AT&T HELP DESK

Giuseppe Di Fabbrizio, Dawn Dutton, Narendra Gupta, Barbara Hollister, Mazin Rahim, Giuseppe Riccardi, Robert Schapire and Juergen Schroeter[1]

AT&T Labs – Research
180 Park Avenue, Florham Park, NJ, 07932 - USA
{pino,dldutton,ngupta,bbh,mazin,dsp3,schapire,jsh}@research.att.com

ABSTRACT

This paper introduces a new breed of natural language dialog applications which we refer to as the *Help Desk*. These voice-enabled applications are an evolution from Help Desk services that are currently available on the web or being supported by human agents. The goals of a voice-enabled Help Desk are to route calls to appropriate agents or departments, provide a wealth of information about various products and services, and conduct problem solving or troubleshooting. In this paper we address the challenges in building this class of applications particularly when speech data is limited or unavailable. We will present the *TTS Help Desk* as an example of a service that has been deployed for automating the customer care component of the AT&T Labs Natural Voices Business.

1. INTRODUCTION

Speech and language processing technologies have the potential of automating a variety of customer care services in large industry sectors such as telecommunication, insurance, finance, etc. In an effort to reduce the cost structure of customer care services, many of these industries have depended more heavily on complex IVR menus for either automating an entire transaction or for routing callers to an appropriate agent or department. Several studies have shown that the "unnatural" and poor user interfaces of these long touch tone menus tend to confuse and frustrate callers, preventing them from accessing information and obtaining the desired service they expect [7]. A recent study by Mobius management systems reveals that over 53% of surveyed consumers say that automated IVR systems are the most frustrating part of a customer service. In this survey, 46% of consumers dropped their credit card provider due to poor customer care.

The advent of speech and language technologies has the potential for improving customer care not only by cutting the huge cost of running call centers but also by providing a more natural communication mode for conversing with users without requiring them to navigate through a laborious touch-tone menu. This has the effect of improving customer satisfaction and increasing customer retention rate. These values which collectively form the foundation for an excellent customer care experience have been evident in the AT&T Call Routing "How May I Help You[SM]" service, which is currently deployed nationally for consumer services [1].

Over the next few years, speech and language technologies will play a more vital role in not only customer care services but also in Help Desk applications where the objective is not only routing of calls or accessing information but also in handling technical problems, sales inquiries, recommendations, and troubleshooting. Many computing and telecommunication companies today provide some form of a Help Desk service through either the World Wide Web or using a human agent. There is an opportunity for spoken natural language interfaces to play a much larger role in this industry.

In this paper, we address the challenges in voice-enabling Help Desks. We present technology extensions that are needed for speech recognition, speech synthesis, language understanding, dialog and user interface design. One key issue that we address is the creation of complex services when speech data is limited or unavailable. A voice-enabled Help Desk service is presented that is currently deployed for the AT&T Labs Natural Voices Business (see *www.naturalvoices.att.com*). Experimental results are presented in terms of recognition accuracy, understanding accuracy and call completion rate on a set of 1000 dialogs that have been collected during system deployment.

2. VOICE-ENABLED HELP DESK SERVICES

There are several technology requirements needed for voice-enabling Help Desk applications, including having a speech recognizer that supports barge-in and is capable of recognizing large-vocabulary spontaneous speech, a text-to-speech synthesizer that is able to generate high-quality synthesized voice prompts, a language understanding unit that parses the natural language input into relevant information, and a dialog manager that operates in a mixed-initiative mode. Other essentials include the availability of vast amounts of transcribed speech data and a call flow design that best optimizes user satisfaction. In this section, we elaborate on each of these requirements for building Help Desk applications.

2.1. Transcription and Annotation

The largest bottleneck when creating spoken natural language dialog applications is the need for domain-specific speech data in building the underlying recognition and understanding models. This process of collecting and annotating speech data is not only expensive and laborious; it delays the deployment cycle of new services. Our process in building Help Desk services begins by "mining" and "reusing" data and models. Mining of data is done not only from other similar application domains (e.g., telecommunication, insurance, airline, etc), but also from relevant emails, web pages and human/agent recordings.

As part of the labeling process, data are annotated for speech understanding purposes. This is done in two phases. The first phase includes identifying and marking domain specific and domain independent value entities such as phone numbers, credit card numbers, dates, times, service offerings, etc. The second phase includes associating each data item with one or more semantic tags (or classes) that identify the "meaning" of a user's request. These tags can be both general and application specific and are structured in a hierarchical manner. For example, phrases such as "*may I hear this again*" and "*yes what products do you offer*" can be tagged as "discourse_repeat'" and "discourse_yes, info_products" respectively.

[1] Authors are in alphabetical order

2.2. Automatic Speech Recognition (ASR)

Accurate recognition of spoken natural-language input for Help Desk applications requires two components: (a) a general-purpose subword-based acoustic model (or a set of specialized acoustic models combined together), and (b) a set of dialog-based stochastic language models. Creating Help Desk applications imposes two challenges in building these models: the ability to *bootstrap* for initial deployment and the ability to *adapt* as task-specific data become available. In the case of acoustic modeling, our Help Desk ASR engine initially uses a general-purpose context-dependent hidden Markov model. This model is then adapted using *Maximum A Posteriori* adaptation once the system is deployed in the field.

The design of the stochastic language model is highly sensitive to the nature of the input language and the number of dialog contexts or prompts. One of the major advantages of using stochastic language models is that they are trained from a sample distribution which mirrors the language patterns and usage in a domain specific language. Their major disadvantage is the need for a large corpus of data for bootstrapping. Task-specific language models tend to have biased statistics on content words or phrases and language style will vary according to the type of human-machine interaction (i.e., system initiated *vs.* mixed initiative). While we believe there are no *universal* statistics to search for, we look for ways to converge to the task-dependent statistics. We look for different sources of data to achieve fast bootstrapping of language models including:

- Language corpus drawn from domain-specific web sites
- Language corpus drawn from emails (task specific)
- Language corpus drawn from a spoken dialog corpus (non task specific).

The first two sources of data can give an estimate of the topics related to the task. However the nature of web and email data does not account for the spontaneous speech speaking style. On the other hand, the third source of data can be a large collection of spoken dialog transcriptions from other applications. In this case although the corpus topics may not be relevant, the speaking style may be closer to the target Help Desk application. The statistics of these different sources of data are combined via a mixture model paradigm to form an *n*-gram language model. These models are adapted once task-specific data becomes available.

2.3. Text-to-Speech Synthesis (TTS)

The extensive call flow in Help Desk applications to support information access and problem solving and the need to rapidly create and maintain these applications make it both difficult and costly to use live voice recordings for prompt generation. TTS plays a critical role in this new breed of natural language services where up-to-the-minute information (e.g., time and weather) and customization to an individual's preferred voice are necessary. Customization means that a TTS system would provide a large variety of distinctive voices, and, within each voice, several speaking-styles of many different languages. This is critical for "branding" of Help Desk services.

Our TTS engine uses AT&T Labs Natural Voices technology and voice fonts [2]. Due to automation of the voice creation process, new and customized voice fonts can be created in less than a month. Including task specific data (i.e., materials relevant to the application) can assure a higher quality TTS voice. For example, the main voice font used in the Help Desk TTS engine, named "Crystal", has been trained with over 12 hours of interactive dialogs between human agents and customers [2]. In the Help Desk application described later in this paper, over eight different voice fonts have been used within the same application for presenting different languages and dialog contexts.

2.4. Spoken Language Understanding (SLU)

2.4.1. Text Normalization

Text normalization in SLU is an essential step for minimizing "noise" variations among words and utterances. This has the potential of increasing the effective size of the training-set and improving the SLU accuracy. The text normalization component is essentially based on using morphology, synonyms and other forms of syntactic normalization. The main steps include stemming, using a synonyms dictionary, and removal of dysfluencies, non-alphanumeric and non-white space characters.

2.4.2. Entity Extraction

An important functionality of an SLU component is the ability to parse the input speech into meaningful phrases. Parsing for Help Desk applications is simplified to a process of identifying task-specific and task-independent entities (such as phone numbers, credit card number, product type, etc.). Each entity module is built using a standard context-free grammar that can be represented by a finite state transducer. Following text normalization, entities are identified by composing each input text string with all active entity modules. For example, the sentence "*my bill for January* 2nd" is parsed as "*my bill for <Date> January 2nd </Date>*". Entity extraction not only helps to provide the dialog manager with the necessary information to generate a desired action but it also provides some form of text normalization for improving the classification accuracy.

2.4.3. Semantic Classification

The next step is to categorize each utterance into one or more semantic classes. A machine learning approach is taken for this problem. A classifier is trained using a corpus of collected utterances that have been annotated using a predefined set of semantic tags.

To train our classifier, we use a technique called *boosting*. The basic idea of boosting is to combine many simple and moderately inaccurate prediction rules into a single rule that is hopefully highly accurate. Each of the base rules is trained on weighted versions of the original training set in which the "hardest" examples - i.e., those that are most often misclassified by the preceding rules - are given the greatest weight. The base rules are then combined into a single rule by taking a kind of majority vote. The first practical and still most widely studied boosting algorithm is Freund and Schapire's AdaBoost [4].

We used an implementation of boosting developed by Schapire and Singer called BoosTexter [5]. In this implementation, each base rule makes its predictions based simply on the presence or absence of a word or short phrase in the utterance.

Like most machine-learning methods, boosting is heavily data driven, and so requires a good number of examples. In developing Help Desk applications, it is often necessary to deploy the system before a sufficient number of examples have been collected. To get around this difficulty, we use human knowledge to compensate for the lack of data. In particular, we use a modification of boosting developed by Rochery et. al [6] that admits the direct incorporation of prior knowledge so that a classifier is built by balancing human-crafted rules against what little data may be available. The human built rules have a simple form and need not be perfectly accurate; for instance, a rule may state that if the word "demo" occurs in the input then the user may want to hear a demonstration of some sort. Incorporating prior knowledge in a probabilistic fashion allows

rapid deployment and a more effective way to instantly add new semantic tags throughout service evolution.

2.4.4. Question/Answering

Help Desk applications that are available on the web often provide an extensive list of Frequently Asked Questions (FAQs) to help users access detailed information in a straight forward manner. Question/answering is frequently used today in text understanding systems. For example, the AT&T IO-NAUT system (see *www.ionaut.com*) can provide answers to queries requesting entity information (such as names and dates).

In our Help Desk architecture, we have incorporated a question/answering module to help users with task-specific FAQs. This is provided in the form of a QA table (Questions and Answers), extracted from previous calls to the systems. The accuracy of this module is improved by partitioning the table into smaller subsets, each corresponding to a semantic tag. During a call, if a user asks a question which matches closely one found in the QA table, the answer is automatically passed to the DM along with any entities and semantic tags from the classifier module. String matching is performed using cosine similarity within the vector space model well known in the information retrieval field. Better matching accuracy has been observed if normalization of the vectors is carried out with the query length as opposed to the entire data set.

2.5. Dialog Management (DM)

Mixed-initiative spoken dialog technology remains at its infancy and there exist significant challenges on how to build and easily maintain large-scale voice-enabled applications. This is a particularly important issue for Help Desks where the nature of the information is constantly changing. The complexity of the dialog modeling and the lack of adequate authoring tools compromise the value and effectiveness of automated Help Desk services which are usually more expensive and time consuming to maintain compared to traditional web-based content.

Our Help Desk DM has been designed to address these challenges. The approach proposes, through general dialog patterns, a unified view to represent a human-machine dialog flow structure of commonly accepted reference models for mixed-initiative systems. A general engine operates at the semantic representation level provided by the SLU and current dialog context to control the interaction flow. To describe the human-machine interaction, we adopted the traditional approach of "sets of contexts" [3] dialog management with few extensions to address the specific domain requirements. At each dialog turn, the DM is focused on accomplishing a concrete task or subtask. The dialog context is maintained by a set of state variables or frames and the dialog history. The interpreter module, shown in Figure 1, is responsible for providing a semantic interpretation of the concept categorization and the named entities generated by the SLU module. Logical predicates described by rules allow the interpreter to rank classes and assign a contextual interpretation to the input based on the current frame content. The interpreter also has access to the state variables and the dialog history. The history mechanism keeps track of previous dialog turns and captures situations where the request is underspecified or too general. This is particularly useful for addressing discourse references such as ellipsis and anaphora. For example, if the current topic has a missed mandatory attribute, the interpreter would first check if the attribute has been previously collected; otherwise it would engage a clarification sub-dialog to obtain the missed information.

A Finite State Machine (FSM) engine controls the actions taken in response to the interpreter output. Each information state provides support for general user interface patterns such as *correction*, *start-over*, *repeat*, *confirmation*, *clarification*, *contextual help*, and *context shifts*. Topic tracking is an important feature in a Help Desk since it provides the infrastructure for rendering information. General conversation topics are managed by a subdialog that (a) handles, in declarative way, new topics, (b) specifies the level of details per topic, and (c) allows context shift to take place at any point in the dialog.

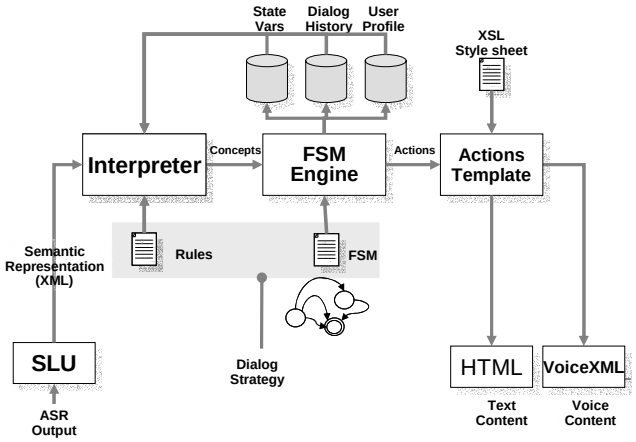


Figure 1. *Dialog Manager Architecture*

Finally, the Action Template module represents a template-based output generator. An XML markup language describes the dialog actions (e.g. prompting, grammar activation, database queries, and variable values updates) and the topic structures. New topics and subtopics can be added, removed or updated at this level without changing the basic service logic. At run-time, the output is translated by an XSL style sheet either to VoiceXML (telephony system) or to HTML for output authoring. In this way the presentation layer and the dialog structure for the topic subdialog are completely separated from the service logic and are easy to maintain with traditional authoring tools.

2.6. User Interface (UI)

User interface planning is a critical phase in the design of Help Desks and a challenge especially when working with synthesized speech. UI is what the customer experiences when interacting with a system and plays a critical role in the success or the failure of a service. There are two challenges in UI design for Help Desk applications: (a) Usability – increasing the likelihood of call completion with minimal user confusion by supporting context shift in the dialog, providing information and help, and by learning how users interact with the system and propagating that knowledge to improve the various technology components; (b) Quality – overcoming the obvious difficulties when working with synthesized speech that often lacks emotions. We propose using a screenwriting dialog technique where a back story is created for the synthesized voice based on a set of desired character traits (e.g., cheerful, trustworthy, etc). A one-page description of the voice's "life" history is described, and prompts are written "in-character". Different synthesized voices are used to convey different information to the user.

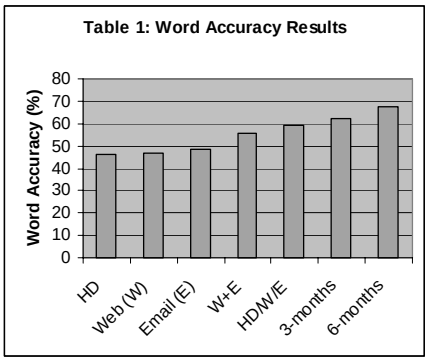
Our dialog strategy begins with the phrase "*How May I Help You?*" It supports natural language input and context shift throughout the application.

3. TTS HELP DESK

A prototype system for a Help Desk application, called *TTS Help Desk*, was developed and deployed for the AT&T Labs Natural

Voices - a business that specializes in selling and marketing TTS products and voice fonts. The *TTS Help Desk*, which was deployed on July 31st, 2001, the day the business was launched, took less than three months to design, develop and test. It currently receives over 1000 calls per month from business customers. The purposes of the service are to perform routing of calls into specialized agents (such as sales, technical support, customer service), and to provide information about the various products and services. The system also provides callers with a variety of demonstrations of the different voice fonts and languages.

The initial data collection effort in building the TTS Help Desk was primarily based on a large set of email interactions that took place prior to launching the business. Relevant messages were manually extracted and annotated using a set of 62 broad semantic tags that describe the types and characteristics of the products and services the business is able to support. These tags were categorized into broader groupings such as agent, general information, help, technical and web site. The rest of the paper presents three sets of results for (a) ASR, (b) question/answering, and (c) task completion rate, on a set of 1000 dialogs.



3.1. ASR Results

Detail analysis of the corpus shows that it exhibits a mixed sample of two language styles: key phrases and spontaneous language. The average number of user turns is 3.3 with 27% of users engaging in longer interactions than the average. Although there are roughly 75 possible prompts at each dialog context, we have clustered those contexts into four categories: *Generic*, *Confirmation*, *Language* and *Help*. Each context corresponded to a stochastic language model and was bootstrapped using three sources of data: web, emails and an inventory of a human-machine database acquired from other dialog applications. Table 1 shows the overall word accuracy of the TTS Help Desk system on 1000 dialog interactions. These results show that we were able to achieve 59% word accuracy without any formal data collection. When sufficient data was available (after 6 months from system deployment), the accuracy jumped to nearly 68%.

3.2. Question/Answering Results

Among the data collected, a small set of 250 questions from one specific tag were identified as potential FAQs and grouped into 81 distinct sets. Thus for each answer there were potentially one or more questions. Given a question with a specific semantic tag, the task was for the system to identify the correct answer. The 81 sets of questions constituted the training set and were indexed using a vector space model. The test set consisted of 336 questions of which only 69 corresponded to valid questions, and the remaining were added to evaluate the robustness of our technique. At a given operating point, precision and recall were computed at 0.9 and 0.94, respectively. These results are encouraging and show the effectiveness of our question-answering system. More details of these experiments on a larger test set including all tags will be published elsewhere.

3.3. Task Completion

Table 2 presents the results of the semantic classifier along with the task completion rate for the initial five revisions of the system that span over a period of 3 months. Although the functionalities of the system were continuously changing, the table shows consistent improvement in the classification accuracy and the task completion rates. The classification accuracy for the V1.4 version of the Help Desk, computed as the percentage of detecting the correct semantic tag, was 84%. The task completion rate, computed as the percentage of correct system action given an input request, was measured between 72% and 96% depending on the task (average at 85%).

Table 2: Classification and task completion results

Release	**V1.0**	**V1.1**	**V1.2**	**V1.3**	**V1.4**
Classification	72	75	83	85	84
Routing	62	74	85	84	83
Demo	74	80	87	94	96
Information	80	78	71	68	72

4. SUMMARY

This paper presented a new breed of natural-language dialog applications which we refer to as Help Desks. The challenges behind building such services when limited data is available were addressed in this paper. A Help Desk application for a TTS business was presented. Results show that (a) the ASR accuracy which was initially at 59% through bootstrapping was improved to 68% following 6 months of system deployment; (b) question/answering results were at 0.9 and 0.94 for precision and recall, respectively; (c) the semantic classification accuracy and the average task completion rate were 84% and 85%, respectively. Extension of the Help Desk to perform troubleshooting and problem solving is currently in progress.

Acknowledgments

The authors would like to thank Ilana Bromberg for processing the initial e-mail data, and Srinivas Bangalore, Bryant Parent, Jim Rowland and Jay Wilpon for fruitful discussions.

References

[1] A.L. Gorin and G. Riccardi and J.H. Wright., "How May I Help You?" *Speech Communication*, pp.113-127, 1997.

[2] M. Beutnagel and A. Conkie and J. Schroeter and Y. Stylianou and A. Syrdal, "The AT&T Next Gen TTS System", Joint Meeting of ASA, EAA and DAGA, 1999.

[3] J. Allen, D. Byron, M. Dzikovska, G. Ferguson, L. Galescu, and A. Stent, "Towards Conversational Human-Computer Interaction" *AI Magazine, 2001.*

[4] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting", Journal of Computer and System Sciences, 1997.

[5] R. E. Schapire and Y. Singer, "BoosTexter: A boosting-based system for text categorization", Machine Learning, 2000.

[6] M. Rochery, R. Schapire, M. Rahim, N. Gupta, G. Riccardi, S. Bangalore, H. Alshawi, S. Douglas, "Combining Prior Knowledge and Boosting for Call Classification in Spoken Language Dialog", ICASSP, 2002.

[7] J. Bers, B. Suhm and D. McCarthy, "Please Tell Me the Reason for Your Call", Speech Technology Magazine, November 2001.

[8] M. Rahim, G. Di Fabbrizio, C. Kamm, M. Walker, A. Pokrovsky, P. Ruscitti, E. Levin, S. Lee, A. Syrdal, K. Schlosser, "Voice-IF: A Mixed initiative spoken dialogue system for AT&T conference services", Eurospeech, 2001.