The AT&T Spoken Language Understanding System

Narendra Gupta, Gokhan Tur, Dilek Hakkani-Tür, *Member, IEEE*, Srinivas Bangalore, Giuseppe Riccardi, *Senior Member, IEEE*, and Mazin Gilbert, *Senior Member, IEEE*

***Abstract*—Spoken language understanding (SLU) aims at extracting *meaning* from natural language speech. Over the past decade, a variety of practical goal-oriented spoken dialog systems have been built for limited domains. SLU in these systems ranges from understanding predetermined phrases through fixed grammars, extracting some predefined *named entities*, extracting users' *intents* for call classification, to combinations of users' intents and named entities. In this paper, we present the SLU system of VoiceTone® (a service provided by AT&T where AT&T develops, deploys and hosts spoken dialog applications for enterprise customers). The SLU system includes extracting both intents and the named entities from the users' utterances. For intent determination, we use statistical classifiers trained from labeled data, and for named entity extraction we use rule-based fixed grammars. The focus of our work is to exploit data and to use machine learning techniques to create scalable SLU systems which can be quickly deployed for new domains with minimal human intervention. These objectives are achieved by 1) using the *predicate-argument* representation of semantic content of an utterance; 2) extending statistical classifiers to seamlessly integrate hand crafted classification rules with the rules learned from data; and 3) developing an *active learning* framework to minimize the human labeling effort for quickly building the classifier models and adapting them to changes. We present an evaluation of this system using two deployed applications of VoiceTone®.**

***Index Terms*—Named entities, semantic classification, semantic representation.**

I. Introduction

In the last decade, a variety of practical goal-oriented spoken language understanding (SLU) systems have been built for limited domains. One characterization of these systems is the way they allow humans to interact with them. On one extreme, there are *machine-initiative* systems, commonly known as interactive voice response (IVR) systems [1]. In IVR systems, the interaction is controlled by the machine. Machine-initiative systems ask users specific questions and expect the users to input one of predetermined keywords or phrases. For example, a mail delivery system may prompt the user to say "schedule a pick up, track a package, get a rate, or order supplies." In such a system, SLU is reduced to detecting one of the allowed keywords or phrases in the users' utterances.

Manuscript received June 1, 2004; revised November 1, 2004. The Associate Editor coordinating the review of this manuscript and approving it for publication was Dr. Timothy J. Hazen.

N. Gupta, G. Tue, D. Hakkani-Tür, S. Bangalore, and M. Gilbert are with AT&T Labs-Research, Florham Park, NJ 07932 USA (e-mail: ngupta@research.att.com; gtur@research.att.com; dtur@research.att.com; srini@research.att.com; rahim@research.att.com).

G. Riccardi was with AT&T Labs-Research, Florham Park, NJ 07932 USA. He is now with the Computer Science Department, University of Trento, Trento, Italy.

Digital Object Identifier 10.1109/TSA.2005.854085

On the other extreme, there are *user-initiative* systems in which users control the flow and a machine simply executes the users' commands. A compromise in any realistic application is to develop a *mixed-initiative* system [2] where both users and the system can assume control of the flow of the dialog. Mixed-initiative systems provide users the flexibility to ask questions and provide information in any sequence they choose. Although such systems are known to be more complex, they have proven to provide more *natural* human/machine interaction. For example, in the DARPA initiated airline travel information system (ATIS) project [3], which was designed to provide flight information, users provide some flight attributes like origination and destination cities, dates, etc. In a mixed-initiative dialog, a user may choose to provide such information in several different ways. For example, a user may say "I want to fly to Boston from New York next week" and another user may express the same information by saying "I am looking to fly from JFK to Boston in the coming week." In spite of this freedom of expression, utterances in such applications have a clear structure that binds together the specific pieces of information. In the case of ATIS, these specific pieces of information, among others include *Destination* and *Departure Date*. SLU in such systems is therefore reduced to the problem of extracting this task specific information. Such crisp and unambiguous pieces of information are called *named entities* [4]. Most ATIS systems employed either a data-driven statistical approach (mostly from the speech processing community) such as AT&T's CHRONUS [5] and BBN's hidden understanding models [6] or a knowledge-based approach (mostly from the computational linguistics community) such as the Massachusetts Institute of Technology's TINA [7], CMU's Phoenix [8], and SRI's Gemini [9].

Although most mixed initiative dialog systems are designed to perform only one task, such as the ATIS systems, one can imagine having more complex dialog applications that are configured to understand and perform several tasks, such as canceling some service, resolving a billing discrepancy, adding a new telephone line, etc. Each task may have different sets of named entities that can be structured and spoken differently by different users. The function of the SLU for such a system is not only to extract the named entities but also to identify which task the user is calling about. In this paper, such tasks will be referred to as users' *intents*. Examples of such systems include the AT&T "How May I Help You?" (HMIHY$^{\mathrm{SM}}$) [10], [11], the Lucent call routing system [12] and the BBN call director [13].

HMIHY$^{\mathrm{SM}}$ is a call routing system. The users are greeted by the open-ended prompt "How May I Help You?," encouraging them to talk naturally. For intent determination, the HMIHY$^{\mathrm{SM}}$ depends heavily on the phrases in the input utterances, which are salient to certain intents or call-types. For example, in the manually transcribed input utterance "I would like to change oh about long distance service to one charge nine cents a minute," the *salient phrase* "cents a minute" is strongly associated with the call-type *Calling_Plans*. The salient phrases are automatically acquired from a corpus of transcribed and labeled training data, and are clustered into salient grammar fragments in the form of finite state machines [14], [10]. In HMIHY^{SM}, utterances are classified into call-types by identifying the grammar fragments present in the recognizer output, and then by applying a statistical decision rule (Naïve Bayes-like) to combine their strengths of associations with different call-types. In summary, the natural language understanding in HMIHY^{SM} is performed by a statistical classifier using salient grammar fragments as features.

One of the main advantages of statistical classification methods is their robustness to both variations in spoken language and to recognition errors introduced by speech recognition. Their disadvantage lies in the need to collect and annotate large volumes of data for training the underlying models. In HMIHY^{SM}, the data was labeled with action oriented call-types, i.e., each label is essentially the destination where a call has to be routed. Such a system has at least two drawbacks. First, changing business needs may require continuous changes in routing destinations. This would in turn require data to be relabeled and classification models to be retrained. Second, since the routes are application specific, data labeled for one application cannot be reused for another application.

In this paper, we present the SLU in VoiceTone®, a spoken dialog service offered by AT&T. As opposed to call routing only, in VoiceTone® we are interested in servicing the call completely. There are three important aspects of our system. First, our semantic representation captures the intents of the speaker and not the actions performed by the system. Therefore, even if the actions taken in an application change, no relabeling of data is required. In addition, our representation promotes labeling consistency by human labelers and allows reuse of data across applications. The second contribution of this work is with respect to the classifier itself. Our classifier model does not completely depend on the labeled data. It allows for seamless integration of hand crafted classification rules with the rules learned from data. This enables development of useful models even in the early phases of an application when little or no data is available for building robust classification models. Lastly, in this work we also address the problems of scalability and adaptability. State of the art statistical classification systems are trained using a large amount of labeled training data. Preparation of this data is labor intensive and time consuming. Furthermore, practical dialog systems are dynamic in nature. This means that the distribution of user intents and the language used to express them change constantly over time. Therefore, a good classification model at one point in time may not be as good at other times. Our models must constantly adapt to such changes. A simple but very expensive way to do this would be to periodically rebuild the model using only the most current data. Instead, in this work we have employed an *active learning framework* to select and label only those data items that improve the classifier performance the most. This minimizes the human labeling effort for building the models.

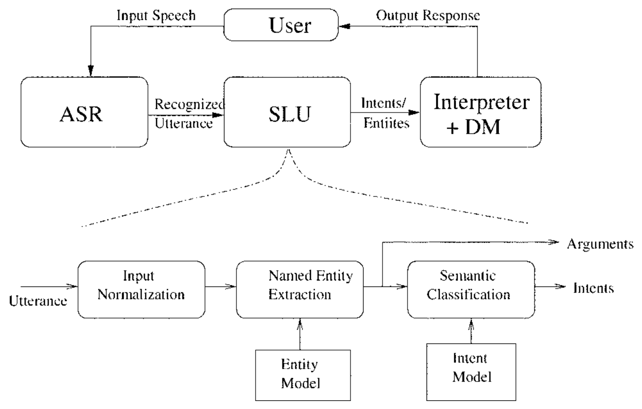


Fig. 1. AT&T SLU run-time system.

The organization of this paper is as follows. In Section II, we provide an overview of VoiceTone®. In Section III, a description of our representation language is provided. In Section IV, the process of labeling the training data using the active learning framework is described. In Sections V–VII, we describe the three main components of the SLU system, namely input normalization, semantic classification, and named entity extraction. To demonstrate the significance of our work, we provide experimental results using two deployed applications of AT&T VoiceTone® spoken dialog services, in Section VIII.

II. SLU System

In Fig. 1, we show a simplified block diagram of the AT&T spoken dialog system. An automatic speech recognizer (ASR) transcribes the speech signal into text and passes that to the SLU unit. The SLU forms a semantic representation of the text and passes it to the dialog manager (DM). The DM interprets the semantic input within the context of the dialog, and responds appropriately to the caller.

The lower section of Fig. 1 shows a block diagram of the SLU system. There are three major processing steps. In the first step, the input normalizer removes filled pauses from the transcribed input. In the second step, named entities are extracted from the normalized text. Finally, the input text is passed to a semantic classifier that assigns it one or more semantic labels capturing the users' intent. These semantic labels together with the named entities constitute the complete semantic description of the utterance and are passed on to the DM.

Both entity extraction and semantic classification modules are driven by semantic models. The process of building these models is a major part of application development. It requires detailed understanding of the task being automated. To understand the types of intents callers have, and how they express them, speech data is collected, transcribed and analyzed. Once a set of intents and named entities needed for the application are documented in a labeling guide, human labelers label the transcribed speech data. This data is then used to train the classification models. In the early phases of the application, there may not be enough data to build good classification models for each semantic label or named entity. However, one may have a good understanding of how those intents and entities are expressed. In our framework, this kind of human knowledge is expressed as rules, which are modified and/or adapted by the learning algorithm, as and when more data become available.

It is important to note that in our architecture, the SLU does not have any contextual information about the dialog. Instead, it is the DM's job to maintain a dialog context and interpret the context independent SLU output within the context of the dialog. Data collection and labeling is a major part of system development cost. This architecture reduces it in a number of ways. First, the labelers can label utterances to best represent the intents within the utterance without using any external knowledge. Besides gains in labelers' productivity, this also results in more consistently labeled data. Second, the changes in the context dependent interpretation of callers' intents can be implemented without relabeling the data. Third, data labeled in a context independent manner can more easily be reused across applications. In this paper, we focus only on the SLU. Details of the DM will not be discussed further in this paper, other than to say that in VoiceTone® the DM represents the dialog context in the form of attribute value pairs, and uses hand written rules to carry out the interpretation task [15].

III. SLU Representation Language

Our interest here is in a representation that can capture information sufficient to fulfill a callers' request. Most of the approaches for semantic annotation consider nested relations among objects. For example, in the FrameNet project, relations among *frames* [16] are described, without an underlying dialog task in mind. In these kinds of semantic annotation schemes, however, once the sentence is semantically parsed, it can be used for various purposes, such as information extraction [17]. In contrast, our approach is driven by the need for a lightweight representation that can be built bottom-up, reliably cover large scale human-machine dialog corpora, and model noisy spontaneous spoken language. In this work, we take the challenge of balancing the coverage of the SLU representation with stability (agreement) of human judgment. In previous work, we have instantiated the problem of SLU based on an input–output association of utterance-actions [10]. More specifically, we used an *action oriented semantic* representation language that mapped one semantic label to each prompt and/or action that could be taken by the system. The advantage of such an approach is that it directly gives a computational model for the input–output associations. The disadvantage is the lack of built-in generalization of the model representation (spoken input-action pairs). Should the prompts/actions taken by the dialog system change over time, the semantic representation must also be changed. In our current framework, we have improved the generalization of the representation language while maintaining its coverage. More specifically our choice of the representation language was made to achieve robustness toward: 1) the *time-varying* nature of spoken language; 2) the *time-varying* nature of the external world (e.g., introduction of new service); and 3) annotator disagreement, following a *stable* labeling scheme across various corpora.

In our representation, utterances are labeled with the *intent* of the speaker as opposed to the action that the system must take in response. For example, in response to the following utterances, the system would prompt the caller about the specific charges in question. Therefore, in an action oriented semantic representation, both these utterances will be labeled as "Charge_on_bill."

I see charges on my bill that I do not recognize.
I want credit for some charges on my bill.

Since intents in these utterances are different, their intent oriented semantic representation will be different. To capture the intent, we have used a predicate-argument $v_i(o_j)$ representation. The predicates v_i are domain independent verbs reflecting the action the caller is trying to perform by the utterance (also known as *dialog acts* [18]). The arguments o_j constitute the domain specific objects, actions or concepts on which these actions are performed. Some examples of predicates v_i are the following.

- Request: user requesting for specific information.
- Report: user reporting specific information.
- Verify: user requesting to verify information.
- Explain: user requesting an explanation of information.

Examples of arguments o_j from a transactional domain are "Credit, Payment, Bill_Charge." Having identified the domain dependent objects and concepts, a list of semantic representation primitives can be generated by joining them with domain independent predicates. For example, "Request(Credit), Verify(Payment), Explain(Bill_Charge), Report(Payment)." Some example utterances and their labels from this set are

I see charges on my bill that I do not understand.	Explain(Bill_Charge)
I want credit for some charges on my bill.	Request(Credit)
I am just wanting to tell you that I have made the payment.	Report(Payment)
I am calling to check if you received my payment.	Verify(Payment)
I dialed a wrong number.	Report(WrongNumber)

.

Values of the arguments are sometimes specified by the user. For example, in the utterance "I want credit for ten dollars," a "Request" for "Credit" is being made. The value of which is of type "monetary amount" and the value of "monetary amount" is \$10. Objects like "monetary amount," "dates," "account numbers," and "phone numbers" are called *named entities* [4] and are part of the semantic representation. Accordingly, the semantic representation of the example given previously is: "Request(Credit) monetary_amount = \$10."

In our representation the following should be noted. 1) Intent (*call-type*) of an utterance is represented in the form of predicate-argument, $v_i(o_j)$. 2) Separating domain dependent aspects from domain independent aspects provides a systematic way of creating the semantic representation for an application. 3) Once labelers understand the semantics of predicates, they only need to learn to spot the domain dependent object/concepts in the utterances from different applications. 4) Since intents of the utterance are captured, changes or extensions to an application only require changes to the DM and do not require relabeling the data. Furthermore, since in our labeling scheme intents consistent across applications are captured, it is possible to reuse the labeled data across applications. For example, data labeled as "Request(Credit)" can be used for all applications where a caller is likely to ask for credit. 5) An utterance may contain many intents. Therefore, multiple labels are allowed for each utterance

We had sent the payment fifteen days back still we cannot make long distance calls.	Report(Payment), Report(Service_Problem)

.

Besides the advantages mentioned previously, our labeling scheme also has other desirable properties. First, in contrast to the action-oriented semantic representation, our scheme discourages labelers from using any knowledge specific to the application or the context of the utterance. Intuitively, this makes labeling more consistent and more accurate semantic classification models can be learned from a smaller amount of training data. A second advantage of intent-oriented semantics is that the classifiers (multiclass, multilabel) trained on such data have relatively better compositional properties than those trained using action oriented semantics. The compositional property allows the assignment of multiple semantic labels to test utterances even when such combinations of labels are not seen in training data.

IV. Data Collection and Labeling

In order to compile application specific semantic labels and entities, application data is collected using the *wizard-of-oz* approach. The purpose of this collection is to gather representative data on how users converse with a machine under some constant application settings. It has been shown that the language characteristics of the responses to machine prompts is significantly different from those to humans [19], [20]. In a wizard-of-oz approach, a human, i.e., a wizard, acts on behalf of the system. Callers to the system do not know about this and believe that they are talking to a machine. To make this process cost effective, we implemented a "ghost wizard" approach that is a slight variation of wizard-of-oz approach. Ghost wizard approach is completely automatic. The goal of this approach is 1) to collect users' responses to opening prompts, and 2) to fine tune the set of initial prompts for full system deployment. The ghost wizard consists of an opening prompt like "How may I help you?" After recording callers' response to such an opening prompt, the callers are informed that their call is being routed to a representative for further handling.

After some data is collected, user experience (UE) designers analyze it to determine a list of semantic labels and named entities needed for the application. Specifically, designers use their knowledge about the application and look for the kinds of requests callers are making. In this manner, high-frequency requests are easily identified. To reliably identify low and medium frequency requests, designers also employ an intent clustering tool [21]. Based on this analysis, designers write an annotation guide, documenting the semantic labels and named entities needed for the application. Such a guide contains positive and negative examples of the different semantic labels and named entities. The purpose of this guide is to help the labelers throughout the process of labeling the data.

Once the semantic labels are designed for an application, the next step is to manually label the data, i.e, assign one or more predefined intent(s) (*call type(s)*) to each utterance and to mark the named entities. Labeling is done using textual input, such as

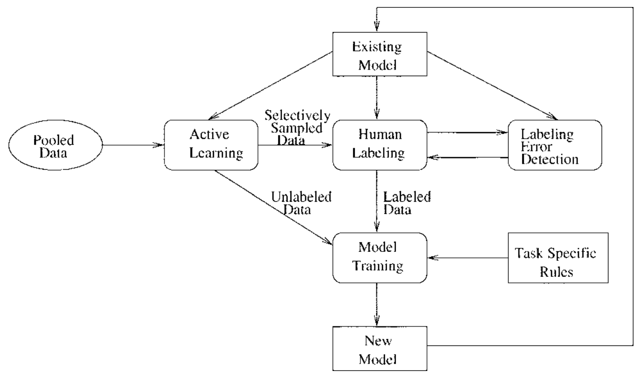


Fig. 2. Active learning framework in the AT&T SLU system.

human or ASR transcriptions, instead of audio. There are three main reasons for this: First, reading is much faster than listening. Second, human transcriptions can also be used to improve ASR models. Third, it is much faster to relabel the transcribed data if needed, for example, due to a change in the semantic labels. We typically use ASR transcriptions if they are recognized with high confidence, otherwise, we revert to manual transcription and labeling [22].

We use statistical methods for training the semantic classification models. To build good models, a large amount of manually labeled data is required. While it is possible to collect a large amount of data, labeling it is an expensive task. Once initial models are developed and deployed, they are constantly updated to adapt to changes in the nature of callers' requests. We employ an active learning framework, shown in Fig. 2, to minimize the human labeling effort for this purpose. The main components of this framework are active and semisupervised learning and labeling error correction.

This framework assumes that there is a constant stream of incoming unlabeled data, which is buffered into a pool, S_p. To select the utterances to be labeled manually, we use active learning, the purpose of which is to sample those examples that improve the classifier performance the most. More specifically, only those examples in the set S_p, for which the confidence values output by the current model are below a certain threshold are selected [23]. The rest of the examples, for which the current model outputs relatively higher confidence values, are used in unsupervised manner, i.e., they are used with the labels assigned by the current model to train a new model [24]. Once a new batch of examples are pooled, this cycle is repeated. The corresponding algorithm for combining active and semisupervised learning is shown in Fig. 3. Active learning has the distinct advantage of efficiently using annotated data and, thus, reducing the human effort. Moreover, it has the intrinsic capability to adapt to nonstationary events by means of a feedback mechanism in the training algorithm. New requests or significant changes in call-type distributions are automatically identified and submitted for manual labeling.

Labeling is performed utterance by utterance to ensure it is context independent. Although labelers follow an annotation guide, this process tends to involve some subjective judgment which can cause inconsistency in labeling. To avoid such inconsistency, each label is verified by a different labeler. Clearly more consistently labeled training data helps in the creation of

```
1) Given some amount of human-labeled training data $S_t$, and a larger amount of unlabeled
   data in the pool $S_p = \{s_1, ..., s_n\}$
2) While (labelers/utterances are available) do
   2.1 Train a classifier using the current training data $S_t$
   2.2 Classify the utterances in $S_p$ using this classifier, and compute the confidence
       scores, $CS(s_i)$,   $i = 1, ..., n$
   2.3 Label the set $S_k = \{s_i : CS(s_i) < th\}$ manually
   2.4 $S_t = S_t \bigcup S_k$ and $S_u = S_u \bigcup (S_p \backslash S_k)$
   2.5 Augment the classifier with the machine-labeled data, $S_u$
   2.6 Get new unlabeled data into the pool, $S_p$
```

Fig. 3. Algorithm for combining active and semisupervised learning. th is the threshold used to select the utterances to label.

better classification models. However, verifying labeled data is also labor intensive. As shown in Fig. 2, we adopt a *labeling error detection* [25] method, inspired by active learning. Our method identifies a small subset of the labeled data that is believed to be erroneous or inconsistent, and must be verified manually. To do this we consider the confidence values in the call-types obtained from the previously trained classifier. We use the Kullback–Leibler (KL) divergence between the first pass of labeler assigned labels P and the outputs of the classifier Q. More formally, we compute

$$\mathrm{KL}(P \parallel Q) = \sum_{i \in L} p_i \times \log\left(\frac{p_i}{q_i}\right) + (1 - p_i) \times \log\left(\frac{1 - p_i}{1 - q_i}\right)$$

where L is the set of all call-types. q_i is the probability of the i^{th} call-type obtained from the classifier. p_i is the probability of the i^{th} call-type as assigned by the labelers in the first pass. Because labelers either assign a call-type or they do not, the value of p_i is either 1 or 0. We then select only those utterances for relabeling that have KL distance above a certain threshold.

V. Input Normalization

A semantic concept can be expressed by using a variety of syntactic, lexical and morphological forms. These variations are compounded by the different kinds of dysfluencies present in the spoken language utterances. Some of these variations might not directly contribute to the semantic content of the utterance. Therefore, we remove them in the *input normalization* step, i.e., before we build the understanding models.

We apply a series of finite-state transductions that normalize certain lexical and morphological variations. These transductions (f_i) take the general form of mapping a word W onto a normalized form W'. More specifically $f_i(W) = W'$, where W is a word from the vocabulary V and W' is a normalized form from the vocabulary $V' \bigcup \{\epsilon\}$, where ϵ is the empty symbol and $|V| \geq |V'|$.

Intuitively, synonyms, morphological processing and stop word transductions have the potential to increase the effective size of the training corpus and decrease data sparseness, and should therefore improve the SLU accuracy, particularly when only a small amount of training data is available. However, as described in Section VIII, our experiments with these transductions proved otherwise.

VI. Semantic Classification

In general, semantic classification is the task of mapping relevant information from the ASR, SLU, DM, and/or the application into one or more semantic labels (or classes). In our context, because of the architectural decisions explained in Section II, we will only consider information from ASR and SLU. More formally, given an instance of information $x_i \in X$, the problem is to associate a set $C_i \in C$ of semantic labels with x_i where C is a finite set of semantic labels. Traditionally this task can be accomplished by a knowledge-based approach, i.e., by manually writing an extensive set of rules [7], [26], [27]. In the absence of any training data, these rules can become stochastic by setting a guess estimate for each conditional probability, $p(c_j \mid x_i)$, that instance x_i belonging to semantic class $c_j \in C$.

Although rule-based methods have the advantage of requiring no data, they suffer from lack of robustness, poor accuracy, and inconsistency when designed by different individuals. Nevertheless, these methods have been widely used in spoken dialog systems such as airline reservation [28] and conference information [29] systems. To achieve more natural human/computer interaction, there has been increasing interest over the past decade in data-driven methods for creating SLU systems [6], [10], [30], [31]. Computing decision rules with learning algorithms has been well studied in the machine learning literature [32]–[34]. It provides the benefits of creating more robust and accurate systems. Unlike rule based systems, such data driven systems are free from individual biases.

Given a collection of labeled examples $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ drawn from a distribution $p(x, y)$, where $y_i \in C$, the learning task becomes essentially of finding rules $h(x) : X \to C$ that yield the lowest cost of the prediction error. This is achieved by using a Bayes classifier [35]. Assuming no loss for a correct prediction and a unit loss for any incorrect prediction: $h(x) = \arg\max_{c \in C} p(y = c \mid x)$. Since the distribution $p(x, y)$ is unknown, learning the classification rule is essentially learning to estimate $p(y \mid x)$. This method of estimation differentiates one classification algorithm from the other.

There are at least two types of classification methods [36]: *Generative* methods, where $p(y \mid x)$ is estimated based on assumptions about the distribution $p(x, y)$. Such classifiers derive optimal parameters of $p(x, y)$ using Bayes rule. Examples include Naïve Bayes and hidden Markov models. Generative classifiers were adopted in the AT&T HMIHY$^{\mathrm{SM}}$ spoken dialog system [11].

Discriminative methods, where $p(y \mid x)$ is directly computed. For such classifiers, no assumptions about the distribution of $p(x, y)$ are necessary. Only assumptions about the form of the discriminant functions H are necessary. Parameters of the discriminant functions are computed by the minimization of prediction error $Err(h)$ over training examples S. $h^* = \arg\min_{h \in H} Err(h)$. Examples of discriminative classifiers are AdaBoost [33] and support vector machines (SVMs) [34].

Because of their superior generalization performance, discriminative classifiers have been successfully used in spoken dialog applications [37], [38]. We also use them in VoiceTone®. In particular, we extended the Boosting algorithm to permit the incorporation of prior knowledge (or rules) as a means to compensate for a shortage of training data. Details of this algorithm are given in [39]. Here, we briefly describe the main idea behind our algorithm.

The basic idea of boosting is to build a highly accurate classifier by combining many "weak" or "simple" *base classi-*

fiers, each one of which may only be moderately accurate. The collection of base classifiers is constructed in rounds. On each round t, the base learner is used to generate a base classifier h_t that minimizes an objective function. Besides the training data, the boosting algorithm also provides the base learner a set of nonnegative weights W_t over the training examples. These weights encode how important it is that h_t correctly classify each training example. Generally, the examples that were most often misclassified by the preceding base classifiers will be given the most weight so as to force the base learner to focus on the "hardest" examples. In standard Adaboost, the objective function is the negative log conditional likelihood of the data, $\sum_i \sum_c \ln(1 + e^{-y_{ic}h(x_i,c)})$. Here y_{ic} is $+1$ if $(x_i, c) \in S$, otherwise, it is -1, and $h(x_i, c)$ is the classifier output for intent c and input x_i. It is computed from the collection of base classifiers: $h(x_i, c) = \sum_t h_{t,c}(x_i)$, and represents the strength of x_i's membership in class c. Using a logistic function $\sigma()$, $h(x_i, c)$ can be converted into probabilities $p(c \mid x_i) = \sigma(h(x_i, c))$.

To accommodate the incorporation of prior knowledge, we added a second term to the objective function. It is a measure of relative entropy between the probabilities predicted by the prior knowledge and those predicted by the data. The modified objective function is as follows:

$$\sum_i \sum_c [\ln(1 + e^{-y_{ic}h(x_i,c)}) + \eta RE(\pi(c \mid x_i) || \sigma(h(x_i, c)))]. \tag{1}$$

Here, η is a parameter which is computed empirically to control the relative importance of the two terms. $\pi(c \mid x_i)$ is the estimate (based on prior knowledge) of the conditional probability that the instance x_i belongs to class c.

At run time, given an input x_i, the classifier outputs $p(c \mid x_i)$ for each class $c \in C$. Using a suitable threshold on these probabilities, classes for the input x_i can be selected.

VII. Named Entity Extraction

There has been a significant amount of research done on identifying the parts of input text that represent named entities and extracting their values [4]. We have applied those techniques in the context of SLU. For example, in the utterance "my phone number is 5 5 5 1 2 3 4 area code 2 0 1," the substring "5 5 5 1 2 3 4 area code 2 0 1" contains the named entity called ⟨PHONE_NO⟩. In the first step, the input string is marked as "my phone number is ⟨PHONE_NO⟩ 5 5 5 1 2 3 4 area code 2 0 1 ⟨/PHONE_NO⟩." In the next step, the value of phone number (201-555-1234) is extracted and the input string is further processed to output "my phone number is ⟨PHONE_NO⟩."

In addition to extracting information necessary for the DM to fulfill the customer request, named entity extraction also helps in normalizing the input. As shown in the previous example, the replacement of a specific digit sequence representing a specific phone number by the token ⟨PHONE_NO⟩ normalizes the utterance and effectively increases the size of the training set.

The set of named entities can be partitioned into two sets – application-independent and application-dependent. Examples of application-independent entities include "phone numbers," "dates," "currency," "credit/calling card numbers." These are

DATE	→	ε:<month> MONTH ε:</month>
		ε:<day> DAY ε:</day>
MONTH	→	january:january | february:february | ...
DAY	→	first:first | second:second | ...

Fig. 4. Sample fragment of a "date" grammar.

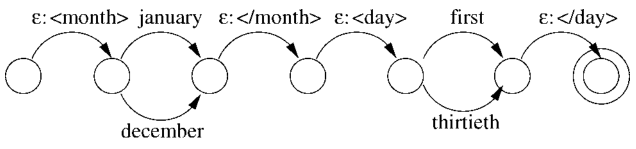


Fig. 5. FST representation of the "date" grammar fragment.

found in many different application. Besides application-independent entities, an application may also need some entities specific to the application. For example names of products and services. Such application-dependent entities are specified by a UE designer in the annotation guide.

We employ a rule-based approach for named entity extraction. Currently, for each entity, a grammar in backus naur form (BNF) is created manually. The creation of a new named entity may involve reusing or extending one of the grammars available in the library of application-independent named entities, or it may involve writing a new grammar from scratch.

As an example, a fragment of a "date" grammar is shown in Fig. 4. Note that the terminals of the BNF are of the form $X : Y$ where $X \in V' \cup \{\epsilon\}$, $Y \in \{V' \cup TAGS\}$, $TAGS = \cup_i \{\langle t_i \rangle, \langle / t_i \rangle\}$ which is the set of start and end symbols representing the entity types, and V' is the vocabulary.

These grammars are typically regular expressions written in a grammar rule notation. They are compiled into finite-state acceptors whose arcs are labeled with the terminals of the grammars. The two components of the arc labels are then interpreted as the input and the output symbols leading to a finite-state transducer representation. The result of compilation of the previous grammar fragment is shown in Fig. 5.

Each entity grammar G_i is compiled into an FST F_i and the final entity extraction model is a transducer F resulting from a union of all the FSTs: $F = \cup_i F_i$.

For entity extraction, the utterance FSM (U), possibly obtained from ASR 1-best, word lattice or a word confusion network, is composed with F resulting in an FSM (M) representing the utterance with entities marked: $M = U \circ F$. It is often the case that the same substring might represent more than one entity type. An example is a sequence of ten digits which could be a phone number or an account number. It is important to note that M holds all possible parses for named entity in the input FSM U. To select the best parse we use longest match (path) criterion. We encode this criterion using weights on the arcs in the FST framework. Although, for the majority of named entities of interest, the grammars can specify the context in which the BNF rules can apply, it is clear that this approach is limited and is unable to deal with other ambiguities that cannot be resolved from a small set of immediate contexts.

The kinds of entities we are interested in can be extracted using the procedure discussed previously. Writing accurate grammars with high precision and high recall is a tedious and time consuming activity. Since it is easier for the DM to detect missing named entities than to recover from a falsely identified named entity (unless it confirms each value explicitly or implicitly as is done in many systems in the DARPA Communicator [40]), in our framework, grammars with higher precision are preferred over those with higher recall. If higher recall is needed for an application, data-driven named entity extraction [41] can be used.

VIII. Experiments and Results

Using VoiceTone® services, AT&T has developed and deployed many spoken dialog applications for its enterprise customers. This is only possible because the SLU system described in this paper is scalable. Development of HMIHY$^{\mathrm{SM}}$ took 12 mo, and another 6 mo thereafter to fine tune its performance. Now the typical turn around time for such an application is only 3 mo, and because of features like intent oriented semantic representation, active learning combined with labeling error detection, fine tuning and adaptation to changing conditions can be done with significantly reduced manual work. While it is hard to experimentally quantify the impact of our semantic representation, it has been shown [24] that the active learning framework can reduce the amount of manually labeled data needed to achieve a given performance by a factor of 2. It has also been shown [25] that the proposed labeling error detection method captures 90% of the errors by manually verifying only half of the data.

In the rest of this section, we will describe the metrics used to evaluate our SLU system and present experimental results on call classification and named entity extraction.

A. Evaluation Metrics

To evaluate the semantic classification performance, we used essentially two metrics. The first one is the top class error rate (TCER). It is the fraction of utterances in which the call-type with maximum probability was not one of the true call-types. The second metric, which is inspired by the information retrieval (IR) community, is the F-Measure (F-M)

$$F\text{Measure} = \frac{2 \times \text{recall} \times \text{precision}}{\text{recall+precision}}$$

where *recall* is defined as the proportion of all the true call-types that are correctly detected by the classifier. *Precision* is defined as the proportion of all the detected call-types that are correct.

The F-Measure changes with respect to the given confidence threshold. For lower thresholds, the precision is lower but recall is higher, and *vice versa* for higher thresholds. In order to select an operating point, we compute the F-Measure for thresholds between 0 and 1 and use the setting that gives the highest F-Measure. One difference between TCER and F-Measure is that the TCER only evaluates the top scoring call-type for an utterance, whereas the F-Measure evaluates all the call-types exceeding the given threshold.

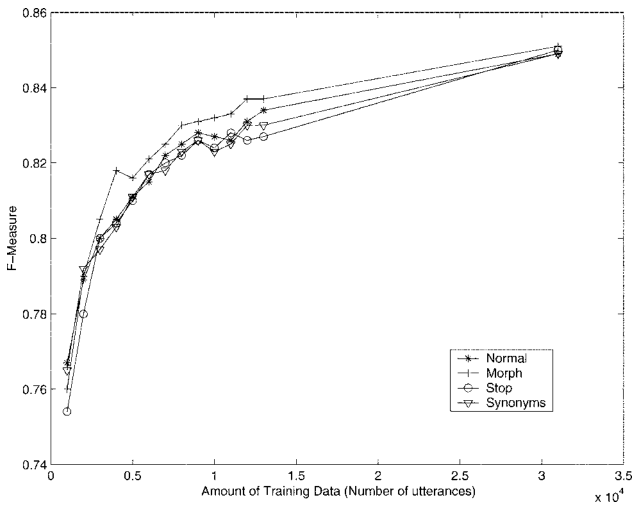


Fig. 6. Effect of different transductions on classifier performance.

B. Effect of Data Normalization

Fig. 6 shows the effect of different normalization transductions on the best F-measure of the SLU trained on increasing number of training examples. The F-measure for each model is computed on the same held out test data. The graph labeled as "Normal" shows the performance when no transduction was performed. Similarly, graphs labeled as "Morph," "Stop," and "Synonyms" show the performance where the corresponding transduction was applied on both training and test data. Although there are some clear trends, the performance changes are generally small.

For each word in the input, the synonym transducer replaced it with its key synonym. Similarly, the morphological transducer replaced a word with the stem, and the stop word transducer removed all stop words. We used the stop word list that is traditionally used by the IR community [42]. This list mainly contains articles, prepositions, pronouns, and conjunctions. Fig. 6 shows that irrespective of the amount of training data, stop word removal actually deteriorates the performance. This is in contrast with what the IR community [43] has shown. The main reason for this is that in IR the size of a document is an order of magnitude larger than the spoken utterances, and stop words do not carry meaningful information for *text categorization*. For semantic understanding of spoken utterances, however, these stop words carry a large amount of information. For example, stop word removal will transform both of the following strings to "mail check" and will lose the crucial semantic distinction among them.

- I would like to know where to mail the check.
- I would like you to mail the check to me.

Intuitively when a small number of training examples are available, both morphological and synonym transductions effectively increase the training data size and allow more reliable estimation of model parameters. This should result in a better F-measure. On the other hand, when a large amount of training data is available these transductions remove some of the discriminating features, and therefore must affect the performance adversely.

TABLE I
DATA CHARACTERISTICS USED IN THE EXPERIMENTS FOR TWO APPLICATIONS, T_1 FROM TELECOMMUNICATIONS DOMAIN AND T_2 FROM PHARMACEUTICAL DOMAIN

	App. T_1	App. T_2
Training Data Size	9094	29561
Test Data Size	5171	5537
Number of Call-Types	84	97
Call-Type Perplexity	32.64	32.81
Vocabulary Size (words)	2821	5350
Language Perplexity	25.06	15.87
Average Length	10.66	10.13
ASR Word Accuracy	69.90	73.80

Contrary to our intuition, Fig. 6 shows that synonym transduction has no effect when the training data size is small.[1] This can be explained by a negligible number of synonym transductions that take place when the data size is small. However, synonym transduction does have a detrimental effect on the performance when a large amount of training data is available. This result demonstrates that a general synonym transduction is not at all useful. On the other hand, in our experience with application development [37], we have noticed that a significant performance improvement can be obtained by carefully selecting the synonyms depending on the application domain.

The performance variation due to morphological transformation, also did not conform to our intuition. The performance is un-affected when the training set is small and it improves when the training set is large. This shows that different morphological word forms do have some discriminating information and transforming them to their root form marginally improves the performance.

C. *Call Classification Results*

In order to evaluate the SLU, we carried out experiments using human-machine dialogs collected from two different VoiceTone® applications. Application T_1 is from telecommunications domain and application T_2 is from pharmaceutical domain.

Table I summarizes the amount of data used for training and testing these two applications. It also provides the total number of call-types, average utterance length, call-type perplexity, and ASR word accuracy. Call-type perplexity is computed using the prior distribution of the call-types in the training data and language perplexity is the perplexity of the test set. ASR word accuracy is obtained using an off-the-shelf acoustic model [44] and a trigram language model trained with the corresponding training data. Language perplexity is computed using the language model.

Table II shows our results using both the output of the ASR and the human transcriptions. As seen, there is a 6%–8% absolute deterioration in the performance due to the recognition errors.

[1]There are 80 semantic labels in the training data, and a corpus of 1000 examples is considered very small to represent all the syntactic forms for each of the semantic labels.

TABLE II
TCER AND F-MEASURES (F-M) USING 2 DIFFERENT APPLICATIONS, NAMELY, T_1 AND T_2 WITH ASR OUTPUT AND HUMAN TRANSCRIPTIONS. ALL NUMBERS ARE IN PERCENTAGES

	App. T_1		App. T_2	
	TCER	F-M	TCER	F-M
ASR Output	30.38	66.84	26.08	73.07
Transcription	22.07	74.42	18.46	78.79

TABLE III
PRECISION AND RECALL FOR NAMED ENTITIES DETECTION

Entity Name	Precision	Recall
13 Digit Account Number	1.0000	0.3721
Dollar Amount	0.8627	0.6486
Date	0.9750	0.5679
Language	0.9000	1.0000
Place	0.8299	0.9053
Page Number	1.0000	0.6234
Line Number	1.0000	1.0000
Rate	0.9928	0.4434
Type of Charges	0.4981	0.7390
Type of Service	0.8746	0.8216
Phone Number	0.9688	0.3333
Mode of Payment	0.9667	0.3494

D. *Named Entity Detection Results*

In Table III, precision and recall for the named entities *detection* in application T_1 are provided. As can be seen from the table, precision for most entities is around 0.9. Only the named entity "Type of Charge" had a very low precision. During the development cycle, such named entities are dropped from consideration, or their grammars are modified to achieve higher precision.

IX. CONCLUSION

In this paper, we have described the AT&T SLU system. The focus of our work has been to exploit data for creating adaptive and scalable systems. Specifically, we use machine learning techniques to build statistical semantic classification models. Statistical classifiers are robust to both variations in spoken language and errors introduced by speech recognition. However, their dependency on large volumes of data can pose serious problems if that data is not available, and changing application requirements can render the existing training data useless. We discussed the semantic representation used by the SLU to capture the intents of the speaker and not the actions performed by the system. Besides robustness to the time-varying nature of spoken language and application requirements, our representation also promotes labeling consistency and therefore, better classification models can be built. We described our extension to the learning algorithm to seamlessly integrate hand crafted classification rules with the rules learned from data. This allows the development of useful models even in the early phases of the application when little or no data is available for building robust classification models. We also described the data labeling process. Specifically, we described how active learning combined with labeling error detection methods could be used to selectively sample the training data and reduce the data labeling effort and at the same time produce better classification models.

Acknowledgment

The authors would like to thank M. Johnston and B. Renger for providing helpful comments.

References

[1] Speech-Enabled Interactive Voice Response Systems. International Engineering Consortium. [Online]. Available: http://www.iec.org/online/tutorials/speech_enabled

[2] G. James, "Challenges for spoken dialogue systems," in *Proc. IEEE Workshop Automatic Speech Recognition and Understanding*, Keystone, CO, 1999.

[3] P. J. Price, "Evaluation of spoken language systems: The ATIS domain," in *Proc. DARPA Workshop on Speech and Natural Language*, Hidden Valley, PA, Jun. 1990.

[4] *Proc. 7th Message Understanding Conf. (MUC-7)*, Fairfax, VA, Apr. 1998.

[5] R. Pieraccini, E. Tzoukermann, Z. Gorelov, J.-L. Gauvain, E. Levin, C.-H. Lee, and J. G. Wilpon, "A speech understanding system based on statistical representation of semantics," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, San Francisco, CA, Mar. 1992.

[6] S. Miller, R. Bobrow, R. Ingria, and R. Schwartz, "Hidden understanding models of natural language," in *Proc. Annu. Meeting Association for Computational Linguistics*, Las Cruces, NM, Jun. 1994.

[7] S. Seneff, "TINA: A natural language system for spoken language applications," *Computat. Linguist.*, vol. 18, no. 1, pp. 61–86, 1992.

[8] W. Ward and S. Issar, "Recent improvements in the CMU spoken language understanding system," in *Proc. ARPA HLT Workshop*, Mar. 1994, pp. 213–216.

[9] J. Dowding, J. M. Gawron, D. Appelt, J. Bear, L. Cherny, R. Moore, and D. Moran, "Gemini: A natural language system for spoken language understanding," in *Proc. ARPA Workshop on Human Language Technology*, Princeton, NJ, Mar. 1993.

[10] A. L. Gorin, G. Riccardi, and J. H. Wright, "How may I help you?," *Speech Commun.*, vol. 23, pp. 113–127, 1997.

[11] A. L. Gorin, A. Abella, T. Alonso, G. Riccardi, and J. H. Wright, "Automated natural spoken dialog," *IEEE Comput. Mag.*, vol. 35, no. 4, pp. 51–56, Apr. 2002.

[12] J. Chu-Carroll and B. Carpenter, "Vector-based natural language call routing," *Computat. Linguist.*, vol. 25, no. 3, pp. 361–388, 1999.

[13] P. Natarajan, R. Prasad, B. Suhm, and D. McCarthy, "Speech enabled natural language call routing: BBN call director," in *Proc. Int. Conf. Spoken Language Processing*, Denver, CO, Sep. 2002.

[14] J. Wright, A. Gorin, and G. Riccardi, "Automatic acquisition of salient grammar fragments for call-type classification," in *Proc. Eur. Conf. Speech Communication and Technology*, Rhodes, Greece, Sep. 1997.

[15] G. Di Fabbrizio and C. Lewis, "Florence: A dialogue manager framework for spoken dialogue systems," in *Proc. 8th Int. Conf. Spoken Language Processing*, Jeju, Korea, Oct. 4–8, 2004.

[16] J. B. Lowe, C. F. Baker, and C. J. Fillmore, "A frame-semantic approach to semantic annotation," in *Proc. ACL – SIGLEX Workshop*, Washington, DC, Apr. 1997.

[17] M. Surdeanu, S. Harabagiu, J. Williams, and P. Aarseth, "Using predicate-argument structures for information extraction," in *Proc. Annu. Meeting Association Computational Linguistics*, Sapporo, Japan, 2003.

[18] J. L. Austin, *How to Do Things with Words*. Cambridge, MA: Harvard Univ. Press, 1962.

[19] G. Riccardi and A. L. Gorin, "Spoken language adaptation over time and state in a natural spoken dialog system," *IEEE Trans. Speech Audio Process.*, vol. 8, no. 1, pp. 3–10, Jan. 2000.

[20] A. Jönsson and N. Dahlbäack, "Talking to a computer is not like talking to your best friend," in *Proc. 1st Scandinavian Conf. Artificial Intelligence*, Tromsø, France, Mar. 1988.

[21] L. Begeja, B. Renger, D. Gibbon, Z. Liu, and B. Shahraray, "Interactive machine learning techniques for improving SLU models," in *Proc. HLT/NAACL Workshop on Spoken Language Understanding for Conversational Systems*, Boston, MA, May 2004.

[22] D. Hakkani-Tür, G. Tur, M. Rahim, and G. Riccardi, "Unsupervised and active learning in automatic speech recognition for call classification," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Montreal, Canada, May 2004.

[23] G. Tur, R. E. Schapire, and D. Hakkani-Tür, "Active learning for spoken language understanding," in *Proc. IEEE Int. Conf. Acoustics, Sp eech, and Signal Processing*, Hong Kong, May 2003.

[24] G. Tur and D. Hakkani-Tür, "Exploiting unlabeled utterances for spoken language understanding," in *Proc. Eur. Conf. Speech Communication and Technology*, Geneva, Switzerland, Sep. 2003.

[25] G. Tur, M. Rahim, and D. Hakkani-Tür, "Active labeling for spoken language understanding," in *Proc. Eur. Conf. Speech Communication and Technology*, Geneva, Switzerland, Sep. 2003.

[26] J. Allen *et al.*, "The TRAINS project: A case study in building a conversational planning agent," *J. Exp. Theor. Artif. Intell.*, vol. 7, pp. 7–48, 1995.

[27] H. Alshawi, D. Carter, R. Crouch, S. Pullman, M. Rayner, and A. Smith, *CLARE – A Contextual Reasoning and Cooperative Response Framework for the Core Language Engine*. Cambridge, U.K.: SRI International, 1992.

[28] 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, "The AT&T-DARPA communicator mixed-initiative spoken dialog system," in *Proc. Int. Conf. Spoken Language Processing*, Beijing, China, Oct. 2000.

[29] M. Rahim, G. Di Fabbrizio, C. Kamm, M. Walker, A. Pokrovsky, P. Ruscitti, E. Levin, S. Lee, A. Syrdal, and K. Schlosser, "VOICE-IF: A mixed-initiative spoken dialogue system for AT&T conference services," in *Proc. Eur. Conf. Speech Communication and Technology*, Aalborg, Denmark, Sep. 2001.

[30] Y.-Y. Wang, A. Acero, C. Chelba, B. Frey, and L. Wong, "Combination of statistical and rule-based approaches for spoken language understanding," in *Proc. Int. Conf. Spoken Language Processing*, Denver, CO, Sep. 2002.

[31] Y. He and S. Young, "Robustness issues in a data-driven spoken language understanding system," in *Proc. HLT/NAACL04 Workshop on Spoken Language Understanding for Conversational Systems*, Boston, MA, 2004.

[32] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann, 1993.

[33] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.

[34] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowl. Discovery*, vol. 2, no. 2, 1998.

[35] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[36] Y. D. Rubinstein and T. Hastie, "Discriminative vs informative learning," in *Proc. Int. Conf. Knowledge Discovery and Data Mining*, Newport Beach, CA, 1997.

[37] G. Di Fabbrizio, D. Dutton, N. Gupta, B. Hollister, M. Rahim, G. Riccardi, R. Schapire, and J. Schroeter, "AT&T help desk," in *Proc. Int. Conf. Spoken Language Processing*, Denver, CO, Sep. 2002.

[38] P. Haffner, G. Tur, and J. Wright, "Optimizing SVM's for complex call classification," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Hong Kong, Apr. 2003.

[39] R. E. Schapire, M. Rochery, M. Rahim, and N. Gupta, "Boosting with prior knowledge for call classification," *IEEE Trans. Speech Audio Process.*, vol. 13, no. 2, pp. 174–181, Mar. 2004.

[40] M. Walker, D. Litman, C. Kamm, and A. Abella, "PARADISE: A framework for evaluating spoken dialogue agents," in *Proc. 35th Annu. Meeting Association of Computational Linguistics*, 1997.

[41] F. Bechet, A. L. Gorin, J. H. Wright, and D. Hakkani-Tür, "Detecting and extracting named entities from spontaneous speech in a mixed initiative spoken dialogue context: How may I help you?," *Speech Commun.*, vol. 42/2, pp. 207–225, 2004.

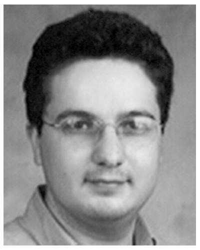
[42] G. Salton, *The SMART Information Retrieval System*. Englewood Cliffs, NJ: Prentice-Hall, 1971.

[43] ——, *Automatic Text Processing-The Transformation, Analysis and Retrieval of Information by Computer*. Reading, MA: Addison-Wesley, 1989.

[44] A. Ljolje, "Multiple task-domain acoustic models," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Hong Kong, Apr. 2003.

Narendra Gupta received the B.Eng. degree form University of Roorkee, India and the M.A.Sc. degree from the University of Waterloo, Canada.

From 1972 to 1983, he worked in various capacities with Siemens India Ltd. and with Siemens A.G. in Germany. From 1985 to 1995, he worked as a Senior Member of Technical Staff with Siemens Corporate Research, Princeton, NJ. During this period, he worked and published on topics like design automation, expert systems, equipment diagnosis systems and information retrieval. In 1995, he joined AT&T Bell Laboratories, Florham Park, NJ. His research interest is in natural language understanding and he is the principal contributor to the spoken language understanding module of the AT&T VoiceTone® Service.



Gokhan Tur was born in Ankara, Turkey in 1972. He received the B.S., M.S., and Ph.D. degrees from the Department of Computer Science, Bilkent University, Turkey in 1994, 1996, and 2000, respectively.

From 1997 to 1999, he visited the Center for Machine Translation, CMU, then the Department of Computer Science, The Johns Hopkins University, and then the Speech Technology and Research Lab of SRI International. He joined AT&T Laboratories–Research as a Senior Technical Staff Member in 2001. His research interests include speech and language processing, machine learning, and information retrieval and extraction. He is currently working on spoken language understanding for spoken dialog systems, such as the AT&T VoiceTone® System. His work has been published in several refereed journals, and presented at more than 25 international conferences.

Dr. Tur is a member of ACL and ISCA.

Dilek Hakkani-Tür (M'00) received the B.Sc. degree in computer engineering from Middle East Technical University in 1994, and the M.Sc. and Ph.D. degrees from the Department of Computer Engineering, Bilkent University, in 1996 and 2000, respectively. Her Ph.D. thesis is on statistical language modeling for agglutinative languages.

She worked on machine translation during her visit to the Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA, in 1997, and to the Computer Science Department, Johns Hopkins University, Baltimore, MD, in 1998. In 1998 and 1999, she visited the Speech Technology and Research Labs, SRI International, and worked on lexical and prosodic information for information extraction from speech. Her research interests include natural language and speech processing, spoken dialog systems, and machine learning. She co-authored more than 20 papers in natural language and speech processing. She is a Senior Technical Staff Member, AT&T Laboratories–Research, Florham Park, NJ.

Dr. Hakkani-Tür is a Member of the Association for Computational Linguistics.

Srinivas Bangalore received the Ph.D. degree in computational linguistics from University of Pennsylvania, Philadelphia, PA, in 1997. His dissertation was awarded the Morris and Dorothy Rubinoff award for outstanding dissertation that has resulted in or could lead to innovative applications of computer technology.

He is currently a Principal Technical Staff Member in the Voice Enabled Services Laboratory at AT&T Laboratories-Research, Florham Park, NJ. He has been at AT&T Labs.-Research since 1997 and has worked on many areas of language processing including spoken language translation, multimodal understanding and language generation issues. He has been an editorial board member of *Computational Linguistics Journal* from 2001 to 2003.

Giuseppe Riccardi (M'96–SM'04) received the Laurea degree in electrical engineering and Master degree in information technology, in 1991, from the University of Padua and CEFRIEL Research Center, respectively. In 1995, he received the Ph.D. degree in electrical engineering from the Department of Electrical Engineering, University of Padua.

He is currently with the Computer Science Department, University of Trento, Trento, Italy. From 1990 to 1993, he collaborated with Alcatel-Telettra Research Laboratories Milan, Italy, and investigated algorithms for speech and audio coding for medium-low bit rates for the half-rate GSM standard (New European Digital Trunking System). In 1993, he spent two years as a Research Fellow at AT&T Bell Laboratories investigating automata learning for stochastic language modeling for speech recognition and understanding. In 1996, he joined AT&T Labs-Research. His research on stochastic finite state machines for speech and language processing has been applied to a wide range of domains for task automation. He participated at the creation of the state-of-the-art AT&T spoken language system used in the 1994 DARPA ATIS evaluation. He has been pioneering the speech and language research in spontaneous speech within the How May I Help You? research program. His research on learning finite state automata and transducers has lead to the creation of the first large scale finite state chain decoding for machine translation (Anuvaad). He has coauthored more than 60 papers in the field of speech and audio coding, speech recognition and understanding and machine translation. His current research interests are stochastic language modeling, language understanding, spoken dialogue, language acquisition and machine translation. He is on the Editorial Board of the ACM *Transactions of Speech and Language*.

Dr. Riccardi has been on the scientific committees of EUROSPEECH, ICASSP, and ACL. He co-organized the IEEE ASRU Workshop in 1993, 1999, and 2001. He is the Guest Editor of the IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING Special Issue on Speech-to-Speech Machine Translation. He is a Senior Member of ACL and the New York Academy of Science.

Mazin Gilbert (SM'97) received the B.Eng. and Ph.D. degrees from the University of Liverpool, U.K.

He joined AT&T Laboratories-Research, Florham Park, NJ, in 1990, performing research in the area of articulatory speech synthesis and was later appointed a Research Professor at Rutgers University, New Brunswick, NJ, where he was engaged in research in the area of neural networks for speech and speaker recognition. In 1993, he joined Bell Labs as a Technical Staff Member pursuing research in noise robustness, acoustic modeling and utterance verification for automatic speech recognition. He is currently the Director of the Natural Language Processing Division at AT&T Labs Research. The focus of his division is to perform research on advanced algorithms and methods for creation of intelligent and complex natural-language dialog agents, and to support AT&T customers and external partners. The division specializes in areas of signal processing, acoustic and language modeling, natural language understanding, speech data mining, machine learning and multimodal user interface. has over seventy publications in the areas of speech and dialog and is the author of the book "Artificial Neural Networks for Speech Analysis/Synthesis" (London: Chapman and Hall). He holds 11 US patents and is a recipient of several national and international awards.

Dr. Rahim was an Associate Editor for the IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING from 1995 to 1999, and a Chair of the 1999 workshop on Automatic Speech Recognition and Understanding, ASRU'99. He is the Chair of the IEEE Speech Technical Committee and the Chair of the CAIP Industrial Board at Rutgers University.