshop, ANLP-NAACL*, Seattle, Washington.
- M. Walker, J. Aberdeen, J. Boland, E. Bratt, J. Garofolo, L. Hirschman, A. Le, S. Lee, S. Narayanan, K. Papineni, B. Pellom, J. Polifroni, A. Potamianos, P. Prabhu, A. Rudnicky, G. Sanders, S. Seneff, D. Stallard and S. Whittaker. 2001. DARPA Communicator Dialog Travel Planning Systems: The June 2000 Data Collection. *Eurospeech 2001*.
- M. Walker, A. Rudnicky, J. Aberdeen, E. Bratt, J. Garofolo, H. Hastie, A. Le, B. Pellom, A. Potamianos, R. Passonneau, R. Prasad, S. Roukos, G. Sanders, S. Seneff and D. Stallard. 2002. DARPA Communicator Evaluation: Progress from 2000 to 2001. *ICSLP 2002*.
- W. Ward and B. Pellom. 1999. The CU Communicator System. *IEEE ASRU*, pp. 341–344.
- W. Xu and A. Rudnicky. 2000. Task-based Dialog Management Using an Agenda. *ANLP/NAACL Workshop on Conversational Systems*, pp. 42–47.