Bolt Beranek and Newman Inc.

Report No. 4557

Research on Narrowband Communications

Quarterly Progress Report No. 1
18 August—17 November 1980

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. 4557
2. GOVT ACCESSION NO.: AD-A102 706
3. RECIPIENT'S CATALOG NUMBER: (progress)

4. TITLE (and Subtitle): RESEARCH ON NARROWBAND COMMUNICATIONS

5. TYPE OF REPORT & PERIOD COVERED: Quarterly Progress Report (no. 1), 18 Aug — 17 Nov 80

6. PERFORMING ORG. REPORT NUMBER: Rep. No. 4557

7. AUTHOR(s): John Makhoul, Richard Schwartz, Salim Roukos

8. CONTRACT OR GRANT NUMBER(s): F19628-80-C-0165, ARPA Order-3515

9. PERFORMING ORGANIZATION NAME AND ADDRESS:
Bolt Beranek and Newman Inc.
10 Moulton Street
Cambridge, MA 02230

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 12 36

11. CONTROLLING OFFICE NAME AND ADDRESS: 14 BBN-4557

12. REPORT DATE: November 1980

13. NUMBER OF PAGES: 32

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

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, Department of Commerce, for sale to the general public.

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.

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

This document reports on work toward a very low rate vocoder. We model speech as a Markov Chain of spectral templates for the unsupervised learning approach to very low rate vocoding. This quarter we compared several clustering techniques. We determined that a hierarchical approach to clustering is economical with minimal loss in performance. Furthermore, we found that a small number of spectral templates (from 128 to 256) is sufficient for vocoding with good

(over)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

060100 LB

intelligibility. Also, in the phoneme recognition approach, we automated the training of the diphone network on labelled speech.

Report No. 4557

RESEARCH ON NARROWBAND COMMUNICATIONS

Quarterly Progress Report No. 1
18 August to 17 November 1980

Prepared by:

Bolt Beranek and Newman Inc.
50 Moulton Street
Cambridge, Massachusetts 02138

Prepared for:

Advanced Research Projects Agency

TABLE OF CONTENTS

1. SUMMARY

In this Quarterly Progress Report, we present our work performed during the period August 19, 1980 to November 18, 1980.

1.1 Introduction

The work in the past quarter was in the areas of phonetic recognition for a very-low-rate (VLR) vocoder, and spectral clustering as a first step toward the unsupervised learning approach to VLR vocoding. Below is a summary of the accomplishments in each of these areas. Details are presented in Sections 2 and 3.

1.2 Spectral clustering

An important phase of our work in this contract is to develop methods for automatically generating the recognition and synthesis networks for the VLR vocoder. This will be desirable to make adaptation of these systems to different speakers and environments practical. One of the principal methods being investigated is to automatically construct a Markov model to represent possible spectral sequences. As a first step, this requires the ability to classify each input spectrum as one of a (relatively) small number of spectral types. This is called spectral clustering.

During this quarter we implemented a variety of spectral clustering algorithms. We also made a few significant improvements in the basic algorithms in order to make the clustering more efficient (i.e., to reduce the number of spectral templates necessary to represent speech.)

Some of these improvements were:

- The use of a non-uniform binary tree to allow selective division of the clusters.
- The use of a simple Euclidean distance on Log-Area-Ratio (LAR) parameters was shown (through informal listening tests) to be as good an error measure as the Itakura-Saito distance measure.
- It was shown that the adaptive binary clustering performs as well as the so-called "optimal" K-means algorithm, which would be far too expensive to implement.
- The use of variable-frame-rate transmission in combination with spectral clustering greatly improves the quality of synthesized speech.
- The results of clustering yield an interesting interpretation in terms of the "dimensionality" of spectral data.

Before using the spectral cluster definitions to generate a Markov model, we needed to know that this spectral quantization *does not affect the intelligibility* of the speech too much. Therefore, the spectral templates were used to quantize speech spectra in a pitch-excited LPC vocoder. The results seem adequate for the VLR vocoder.

1.3 Phonetic Recognition

The most significant change to the phonetic recognition program this quarter was to automate the procedure for training the diphone network on labelled (phonetically transcribed) speech. Also, debugging aids were added to enable us to find bugs in this very complex system. Several significant bugs were detected and corrected.

In preparation for training the diphone network, we also manually transcribed a large data base of speech.

A significant amount of time had to be spent to get back up to speed in the program development due to the break in the research between June 6 and August 19, 1980. Recovering from this lapse was particularly difficult in light of the fact that one of the key personnel involved in the programming of the system left BBN on June 6. Consequently, some time had to be spent for other personnel to become familiar with the many programs involved. At this time, this familiarization phase is mostly complete - as evidenced by the successful detection and correction of many program bugs.

The phonetic recognition work is discussed in more detail in Section 3.

2. CLUSTERING OF SPEECH SPECTRA

One of our primary tasks in this contract is to develop a method of very-low-rate (VLR) vocoding that does not require large amounts of transcribed speech and human direction. The idea here is that the program should have an algorithm such that upon being presented with large amounts of analyzed speech data, it can produce a model sufficient for VLR coding of speech. This should be possible with little or no interaction from a human once the process has begun. We call this general method "Unsupervised Learning".

A useful representation of speech spectra, for the unsupervised learning approach to speech recognition, is obtained by using a finite set of spectral templates. This representation allows the modelling of speech as a Markov chain; this model can be used to identify important sequences of spectra that will be considered as basic speech units. Also this representation is very economical since we only need to deal with a template label rather than a model spectrum. We believe that as few as 128 to 256 spectral templates may be adequate for a single speaker. One method for automatically determining these templates is clustering. The remainder of this section will therefore be devoted to clustering of LPC speech spectra.

2.1 Introduction to Clustering

The purpose of clustering is to reduce the continuum of possible speech spectra to a finite set of spectral templates. All spectra that belong to the same cluster are represented by the same template. A common and simple definition of a cluster is based on a minimum distance classification rule: all spectra that are nearest[1] to a template are assigned to the cluster corresponding to that template. Usually a training set of spectra is used to define the partition into clusters. We shall consider two methods for clustering the training data into K clusters. Both methods optimize a "clustering criterion", that is, a measure of "goodness" of the resulting clusters.

A partition $P=\{C_i\}_{i=1}^{K}$ of the continuum of spectra into K clusters is defined by a minimum-distance rule[2]. Each cluster C_i is represented by a template $\underline{z}_i$; the vector $\underline{z}_i$ is an ordered set of parameters that define a model spectrum (14 log area ratios in our case). A spectrum $\underline{x}$ belongs to a cluster C_i if:

---

[1] using an appropriate distance or dissimilarity measure.

[2] ties are resolved appropriately

$$d(\underline{z}_i,\underline{x}) \leq d(\underline{z}_j,\underline{x}) \qquad 1\leq j\leq K \tag{1}$$

where $d(\underline{z},\underline{x})$ is a distance function from the template $\underline{z}$ to the spectrum $\underline{x}$. We use the word distance in a loose sense, since we only require that $d(.,.)$ is nonnegative and finite (e.g., it can be nonsymmetric). The resulting clusters will have piecewise linear boundaries (hyperplanes) if a Euclidean distance is used. For every cluster C_i, we define a measure of the spread within the cluster as the cluster average distance

$$D(C_i) = \frac{1}{M_i} \sum_{\underline{x}\epsilon C_i} d(\underline{z}_i,\underline{x}) \tag{2}$$

The summation is for all $\underline{x}$ in the training data classified to belong to cluster C_i; M_i is the number of training spectra assigned to C_i by (1). One could use other definitions for cluster spread, but this is the only one we consider. The clustering criterion used to evaluate a partition P is defined by:

$$J(P) = F(M_1, M_2, \ldots, M_K, D(C_1), D(C_2), \ldots D(C_K))$$

The function F may, in general, depend on other variables: Usually, we shall consider the following criterion

$$J(P) = \frac{1}{M} \sum_{i=1}^{K} M_i D(C_i) \tag{3}$$

Other choices will be discussed later.

The clustering problem, therefore, is to find the partition (or the templates) that minimize the clustering criterion, J(P), by using a training set of M speech spectra. In the next two sections, we discuss two different clustering algorithms.

2.2 The Optimal Method: K-means algorithm

This is a hill-climbing algorithm that updates the spectral templates by minimizing the clustering criterion. Below, k is the iteration index and $\underline{z}_i(k)$ is the value of $\underline{z}_i$ at iteration k. The steps of the algorithm are:

1. Initialization: Set k=0. Choose by an adequate method a set of initial spectral templates $\underline{z}_i(1)$ for $1 \leq i \leq K$.
2. Classification: $k \leftarrow k+1$. Classify all the training data (M spectra) to the corresponding nearest template. This defines the clusters $C_i(k)$:

$$\underline{x} \in C_i(k) \quad \text{if} \quad d(\underline{z}_i(k), \underline{x}) \leq d(\underline{z}_j(k), \underline{x}) \qquad 1 \leq j \leq K$$

3. Template Updating: Update the template of every cluster using all model spectra assigned to that cluster in Step 2. For cluster i, the new template $\underline{z}_i(k+1)$ is the vector $\underline{z}$ that minimizes the cluster average distance given by

$$D(C_i(k)) = \frac{1}{M_i(k)} \sum_{\underline{x} \in C_i(k)} d(\underline{z}, \underline{x})$$

4. Termination Test: If the templates $\underline{z}_i(k+1)$ are significantly different from $\underline{z}_i(k)$, go to Step 2; otherwise, stop.

The above algorithm can be shown to converge. However, the K-means algorithm may converge to a local optimum. A classical solution to get global optimality has been to use different sets

of initial templates (Step 1), and then to choose the best final result.

2.2.1 Variants of the K-means algorithm

Several variants of this algorithm have been proposed. We note that, to guarantee convergence, we only need to reduce J(P) iteratively. The following are alternative modifications:

1. Instead of classifying all spectra before the templates are updated, the templates are updated as each speech spectrum is classified.

2. The update rule can be modified to:

$$\underline{z}_n(k+1) = \underline{z}(k) + \alpha(z(k+1) - z(k))$$

where $\alpha > 1$ and $z_n(k+1)$ is used as a new template. Basically, the idea is to increase the size of the template correction hoping to achieve faster convergence and/or global optimality. However, in this case, one cannot guarantee convergence.

3. Finally, Hartigan [1] proposes to combine both steps 2 and 3 in the case of Euclidean distance. For every datum $\underline{x}$, let $x \in C_i(k)$ and consider

$$\min_{\substack{1 \le j \le K \\ j \neq i}} \left(\frac{M_j(k)\, d(\underline{z}_j(k), \underline{x})}{M_j(k)+1} - \frac{M_i(k)\, d(\underline{z}_i(k), \underline{x})}{M_i(k)-1} \right)$$

If the above minimum is negative, we reclassify $\underline{x}$ to cluster C_j that corresponds to this minimum. We also update both templates of clusters C_i and C_j. Basically, the above quantity is the variation in the clustering criterion due to the reclassification of $\underline{x}$ from C_i to C_j. For a general distance function, and the corresponding clustering criterion, one has to compute the change in the criterion and iterate accordingly.

It is not clear that the above algorithms differ

significantly from each other. However, we shall use the algorithm described in Section 1.2 because it is slightly simpler and requires less computations. We have not discussed how the initial templates are chosen. There are several methods as discussed in Hartigan [1] and Anderberg [2]. The most promising was suggested by Duda and Hart [3]. It consists of a hierarchical clustering algorithm to obtain the templates. We discuss this next.

2.3 Hierarchical Clustering

The K-means algorithm suffers from two drawbacks:

1. It has a large computational load; for M training spectra and K clusters, MK distance calculations are required per iteration.
2. A reasonably good initial estimate of the K templates is required to have reasonable confidence in the optimality of the solution.

In fact, the hierarchical method has been suggested to obtain the initial estimate for the K-means by Duda and Hart [3]. Also, it does require fewer computations, as we discuss later. The basic idea is to impose a hierarchy (grouping) on the clusters; each cluster is contained in a larger cluster. In the binary hierarchical case, the training data is originally divided into two clusters. Then each cluster is further subdivided into two clusters successively until we have the desired number of

clusters. This process defines a binary tree: each node of the tree is associated with a cluster and the corresponding template. When a cluster is divided into two, then the node will have two sons (two nodes branching from this node).

We associate with every node N_j of the tree a value given by the function $f(N_j)$. We call $f(\cdot)$ the propagation function. This propagation function is used to decide which cluster is divided. We choose the terminal node with the largest value for $f(N_j)$ as the next node for propagation. When the required number of cluster is obtained, we evaluate the clustering criterion given by (3); the value of the clustering criterion is used to assess the usefulness of the templates as was done for the optimal algorithm.

A cluster is divided in two clusters by using the K-means algorithm (K=2). There are several choices for the propagation function; but since we propagate the node with the largest $f(N_j)$, we obtain $f(N_f)$ as an upper bound on f for all clusters. N_f is the last divided cluster. Suppose we use (3) to define the clustering criterion; if for the propagation function, we use $f(N_j) = D(C_j)$ as given by (2), then we can draw that the resulting clusters minimize J(P), the clustering criterion, under the constraint of a binary tree on the clusters. This is true

because both steps (the subdivision of a cluster and tree propagation) minimize the criterion. For other choices of $f(N_j)$, $J(P)$ *is not minimized and in fact,* as we shall discuss later, $J(P)$ will be far from optimum.

2.4 Distance Functions

There are several choices for a distance function. In fact, we propose to use clustering as a method to evaluate the usefulness of the different distance functions.

2.4.1 Euclidean distance

Let a speech model spectrum $\underline{x}$ be represented by L log area ratios (we use L=14). The (squared) Euclidean distance between two model spectra $\underline{x}$ and $\underline{y}$ is

$$d(\underline{x},\underline{y}) = (\underline{x}-\underline{y})'\underline{x}-\underline{y}) = \sum_{i=1}^{L}(x_i - y_i)^2$$

where $'$ denotes transpose and x_i is the i-th component of the vector $\underline{x}$. The spread of a cluster as defined by (2) is the mean square distance. Given a set of spectra that belong to the same cluster, then a template that minimizes the spread is given by the center of mass of the cluster; i.e., the arithmetic mean:

$$\underline{z}_i = \frac{1}{M_i} \sum_{\underline{x}\epsilon C_i} \underline{x}$$

The general weighted Euclidean distance is

$$d_1(\underline{x},\underline{y}) = (\underline{x} - \underline{y})'A(\underline{x} - \underline{y})$$

where A is a positive semi-definite matrix. It is well known that A can be factored as A=T'T. Defining $\underline{x}_1 = T\underline{x}$ and $\underline{y}_1 = T\underline{y}$, we have

$$d_1(\underline{x},\underline{y}) = (T\underline{x} - T\underline{y})'(T\underline{x}-T\underline{y}) = (\underline{x}_1-\underline{y}_1)'(\underline{x}_1-\underline{y}_1) = d(\underline{x}_1,\underline{y}_1)$$

Hence, we can implement this general metric by using the Euclidean distance on the transformed vectors $\underline{x}_1$ and $\underline{y}_1$.

2.4.2 Itakura-Saito Distance

This distance is based on a maximum likelihood approach. Several distance functions have been suggested; namely:

$$d_1(\underline{z},\underline{x}) = \frac{\underline{z}'R_x\underline{z}}{\underline{x}'R_x\underline{x}}$$

$$d_2(\underline{z},\underline{x}) = \ln d_1(\underline{z},\underline{x})$$

$$d_3(\underline{z},\underline{x}) = (\underline{z} - \underline{x})' R_x(\underline{z} - \underline{x})$$

where $\underline{z}$ is the all zero inverse filter. An interesting problem is the center of the mass determination for a cluster. The spread of a cluster as defined in (2) is:

$$D(C) = \frac{1}{M} \sum_{\underline{x}\in C} d(\underline{z},\underline{x})$$

Using the distance $d_1(\underline{z},\underline{x})$ we get:

$$D(c) = \underline{z}' \frac{1}{M} \sum_{\underline{x}\in C} \frac{R_{xn}}{V_p} \underline{z}$$

$R_{xn} = R_x/R_0$ is the normalized autocorrelation matrix and V_p is

the normalized error. If d_3 is used, then

$$D(C) = \underline{z}' \frac{1}{M} \sum_{x \in C} R_x \underline{z}$$

In both cases, the template is the inverse filter of an average autocorrelation matrix (a weighted average in the first case). However, we note that d_3 does not have a gain normalization. Hence, normalized autocorrelation matrices are used when the distance d_3 is used.

2.5 Tests and Results

2.5.1 Binary Clustering

We discuss first the hierarchical (binary) clustering algorithm. We used the Euclidean distance defined on log area ratios. The tree is propagated by selecting the node or cluster with the largest intracluster error defined by:

$$E(N_j) = \frac{1}{\sqrt{M_j}} \sum_{\underline{x} \in C_j} d(\underline{z}_j, \underline{x}) = \sqrt{M_j}\ \overline{e^2}(N_j)$$

where $d(\underline{z}_j, \underline{x})$ is the squared Euclidean distance, C_j is the cluster associated with node N_j and has M_j spectra. The final set of clusters is used to evaluate the mean square error of the partition:

$$\overline{e^2} = J(P) = \frac{1}{M} \sum_{C_j \in P} \sum_{x \in C_j} d(\underline{z}_j, \underline{x})$$

P is the set of final clusters, M is the total number of training spectra (in our case, M = 5000 frames).

We performed the above clustering to obtain 2^n clusters for n = 1,2,...8. Note that by increasing n we propagate the same tree. Hence, by increasing the number of clusters, we are only refining the clusters. We show in Fig. 1 the tree for n=4 bits and the subtrees that correspond to n=1,2,3. Table 1 gives the mean square error and the corresponding values of n.

n	0	1	2	3	4	5	6	7	8
$\overline{e^2}$, L=14	163.9	98	70	50	38	31.5	25.9	21.6	17.5
$\overline{e^2}$, L=10	155.9	91.2	62.8	43.5	32.5	26.1	20.7	16.7	13.3
$\overline{e^2}$, L=6	143.18	79.4	49.3	33.2	23.4	17.3	13.2	9.76	7.31

TABLE 1. Mean square error for binary clustering.

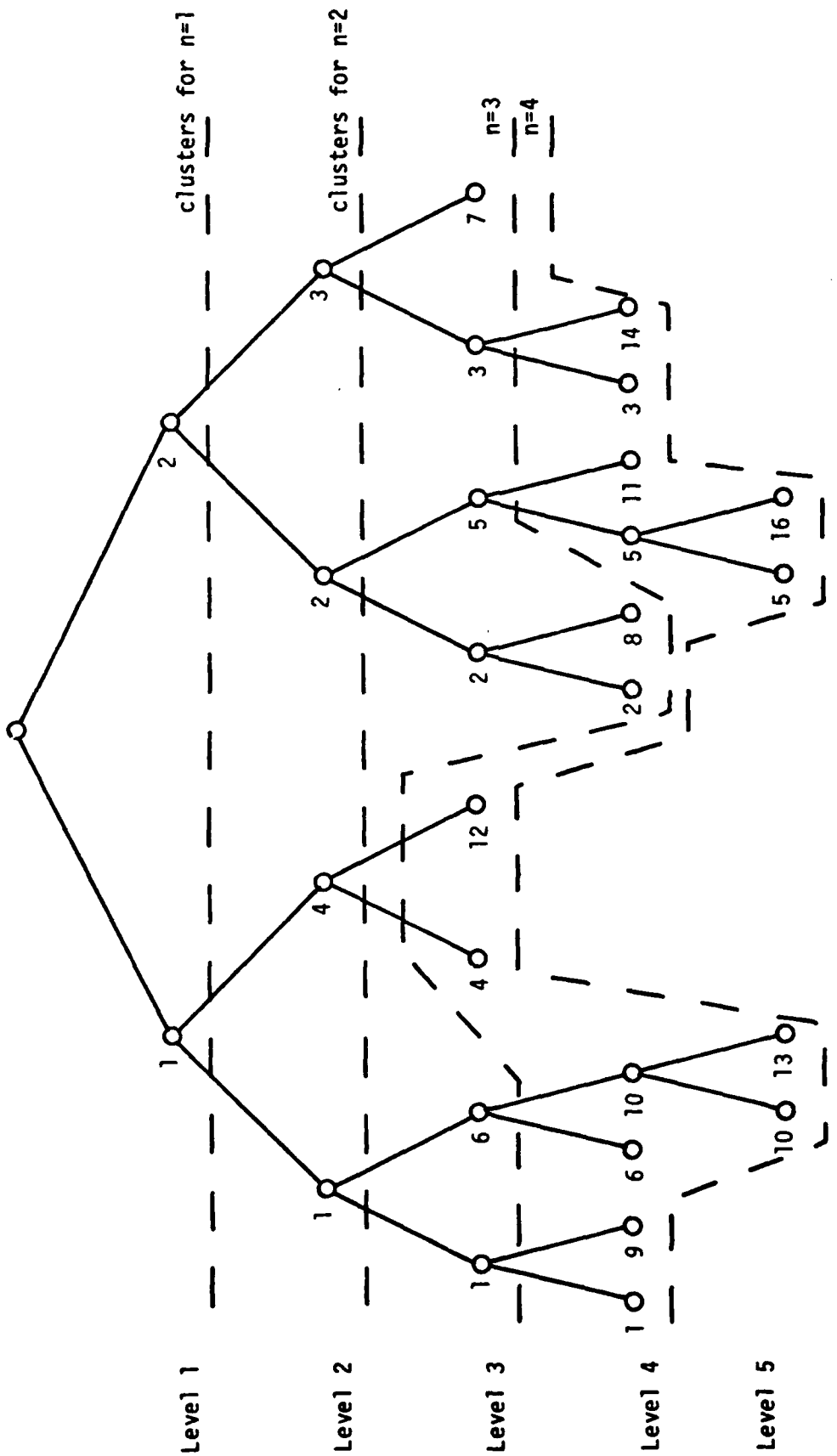

FIG. 1 Tree of binary clustering. The terminal nodes above dotted boundary represent the clusters for the corresponding value of n (number of bits).

To better understand how the mean square error varies as the number of clusters increase, we consider a L-dimensional cube with a uniform distribution within it. Let a be the side of the cube, then its volume is a^L. The mean square error from the center of mass of the cube is

$$\overline{e_v^2} = \frac{L}{12} a^2$$

Suppose we divide the cube into 2^n equal cubic cells (for the cells to fit, n must be a multiple of L). The volume of each cell is

$$v_c = \frac{a^L}{2^n} = \left(\frac{a}{2^{n/L}}\right)^L$$

The corresponding mean square error from the center of mass of each cubic cell is:

$$\overline{e_c^2} = \frac{L}{12} \frac{a^2}{2^{2n/L}} = (0.25)^{n/L} \overline{e_v^2}$$

If we consider the error in dB, we get:

$$10 \log e_c^2 = \frac{n}{L} 10 \log (0.25) + 10 \log e_v^2$$

$$e_f = - \frac{n}{L} 6.03\text{dB} + e_0$$

Hence, the error in dB decreases linearly with n with a slope of $- \frac{6.03}{L}$ dB. For L=14, the slope should be -0.43 dB/bit.

In Fig. 2, we show the error in dB for binary clustering for two cases L=14 and L=6; L is the order of the LPC. In both cases, there is a linear decrease in error beyond n=4. However, the slope is -0.86 dB/bit for L=14 instead of 0.43 dB/bit for the cube example, this corresponds to clusters with an intrinsic dimensionality of 7. For L=6, the slope was -1.29 dB/bit for a corresponding dimensionality of 4.9.

Finally, we note that for n=1,2,3, the mean-squared error was significantly reduced; presumably because natural clusters were obtained.

2.5.2 Tree Propagation Rules

We used other propagation rules by using different intracluster error functions, namely:

$$E_1(N_j) = \overline{e^2}(N_j) \qquad \text{(mean square error)}$$

$$E_2(N_j) = M_j\overline{e^2}(N_j) \qquad \text{(total squared error)}$$

$$E_3(N_j) = \overline{R}_0^n\ \overline{e^2}(N_j) \qquad \text{(energy weighting)}$$

where R_o is the average of the logarithm of energy of all training data in the cluster; we used n=1,2 or 3. The choice of n was unimportant, as determined by subjective listening tests. E_1 and E_3 were found to be only slightly better than E_2. The

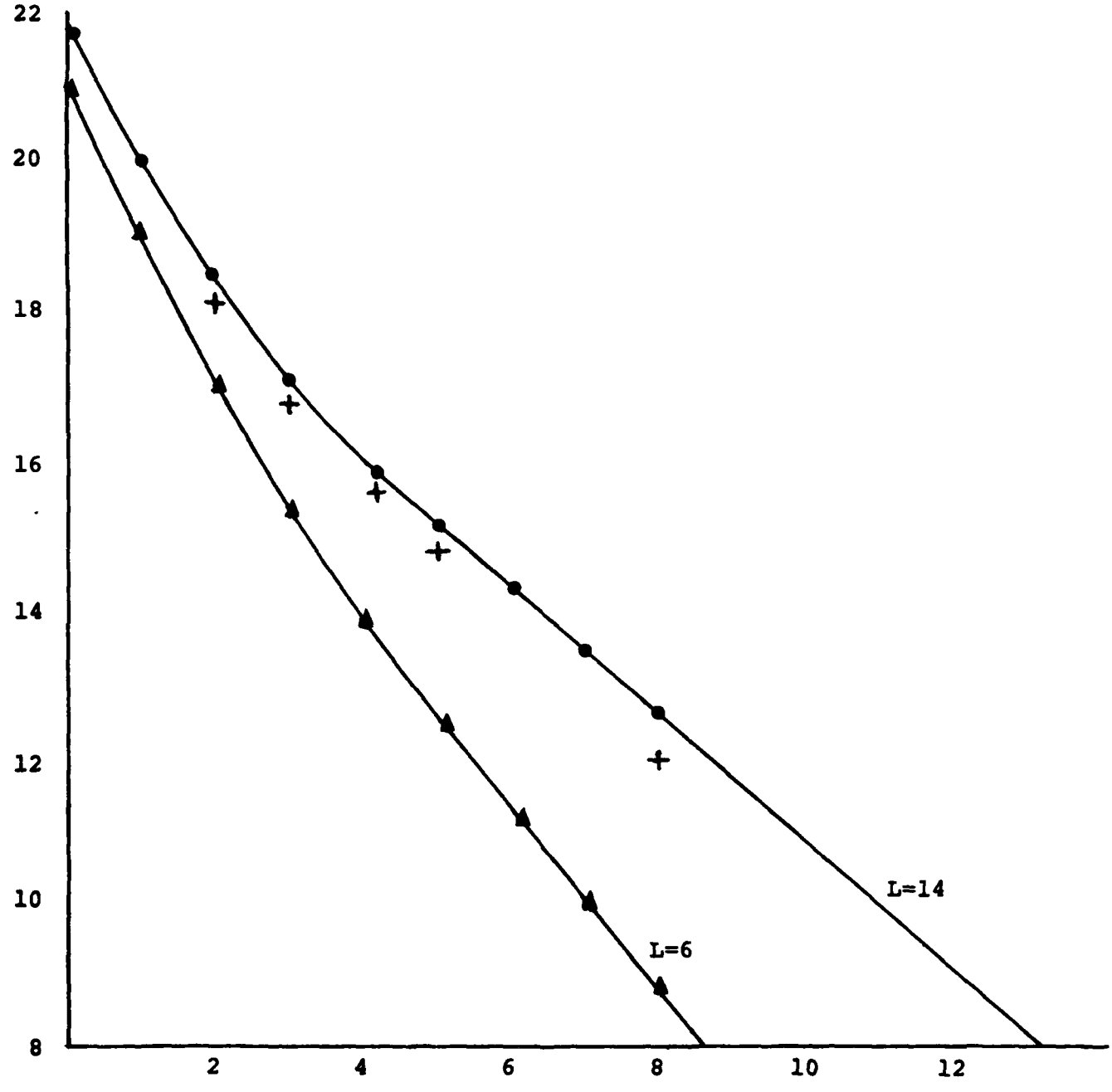

FIG. 2. Mean square error in dB of log area ratios (LAR). ● Binary tree for 14 LAR, ▲ Binary tree for 6 LAR and + optimal for 14 LAR.

tests were conducted using 256 clusters. It is important to realize that, independent of what propagation function is used, the final templates will be the same if the final tree is the same. This is true because the cluster division algorithm (K-means) is unchanged. At 256 clusters, E_1 and E_2 result in very similar trees. We feel the tests have to be done using more clusters to allow the trees to be different. However, because of storage, we cannot implement adequately more than 256 clusters. That will change when we move to the VAX.

To conclude the discussion on binary clustering, we present some of the statistics of the clusters obtained above. The results are tabulated in Table 2. Each row indicates the value of the corresponding parameter such that a certain percentage of the clusters (indicated in first column) have an equal or higher value for that parameter. Three parameters are tabulated. These parameters are the number of levels of the tree (LVL) to get to the cluster, the cluster average distance (D) (cluster mean square error) and the number of spectra (n) in a cluster. Further, we show the statistics for 3 choices of the propagation function and we show the value of the clustering criterion (J) for each set of templates.

Because of the above cluster statistics (25% of clusters in

some cases have less than four training spectra), we increased the training data set by packing the data in core memory (around 15000 frames). However, the required unpacking slowed execution by a factor of 45. It takes 45 minutes of CPU for the binary program to execute instead of 1 minute. The result is improved statistics for those poor 25% clusters as shown in Table 3. Also, the new templates are marginally better on new speech not used in training but from the same speaker. This indicates that in the future we should use the larger training set ($M \gtrsim 15000$). One final remark regarding binary clustering, is that we do not know how far from optimum we are. In the next section we shall discuss the optimal algorithm.

2.5.3 The Optimal Algorithm

We use the templates obtained from the binary clustering algorithm as the initial templates for the K-means algorithm. Note that the templates were obtained from a $N\ \overline{e^2}$ propagation. In Fig. 2 we show the mean square error for the optimal algorithm with Euclidean distance for several numbers of clusters. The reduction in the mean square error from the binary tree case is small. Furthermore, there is little perceptual difference. This small improvement does not warrant the excessive computational load of the K-means algorithm. Hence, in the future we shall use the binary clustering method only.

Propagation Function	$\overline{Ne^2}$			$\overline{e^2}$			$\overline{R_0 e^2}$		
	LVL	D	n	LVL	D	n	LVL	D	n
5%	10	32.0	35	11	23.6	66	12	28.1	53
25%	9	23.6	22	10	22.1	19	10	20.6	21
50%	8	19.4	17	9	20.0	9	9	17.3	10
75%	8	15.7	13	8	16.1	4	8	13.2	4
95%	7	10.4	8	7	0	1	6	0	1
	J = 17.38			J = 20.65			J = 22.65		

TABLE 2. Cluster Statistics.

	LVL	D	n
5%	11	30.6	37
25%	10	23.0	20
50%	9	18.7	15
75%	8	15.2	11
95%	7	11.6	6

TABLE 3. Large Training Set (15000 frames). The propagation function is $R_0 \sqrt{N}\, \overline{e^2}$. J = 18.54.

We also used the Itakura-Saito distance, $d_3(x,y)$, since it is widely used. From listening experiments, we did not find any significant difference from the Euclidean distance on log area ratios. However, we still feel that those results are still inconclusive until we reach about 10000 clusters. We need the address space of the VAX to accomplish this task.

2.5.4 Spectral Quantization

The principal method for evaluating the different clustering results was by conducting informal listening tests. Basically, for every frame (100 frames/sec) we used pitch, gain and a spectral template to resynthesize speech from the same speaker. We used as a test set a total of 6 sentences: 3 new sentences and 3 sentences from the training data. The spectra used for synthesis can be obtained by either of two methods for spectral quantization.

Method 1: Optimal.

Given the analyzed frame spectrum, we replaced it by the nearest template. We used the same distance that was used in obtaining the templates in clustering. The disadvantage of this method is that 2^n distance calculations are required for 2^n clusters. However, it can be used for templates obtained by either optimal or binary clustering.

Method 2: Hierarchical.

Given the analyzed spectrum and the binary tree (the hierarchy on the clusters), we classify it by successive dichotomy. (This in general will not yield the nearest template. However, only n distance calculations are required on the average for 2^n clusters.) This method can only be used with the binary clustering templates.

The two methods give almost identical performance: The hierarchical spectral quantization resulted in slightly higher mean square error (within 6%) than the optimal quantization method. But the listening tests show their equivalence. Since in both clustering and quantization, the performance degradation is small for the hierarchical approach as compared to the optimal method, we shall be using the former in the future.

2.5.5 Variable Frame Rate

Since we only used 256 spectral templates, large "discontinuities" were artificially introduced in the spectrum from frame to frame. These discontinuities degraded speech quality to the extent that spectral differences due to clustering were effectively masked. Hence, to perform reasonable listening tests, we implemented a spectral smoothing algorithm to improve quality by reducing false sharp transitions. This is basically

the variable-frame-rate algorithm. The idea is to select which frames are to be replaced by spectral templates, with linear interpolation (on log area ratios) for the spectrum between those frames. The choice of the location of the template is determined sequentially by a threshold on an average error. The average error is

$$e^2(n,m) = \frac{1}{m-n} \sum_{i=n+1}^{m} d(\underline{z}_n + \frac{i-n}{m-n} (\underline{z}_m - \underline{z}_n), \underline{x}_i)$$

This is the average error between the true speech spectrum $\underline{x}_i$ and the linearly interpolated spectrum (note d(.,.) is a squared distance); $\underline{z}_n$, $\underline{z}_m$ are the templates used at times n, m. The time index m is chosen when $e^2(n,p)$ exceeds the threshold for 3 consecutive time points (i.e. p=m+1, m+2 and m+3). Linear interpolation (on log area ratios) between the templates at times n and m is used to obtain the spectra for synthesis.

This smoothing operation improved speech quality significantly, and we routinely used it in our listening tests to evaluate different clustering results.

Based on our experiments with clustering, we have found that representing LPC spectra by a small number of templates preserves the intelligibility of the speech. Therefore, we feel

comfortable with using this representation to determine a Markov chain for speech spectra. We will next examine methods to derive this chain automatically.

3. PHONETIC RECOGNITION

The VLR phonetic vocoder is intended to transmit speech at about 100 b/s that would be intelligible within the context of a conversation. The transmitter analyzes input speech in terms of a sequence of phonemes - each with an associated duration and pitch target value. This information, which can be transmitted with minimal loss at 100 b/s, is used by the receiver as input to a phonetic synthesizer to generate speech.

The phonetic synthesizer has been completed (for a single speaker) under the previous contract [4]. The synthesizer produces speech by concatenation of diphone templates. A diphone is defined as the region from the middle of one phoneme to the middle of the next. A diphone template consists of the parameters necessary for LPC synthesis of a diphone (LAR parameters, gain, voicing, cutoff frequency). The speech produced by the phonetic synthesizer is highly intelligible, and has a naturalness approaching that of LPC vocoded speech.

During the second half of that contract, work was begun on the phonetic recognition program. The phonetic recognizer uses a network of diphone templates to recognize speech. A compiler first produces a network that represents the allowable sequences

of diphones. The recognizer then determines the path through the network that most closely matches the input speech. It is important that the network be tuned by the inclusion of alternate diphone templates. We will refer to this general process as "training."

3.1 Automatic Training

As described in Section 3.3 of the final report from the previous contract, the diphone recognition program was designed to allow interactive training of the diphone network. The user could cause the recognition program (matcher) to find the best path alignment through the network for a known utterance. The program is constrained to match the input utterance to the "correct" path through the network. The user can then instruct the program to augment the network in one of two basic ways. For those parts of the input utterance that match the network well, the input parameters can be used to update the network statistics (LAR means, variances, node duration PDFs). For those regions that are not well represented, the input can be used to add new, alternate paths in the network. Eventually, the network should be able to match most input speech with a small error.

During this quarter, we implemented the automatic training

capability suggested in Section 3.3.3 of the Final Report. This procedure automatically decides which parts of the input to use for updating statistics or for adding alternate paths. Initially, more than half of a training utterance is typically used for new paths. This portion will gradually decrease as the network becomes more rich.

When we implemented and tested the above automatic training procedure, we found and corrected several bugs in the original version of the training programs. These were difficult to find since the person who programmed the original version of this very complex program is no longer at BBN. We are now more familiar with the program so that future problems may be more easily solved.

3.2 Transcribed Data Base for Training

In order to train the diphone network, we need a data base of speech for which we know the "correct answer". We have, therefore, transcribed a large data base of speech for this purpose. Our data base currently consists of 60 of the sentences from the Harvard Phonetically Balanced list, and 200 military-type phrases and sentences of varying lengths (all spoken by one speaker) that we had recently recorded for another

project. These 260 utterance have been carefully transcribed in a manner consistent with that used for our diphone data base. The transcription includes the detailed phonetic labels, as well as an accurate indication of the phoneme boundaries.

This labelled data base will be used for the first phase of training. We intend to monitor the effect of different amounts of training on the performance of the recognizer so that we may predict how much training will be necessary. If, after using all 260 utterances, the performance is still improving with additional training, we will then increase the size of the data base.

3.3 Future Work

In the next quarter, we will be using the data base for training as described above. We also expect to spend some amount of time moving our programs from TOPS-20 to the VAX-VMS system. This will entail both modification of our large subroutine library (in particular those routines written in PDP-10 MACRO or BCPL. It will also require that our diphone compilers, synthesis, and recognition programs, which are all written in BCPL, be translated to a language available on VAX-VMS. We are currently considering PASCAL as the prime candidate - primarily

because it is a widely used language, is fully supported by DEC, and is closer to ADA than other languages considered.

4. REFERENCES

1. J.A. Hartigan, Clustering Algorithms, John Wiley, New York, 1975.

2. M.R. Anderberg, Cluster Analysis for Applications, Academic Press, New York, 1973.

3. R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis, John Wiley, New York, 1973.

4. Berouti, M, Makhoul, J. Schwartz, R. and Sorensen, J. "Speech Compression and Synthesis," Final Report, BBN Report No. 4414, Contract No. F19628-78-C-0136, Oct. 1980.