Speech Recognition: Statistical Methods

L R Rabiner, Rutgers University, New Brunswick, NJ, USA and University of California, Santa Barbara, CA, USA
B-H Juang, Georgia Institute of Technology, Atlanta, GA, USA

Introduction

The goal of getting a machine to understand fluently spoken speech and respond in a natural voice has been driving speech research for more than 50 years. Although the personification of an intelligent machine such as HAL in the movie *2001, A Space Odyssey*, or R2D2 in the *Star Wars* series, has been around for more than 35 years, we are still not yet at the point where machines reliably understand fluent speech, spoken by anyone, and in any acoustic environment. In spite of the remaining technical problems that need to be solved, the fields of automatic speech recognition and understanding have made tremendous advances and the technology is now readily available and used on a day-to-day basis in a number of applications and services – especially those conducted over the public-switched telephone network (PSTN) (Cox *et al*., 2000). This article aims at reviewing the technology that has made these applications possible.

Speech recognition and language understanding are two major research thrusts that have traditionally been approached as problems in linguistics and acoustic phonetics, where a range of acoustic phonetic knowledge has been brought to bear on the problem with remarkably little success. In this article, however, we focus on statistical methods for speech and language processing, where the knowledge about a speech signal and the language that it expresses, together with practical uses of the knowledge, is developed from actual realizations of speech data through a well-defined mathematical and statistical formalism. We review how the statistical methods are used for speech recognition and language understanding, show current performance on a number of task-specific applications and services, and discuss the challenges that remain to be solved before the technology becomes ubiquitous.

The Speech Advantage

There are fundamentally three major reasons why so much research and effort has gone into the problem of trying to teach machines to recognize and understand fluent speech, and these are the following:

- Cost reduction. Among the earliest goals for speech recognition systems was to replace humans performing certain simple tasks with automated machines, thereby reducing labor expenses while still providing customers with a natural and convenient way to access information and services. One simple example of a cost reduction system was the Voice Recognition Call Processing (VRCP) system introduced by AT&T in 1992 (Roe *et al*., 1996), which essentially automated so-called operator-assisted calls, such as person-to-person calls, reverse-billing calls, third-party billing calls, collect calls (by far the most common class of such calls), and operator-assisted calls. The resulting automation eliminated about 6600 jobs, while providing a quality of service that matched or exceeded that provided by the live attendants, saving AT&T on the order of $300 million per year.
- New revenue opportunities. Speech recognition and understanding systems enabled service providers to have a 24/7 high-quality customer care automation capability, without the need for access to information by keyboard or touch-tone button pushes. An example of such a service was the How May I Help You (HMIHY)© service introduced by AT&T late in 2000 (Gorin *et al*., 1996), which automated the customer care for AT&T Consumer Services. This system will be discussed further in the section on speech understanding. A second example of such a service was the NTT ANSER service for voice banking in Japan [Sugamura *et al*., 1994], which enabled Japanese banking customers to access bank account records from an ordinary telephone without having to go to the bank. (Of course, today we utilize the Internet for such information, but in 1981, when this system was introduced, the only way to access such records was a physical trip to the bank and a wait in lines to speak to a banking clerk.)
- Customer retention. Speech recognition provides the potential for personalized services based on customer preferences, and thereby the potential to improve the customer experience. A trivial example of such a service is the voice-controlled automotive environment that recognizes the identity of the driver from voice commands and adjusts the automobile's features (seat position, radio station, mirror positions, etc.) to suit the customer's preference (which is established in an enrollment session).

The Speech Dialog Circle

When we consider the problem of communicating with a machine, we must consider the cycle of events that occurs between a spoken utterance (as part of

a dialog between a person and a machine) and the response to that utterance from the machine. Figure 1 shows such a sequence of events, which is often referred to as the speech dialog circle, using an example in the telecommunications context.

The customer initially makes a request by speaking an utterance that is sent to a machine, which attempts to recognize, on a word-by-word basis, the spoken speech. The process of recognizing the words in the speech is called automatic speech recognition (ASR) and its output is an orthographic representation of the recognized spoken input. The ASR process will be discussed in the next section. Next the spoken words are analyzed by a spoken language understanding (SLU) module, which attempts to attribute meaning to the spoken words. The meaning that is attributed is in the context of the task being handled by the speech dialog system. (What is described here is traditionally referred to as a limited domain understanding system or application.) Once meaning has been determined, the dialog management (DM) module examines the state of the dialog according to a prescribed operational workflow and determines the course of action that would be most appropriate to take. The action may be as simple as a request for further information or confirmation of an action that is taken. Thus if there were confusion as to how best to proceed, a text query would be generated by the spoken language generation module to hopefully clarify the meaning and help determine what to do next. The query text is then sent to the final module, the text-to-speech synthesis (TTS) module, and then converted into intelligible and highly natural speech, which is sent to the customer who decides what to say next based on what action was taken, or based on previous dialogs with the machine. All of the modules in the speech dialog circle can be 'data-driven' in both the learning and active use phases, as indicated by the central Data block in Figure 1.

A typical task scenario, e.g., booking an airline reservation, requires navigating the speech dialog circle many times – each time being referred to as one 'turn' – to complete a transaction. (The average number of turns a machine takes to complete a prescribed task is a measure of the effectiveness of the machine in many applications.) Hopefully, each time through the dialog circle enables the customer to get closer to the desired action either via proper understanding of the spoken request or via a series of clarification steps. The speech dialog circle is a powerful concept in modern speech recognition and understanding systems, and is at the heart of most speech understanding systems that are in use today.

Basic ASR Formulation

The goal of an ASR system is to accurately and efficiently convert a speech signal into a text message transcription of the spoken words, independent of the device used to record the speech (i.e., the

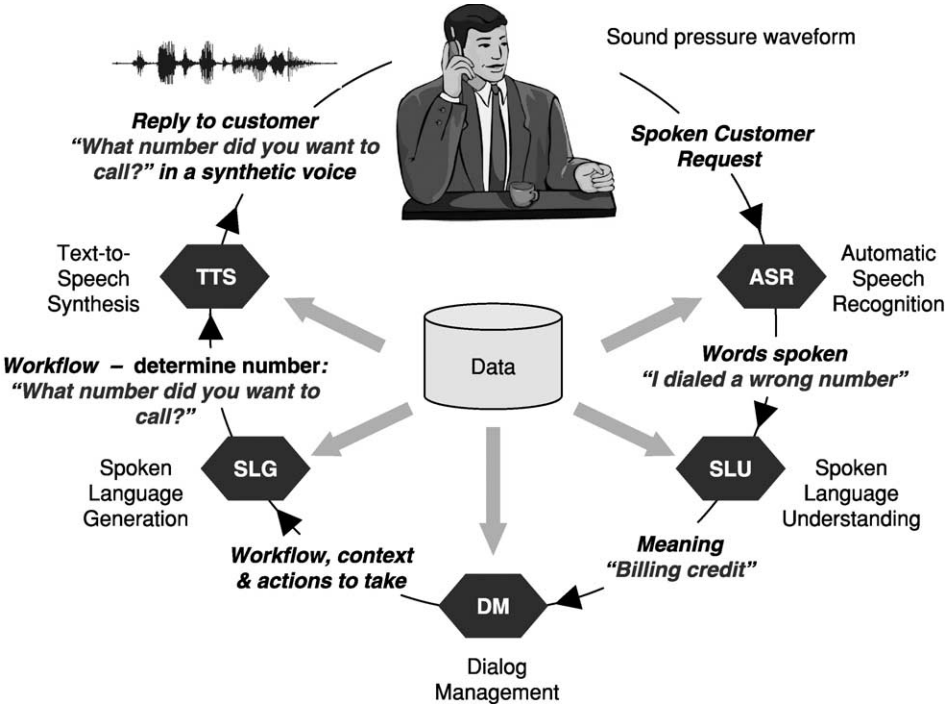

Figure 1 The conventional speech dialog circle.

transducer or microphone), the speaker, or the environment. A simple model of the speech generation process, as used to convey a speaker's intention is shown in Figure 2.

It is assumed that the speaker decides what to say and then embeds the concept in a sentence, W, which is a sequence of words (possibly with pauses and other acoustic events such as uh's, um's, er's, etc.). The speech production mechanisms then produce a speech waveform, $s(n)$, which embodies the words of W as well as the extraneous sounds and pauses in the spoken input. A conventional automatic speech recognizer attempts to decode the speech, $s(n)$, into the best estimate of the sentence, $\hat{W}$, using a two-step process, as shown in Figure 3.

The first step in the process is to convert the speech signal, $s(n)$, into a sequence of spectral feature vectors, X, where the feature vectors are measured every 10 ms (or so) throughout the duration of the speech signal. The second step in the process is to use a syntactic decoder to generate every possible valid sentence (as a sequence of orthographic representations) in the task language, and to evaluate the score (i.e., the *a posteriori* probability of the word string given the realized acoustic signal as measured by the feature vector) for each such string, choosing as the recognized string, $\hat{W}$, the one with the highest score. This is the so-called maximum *a posteriori* probability (MAP) decision principle, originally suggested by Bayes. Additional linguistic processing can be done to try to determine side information about the speaker, such as the speaker's intention, as indicated in Figure 3.

Mathematically, we seek to find the string $\hat{W}$ that maximizes the *a posteriori* probability of that string, when given the measured feature vector X, i.e.,

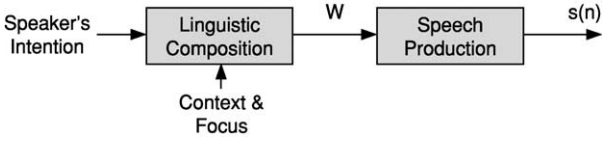

Figure 2 Model of spoken speech.

$$\hat{W} = \arg\max_{W} P(W|X)$$

Using Bayes Law, we can rewrite this expression as:

$$\hat{W} = \arg\max_{W} \frac{P(X|W)P(W)}{P(X)}$$

Thus, calculation of the *a posteriori* probability is decomposed into two main components, one that defines the *a priori* probability of a word sequence W, $P(W)$, and the other the likelihood of the word string W in producing the measured feature vector, $P(X|W)$. (We disregard the denominator term, $P(X)$, since it is independent of the unknown W). The latter is referred to as the acoustic model, $P_A(X|W)$, and the former the language model, $P_L(W)$ (Rabiner *et al.*, 1996; Gauvain and Lamel, 2003). We note that these quantities are not given directly, but instead are usually estimated or inferred from a set of training data that have been labeled by a knowledge source, i.e., a human expert. The decoding equation is then rewritten as:

$$\hat{W} = \arg\max_{W} P_A(X|W)P_L(W)$$

We explicitly write the sequence of feature vectors (the acoustic observations) as:

$$X = \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$$

where the speech signal duration is N frames (or N times 10 ms when the frame shift is 10 ms). Similarly, we explicitly write the optimally decoded word sequence as:

$$\hat{W} = w_1 w_2 \ldots w_M$$

where there are M words in the decoded string. The above decoding equation defines the fundamental statistical approach to the problem of automatic speech recognition.

It can be seen that there are three steps to the basic ASR formulation, namely:

- Step 1: acoustic modeling for assigning probabilities to acoustic (spectral) realizations of a sequence

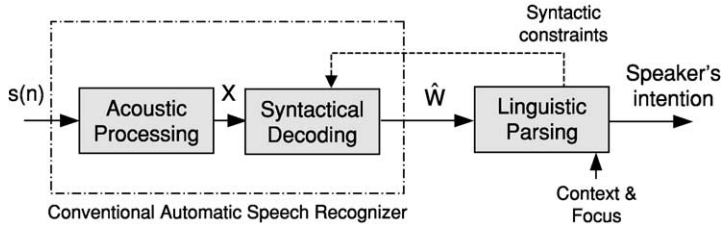

Figure 3 ASR decoder from speech to sentence.

of words. For this step we use a statistical model (called the hidden Markov model or HMM) of the acoustic signals of either individual words or subword units (e.g., phonemes) to compute the quantity $P_A(X|W)$. We train the acoustic models from a training set of speech utterances, which have been appropriately labeled to establish the statistical relationship between X and W.

- Step 2: language modeling for assigning probabilities, $P_L(W)$, to sequences of words that form valid sentences in the language and are consistent with the recognition task being performed. We train such language models from generic text sequences, or from transcriptions of task-specific dialogues. (Note that a deterministic grammar, as is used in many simple tasks, can be considered a degenerate form of a statistical language model. The 'coverage' of a deterministic grammar is the set of permissible word sequences, i.e., expressions that are deemed legitimate.)
- Step 3: hypothesis search whereby we find the word sequence with the maximum *a posteriori* probability by searching through all possible word sequences in the language.

In step 1, acoustic modeling (Young, 1996; Rabiner *et al*., 1986), we train a set of acoustic models for the words or sounds of the language by learning the statistics of the acoustic features, X, for each word or sound, from a speech training set, where we compute the variability of the acoustic features during the production of the words or sounds, as represented by the models. For large vocabulary tasks, it is impractical to create a separate acoustic model for every possible word in the language since it requires far too much training data to measure the variability in every possible context. Instead, we train a set of about 50 acoustic-phonetic subword models for the approximately 50 phonemes in the English language, and construct a model for a word by concatenating (stringing together sequentially) the models for the constituent subword sounds in the word, as defined in a word lexicon or dictionary, where multiple pronunciations are allowed). Similarly, we build sentences (sequences of words) by concatenating word models. Since the actual pronunciation of a phoneme may be influenced by neighboring phonemes (those occurring before and after the phoneme), the set of so-called context-dependent phoneme models are often used as the speech models, as long as sufficient data are collected for proper training of these models.

In step 2, the language model (Jelinek, 1997; Rosenfeld, 2000) describes the probability of a sequence of words that form a valid sentence in the task language. A simple statistical method works well, based on a Markovian assumption, namely that the probability of a word in a sentence is conditioned on **only** the previous N–1 words, namely an N-gram language model, of the form:

$$P_L(W) = P_L(w_1, w_2, \ldots, w_M) = \prod_{m=1}^{M} P_L(w_m | w_{m-1}, w_{m-2}, \ldots, w_{m-N+1})$$

where $P_L(w_m|w_{m-1}, w_{m-2}, \ldots, w_{m-N+1})$ is estimated by simply counting up the relative frequencies of N-tuples in a large corpus of text.

In step 3, the search problem (Ney, 1984; Paul, 2001) is one of searching the space of all valid sound sequences, conditioned on the word grammar, the language syntax, and the task constraints, to find the word sequence with the maximum likelihood. The size of the search space can be astronomically large and take inordinate amounts of computing power to solve by heuristic methods. The use of methods from the field of Finite State Automata Theory provide finite state networks (FSNs) (Mohri, 1997), along with the associated search policy based on dynamic programming, that reduce the computational burden by orders of magnitude, thereby enabling exact solutions in computationally feasible times, for large speech recognition problems.

Development of a Speech Recognition System for a Task or an Application

Before going into more detail on the various aspects of the process of automatic speech recognition by machine, we review the three steps that must occur in order to define, train, and build an ASR system (Juang *et al*., 1995; Kam and Helander, 1997). These steps are the following:

- Step 1: choose the recognition task. Specify the word vocabulary for the task, the set of units that will be modeled by the acoustic models (e.g., whole words, phonemes, etc.), the word pronunciation lexicon (or dictionary) that describes the variations in word pronunciation, the task syntax (grammar), and the task semantics. By way of example, for a simple speech recognition system capable of recognizing a spoken credit card number using isolated digits (i.e., single digits spoken one at a time), the sounds to be recognized are either whole words or the set of subword units that appear in the digits /zero/ to /nine/ plus the word /oh/. The word vocabulary is the set of 11 digits. The task syntax allows any single digit to be spoken, and the task

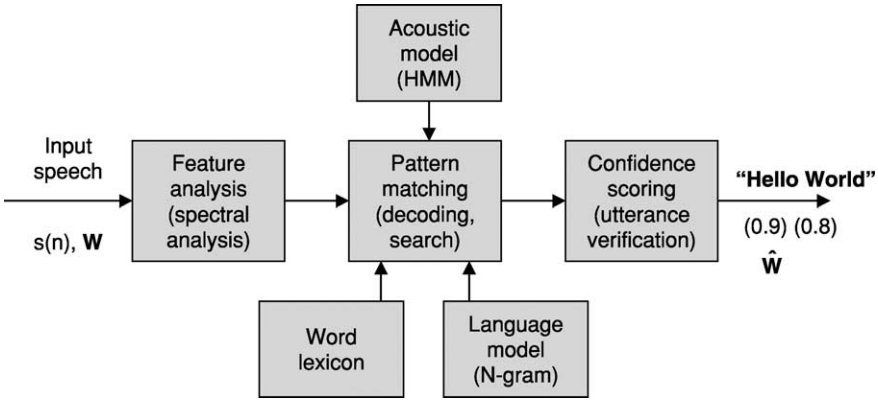

Figure 4 Framework of ASR system.

semantics specify that a sequence of isolated digits must form a valid credit card code for identifying the user.

- Step 2: train the models. Create a method for building acoustic word models (or subword models) from a labeled speech training data set of multiple occurrences of each of the vocabulary words by one or more speakers. We also must use a text training data set to create a word lexicon (dictionary) describing the ways that each word can be pronounced (assuming we are using subword units to characterize individual words), a word grammar (or language model) that describes how words are concatenated to form valid sentences (i.e., credit card numbers), and finally a task grammar that describes which valid word strings are meaningful in the task application (e.g., valid credit card numbers).
- Step 3: evaluate recognizer performance. We need to determine the word error rate and the task error rate for the recognizer on the desired task. For an isolated digit recognition task, the word error rate is just the isolated digit error rate, whereas the task error rate would be the number of credit card errors that lead to misidentification of the user. Evaluation of the recognizer performance often includes an analysis of the types of recognition errors made by the system. This analysis can lead to revision of the task in a number of ways, ranging from changing the vocabulary words or the grammar (i.e., to eliminate highly confusable words) to the use of word spotting, as opposed to word transcription. As an example, in limited vocabulary applications, if the recognizer encounters frequent confusions between words like 'freight' and 'flight,' it may be advisable to change 'freight' to 'cargo' to maximize its distinction from 'flight.' Revision of the task grammar often becomes necessary if the recognizer experiences substantial amounts of what is called 'out of grammar' (OOG) utterances, namely the use of words and phrases that are not directly included in the task vocabulary (ISCA, 2001).

The Speech Recognition Process

In this section, we provide some technical aspects of a typical speech recognition system. Figure 4 shows a block diagram of a speech recognizer that follows the Bayesian framework discussed above.

The recognizer consists of three processing steps, namely feature analysis, pattern matching, and confidence scoring, along with three trained databases, the set of acoustic models, the word lexicon, and the language model. In this section, we briefly describe each of the processing steps and each of the trained model databases.

Feature Analysis

The goal of feature analysis is to extract a set of salient features that characterize the spectral properties of the various speech sounds (the subword units) and that can be efficiently measured. The 'standard' feature set for speech recognition is a set of mel-frequency cepstral coefficients (MFCCs) (which perceptually match some of the characteristics of the spectral analysis done in the human auditory system) (Davis and Mermelstein, 1980), along with the first- and second-order derivatives of these features. Typically about 13 MFCCs and their first and second derivatives (Furai, 1981) are calculated every 10 ms, leading to a spectral vector with 39 coefficients every 10 ms. A block diagram of a typical feature analysis process is shown in Figure 5.

The speech signal is sampled and quantized, pre-emphasized by a first-order (highpass) digital filter with pre-emphasis factor α (to reduce the influence of glottal coupling and lip radiation on the estimated

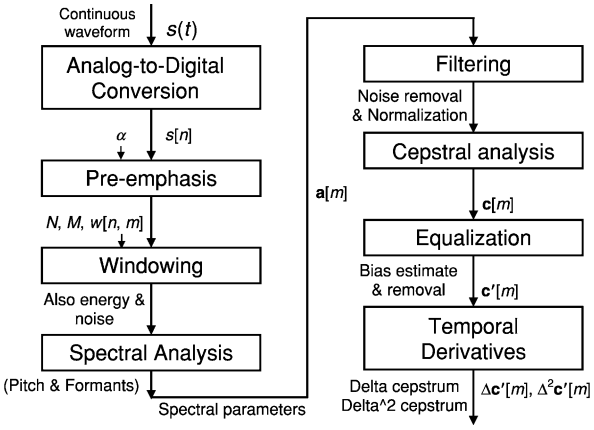

Figure 5 Block diagram of feature analysis computation.

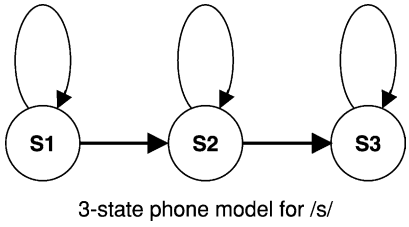

Figure 6 Three-state HMM for the sound /s/.

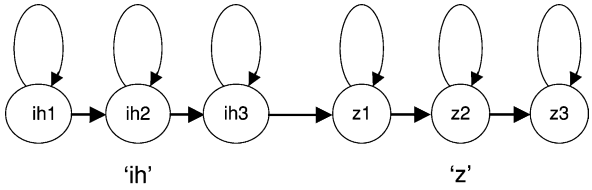

Figure 7 Concatenated model for the word 'is.'

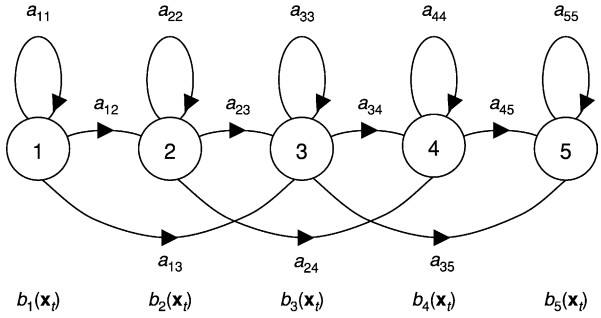

Figure 8 HMM for whole word model with five states.

vocal tract characteristics), segmented into frames, windowed, and then a spectral analysis is performed using a fast Fourier transform (FFT) (Rabiner and Gold, 1975) or linear predictive coding (LPC) method (Atal and Hanauer, 1971; Markel and Gray, 1976). The frequency conversion from a linear frequency scale to a mel frequency scale is performed in the filtering block, followed by cepstral analysis yielding the MFCCs (Davis and Mermelstein, 1980), equalization to remove any bias and to normalize the cepstral coefficients (Rahim and Juang, 1996), and finally the computation of first- and second-order (via temporal derivative) MFCCs is made, completing the feature extraction process.

Acoustic Models

The goal of acoustic modeling is to characterize the statistical variability of the feature set determined above for each of the basic sounds (or words) of the language. Acoustic modeling uses probability measures to characterize sound realization using statistical models. A statistical method, known as the hidden Markov model (HMM) (Levinson *et al.*, 1983; Ferguson, 1980; Rabiner, 1989; Rabiner and Juang, 1985), is used to model the spectral variability of each of the basic sounds of the language using a mixture density Gaussian distribution (Juang *et al.*, 1986; Juang, 1985), which is optimally aligned with a speech training set and iteratively updated and improved (the means, variances, and mixture gains are iteratively updated) until an optimal alignment and match is achieved.

Figure 6 shows a simple three-state HMM for modeling the subword unit /s/ as spoken at the beginning of the word /six/. Each HMM state is characterized by a probability density function (usually a mixture Gaussian density) that characterizes the statistical behavior of the feature vectors at the beginning (state s1), middle (state s2), and end (state s3) of the sound /s/. In order to train the HMM for each subword unit, we use a labeled training set of words and sentences and utilize an efficient training procedure known as the Baum-Welch algorithm (Rabiner, 1989; Baum, 1972; Baum *et al.*, 1970) to align each of the various subword units with the spoken inputs, and then estimate the appropriate means, covariances, and mixture gains for the distributions in each subword unit state. The algorithm is a hill-climbing algorithm and is iterated until a stable alignment of subword unit models and speech is obtained, enabling the creation of stable models for each subword unit.

Figure 7 shows how a simple two-sound word, 'is,' which consists of the sounds /ih/ and /z/, is created by concatenating the models (Lee, 1989) for the /ih/ sound with the model for the /z/ sound, thereby creating a six-state model for the word 'is.'

Figure 8 shows how an HMM can be used to characterize a whole-word model (Lee *et al.*, 1989). In this

case, the word is modeled as a sequence of $M = 5$ HMM states, where each state is characterized by a mixture density, denoted as $b_j(\mathbf{x}_t)$ where the model state is the index j, the feature vector at time t is denoted as $\mathbf{x}_t$, and the mixture density is of the form:

$$b_j(\mathbf{x}_t) = \sum_{k=1}^{K} c_{jk}\mathcal{N}[\mathbf{x}_t, \mu_{jk}, U_{jk}]$$

$$\mathbf{x}_t = (x_{t1}, x_{t2}, \ldots, x_{tD}),\ D = 39$$

$K =$ number of mixture components in the density function

$c_{jk} =$ weight of kth mixture component in state $j, c_{jk} \geq 0$

$\mathcal{N} =$ Gaussian density function

$\mu_{jk} =$ mean vector for mixture k, state j

$U_{jk} =$ covariance matrix for mixture k, state j

$$\sum_{k=1}^{K} c_{jk} = 1, \quad 1 \leq j \leq M$$

$$\int_{-\infty}^{\infty} b_j(\mathbf{x}_t)d\mathbf{x}_t = 1, \quad 1 \leq j \leq M$$

Included in Figure 8 is an explicit set of state transitions, a_{ij}, which specify the probability of making a transition from state i to state j at each frame, thereby defining the time sequence of the feature vectors over the duration of the word. Usually the self-transitions, a_{ii}, are large (close to 1.0), and the skip-state transitions, a_{13}, a_{24}, a_{35}, are small (close to 0).

Once the set of state transitions and state probability densities are specified, we say that a model λ (which is also used to denote the set of parameters that define the probability measure) has been created for the word or subword unit. (The model λ is often written as $\lambda(A, B, \pi)$ to explicitly denote the model parameters, namely $A = \{a_{ij}, 1 \leq i, j \leq M\}$, which is the state transition matrix, $B = \{b_j(x_t), 1 \leq j \leq M\}$, which is the state observation probability density, and $\pi = \{\pi_i, 1 \leq i \leq M\}$, which is the initial state distribution). In order to optimally train the various models (for each word unit [Lee *et al.*, 1989] or subword unit [Lee, 1989]), we need to have algorithms that perform the following three steps or tasks (Rabiner and Juang, 1985) using the acoustic observation sequence, X, and the model λ:

a. likelihood evaluation: compute $P(X|\lambda)$
b. decoding: choose the optimal state sequence for a given speech utterance
c. re-estimation: adjust the parameters of λ to maximize $P(X|\lambda)$.

Each of these three steps is essential to defining the optimal HMM models for speech recognition based on the available training data and each task if approached in a brute force manner would be computationally costly. Fortunately, efficient algorithms have been developed to enable efficient and accurate solutions to each of the three steps that must be performed to train and utilize HMM models in a speech recognition system. These are generally referred to as the forward-backward algorithm or the Baum-Welch re-estimation method (Levinson *et al.*, 1983). Details of the Baum-Welch procedure are beyond the scope of this article. The heart of the training procedure for re-estimating model parameters using the Baum-Welch procedure is shown in Figure 9.

Recently, the fundamental statistical method, while successful for a range of conditions, has been augmented with a number of techniques that attempt to further enhance the recognition accuracy and make the recognizer more robust to different talkers, background noise conditions, and channel effects. One family of such techniques focuses on transformation of the observed or measured features. The transformation is motivated by the need for vocal tract length normalization (e.g., reducing the impact of differences in vocal tract length of various speakers). Another such transformation (called the maximum likelihood linear regression method) can be embedded in the statistical model to account for a potential mismatch between the statistical characteristics of the training data and the actual unknown utterances to be recognized. Yet another family of techniques (e.g., the discriminative training method based on minimum classification error [MCE] or maximum mutual information [MMI]) aims at direct minimization of the recognition error during the parameter optimization stage.

Word Lexicon

The purpose of the word lexicon, or dictionary, is to define the range of pronunciation of words in the task vocabulary (Jurafsky and Martin, 2000; Riley *et al.*, 1999). The reason that such a word lexicon is necessary is because the same orthography can be pronounced differently by people with different accents, or because the word has multiple meanings that change the pronunciation by the context of its use. For example, the word 'data' can be pronounced as: /d/ /ae/ /t/ /ax/ or as /d/ /ey/ /t/ /ax/, and we would need

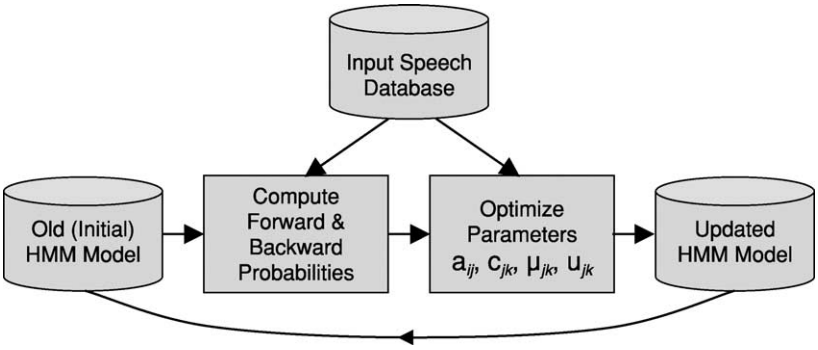

Figure 9 The Baum-Welch training procedure.

both pronunciations in the dictionary to properly train the recognizer models and to properly recognize the word when spoken by different individuals. Another example of variability in pronunciation from orthography is the word 'record,' which can be either a disk that goes on a player, or the process of capturing and storing a signal (e.g., audio or video). The different meanings have significantly different pronunciations. As in the statistical language model, the word lexicon (consisting of sequences of symbols) can be associated with probability assignments, resulting in a probabilistic word lexicon.

Language Model

The purpose of the language model (Rosenfeld, 2000; Jelinek *et al.*, 1991), or grammar, is to provide a task syntax that defines acceptable spoken input sentences and enables the computation of the probability of the word string, W, given the language model, i.e., $P_L(W)$. There are several methods of creating word grammars, including the use of rule-based systems (i.e., deterministic grammars that are knowledge-driven), and statistical methods that compute an estimate of word probabilities from large training sets of textual material. We describe the way in which a statistical N-gram word grammar is constructed from a large training set of text.

Assume we have a large text training set of labeled words. Thus for every sentence in the training set, we have a text file that identifies the words in that sentence. If we consider the class of N-gram word grammars, then we can estimate the word probabilities from the labeled text training set using counting methods. Thus to estimate word trigram probabilities (that is the probability that a word w_i was preceded by the pair of words (w_{i-1}, w_{i-2})), we compute this quantity as:

$$P(w_i|w_{i-1}, w_{i-2}) = \frac{C(w_{i-2}, w_{i-1}, w_i)}{C(w_{i-2}, w_{i-1})}$$

where $C(w_{i-2}, w_{i-1}, w_i)$ is the frequency count of the word triplet (i.e., trigram) consisting of (w_{i-2}, w_{i-1}, w_i) that occurred in the training set, and $C(w_{i-2}, w_{i-1})$ is the frequency count of the word duplet (i.e., bigram) (w_{i-2}, w_{i-1}) that occurred in the training set.

Although the method of training N-gram word grammars, as described above, generally works quite well, it suffers from the problem that the counts of N-grams are often highly in error due to problems of data sparseness in the training set. Hence for a text training set of millions of words, and a word vocabulary of several thousand words, more than 50% of word trigrams are likely to occur either once or not at all in the training set. This leads to gross distortions in the computation of the probability of a word string, as required by the basic Bayesian recognition algorithm. In the cases when a word trigram does not occur at all in the training set, it is unacceptable to define the trigram probability as 0 (as would be required by the direct definition above), since this leads to effectively invalidating all strings with that particular trigram from occurring in recognition. Instead, in the case of estimating trigram word probabilities (or similarly extended to N-grams where N is more than three), a smoothing algorithm (Bahl *et al.*, 1983) is applied by interpolating trigram, bigram, and unigram relative frequencies, i.e.,

$$\hat{P}(w_i|w_{i-1}, w_{i-2}) = p_3 \frac{C(w_{i-2}, w_{i-1}, w_i)}{C(w_{i-2}, w_{i-1})} + p_2 \frac{C(w_{i-1}, w_i)}{C(w_{i-1})} + p_1 \frac{C(w_i)}{\sum_i C(w_i)}$$

$$p_3 + p_2 + p_1 = 1$$

$$\sum_i C(w_i) = \text{size of text training corpus}$$

where the smoothing probabilities, p_3, p_2, p_1 are obtained by applying the principle of cross-validation. Other schemes such as the Turing-Good

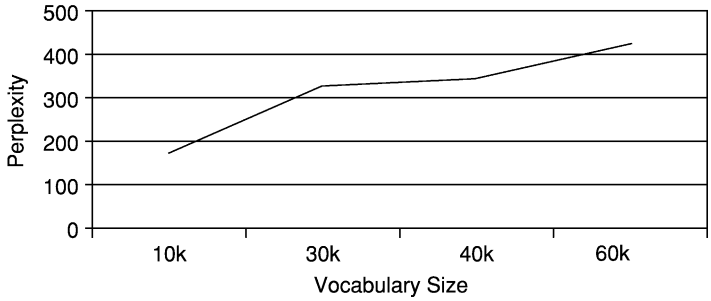

Figure 10 Bigram language perplexity for Encarta Encyclopedia.

estimator, which deals with unseen classes of observations in distribution estimation, have also been proposed (Nadas, 1985).

Worth mentioning here are two important notions that are associated with language models: perplexity of the language model and the rate of occurrences of out-of-vocabulary words in real data sets. We elaborate them below.

Language Perplexity A measure of the complexity of the language model is the mathematical quantity known as language perplexity (which is actually the geometric mean of the word branching factor, or the average number of words that follow any given word of the language) (Roukos, 1998). We can compute language perplexity, as embodied in the language model, $P_L(W)$, where $W = (w_1, w_2, \ldots, w_Q)$ is a length-Q word sequence, by first defining the entropy (Cover and Thomas, 1991) as:

$$H(W) = -\frac{1}{Q}\log_2 P(W)$$

Using a trigram language model, we can write the entropy as:

$$H(W) = -\frac{1}{Q}\sum_{i=1}^{Q}\log_2 P(w_i|w_{i-1}, w_{i-2})$$

where we suitably define the first couple of probabilities as the unigram and bigram probabilities. Note that as Q approaches infinity, the above entropy approaches the asymptotic entropy of the source defined by the measure $P_L(W)$. The perplexity of the language is then defined as:

$$PP(W) = 2^{H(W)} = P(w_1, w_2, \ldots, w_Q)^{-1/Q} \quad \text{as } Q \to \infty.$$

Some examples of language perplexity for specific speech recognition tasks are the following:

- For an 11-digit vocabulary ('zero' to 'nine' plus 'oh') where every digit can occur independently of every other digit, the language perplexity (average word branching factor) is 11.
- For a 2000-word Airline Travel Information System (ATIS) [Ward, 1991], the language perplexity (using a trigram language model) is 20 [Price, 1990].
- For a 5000-word *Wall Street Journal* task (reading articles aloud), the language perplexity (using a bigram language model) is 130 [Paul *et al*., 1992].

A plot of the bigram perplexity for a training set of 500 million words, tested on the Encarta Encyclopedia is shown in Figure 10. It can be seen that language perplexity grows only slowly with the vocabulary size and is only about 400 for a 60 000-word vocabulary. (Language perplexity is a complicated function of vocabulary size and vocabulary predictability, and is not in any way directly proportional to vocabulary size).

Out-of-Vocabulary Rate Another interesting aspect of language models is their coverage of the language as exemplified by the concept of an out-of-vocabulary (OOV) (Kawahara and Lee, 1998) rate, which measures how often a new word appears for a specific task, given that a language model of a given vocabulary size for the task has been created. Figure 11 shows the OOV rate for sentences from the Encarta Encyclopedia, again trained on 500 million words of text, as a function of the vocabulary size. It can be seen that even for a 60 000-word vocabulary, about 4% of the words that are encountered have not been seen previously and thus are considered OOV words (which, by definition, cannot be recognized correctly by the recognition system).

Pattern Matching

The job of the pattern matching module is to combine information (probabilities) from the acoustic model, the language model, and the word lexicon to find the

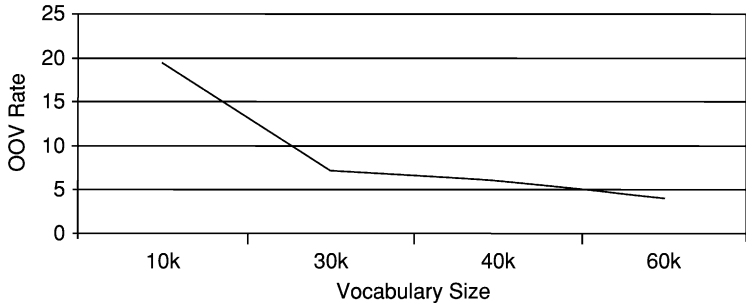

Figure 11 Out-of-vocabulary rate of Encarta Encyclopedia as a function of the vocabulary size.

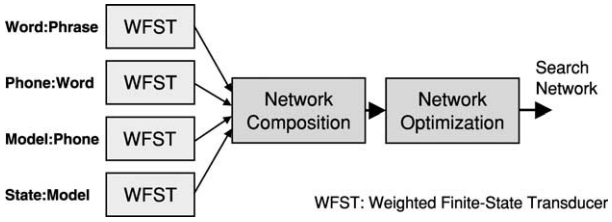

Figure 12 Use of WFSTs to compile FSN to minimize redundancy in the network.

'optimal' word sequence, i.e., the word sequence that is consistent with the language model and that has the highest probability among all possible word sequences in the language (i.e., best matches the spectral feature vectors of the input signal). To achieve this goal, the pattern matching system is actually a decoder (Ney, 1984; Paul, 2001; Mohri, 1997) that searches through all possible word strings and assigns a probability score to each string, using a Viterbi decoding algorithm (Forney, 1973) or its variants.

The challenge for the pattern matching module is to build an efficient structure (via an appropriate finite state network or FSN) (Mohri, 1997) for decoding and searching large-vocabulary complex-language models for a range of speech recognition tasks. The resulting composite FSNs represent the cross-product of the features (from the input signal), with the HMM states (for each sound), with the HMM units (for each sound), with the sounds (for each word), with the words (for each sentence), and with the sentences (those valid within the syntax and semantics of the task and language). For large-vocabulary high-perplexity speech recognition tasks, the size of the network can become astronomically large and has been shown to be on the order of 10^{22} states for some tasks. Such networks are prohibitively large and cannot be exhaustively searched by any known method or machine. Fortunately there are methods (Mohri, 1997) for compiling such large networks and reducing the size significantly due to inherent redundancies and overlaps across each of the levels of the network. (One earlier example of taking advantage of the search redundancy is the dynamic programming method (Bellman, 1957), which turns an otherwise exhaustive search problem into an incremental one.) Hence the network that started with 10^{22} states was able to be compiled down to a mathematically equivalent network of 10^{8} states that was readily searched for the optimum word string with no loss of performance or word accuracy.

The way in which such a large network can be theoretically (and practically) compiled to a much smaller network is via the method of weighted finite state transducers (WFST), which combine the various representations of speech and language and optimize the resulting network to minimize the number of search states. A simple example of such a WFST is given in Figure 12, and an example of a simple word pronunciation transducer (for two versions of the word 'data') is given in Figure 13.

Using the techniques of composition and optimization, the WFST uses a unified mathematical framework to efficiently compile a large network into a minimal representation that is readily searched using standard Viterbi decoding methods. The example of Figure 13 shows how all redundancy is removed and a minimal search network is obtained, even for as simple an example as two pronunciations of the word 'data.'

Confidence Scoring

The goal of the confidence scoring module is to post-process the speech feature set in order to identify possible recognition errors as well as out-of-vocabulary events and thereby to potentially improve the performance of the recognition algorithm. To achieve this goal, a word confidence score (Rahim *et al.*, 1997) based on a simple likelihood ratio hypothesis testing associated with each recognized word, is performed and the word confidence score is used to determine which, if any, words are likely to be incorrect because of either a recognition error or because it

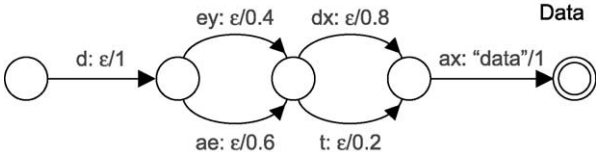

Figure 13 Word pronunciation transducer for two pronunciations of the word 'data.'

was an OOV word (that could never be correctly recognized). A simple example of a two-word phrase and the resulting confidence scores is as follows:

Spoken Input: credit please
Recognized String: credit fees
Confidence Scores: (0.9) (0.3)

Based on the confidence scores (derived using a likelihood ratio test), the recognition system would realize which word or words are likely to be in error and take appropriate steps (in the ensuing dialog) to determine whether an error had been made and how to fix it so that the dialog moves forward to the task goal in an orderly and proper manner. (We will discuss how this happens in the discussion of dialog management later in this article.)

Simple Example of ASR System: Isolated Digit Recognition

To illustrate some of the ideas presented above, consider a simple isolated word speech recognition system where the vocabulary is the set of 11 digits ('zero' to 'nine' plus the word 'oh' as an alternative for 'zero') and the basic recognition unit is a whole word model. For each of the 11 vocabulary words, we must collect a training set with sufficient, say K, occurrences of each spoken word so as to be able to train reliable and stable acoustic models (the HMMs) for each word. Typically a value of $K = 5$ is sufficient for a speaker-trained system (that is a recognizer that works only for the speech of the speaker who trained the system). For a speaker-independent recognizer, a significantly larger value of K is required to completely characterize the variability in accents, speakers, transducers, environments, etc. For a speaker-independent system based on using only a single transducer (e.g., a telephone line input), and a carefully controlled acoustic environment (low noise), reasonable values of K are on the order of 100–500 for training reliable word models and obtaining good recognition performance.

For implementing an isolated-word recognition system, we do the following:

1. For each word, v, in the vocabulary, we build a word-based HMM, λ_v, i.e., we must (re-)estimate the model parameters λ_v that optimize the likelihood of the K training vectors for the v-th word. This is the training phase of the system.
2. For each unknown (newly spoken) test word that is to be recognized, we measure the feature vectors (the observation sequence), $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$ (where each observation vector, $\mathbf{x}_i$ is the set of MFCCs and their first- and second-order derivatives), we calculate model likelihoods, $P(X|\lambda_v)$, $1 \le v \le V$ for each individual word model (where V is 11 for the digits case), and then we select as the recognized word the word whose model likelihood score is highest, i.e., $v = \arg\max_{1 \le v \le V} P(X|\lambda_v)$. This is the testing phase of the system.

Figure 14 shows a block diagram of a simple HMM-based isolated word recognition system.

Performance of Speech Recognition Systems

A key issue in speech recognition (and understanding) system design is how to evaluate the system's performance. For simple recognition systems, such as the isolated word recognition system described in the previous section, the performance is simply the word error rate of the system. For more complex speech recognition tasks, such as for dictation applications, we must take into account the three types of errors that can occur in recognition, namely word insertions (recognizing more words than were actually spoken), word substitutions (recognizing an incorrect word in place of the correctly spoken word), and word deletions (recognizing fewer words than were actually spoken) (Pallet and Fiscus, 1997). Based on the criterion of equally weighting all three types of errors, the conventional definition of word error rate for most speech recognition tasks is:

$$WER = \frac{NI + NS + ND}{|W|}$$

where NI is the number of word insertions, NS is the number of word substitutions, ND is the number of word deletions, and $|W|$ is the number of words in the sentence W being scored. Based on the above definition of word error rate, the performance of a range of speech recognition and understanding systems is shown in Table 1.

It can be seen that for a small vocabulary (11 digits), the word error rates are very low (0.3%) for a connected digit recognition task in a very clean environment (TI database) (Leonard, 1984), but we

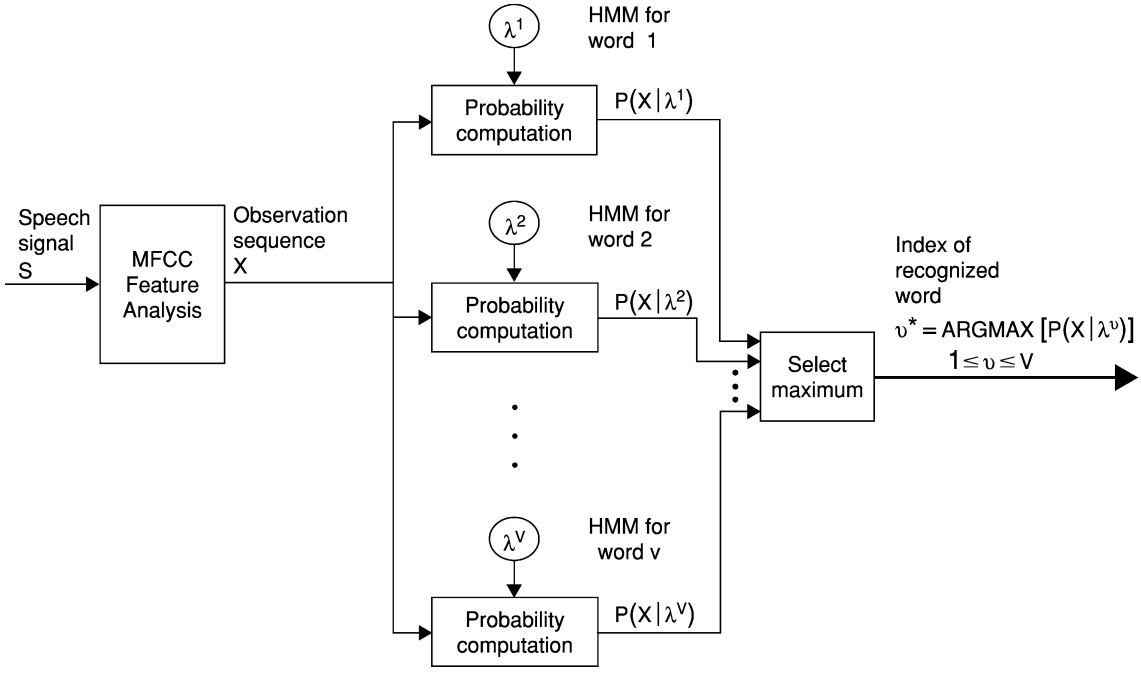

Figure 14 HMM-based Isolated word recognizer.

Table 1 Word error rates for a range of speech recognition systems

Corpus	*Type of speech*	*Vocabulary size*	*Word error rate*
Connect digit string (TI database)	Spontaneous	11 (0–9, oh)	0.3%
Connect digit string (AT&T mall recordings)	Spontaneous	11 (0–9, oh)	2.0%
Connected digit string (AT&T HMIHY)	Conversational	11 (0–9, oh)	5.0%
Resource management (RM)	Read speech	1000	2.0%
Airline travel information system (ATIS)	Spontaneous	2500	2.5%
North American business (NAB & WSJ)	Read text	64 000	6.6%
Broadcast news	Narrated news	210 000	~15%
Switchboard	Telephone conversation	45 000	~27%
Call-home	Telephone conversation	28 000	~35%

see that the digit word error rate rises significantly (to 5.0%) for connected digit strings recorded in the context of a conversation as part of a speech understanding system (HMIHY©) (Gorin *et al.*, 1996). We also see that word error rates are fairly low for 1000- to 2500-word vocabulary tasks (RM [Linguistic Data Consortium, 1992–2000] and ATIS [Ward, 1991]) but increase significantly as the vocabulary size rises (6.6% for a 64 000-word NAB vocabulary, and 13–17% for a 210 000-word broadcast news vocabulary), as well as for more colloquially spoken speech (Switchboard and Call-home [Godfrey *et al.*, 1992]), where the word error rates are much higher than comparable tasks where the speech is more formally spoken.

Figure 15 illustrates the reduction in word error rate that has been achieved over time for several of the tasks from Table 1 (as well as other tasks not covered in Table 1). It can be seen that there is a steady and systematic decrease in word error rate (shown on a logarithmic scale) over time for every system that has been extensively studied. Hence it is generally believed that virtually any (task-oriented) speech recognition system can achieve arbitrarily low error (over time) if sufficient effort is put into finding appropriate techniques for reducing the word error rate.

If one compares the best ASR performance for machines on any given task with human performance (which often is hard to measure), the resulting comparison (as seen in Figure 16) shows that humans outperform machines by factors of between 10 and 50; that is the machine achieves word error rates that are larger by factors of 10–50. Hence we still have a long way to go before machines outperform humans on speech recognition tasks. However, one should

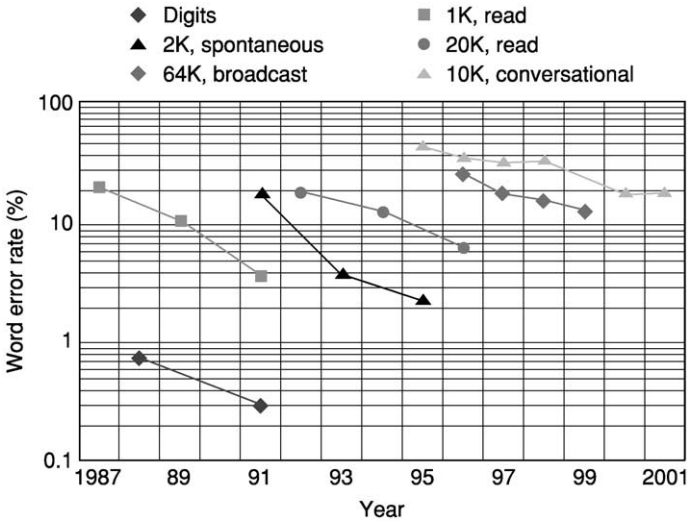

Figure 15 Reductions in speech recognition word error rates over time for a range of task-oriented systems (Pallet *et al.*, 1995).

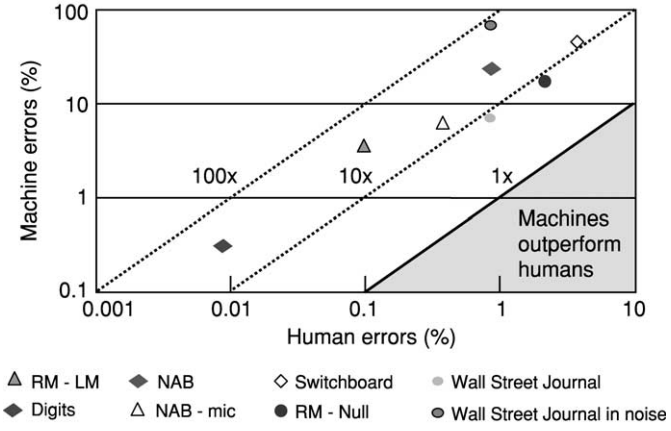

Figure 16 Comparison of human and machine speech recognition performance for a range of speech recognition tasks (Lippman, 1997).

also note that under a certain condition an automatic speech recognition system could deliver a better service than a human. One such example is the recognition of a long connected digit string, such as a credit card's 16-digit number, that is uttered all at once; a human listener would not be able to memorize or jot down the spoken string without losing track of all the digits.

Spoken Language Understanding

The goal of the spoken language understanding module of the speech dialog circle is to interpret the meaning of key words and phrases in the recognized speech string, and to map them to actions that the speech understanding system should take. For speech understanding, it is important to recognize that in domain-specific applications highly accurate understanding can be achieved without correctly recognizing every word in the sentence. Hence a speaker can have spoken the sentence: *I need some help with my computer hard drive* and so long as the machine correctly recognized the words *help* and *hard drive*, it basically understands the context of the sentence (needing help) and the object of the context (hard drive). All of the other words in the sentence can often be misrecognized (although not so badly that other contextually significant words are recognized) without affecting the understanding of the meaning of the sentence. In this sense, keyword spotting (Wilpon *et al.*, 1990) can be considered a primitive form of speech understanding, without involving sophisticated semantic analysis.

Spoken language understanding makes it possible to offer services where the customer can speak naturally without having to learn a specific vocabulary

and task syntax in order to complete a transaction and interact with a machine (Juang and Furui, 2000). It performs this task by exploiting the task grammar and task semantics to restrict the range of meanings associated with the recognized word string, and by exploiting a predefined set of 'salient' words and phrases that map high-information word sequences to this restricted set of meanings. Spoken language understanding is especially useful when the range of meanings is naturally restricted and easily cataloged so that a Bayesian formulation can be used to optimally determine the meaning of the sentence from the word sequence. This Bayesian approach utilizes the recognized sequence of words, W, and the underlying meaning, C, to determine the probability of each possible meaning, given the word sequence, namely:

$$P(C|W) = P(W|C)P(C)/P(W)$$

and then finding the best conceptual structure (meaning) using a combination of acoustic, linguistic and semantic scores, namely:

$$C^* = \arg\max_C P(W|C)P(C)$$

This approach makes extensive use of the statistical relationship between the word sequence and the intended meaning.

One of the most successful (commercial) speech understanding systems to date has been the AT&T How May I Help You (HMIHY) task for customer care. For this task, the customer dials into an AT&T 800 number for help on tasks related to his or her long distance or local billing account. The prompt to the customer is simply: 'AT&T. How May I Help You?' The customer responds to this prompt with totally unconstrained fluent speech describing the reason for calling the customer care help line. The system tries to recognize every spoken word (but invariably makes a very high percentage of word errors), and then utilizes the Bayesian concept framework to determine the meaning of the speech. Fortunately, the potential meaning of the spoken input is restricted to one of several possible outcomes, such as asking about Account Balances, or new Calling Plans, or changes in local service, or help for an Unrecognized Number, etc. Based on this highly limited set of outcomes, the spoken language component determines which meaning is most appropriate (or else decides not to make a decision but instead to defer the decision to the next cycle of the dialog circle), and appropriately routes the call. The dialog manager, spoken language generation, and text-to-speech modules complete the cycle based on the meaning determined by the spoken language understanding box. A simple characterization of the HMIHY system is shown in Figure 17.

The major challenge in spoken language understanding is to go beyond the simple classification task of the HMIHY system (where the conceptual meaning is restricted to one of a fixed, often small, set of choices) and to create a true concept and meaning understanding system.

While this challenge remains in an embryonic stage, an early attempt, namely the Air Travel Information System (ATIS), was made in embedding speech recognition in a stylized semantic structure to mimic a natural language interaction between human and a machine. In such a system, the semantic notions encapsulated in the system are rather limited, mostly in terms of originating city and destination city names, fares, airport names, travel times and so on, and can be directly instantiated in a semantic template without much text analysis for understanding. For example, a typical semantic template or network is shown in Figure 18 where the relevant notions, such as the departing city, can be easily identified and used in dialog management to create the desired user interaction with the system.

Dialog Management, Spoken Language Generation, and Text-to-Speech Synthesis

The goal of the dialog management module is to combine the meaning of the current input speech

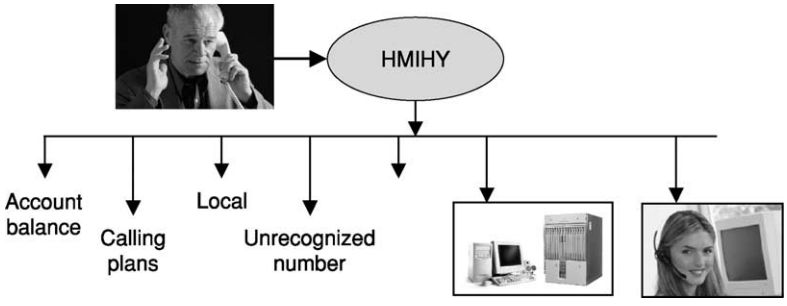

Figure 17 Conceptual representation of HMIHY (How May I Help You?) system.

with the current state of the system (which is based on the interaction history with the user) in order to decide what the next step in the interaction should be. In this manner, the dialog management module makes viable fairly complex services that require multiple exchanges between the system and the customer. Such dialog systems can also handle user-initiated topic switching within the domain of the application.

The dialog management module is one of the most crucial steps in the speech dialog circle for a successful transaction as it enables the customer to accomplish the desired task. The way in which the dialog management module works is by exploiting models of dialog to determine the most appropriate spoken text string to guide the dialog forward toward a clear and well-understood goal or system interaction. The computational models for dialog management include both structure-based approaches (which models dialog as a predefined state transition network that is followed from an initial goal state to a set of final goal states), or plan-based approaches (which consider communication as executing a set of plans that are oriented toward goal achievement).

The key tools of dialog strategy are the following:

- Confirmation: used to ascertain correctness of the recognized and understood utterances
- Error recovery: used to get the dialog back on track after a user indicates that the system has misunderstood something
- Reprompting: used when the system expected input but did not receive any input
- Completion: used to elicit missing input information from the user
- Constraining: used to reduce the scope of the request so that a reasonable amount of information is retrieved, presented to the user, or otherwise acted upon
- Relaxation: used to increase the scope of the request when no information has been retrieved
- Disambiguation: used to resolve inconsistent input from the user
- Greeting/Closing: used to maintain social protocol at the beginning and end of an interaction
- Mixed initiative: allows users to manage the dialog flow.

Although most of the tools of dialog strategy are straightforward and the conditions for their use are fairly clear, the mixed initiative tool is perhaps the most interesting one, as it enables a user to manage the dialog and get it back on track whenever the user feels the need to take over and lead the interactions with the machine. Figure 19 shows a simple chart that illustrates the two extremes of mixed initiative for a simple operator services scenario. At the one extreme, where the **system** manages the dialog totally, the system responses are simple declarative requests to elicit information, as exemplified by the system command "Please say collect, calling card, third number". At the other extreme is **user** management of the dialog where the system responses are open ended and the customer can freely respond to the system command 'How may I help you?'

Figure 20 illustrates some simple examples of the use of system initiative, mixed initiative, and user initiative for an airlines reservation task. It can be seen that system initiative leads to long dialogs (due to the limited information retrieval at each query), but the dialogs are relatively easy to design, whereas user initiative leads to shorter dialogs (and hence a better user experience), but the dialogs are more difficult to design. (Most practical natural language systems need to be mixed initiative so as to be able to change initiatives from one extreme to another, depending on the state of the dialog and how successfully things have progressed toward the ultimate understanding goal.)

Dialog management systems are evaluated based on the speed and accuracy of attaining a well-defined

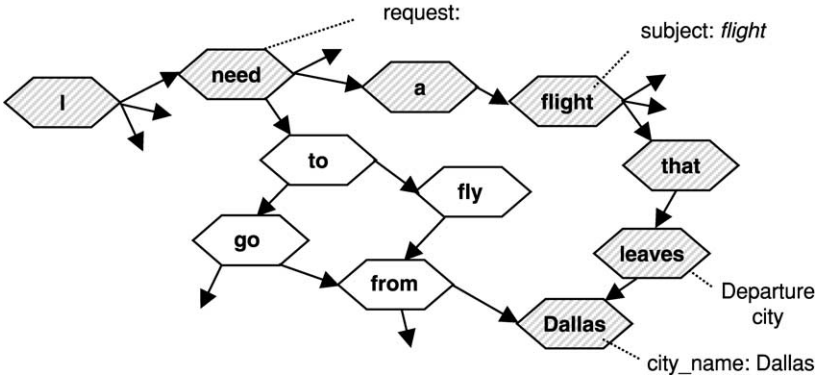

Figure 18 An example of a word grammar with embedded semantic notions in ATIS.

Figure 19 Illustration of mixed initiative for operator services scenario.

task goal, such as booking an airline reservation, renting a car, purchasing a stock, or obtaining help with a service.

The spoken language generation module translates the action of the dialog manager into a textual representation and the text-to-speech modules convert the textual representation into natural-sounding speech to be played to the user so as to initiate another round of dialog discussion or to end the query (hopefully successfully).

User Interfaces and Multimodal Systems

The user interface for a speech communications system is defined by the performance of each of the blocks in the speech dialog circle. A good user interface is essential to the success of any task-oriented system, providing the following capabilities:

- It makes the application easy to use and robust to the kinds of confusion that arise in human–machine communications by voice.
- It keeps the conversation moving forward, even in periods of great uncertainty on the parts of either the user or the machine.
- Although it cannot save a system with poor speech recognition or speech understanding performance, it can make or break a system with excellent speech recognition and speech understanding performance.

Although we have primarily been concerned with speech recognition and understanding interfaces to machines, there are times when a multimodal approach to human–machine communications is both necessary and essential. The potential modalities that can work in concert with speech include gesture and pointing devices (e.g., a mouse, keypad, or stylus). The selection of the most appropriate user interface mode (or combination of modes) depends on the device, the task, the environment, and the user's abilities and preferences. Hence, when trying to identify objects on a map (e.g., restaurants, locations of subway stations, historical sites), the use of a pointing device (to indicate the area of interest) along with speech (to indicate the topic of interest) often is a good user interface, especially for small computing devices like tablet PCs or PDAs. Similarly, when entering PDA-like information (e.g., appointments, reminders, dates, times, etc.) onto a small handheld device, the use of a stylus to indicate the appropriate

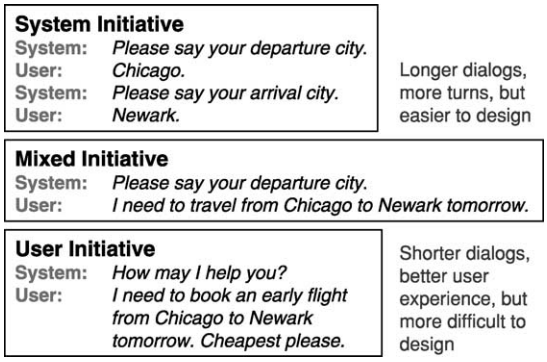

Figure 20 Examples of mixed initiative dialogs.

type of information with voice filling in the data field is often the most natural way of entering such information (especially as contrasted with stylus-based text input systems such as graffiti for Palm-like devices). Microsoft research has shown the efficacy of such a solution with the MIPad (Multimodel Interactive Pad) demonstration, and they claim to have achieved double the throughput for English using the multimodal interface over that achieved with just a pen stylus and the graffiti language.

Summary

In this article we have outlined the major components of a modern speech recognition and spoken language understanding system, as used within a voice dialog system. We have shown the role of signal processing in creating a reliable feature set for the recognizer and the role of statistical methods in enabling the recognizer to recognize the words of the spoken input sentence as well as the meaning associated with the recognized word sequence. We have shown how a dialog manager utilizes the meaning accrued from the current as well as previous spoken inputs to create an appropriate response (as well as potentially taking some appropriate actions) to the customer request(s), and finally how the spoken language generation and text-to-speech synthesis parts of the dialog complete the dialog circle by providing feedback to the user as to actions taken and further information that is required to complete the transaction that is requested.

Although we have come a long way toward the vision of Hal, the machine that both recognizes words reliably and understands their meaning almost flawlessly, we still have a long way to go before this vision is fully achieved. The major problem that must yet be tackled is robustness of the recognizer and the language understanding system to variability in speakers, accents, devices, and environments in which the speech is recorded. Systems that appear to

work almost flawlessly under laboratory conditions often fail miserably in noisy train or airplane stations, when used with a cellphone or a speakerphone, when used in an automobile environment, or when used in noisy offices. There are many ideas that have been advanced for making speech recognition more robust, but to date none of these ideas has been able to fully combat the degradation in performance that occurs under these nonideal conditions.

Speech recognition and speech understanding systems have made their way into mainstream applications and almost everybody has used a speech recognition device at one time or another. They are widely used in telephony applications (operator services, customer care), in help desks, in desktop dictation applications, and especially in office environments as an aid to digitizing reports, memos, briefs, and other office information. As speech recognition and speech understanding systems become more robust, they will find their way into cellphone and automotive applications, as well as into small devices, providing a natural and intuitive way to control the operation of these devices as well as to access and enter information.

See also: Speech Recognition, Audio-Visual; Speech Recognition, Automatic: History.

Bibliography

Atal B S & Hanauer S L (1971). 'Speech analysis and synthesis by linear prediction of the speech wave.' *Journal of the Acoustical Society of America 50(2),* 637–655.

Bahl L R, Jelinek F & Mercer R L (1983). 'A maximum likelihood approach to continuous speech recognition.' *IEEE Transactions on Pattern Analysis & Machine Intelligence, PAMI-5(2),* 179–190.

Baum L E (1972). 'An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes.' *Inequalities 3,* 1–8.

Baum L E, Petri T, Soules G & Weiss N (1970). 'A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains.' *Annals in Mathematical Statistics 41,* 164–171.

Bellman R (1957). *Dynamic programming.* Boston: Princeton University Press.

Cover T & Thomas J (1991). *Wiley series in telecommunications: Elements of information theory.* John Wiley and Sons.

Cox R V, Kamm C A, Rabiner L R, Schroeter J & Wilpon G J (2000). 'Speech and language processing for next-millennium communications services.' *Proceedings of the IEEE 88(8),* 1314–1337.

Davis S & Mermelstein P (1980). 'Comparison of parametric representations of monosyllabic word recognition in continuously spoken sentences.' *IEEE Transactions on Acoustics, Speech and Signal Processing 28(4),* 357–366.

Ferguson J D (1980). 'Hidden Markov analysis: an introduction.' In *Hidden Markov models for speech.* Princeton: Institute for Defense Analyses.

Forney D (1973). 'The Viterbi algorithm.' *Proceedings IEEE 61,* 268–278.

Furui S (1981). 'Cepstral analysis techniques for automatic speaker verification.' *IEEE Transactions on Acoustics, Speech and Signal Processing, ASSP-29(2),* 254–272.

Gauvain J-L & Lamel L (2003). 'Large vocabulary speech recognition based on statistical methods.' In Chou W & Juang B H (eds.) *Pattern recognition in speech & language processing.* New York: CRC Press. 149–189.

Godfrey J J, Holliman E C & McDaniel J (1992). 'SWITCHBOARD: telephone speech corpus for research and development.' In *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing I.* 517–520.

Gorin A L, Parker B A, Sachs R M & Wilpon J G (1996). 'How may I help you?' *Proceedings of the Interactive Voice Technology for Telecommunications Applications (IVTTA).* 57–60.

ISCA Archive (2001). *Disfluency in spontaneous speech (DiSS'01),* ISCA Tutorial and Research Workshop (ITRW), Edinburgh, Scotland, UK, August 29–31, 2001. http://www.isca-speech.org/archive/diss_01.

Jelinek F (1997). *Statistical methods for speech recognition.* Cambridge: MIT Press, Cambridge.

Jelinek F, Mercer R L & Roukos S (1991). 'Principles of lexical language modeling for speech recognition.' In Furui & Sondhi (eds.) *Advances in speech signal processing.* New York: Mercer Dekker. 651–699.

Juang B H (1985). 'Maximum likelihood estimation for mixture multivariate stochastic observations of Markov chains.' *AT&T Technology Journal 64(6),* 1235–1249.

Juang B H & Furui S (2000). 'Automatic recognition and understanding of spoken language – a first step towards natural human-machine communication.' *Proceedings of the IEEE* .

Juang B H, Levinson S E & Sondhi M M (1986). 'Maximum likelihood estimation for multivariate mixture observations of Markov chains.' *IEEE Transactions in Information Theory It-32(2),* 307–309.

Juang B H, Thomson D & Perdue R J (1995). 'Deployable automatic speech recognition systems – advances and challenges.' *AT&T Technical Journal 74(2).*

Jurafsky D S & Martin J H (2000). *Speech and language processing.* Englewood: Prentice Hall.

Kamm C & Helander M (1997). 'Design issues for interfaces using voice input.' In Helander M, Landauer T K & Prabhu P (eds.) *Handbook of human-computer interaction.* Amsterdam: Elsevier. 1043–1059.

Kawahara T & Lee C H (1998). 'Flexible speech understanding based on combined key-phrase detection and verification.' *IEEE Transactions on Speech and Audio Processing, T-SA 6(6),* 558–568.

Lee C H, Juang B H, Soong F K & Rabiner L R (1989). 'Word recognition using whole word and subword

models.' *Conference Record 1989 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Paper S 12.2. 683–686.

Lee K-F (1989). *The development of the Sphinx System*. Kluwer.

Leonard R G (1984). 'A database for speaker-independent digit recognition.' *Proceedings of ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing*. 42.11.1–42.11.4.

Levinson S E, Rabiner L R & Sondhi M M (1983). 'An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition.' *Bell Systems Technical Journal 62(4)*, 1035–1074.

Linguistic Data Consortium (1992–2000). LDC Catalog Resource Management RM 2 2.0, http://wave.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC93S3C.

Lippman R P (1997). 'Speech recognition by machines and humans.' *Speech Communication 22(1)*, 1–15.

Markel J D & Gray A H Jr (1996). *Linear prediction of speech*. Springer-Verlag.

Rahim M & Juang B H (1996). 'Signal bias removal by maximum likelihood estimation for robust telephone speech recognition.' *IEEE Transactions Speech and Audio Processing 4(1)*, 19–30.

Mohri M (1997). 'Finite-state transducers in language and speech processing.' *Computational Linguistics 23(2)*, 269–312.

Nadas A (1985). 'On Turing's formula for word probabilities.' *IEEE Transactions on Acoustics, Speech, and Signal Processing ASSP-33(6)*, 1414–1416.

Ney H (1984). 'The use of a one stage dynamic programming algorithm for connected word recognition.' *IEEE Transactions, Acoustics, Speech and Signal Processing, ASSP-32(2)*, 263–271.

Pallett D & Fiscus J (1997). '1996 Preliminary broadcast news benchmark tests.' In *DARPA 1997 speech recognition workshop*.

Pallett D S *et al.* (1995). '1994 benchmark tests for the ARPA spoken language program.' *Proceedings of the 1995 ARPA Human Language Technology Workshop* 5–36.

Paul D B (2001). 'An efficient A* stack decoder algorithm for continuous speech recognition with a stochastic language model.' *Proceedings IEEE ICASSP-01*, Salt Lake City, May 2001. 357–362.

Paul D B & Baker J M (1992). 'The design for the Wall Street Journal-based CSR corpus.' In *Proceedings of the DARPA SLS Workshop*.

Price P (1990). 'Evaluation of spoken language systems: the ATIS domain.' In Price P (ed.) *Proceedings of the Third DARPA SLS Workshop*. Morgan Kaufmann. 91–95.

Rabiner L R (1989). 'A tutorial on hidden Markov models and selected applications in speech recognition.' *Proceedings of the IEEE 77(2)*, 257–286.

Rabiner L R & Gold B (1975). *Theory and applications of digital signal processing*. Englewood Cliffs, NJ: Prentice-Hall.

Rabiner L R & Juang B H (1985). 'An introduction to hidden Markov models.' *IEEE Signal Processing Magazine 3(1)*, 4–16.

Rabiner L R, Wilpon J G & Juang B H (1986). 'A model-based connected-digit recognition system using either hidden Markov models or templates.' *Computer Speech & Language 1(2)*, December, 167–197.

Rabiner L R, Juang B H & Lee C H (1996). 'An overview of automatic speech recognition, in automatic speech & speaker recognition – advanced topics.' In Lee *et al.* (eds.). Norwell: Kluwer Academic. 1–30.

Rahim M, Lee C-H & Juang B-H (1997). 'Discriminative utterance verification for connected digit recognition.' *IEEE Transactions on Speech and Audio Processing 5(3)*, 266–277.

Riley M D *et al.* (1999). 'Stochastic pronunciation modeling from hand-labelled phonetic corpora.' *Speech Communication 29(2–4)*, 209–224.

Roe D B, Wilpon J G, Mikkilineni P & Prezas D (1991). 'AT&T's speech recognition in the telephone network.' *Speech Technology Mag 5(3)*, February/March, 16–22.

Rosenfeld R (2000). 'Two decades of statistical language modeling: where do we go from here?' *Proceedings of the IEEE, Special Issue on Spoken Language Processing 88(8)*, 1270–1278.

Roukos S (1998). 'Language representation.' In Varile G B & Zampolli A (eds.) *Survey of the State of the Art in Human Language Technology*. Cambridge University Express.

Sugamura N, Hirokawa T, Sagayama S & Furui S (1994). 'Speech processing technologies and telecommunications applications at NTT.' *Proceedings of the IVTTA 94*, 37–42.

Ward W (1991). 'Evaluation of the CMU ATIS System.' *Proceedings of the DARPA Speech and Natural Language Workshop*, February 19–22, 1991. 101–105.

Wilpon J G, Rabiner L R, Lee C-H & Goldman E (1990). 'Automatic recognition of keywords in unconstrained speech using hidden Markov models.' *IEEE Transactions on Acoustics, Speech and Signal Processing 38(11)*, 1870–1878.

Young S J (1996). 'A review of large vocabulary continuous speech recognition.' *IEEE Signal Processing Magazine 13(5)*, September, 45–57.