LARGE VOCABULARY SPEECH RECOGNITION USING SUBWORD UNITS

C. H. Lee, J. L. Gauvain, R. Pieraccini and L. R. Rabiner

AT&T Bell Laboratories
Murray Hill, New Jersey 07974

ABSTRACT – Research in large vocabulary speech recognition has been intensively carried out worldwide, in the past several years, spurred on by advances in algorithms, architectures, and hardware. In the United States, the DARPA community has focused efforts on studying several systems including Resource Management, a 991 word task, ATIS (Air Travel Information System), a task with an open vocabulary (in practice on the order of several thousand words) and a natural language component, and Wall Street Journal, a task with a vocabulary on the order of 20,000 words. Although we have learned a great deal about how to build and efficiently implement large vocabulary speech recognition systems, there remain a whole range of fundamental questions for which we have no definitive answers. For example we do not yet know the best way to build and train the fundamental subword units from which word models are created. We do not yet know the best way to impose language constraints on the recognizer so as to utilize all available knowledge in the most computationally efficient manner. We do not yet even understand the best way to implement a recognition system so as to maximize the probability of recognizing the spoken string while minimizing the computation for string comparison and searching through the recognition network. In this paper we review the basic structure of a large vocabulary speech recognition system, discuss the considerations in the choice of subword unit, method of training, integration of language model, and implementation of overall system, and report on some recent results, obtained on at AT&T Bell Laboratories and elsewhere, on the DARPA Resource Management Task.

1 INTRODUCTION

In the past few years a significant portion of the research in speech recognition has gone into studying the problem of how to build and implement a large vocabulary, continuous speech recognition system. Much of this effort has been stimulated by DARPA which has funded research on three recognition tasks (Lee, 1989 & Lee, et al., 1990); however there is worldwide interest in large vocabulary speech recognition because of the potential applications to voice database access and management, voice dictation, and to language translation (Jelinek, 1985 & Roe, et al., 1992). Although some of the systems have been trained to individual speakers (Jelinek, 1985 & Roe, et al., 1992), most current large vocabulary recognition systems have the goal of performing speech recognition on fluent input (continuous speech) by any talker (speaker independent systems).

The approach that is conventionally taken to large vocabulary speech recognition is fundamentally a statistical pattern recognition approach. The fundamental speech units use phonetic labels but are modeled acoustically based on a lexical description of the words in the vocabulary. In general, no assumption is made, a priori, about the mapping between acoustic measurements and subword linguistic units such as phonemes; such a mapping is entirely learned via a finite labeled training set of speech utterances. The resulting speech units, which we call phone-like units or PLU's are essentially acoustic descriptions of linguistically-based units as represented in the words occurring in the given training set. (We will return to this important point later in this paper when we discuss creation of so-called vocabulary independent subword units.)

A block diagram of the large vocabulary continuous speech recognition system developed at AT&T Bell Laboratories is shown in Fig. 1. The system consists of three main modules, namely a feature analysis (or spectral analysis) module, a word-level acoustic match module, and a sentence-level language match module. The feature analysis module provides the acoustic feature vectors used to characterize the spectral properties of the time varying speech signal. The word-level acoustic match module evaluates

the similarity between the input feature vector sequence (corresponding to the input speech) and a set of acoustic word models to determine what words were most likely spoken. The sentence-level match module uses a language model (based on a set of syntactic and semantic rules) to determine the word sequence for the most likely sentence.

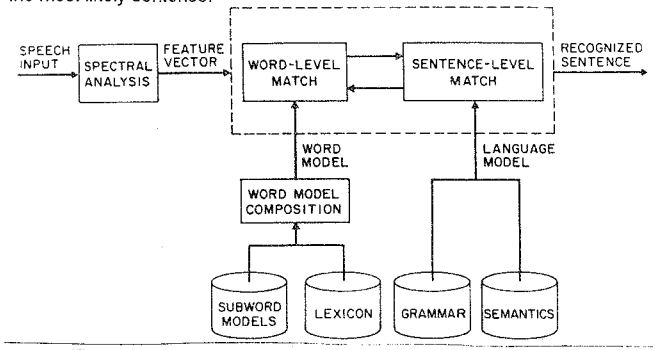

Figure 1. Block diagram of the continuous speech recognizer.

In the following sections we discuss each module of the baseline system of Fig. 1 in more detail. We will attempt to explain what is understood about each module, and where active research is ongoing in order to resolve differences of opinion as to the best way to implement the desired processing. Following these discussions we present some recent results on performance of the DARPA Resource Management System based on using so-called vocabulary independent units, and based on adaptive training methods.

2 THE BASELINE SPEECH RECOGNITION SYSTEM

2.1 Acoustic Analysis Module

The purpose of the acoustic analysis module is to parameterize the speech into a series of spectral vectors that contain the relevant (for recognition) information about the sounds within the utterance. Although there is no consensus as to what constitutes the optimal spectral analysis, there are generally several aspects of the analysis that are common to most recognition systems. For example most systems use LPC spectral analysis methods based on fixed sized frames, e.g. every 10 msec an analysis of a fixed frame of 30 msec of signal is performed. Typically the LPC analysis provides a set of cepstral coefficients for the frame. Sometimes non-uniform frequency scales are used giving the so-called mel frequency cepstral coefficients (Davis & Mermelstein, 1980). The rationale here is that since the human ear perceives frequencies on a non-uniform scale, it would be desirable to represent the spectral information of sounds on the same perceptual scale.

In the last few years the spectral feature set for each frame has been extended to include dynamic information about the derivatives (first and second order) of the cepstral vector, as well as the static information about the cepstrum (Furui, 1986 & Lee, et al., 1992). Also scalars representing frame energy and its derivatives are often used as part of the representation for each frame. For the system implemented at Bell Labs, each 30 msec of speech (8 kHz sampling rate) was analyzed 100 times per second (10 msec shift) to give a spectral vector with 12 cepstral coefficients (on a uniform frequency scale), 12 first order cepstral derivatives, 12 second order cepstral derivatives, and first and second order log energy derivatives. (Absolute log energy was not used directly because of its sensitivity to absolute level in the the recording.) Hence a spectral vector with 38 components was created every 10 msec throughout the signal.

Number	Symbol	Word	Number	Symbol	Word	Number	Symbol	Word
1	h#	silence	17	er	bird	33	p	pop
2	aa	father	18	ey	bait	34	r	red
3	ae	bat	19	f	fief	35	s	sis
4	ah	butt	20	g	gag	36	sh	shoe
5	ao	bought	21	hh	hag	37	t	tot
6	aw	bough	22	ih	bit	38	th	thief
7	ax	again	23	ix	roses	39	uh	book
8	axr	diner	24	iy	beat	40	uw	boot
9	ay	bite	25	jh	judge	41	v	very
10	b	bob	26	k	kick	42	w	wet
11	ch	church	27	l	led	43	y	yet
12	d	dad	28	m	mom	44	z	zoo
13	dh	they	29	n	no	45	zh	measure
14	eh	bet	30	ng	sing	46	dx	butter
15	el	bottle	31	ow	boat	47	nx	center
16	en	button	32	oy	boy			

Table 1. The 47 Context-Independent PLUs.

2.2 Word Level Match Module

The essence of the word level match module is the set of subword models and the lexicon, as seen in Fig. 1. The subword models are the representation of the fundamental speech units used as the building blocks for words, phrases, and sentences. Probably the most research in large vocabulary speech recognition has gone into defining these subword units in a manner such that they can be easily trained from finite training sets of speech material, such that they are robust to natural variations in accent, word pronunciation, and test material, and such that they provide high recognition accuracy for the required speech task. To date no one has defined the "ideal" set of subword units. However a great deal of thought has gone into deciding what the real issues are in defining and using various alternatives for the subword units.

Perhaps the simplest set of subword units, which are widely used, is the set of basic phonemes of the language. Although there is no complete agreement as to what sounds are part of this basic set, Table 1 shows one representative set of 47 such phonemes with typical words in which the phonemes appear. These basic units, when trained from real speech material, are called context-independent phone-like units (CI-PLU) since the sounds are represented independent of the linguistic context in which they occur, and since the spectral properties of the sounds are learned from a training set, rather than postulated on the basis of the linguistic features of the units.

In contrast to the 47 CI-PLU's of Table 1, one could consider subword units which were context dependent (CD). Thus, for example, a separate unit could exist for the sound /ae/ when preceded by /f/ and followed by /t/ (as in fat), then for /ae/ when preceded by /b/ and followed by /t/ (as in bat). In theory there could be as many as $(47)^3$ CD-PLU's when considering all preceding and following sounds; in practice there are on the order of 10,000 such possibilities, a number significantly less than the 100,000 count of $(47)^3$, but significantly more than the 47 CI-PLU's of Table 1. Such CD-PLU's have been extensively used for large vocabulary speech recognition (Lee, 1989 & Morimoto, et al., 1990), but practical methods are generally used to restrict the number of units to something on the order of 1000-2000 units.

The second basic component of the word-level match module is the lexicon which provides a linguistic description of the words in the task vocabulary in terms of the basic set of subword units. Among the issues in the creation of a suitable word lexicon is the base (or standard) pronunciation of each word and the number of alternative pronunciations provided for each word. The base pronunciation is the equivalent, in some sense, of a pronunciation guide to the word; the number of alternative pronunciations is a measure of word variability across different regional accents and talker populations. Although there have been some very interesting experiments based on multiple word pronunciation lexicons (Weintraub, et al., 1989), most large vocabulary speech recognition systems rely on a lexicon with only a single pronunciation provided

for each word. This "canonic" representation of each word must be consistent with the subword units; hence its form changes as different sets of CD or CI subword units are used. Also, for function words like "the", "and", "to", etc., it is well known that there is no "canonic" or standard pronunciation. Hence a single representation for such function words will invariantly lead to some problems with recognition.

The word model composition component of the word-level match module is simply the process of retrieving the word pronunciation from the lexicon, and then concatenating appropriate subword units to create individual word models which are then matched against the spectral vectors of the input speech signal. In order to understand how such matching takes place, we must first discuss how subword units are modeled and how the models are trained from finite training sets of speech.

2.2.1 Subword Unit Models

A key to the success of modern speech recognition systems is the use of statistical modeling techniques (e.g. hidden Markov models – HMM's) to represent the basic subword units (Rabiner, 1989). Although many variants exist, perhaps the simplest way subword units are modeled is as a left-to-right HMM, of the type shown in Fig. 2. Each unit is represented by a simple first-order, left-to-right HMM having N states, $S_1, S_2, \ldots, S_N$, with only self and forward transitions.

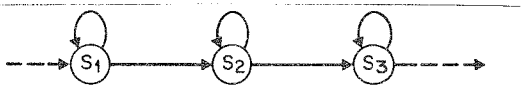

Figure 2. HMM representation of subword model.

Within each state of the model there is an observation density which specifies the likelihood (probability) of a spectral vector from the speech signal occurring within the model state. This observation density can either be a discrete density (implying the use of a common codebook to discretize the input spectral vector), or a continuous density (or even what is called a semi-continuous density (Huang, et al., 1990) which is a codebook of continuous densities whose weights are chosen according to the model state). Although continuous density modeling usually provides the highest performance recognition systems, it requires the most computation to implement. The performance obtained with discrete or semi-continuous densities is often comparable to or only slightly lower than that obtained with continuous densities; often at significantly reduced computation rates.

For continuous density modeling the Bell Labs system uses both an observation probability density function (for each state) represented by a weighted sum of M multivariate Gaussian density functions with a diagonal covariance matrix, and an energy histogram representing the log probability of observing a frame with a given log-energy. No duration information is used. All subword unit models have three states except the model for silence which has only one state. Furthermore no transition probabilities are used; forward and self transitions from a state are assumed equally likely.

2.2.2 Training of Subword Unit Models

Training of subword unit models consists of estimating the HMM parameters from a labeled training set of continuous speech utterances where all of the relevant subword units are known to occur "sufficiently" often. The training problem is another key aspect of the system, as the way in which training is performed affects greatly the overall recognition system performance.

The first issue of note is the size of the training set. The optimal training set size is infinity – i.e. the more training material that is used, the higher the reliability of the resulting speech models. Since infinite size training sets are impossible to obtain (and computationally unmanageable), we must use a finite size training set. This immediately implies that some subword units will occur much less often than others (at least in any natural recording this will be the case). Hence we immediately see a tradeoff between using

fewer subword units (where we get better coverage of individual units, but poor resolution as to linguistic context), and more subword units (where we get poor coverage of the infrequently occurring units, but improved resolution of linguistic context).

A second issue is the choice of training material. For a given amount of training material, the best coverage is obtained when the statistics of occurrence of the training set units match those of the recognition task; i.e. the training set sentences should come from the same linguistic material as the recognition task (i.e. same vocabulary, same language model). However, in such a case, the universality of the resulting speech models is poor; i.e. the same models may perform poorly on a totally different recognition task because of poor coverage of subword units for the new task. Hence the concept of "task dependent" training, which maximizes performance for a given task, versus "task independent" training, which maximizes performance for any task. Most systems use task dependent training – we will present results on both types of training in this paper.

Finally, an alternative to using a large training set is to use some initial set of subword unit models and adapt them over time (with new training material, possibly derived from actual test utterances) to the speaker or environment. Such methods of adaptive training are reasonable for new speakers, vocabularies or environments, and will be shown later to be an effective way of bootstrapping a good set of specific models from a more general set of models.

2.3 Sentence Level Match Module

The sentence level match module uses the constraints imposed by a grammar (a set of syntactic rules on which words are allowed in given contexts) and a set of semantic rules (which eliminate meaningless sentences) to determine the optimal sentence in the language – i.e. the best word sequence, consistent with the grammar and the semantics, that matches the input speech. Although there have been proposed a number of different forms for the grammar (e.g. formal grammar, N-gram word probabilities, word pair, etc.), we assume a simple grammar that can be represented as a finite state network (FSN). In this manner it is relatively straightforward to implement the grammar directly with the word-level match module. In particular, for the DARPA RM task (991 words), we have used either a word-pair (WP) grammar, which specifies explicitly, for each word in the vocabulary, which words are allowed to follow that word, or a no-grammar (NG) grammar, in which we assume that every word can follow every other word in the vocabulary. The perplexities (average word branching factor) of these two grammars is 60 for the WP case and 991 for the NG case. The implementations of these grammars as FSN's is shown in Figs. 3 and 4. For the WP case we exploit the fact that only a subset of the vocabulary occurs as the first word in a sentence (condition B for beginning words), and only a subset of the vocabulary occurs as the last word in a sentence (condition E for ending words); hence we can partition the vocabulary into 4 non-overlapping sets of words, namely:

$\{BE\}$ = set of words which can either begin or end a sentence, $|BE| = 117$

$\{B\bar{E}\}$ = set of words which can begin but which cannot end a sentence, $|B\bar{E}| = 64$

$\{\bar{B}E\}$ = set of words which cannot begin but can end a sentence, $|\bar{B}E| = 488$

$\{\bar{B}\bar{E}\}$ = set of words which cannot begin or end a sentence, $|\bar{B}\bar{E}| = 322$

The resulting FSN of Fig. 3 has 995 real arcs and 18 null arcs. To account for silence between words (which is optional) each word arc bundle (nodes 1 to 4) is expanded to individual words followed by optional silence, as shown at the bottom of Fig. 3. Hence the overall FSN allows recognition of sentences of the form:

$$S: \text{(silence)} - \{B\bar{E}, BE\} - \text{(silence)} - (\{W\}) \ldots (\{W\}) - \text{(silence)} - \{\bar{B}E, BE\} - \text{(silence)}$$

where $\{W\}$ is any word which is allowed to follow the previous word and includes optional silence.

The FSN for the NG case, as shown in Fig. 4, is considerably simpler than the FSN for the WP case. The sentences allowed by this grammar are of the form:

$$S: \text{(silence)} - (\{W\}) \ldots (\{W\}) - \text{(silence)}$$

where $\{W\}$ is now any word in the task vocabulary.

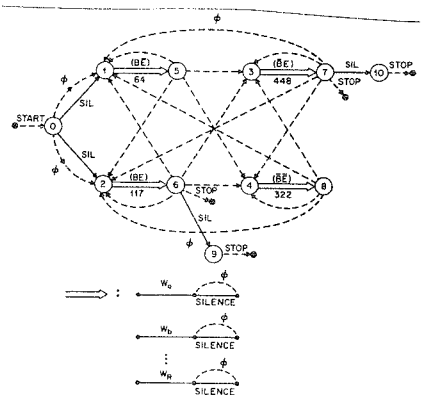

Figure 3. Network representation of WP syntax.

The grammar FSN's of Figs. 3 and 4 have the property that they can produce any valid sentence in the task language. Unfortunately they also have the property that they can produce a large number of sentences which are not valid in the task language, e.g. the sentence S : "and" "and" "and" is valid for the NG network but not for the RM task. The overcoverage (ratio of sentences generated by the FSN to sentences valid in the task language) of the FSN's is often extremely large and this is a negative feature of using these simple networks as the grammar network. On the other hand, using a full grammar (i.e. no overcoverage) is generally prohibitive from a computational point of view.

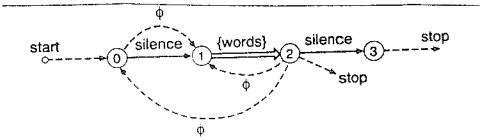

Figure 4. Network representation of NG syntax.

One way to compensate for the overcoverage of the FSN grammar implementations is to use a semantic processor to detect and correct invalid sentences. In a sense the semantic processor exploits the fact that the syntax used in recognition has a great deal of overcoverage, i.e. it allows meaningless sentences to be passed to the semantic module. The semantic processor can use the actual task perplexity (generally much lower than the perplexity of the syntax) to convert the recognized output to a semantically valid string (Pieraccini & Lee, 1992).

In theory, the semantic processor should be able to communicate back to the recognizer to request a new string whenever the resulting string from the syntactic FSN is deemed invalid. In practice, one of two simple strategies can be used; either the recognizer can generate a list of the best N sentences ($N = 500 - 1000$) that the semantic processor can search until a semantically valid string is found, or it can assume that the best (recognized) string is semantically "close" to the correct string, and therefore process it appropriately to determine a semantically valid approximation.