Bolt Beranek and Newman Inc.

Report No. 4665

Research on Narrowband Communications

Quarterly Progress Report No. 3
18 February—17 May 1981

Prepared for:
Defense Advanced Research Projects Agency

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: Report No. 4665 ✓
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle): RESEARCH ON NARROWBAND COMMUNICATIONS
5. TYPE OF REPORT & PERIOD COVERED: Quarterly Prog. Rep. No. 3, 18 Feb. - 17 May 1981
6. PERFORMING ORG. REPORT NUMBER: BBN Report No. 4665

7. AUTHOR(s): John Makhoul, Michael Krasner, Salim Roukos, Richard Schwartz, John Sorensen
8. CONTRACT OR GRANT NUMBER(s): F19628-80-C-0165, ARPA Order 3515

9. PERFORMING ORGANIZATION NAME AND ADDRESS: Quarterly progress rept. no. 3, 18 Feb - 17 May 81
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
12. REPORT DATE: March 1981
13. NUMBER OF PAGES: 35

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Deputy for Electronic Technology (RADC/EEV), Hanscom AFB, MA 01731, Mr. Anton Segota, Contract Monitor (1237)
15. SECURITY CLASS. (of this report): Unclassified
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this document is unlimited. It may be released to the Clearinghouse, Dept. of Commerce, for sale to the general public.

(11) May 81

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

This research was supported by the Defense Advanced Research Projects Agency under ARPA Order No. 3515, AMD. 4.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Speech compression, linear prediction, clustering, spectral template, vocoder, hierarchical clustering, unsupervised learning, diphone, phonetic vocoder, phoneme recognition, multiple speaker phonetic synthesis.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

We report on research toward a very-low-rate vocoder. This quarter we continued investigation in three areas. The first area of research is multi-speaker synthesis: speech synthesis from the transmitted vocoder parameters with the voice quality of the vocoder user. This processing entails speaker-specific spectral transformation of the vocoder diphone database. The second area of research is to improve the accuracy of the phonetic recognition. Our work this quarter concentrated on training the

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD-A100100

recognizer by augmentation of the diphone database with diphones extracted from natural, continuous speech. The third area of research is the development of an efficient model of continuous speech. We have developed a novel method of a variable-order Markov chain. We are continuing evaluation of this method.

Report No. 4665

RESEARCH ON NARROWBAND COMMUNICATIONS

Quarterly Progress Report No. 3
18 February - 17 May 1981

Prepared by:

Bolt Beranek and Newman Inc.
10 Moulton Street
Cambridge, Massachusetts 02238

Prepared for:

Defense Advanced Research Projects Agency

TABLE OF CONTENTS

1. OVERVIEW

In this Quarterly Progress Report, we present our work performed during the period 18 Feb. to 17 May 1981.

Our work during the past quarter concentrated on three main topics:

1. synthesis of the voice of a "vocoder user" by speaker-specific transformation of the diphone database;
2. improvement and debugging of the phonetic recognition algorithm; and
3. modeling of speech as a Markov chain to reduce the bit rate necessary for coding of the sequences of the speech spectra.

1.1 Multiple Speaker Synthesis

The input to the phonetic synthesizer is a sequence of phonemes, durations, and pitch values produced by the phonetic recognizer by analysis of the speech of the "vocoder user". The translation of this sequence to frame-by-frame values of spectra and pitch suitable as input to an LPC synthesizer is performed using the diphone database. For each possible diphone, this database contains a sequence of spectra derived from variable-frame rate LPC analysis of a prototypical speaker: the "database speaker". Synthesis using the database will have the same

prosody as the analyzed speech of the vocoder user since the prosodic characteristics of the speech is contained in the phoneme, duration, and pitch sequence from the recognizer. The spectral characteristics of the speech of the vocoder user, however, are not captured in that sequence. Thus, if no modification of the database spectral information is made, the synthetic speech would have the prosodic characteristics of the vocoder user, but the spectral characteristics of the database speaker.

During this quarter, we continued our research on characterization and transformation of the spectral characteristics of speech. As discussed in detail in Section 2, the speaker-specific spectral parameters including long-term average (LTA) magnitude spectrum and vocal tract length (VTL) are estimated for each speaker for each category of speech: voiced, unvoiced, and silence. These spectral parameters are then used for modification of the diphone database spectral sequences. Informal evaluation of the method shows that for some vocoder users, the resultant synthetic speech sounds very similar to the user's actual speech.

1.2 Phonetic Recognition

The phonetic recognition is performed by finding the best path through the diphone network. The basic diphone network is compiled from a set of temporal sequences of spectral parameters, one sequence for each of 2800 diphones. The sequence is generated by variable-frame-rate LPC analysis of diphones extracted from "nonsense" utterances. Phoneme identification accuracy is thus dependent on how well a sequence of diphones in the network, each derived from nonsense utterances, models natural speech. Examination of the diphones show that the "nonsense" diphone prototypes may differ significantly from the diphones' occurrences in natural speech. This difference leads to errors in the phoneme recognition.

A procedure to improve the modeling of natural speech by the diphone network is to use natural speech to "train" the network. The methods of training we have investigated are to modify the "nonsense" diphone template by averaging the spectra with the spectra of occurrences of the diphone from natural speech and by augmenting the network by adding additional diphone paths for occurrences of the diphone from natural speech. The results of investigating these procedures are described in Section 3.

1.3 Modeling of Speech

An important component of the recognition algorithm is the model of speech. Phoneme identification accuracy is directly related to the accuracy to which the model of speech embedded in the algorithm does model the input speech signal. To refine the model, it is desirable to "train" the model with a large amount of natural speech. This task is facilitated by the use of methods that require little human interaction. For this purpose, we have investigated the Markov chain model of speech.

Two Markov chain models are discussed: a first-order Markov chain model and a variable-order Markov chain model. Use of the first order model results in a savings of 1.2 bits per frame (bpf). Since the resultant bit rate is still too high for the vocoder, the novel concept of a variable-order Markov chain was developed. Although preliminary results are encouraging, it is necessary to have a database that is larger than our present database in order to accurately estimate the model parameters. We are currently expanding our database and are continuing research on the variable-order Markov chain model.

2. MULTIPLE SPEAKER SYNTHESIS

We recently completed the design, implementation and testing of a multi-speaker synthesizer. This synthesizer can be used in our present VLR vocoder to produce speech which sounds like the speaker who is talking (the vocoder user) rather than the speaker who produced the database of diphone templates. The basic technique, described in more detail below, can be summarized as follows. A sample of speech from a speaker other than our diphone-database speaker is analyzed for speaker-specific characteristics, including estimated vocal tract length (VTL) and the long-term average (LTA) magnitude spectrum for three classes of speech: voiced (V), unvoiced (UV), and silence (SIL). These speaker parameters, in conjunction with the same parameters for the database speaker, are then used to "reshape" or "transform" the diphone template spectra during synthesis. For about half the speakers tested, the resulting output speech sounds like the new speaker.

2.1 Extracting Speaker Specific Parameters

The first task in this method of multiple speaker synthesis is to extract the speaker parameters from a speech sample. In experimenting with samples of varying length, we have found that

at least twenty seconds of speech (excluding silences) should be analyzed in order to obtain reliable estimates for the speaker parameters.

2.1.1 Estimated Vocal Tract Length

The speech parameter analyzer uses an implementation of an algorithm developed by Paige and Zue[1] to estimate VTL. This algorithm calculates VTL given values of the formant frequencies and bandwidths. These formants and bandwidths are obtained by solving the roots of the all-pole model. The formants are smoothed by several heuristics before being used in the VTL algorithm.

The VTL algorithm will produce reliable VTL estimates when the all-pole model of speech production is valid. Specifically, we want to calculate VTL during voiced, non-nasal, non-"r" colored vowels. Furthermore, the VTL measures will be more reliable near syllabic nuclei, in regions of high total energy. To satisfy these requirements, the analysis program computes an estimate of VTL for a frame of speech only when the following conditions are met:

1. Pitch is non-zero.
2. Total energy and energy in the 1kHz to 3 kHz region are both within 5% of the nearby maximum energy.

3. First and third formants are above specified thresholds.

4. All formants and bandwidths are non-zero.

Estimates for VTL are discarded if they fall outside the range of 10-20 cm, and first-order statistics are kept on the remaining VTL values.

2.1.2 Long-Term Average Spectra

The intent in calculating the long-term average spectra for a speaker is to produce an estimate of the source spectral slope and average vocal tract transfer function for that speaker. The speech analyzer makes a V-UV-SIL decision for each frame based on the energy and number of zero-crossings in the frame. The spectrum for the frame is then averaged in with the other frames of that type. Some smoothing of these spectra is necessary (even though they are average spectra). Currently we utilize a 13-point raised cosine window to smooth a 129-point long-term average spectrum. For each speaker, the result is three LTA average spectra, one for each category of speech: V, UV, or SIL.

2.2 Synthesis Using Speaker-Specific Parameters

The diphone synthesizer needs the speaker parameters of average VTL and LTA Spectra for both the database speaker, whose

speech was used to create the diphone database, and the vocoder-user speaker, whose voice the synthesizer is trying to duplicate. Given these speaker-specific, average speech parameters, and the sequence of phonemes, durations and pitches generated by the phonetic recognizer, the phonetic synthesizer can produce speech which sounds like the vocoder user.

The spectral parameters of the diphone templates are modified by two transformations that are performed together. The spectral transformation accounts for differences in glottal source spectrum and average vocal tract transfer function. This filtering is performed by multiplying each diphone template spectrum by the ratio of the long-term average spectra of the database speaker and vocoder user. We choose the appropriate long-term average spectra from each of the two speakers depending on whether the phoneme being synthesized is voiced, unvoiced or silent. The vocal tract length transformation is performed by scaling the frequency axis of the spectrum by the ratio of the two speakers' average vocal tract lengths. The overall effect of the two transformations is to "remove" the spectral information characteristic of the database speaker and "add in" the characteristics of the vocoder user. In the vocoder operation, the vocoder user's intonation and durational characteristics are already present in the sequence of phonemes, durations, and

pitches produced by the phonetic recognizer. Thus, with the modification of the spectral parameters of the stored diphones, the synthetic speech has both the spectral and prosodic characteristics of the vocoder user.

2.3 Evaluation of Multiple Speaker Synthesis

The results of our effort in multiple speaker synthesis are encouraging, and there are basically two conclusions we can make.

We have analyzed speech samples for about 10 "vocoder users", and have used their long-term average spectra and average vocal tract length to transform the spectral information of phonetically synthesized speech. For speakers whose long-term spectra were markedly different from the database speaker, there is an audible change in the synthetic output, and the speech can sound very similar to the intended speaker. However, some of the "vocoder users", even though they sound quite different from the database speaker, exhibit similar LTA spectra and average VTL. Hence, the transformed speech for these speakers still sounds like the database speaker. We postulate that the differences between these speakers' voices may have to do with features at a more "micro-level", such as particular pronunciations of classes of phonemes, or features such as nasalization. We also know that

dialect causes large changes in voice quality, and it seems that these differences are not always captured by long-term average analysis.

3. PHONETIC RECOGNITION

During this quarter, the work in phonetic recognition consisted of three main efforts. First, we used the training capability of the diphone network compiler to augment the diphone network with alternate diphone paths. We then ran some tests to determine the effectiveness of this training. Second, we made some minor changes in the duration scoring algorithms and added some helpful diagnostics to the program output. Third, we began the large effort of moving all the recognition, synthesis and associated signal processing routines to the VAX.

3.1 Diphone Network Training

As mentioned in the previous QPR[2], we now have a substantial database of 255 sentences of natural speech that have been carefully transcribed (labelled with phoneme boundaries and phoneme identification). This data can be used as "training" for the phonetic recognition program to improve the phoneme identification accuracy of the recognizer.

There are two basic methods of using labelled speech to train the diphone network. One method uses new occurrences of each diphone to modify the diphone spectral template. Then, the

stored template will be a better model for occurrences of the diphone in natural speech. The other method is to augment the diphone network with each new occurrence of each diphone as an alternate path representing the diphone. As reported in the last QPR, the latter method yields higher accuracy phoneme recognition. (A combination of the two methods, however, would be optimal.) Therefore, we divided the training speech into three equal sets of approximately 1500 phonemes each. These were used incrementally to produce three diphone networks with different numbers of alternate paths.

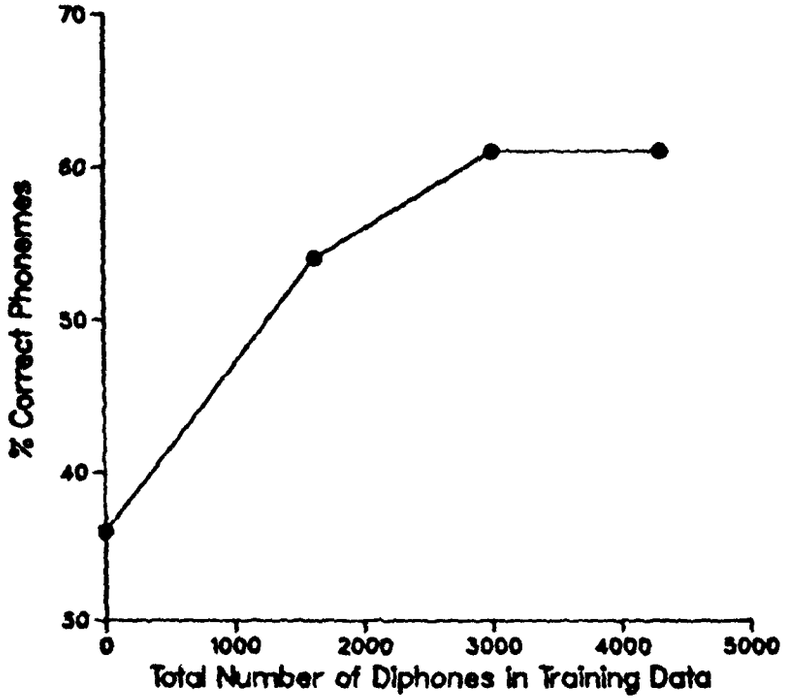
In order to evaluate the effectiveness of this training, we used four different diphone networks in a recognition experiment. The first network had just one sample of each diphone taken from the phonetic synthesis database of nonsense utterances. We shall call this network "untrained." For each of the other three diphone networks, we determined the total number of diphones used to train it, the number of unique diphones used to train it (i.e., the number of diphones for which there was now at least one additional template), and the percentage of correctly recognized phonemes.

The test material consisted of 10 sentences from the Harvard phonetically balanced list. These sentences had <u>not</u> been used in training. The total number of phonemes in the test sentences was 234.

Figure 1 shows the recognition performance as a function of the amount of training. Performance is given as a function of each of the two parameters described above: the total number of training diphones and the number of distinct training diphones. As the figure shows, the recognition performance improves considerably with additional training, improving from a recognition accuracy of 36% correct with no training (the "untrained" network) to 61% correct with 3000 total diphones of training . However, as the last point indicates, further training by the network augmentation method does not seem to make any significant improvement.

Careful examination of the training data indicated that even though only approximately 1200 of the 2800 possible diphones in the network had been augmented by the training with one or more alternate paths, over 90% of those diphones appearing in the test sentences were of diphones that had been augmented by additional paths. Thus, adding additional paths to diphones that were not needed in the test would not help at all. We looked at the subset of phonemes in the test for which two conditions were met: (1) the matcher had correctly identified both adjacent phonemes, and (2) the two diphones that span the phoneme had been trained. That is, if the correct phoneme string in the test sentence were

A B C



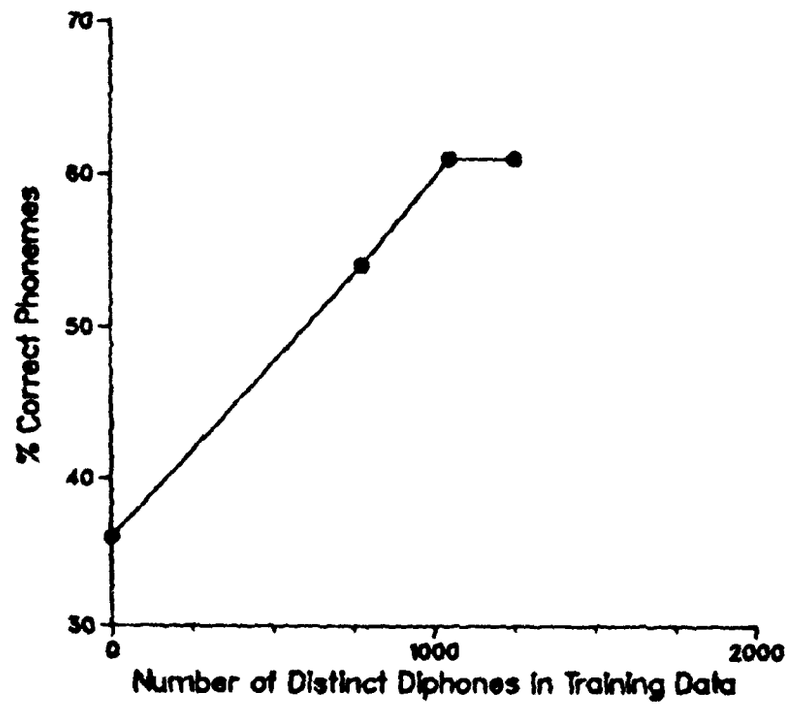


FIG. 1. The effect of training on phoneme recognition accuracy.

we only considered phoneme B if both A and C were correctly recognized, and the diphones A-B and B-C had been augmented by training. In these cases, we found that 85% of the phonemes were correctly recognized. This result indicates that the matcher tends to get long strings of phonemes correct. When a phoneme is incorrectly identified, it will usually be part of a string of several contiguous, incorrectly identified phonemes. Unfortunately, this may be an inherent quality of a matcher such as ours that finds a globally optimal scoring path. We are considering possible steps to alleviate this behavior. One possible solution to this problem is to incorporate into the score of a diphone match the probability of the associated phoneme sequence. This capability was designed as an option in the recognition program, but has not yet been tested. In our feasibility study for this project, we found that by inclusion of first order phoneme statistics (probability of phoneme pairs or diphones) into the recognition process, phoneme identification accuracy improved by 15%. Next quarter we will implement and test the phoneme sequence probabilities as part of the scoring procedure.

There are several conclusions to be drawn from these experiments. First, training alone does not improve phoneme recognition accuracy sufficiently for intelligible vocoded

speech. We need to improve the basic algorithm and spectral distance metric to obtain the desired performance. We have discussed several possible changes in previous QPR's. Second, it appears that the amount of speech necessary for training of the system may be relatively small. Since most of the commonly occurring diphones come from a relatively small subset of all possible diphones, a moderate amount of training data (300-500 sentences) would probably be sufficient.

3.2 Program Changes

In order to evaluate the performance of the network matcher, and to help in detecting problems with the algorithm we added several diagnostic printouts to the matcher output. First, as the best path is reported to the user, the spectral score, the duration score, and which network node the frame was aligned with is printed out for each input frame. In particular, the program types out which training sentence was the source of the data for each node. There are several functions available in the matcher that can be called interactively from the debugger. These functions allow the user to examine a part of the diphone network, or to print out the current theory list. These options make it possible to trace the evolution of particular theories in order to follow the complex program operation.

Another change was made in the way the duration score is added to theories. The duration score is evaluated for each network node. This duration score reflects the probability that a particular number of input spectral frames would be aligned with the network node. (Remember that the network node is the result of a variable frame rate (VFR) spectral analysis.) For any given theory (partial path through the network) the matcher can only compute this score at the point where the theory advances from one node to the next. However, this could cause a large variation in the scores between theories depending only on how recently they progressed from one node to the next. To reduce this variation, the matcher assigns part of this score to the theory with the addition of each frame. The partial score is the expected total duration score for the current node - given the duration so far - minus the duration score already given for this node, divided by the expected number of remaining frames to be aligned with this node. The end result of this change is that the duration scores assigned on each frame vary slowly, and fewer correct theories are accidentally dropped due to a sudden large duration penalty.

3.3 System Problems

One problem that has hampered our progress has been an

operating system bug in the TOPS-20 Release 4 monitor. Since the diphone network is very large, we were using the extended addressing capability afforded by the field test of Release 4 of TOPS-20. This allowed us to use up to 30 PDP-10 address spaces for the network. The largest network we have used so far fills 2 1/2 address spaces.

We spent approximately one month of time during the spring of 1980 converting our programs and internal data format to be able to use this feature. Unfortunately, the official Release 4 monitor no longer allows extended addressing for user programs. We made some quick modifications to the monitor to eliminate this problem. However, this caused other system problems.

These problems have increased the need for us to move our programs to the VAX as quickly as possible.

3.4 Transfer to VAX

We have begun to transfer our recognition and synthesis programs to the VAX. We have decided to adopt PRAXIS as the programming language for our programs presently in BCPL. The similarity between the two languages will ease this process. Another reason for choosing PRAXIS as our new language is that it will be implemented on our Jericho personal computers as a first

step toward implementing ADA. We have, at present, completed the conversion of our FORTRAN library routines to VAX FORTRAN 77. The conversion of our signal processing programs and our PDP11 Real-Time programs is now underway.

While we expect the eventual system on the VAX to be more flexible and easier to use, we will have to spend a substantial effort in converting roughly 9,000 lines of BCPL code, specific to this project, into PRAXIS.

4. A MARKOV CHAIN MODEL OF SPEECH

During this quarter, we have investigated a method based on modelling speech as a Markov chain. The Markov chain was used to model speech as analyzed by a variable frame rate (VFR) linear prediction algorithm. The output of the VFR analysis is a sequence of spectral templates and durations. We investigated how well this sequence is modeled by each of several different Markov models. The primary difference between the Markov models is the order of the model.

We are currently using 64 templates (6 bits) with an average frame rate of 30 fps (frames per sec). Hence, a total of 180 bits would be needed to encode the spectrum. The Markov model will be used to reduce the encoding bit rate without any loss in quality. We will discuss two models: a first-order Markov chain model and a variable-order Markov chain model. The Markov model is used to generate a network of possible spectral sequences as a model of speech. To reduce the encoding bit rate further, similar spectral sequences can be merged, reducing the number of sequences to encode. Merging, however, reduces the accuracy of the model and, hence, the resultant speech quality.

4.1 First Order Markov Model

It is reasonable to assume that not all spectral templates are equally likely to follow a given spectrum. A first order Markov chain model of speech makes use of the dependence to reduce the encoding bit rate. We present in this section a first order, ergodic, and stationary Markov chain model for speech.

Let x_n denote the spectral template at time n. The random variable x_n has 64 possible values. The entropy of x_n, denoted by h_x, is 5.92 bits for our multispeaker database. Let $P(x_{n+1}=j|x_n=i_1, x_{n-1}=i_2...)$ be the conditional probability that the next symbol (spectral template) is j, given the current and past values. Then, the random process $\{x_n\}$ is a first order Markov chain if

$$P(x_{n+1}=j|x_n=i_1, x_{n-1}=i_2...) = P(x_{n+1}=j|x_n=i) = P_{ij}(n) \quad (1)$$

where $i=i_1$. $P_{ij}(n)$ is the transition probability from symbol i to symbol j at time n. If we assume a time homogeneous process, then $P_{ij}(n)=P_{ij}$. The matrix $[P_{ij}]$, $1 \le i, j \le 64$ is called the transition matrix. Let $\underline{p}_0$ be a vector whose components are the probabilities that the initial symbol at time zero has a given value. If $\underline{p}_0$ is an eigenvector of $[P_{ij}]$ with unit eigenvalue,

then the Markov chain is stationary. Further, if $[P_{ij}]$ satisfies some conditions as in Bhat, 1972[3], then the chain is ergodic. We assume that the output sequence of the VFR algorithm is stationary and ergodic. Hence, we need to estimate the transition matrix from an observed sequence of n symbols.

The maximum likelihood estimate of P_{ij} is

$$\hat{P}_{ij} = \frac{n_{ij}}{n_i} \tag{2}$$

where n_{ij} is the number of times symbol j is observed directly following symbol i, and n_i is the total number of times symbol i is observed.[3] For a large n, the random variable $n_i(\hat{P}_{ij}-P_{ij})$ is asymptotically distributed as a Gaussian with a zero mean and a variance of $P_{ij}(1-P_{ij})$. In other words, approximately, $\hat{P}_{ij}$ has a variance of the order of $\frac{1}{n_i}P_{ij}(1-P_{ij})$. A rough estimate of the variance of our estimates can be obtained as follows. We have 64 states and 4096 possible transitions. For the training sequence of 32800 symbols in our speech database, we have an average of 8 observations per transition. Also, a good estimate for n_i is $(32800/64)\simeq 500$. Hence, the average variance of $\hat{P}_{ij}$ is $\frac{P_{ij}}{500}$. The entropy of this first order model was estimated to be 4.74 bits. Hence, the entropy of the Markov model is 1.2 bits less than the entropy of x_n. This substantial savings, however, is not large enough for our application. Thus, we investigated the variable order Markov model described below.

4.2 Variable Order Markov Model

One method to decrease the entropy of the Markov chain is to increase the order of the model. In fact, the conditional entropy of a random variable is monotone decreasing with the number of conditioning variables, i.e.,

$$h(x|y,z) \leq h(x|y). \tag{3}$$

The difficulty in estimating a high order Markov model for speech is due to the limited amount of training data. For a k-th order model, there are N^k possible states for a Markov chain with an alphabet of size N. Further, for every state there are N possible transitions. Hence, we need to estimate N^{k+1} transition probabilities. For N=64 and k=2, we get 256000 transitions. If we require a minimum of 10 observations per transition, we require 20 hours of speech (at 30 fps). The severity of the problem is due to the exponential growth of the number of states with the order of the model. To reduce the training set size problem, we must limit the allowed number of states. Given the amount of training data available, we can determine the maximum number of states our model should have. We will present the variable order Markov model as a method to select the required conditioning states.

4.2.1 Definition of State

We investigated two methods for the selection of the conditioning states. We need to define a new notation to present the two methods. The sequence of spectral templates, $\{x_n\}$, will be considered as a string of letters from the beginning of the alphabet. At time n, the state s_n is a finite length string. For a state of length k, the string is $x_n x_{n-1} \cdots x_{n-k+1}$. We note that for states, symbols are concatenated in a time reversed order.

Let Σ be the set of all states of the model. Σ is a set of strings of letters. In particular, Σ contains the empty state (or string). For the type of states we consider, it is useful to define a tree or network of states as a tool in grouping the states of a model. Every node in the tree has a label that is a possible letter from the alphabet of the Markov chain except the root node. The root node is associated with the null state (the empty string). Further, every node defines a state. The state defined by a node is the string obtained by concatenating the labels of the nodes traversed in going from the root node to the node in question. As one goes deeper in the tree, one is including more of the past. Figure 1 gives the sequence of states s_n for the given state tree and for a typical sequence of the Markov chain.

time	0	1	2	3	4	5	6
VFR, x_n	a	b	a	c	d	a	b
state, s_n		a	null	aba	c	null	ad
state node		2	1	6	3	1	5

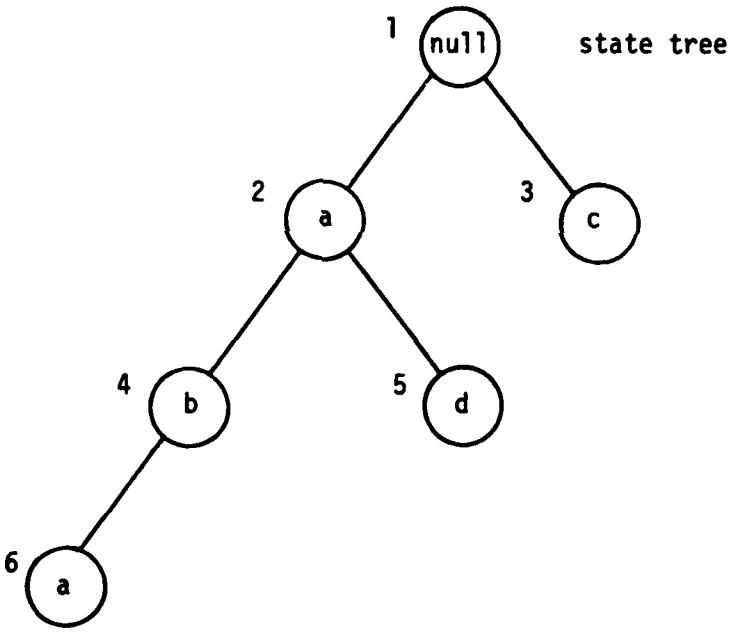


Figure 2. State tree and an example of a sequence of symbols and the corresponding state sequence, s_n, and the state nodes on the tree.

We present in the next section a method for generating a state tree and estimating the corresponding Markov model. However, we should stress that the state tree representation does not allow all possible state sets. Since for every string that is a state, the state tree requires that all its prefixes are also states of the chain.

4.2.2 Variable Order Model Estimation

We present one method for determining a Markov model for speech. The approach is to sequentially add states to the state tree until the required number of states has been found. Initially the state set has the root node (null state) only. The algorithm consists of the following:

1. Initialize the state tree to have one node only: the null state.
2. Using the training data, estimate the transition probabilities of all transitions from the states currently in the tree.
3. Test for highly probable state transition pairs. We used a count of 30 for a specific state transition pair as a threshold (the training data size was 8 counts/transitions). Let s_n and x_{n+1} be such a pair. Create a new state $s'=x_{n+1}s_n$ obtained by concatenating x_{n+1} and s_n.
4. When the number of created states equals the required number of states, then stop adding states and reestimate the transition probabilities using all the training data set. Otherwise, go to Step 2.

We implemented the above algorithm with two variations. In Step 2, it is not clear how much training data should be used before going to Step 3. To see the difficulty, we note that the transition counts for recently created states will be underestimated as compared to older states. One method is to loop through steps 2, 3, and 4 for every observed symbol. Another method is to analyze a block of data, then create a set of new states, then zero all estimates and go to step 2 again. The latter method, though computationally more expensive, results in a model with slightly lower entropy (by 0.1 bit).

For a model with 100 states, we got an entropy of 4.5 bits. The average number of transitions associated with a state was 33 transitions. Hence, given a state, not all spectral templates can follow. Also, 57 of the states were of length one. Those are similar to the 64 states of a first-order Markov model. The states of length 2, 3 and 4 were 36, 5 and 2, respectively.

Due to the limited amount of training (we had 10 observations per transition), we did not investigate the full potential of this method yet. We are planning to acquire a large database to investigate a model with around 1000 states. At the moment, the model with 100 states is not significantly different from a first-order model.

The state of a variable order model can also be interpreted as an equivalence class of states of a high order, yet fixed, Markov model. Let k be the length of the longest state in the variable order model. Consider a k-th order Markov chain. Let s be a state of the variable order Markov model. An equivalence class of the states of a k-th order model consists of all strings that have s as prefix but no state, in the state tree, longer than s as a prefix. Then the fixed k-th Markov model with the equivalence classes used for conditioning is exactly the variable order model we described earlier. This notion of grouping states into an equivalence class to get a reduced state set for a fixed order Markov model will be used to generate another model for speech.

4.2.3 Variable Resolution States

A state of the variable order Markov model can be considered as an equivalence class that is effective in conditioning the next possible symbol. The purpose of the modeling is to find the minimal number of equivalence classes (or states) needed to condition speech to get the lowest entropy. One method of decreasing the number of states with minimal loss in the effectiveness of state conditioning is based on a variable spectral resolution representation of the states. The idea is

that strings that differ only in the "remote" past by small spectral distances should belong to the same equivalence class. One method to implement the above is to use a different size codebook (set of spectral templates) for the symbols in a state string that depends on the position of the symbol in the string. The codebook size decreases as the position corresponds to a more distant past. Hence, the spectral resolution decreases with the past. For example, let $s=x_1x_2x_3$ be a state string. Then x_1 has 64 possible values (6 bits), x_2 has 32 values (5 bits) and x_3 has 16 values (4 bits) is one way of defining the equivalence classes. On the state tree, this means the number of labels of a node depend on the level of the node. This number decreases as the level of the node increases.

We tested this method with a codebook size of 32 at level 1 and size 16 at levels 2, 3, 4 and 5. The resulting entropy for a 100 state model is 4.81. The average number of transitions per state was 42. Thus, the entropy has increased. Hence, even though this method allows more higher-order states, the loss in spectral resolution for the current symbol, from 64 templates to 32, reduces the effectiveness of predicting the next symbol.

The two approaches to get a Markov chain model for speech have yet to be tested with a large database. In the next quarter, we will be acquiring this database. We will also

investigate a method of spectral sequence clustering that we are currently developing.

REFERENCES

1. Paige, A. and Zue, V., ``Calculation of Vocal Tract Length,'' IEEE Trans. on Audio & Electroacoustics, Vol. AU-18, No. 3, Sept. 1970, .

2. Makhoul, J. Krasner, M., Roukos, S., Schwartz, R. and Sorensen, J., ``Research on Narrowband Communications,'' Tech. report Quarterly Progress Report No. 4620, Bolt Beranek and Newman Inc., 18 Nov. 1980-17 Feb. 1981.

3. Bhat, U.N., Elements of Applied Stochastic Processes, John Wiley, New York, 1972.