Time Delay Neural Networks and Speech Recognition: Context Independence of Stops in Different Vowel Environments

Gregory Andrew Makowski
Western Michigan University

Follow this and additional works at: http://scholarworks.wmich.edu/masters_theses
Part of the Communication Sciences and Disorders Commons, and the Computer Sciences Commons

Recommended Citation
http://scholarworks.wmich.edu/masters_theses/646

This Masters Thesis-Open Access is brought to you for free and open access by the Graduate College at ScholarWorks at WMU. It has been accepted for inclusion in Master's Theses by an authorized administrator of ScholarWorks at WMU. For more information, please contact maira.bundza@wmich.edu.
TIME DELAY NEURAL NETWORKS AND SPEECH RECOGNITION: CONTEXT INDEPENDENCE OF STOPS IN DIFFERENT VOWEL ENVIRONMENTS

by

Gregory Andrew Makowski

A Thesis
Submitted to the
Faculty of The Graduate College
in partial fulfillment of the
requirements for the
Degree of Master of Science
Department of Computer Science

Western Michigan University
Kalamazoo, Michigan
June 1991
A series of speech recognition experiments was conducted to investigate time-dynamic speech recognition of stop consonants invariant of vowel environment using data from six talkers. The speech preprocessing was based on previous studies investigating acoustic characteristics which correlate to the place of articulation (Blumstein and Stevens 1979). The place of articulation features were statistically abstracted using four moments and the energy level of the speech sample.

Both statistical and neural network pattern recognition methods were used. Statistical methods included linear and quadratic discriminant functions, maximum likelihood estimator (MLE) and K-nearest neighbors (KNN). The neural network approach used was Time Delay Neural Networks (TDNN), a time shift-invariant version of backpropagation (Waibel 1989). The classification error rates ranged from 6.1% for quadratic resubstitution, 15.6% for KNN, 18.0% for MLE, 19.0% for the TDNN and 19.1% for linear resubstitution.
Time delay neural networks and speech recognition: Context independence of stops in different vowel environments

Makowski, Gregory Andrew, M.S.
Western Michigan University, 1991
INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.
ACKNOWLEDGEMENTS

I would like to thank my Thesis Committee, Dr. Ben Pinkowski, Dr. Jim Hillenbrand and Dr. Robert Trenal. Specifically I would like to thank my main advisor, Dr. Ben Pinkowski, for his supervision, continuing support, and mentoring in pattern recognition throughout my thesis; and Dr. Jim Hillenbrand who was involved early in this project by making available a variety of software and hardware for speech processing as well as guiding my review of speech literature. I appreciate the academic advice from Dr. Robert Trenal and Dr. Derek Stubbs as well as the conscientious proof-reading by Monica Malamud, my fiancee. I would like to thank Dr. Sheila Blumstein and Dr. Kenneth Stevens for sending audio tapes of the speech samples from their 1979 study. Two students who provided many weeks of helpful assistance digitizing audio tapes were Ravi Kosaraju and Yair Mendelovitsch. The President of Hyperception Inc., (Dallas, Texas) Jim Zachman, donated the software package Hypersignal Workstation, which was a significant help for fast and convenient hand-processing of the speech samples. I would like to acknowledge The Graduate College, Western Michigan University for assistance in the form of a financial grant to help to defray hardware expenses for this project.

Gregory Andrew Makowski
TABLE OF CONTENTS

ACKNOWLEDGEMENTS ................................................................. ii
LIST OF TABLES ................................................................. v
LIST OF FIGURES ................................................................. vi

CHAPTER

I. INTRODUCTION ................................................................. 1

II. A REVIEW OF PATTERN RECOGNITION AND NEURAL NETWORKS ... 3

2.1 Pattern Recognition .......................................................... 3
2.2 Supervised Learning Methods ................................................. 4
2.3 Comparisons of Neural Nets With Traditional Classification Methods ... 6
2.4 Neural Networks and Speech Recognition .................................... 8
2.5 The Backpropagation Neural Network Model ............................... 16
2.6 Setting Backpropagation Parameters ......................................... 20
2.7 Testing Backpropagation for Shift Invariance ............................... 26
2.8 Time Delay Neural Networks ................................................ 28

III. A REVIEW OF SPEECH PROCESSING AND RECOGNITION ............ 32

3.1 Speech Recognition Background and Challenges ......................... 32
3.2 Speech Preprocessing ....................................................... 36
3.3 Acoustic Invariance in Stop Consonants .................................... 41
3.4 Time-Varying Feature Extraction ............................................ 45

IV. PROJECT DESIGN AND DISCUSSION .......................................... 52

4.1 Project and Model Selection ................................................ 52
4.2 Methods ................................................................. 53
4.3 Classification Tests .......................................................... 55
Table of Contents--Continued

CHAPTER

4.4 Analysis of Results ............................................................. 59
4.5 Directions for Future Research ........................................... 62

APPENDICES

A. Human Subjects Institutional Review Board Acceptance Letter .......... 64
B. Parallel Distributed Processing Configuration Files for Time Delay Neural Networks ........................................ 66

BIBLIOGRAPHY ................................................................. 69
INDEX .................................................................................. 82
LIST OF TABLES

1. Lippmann's Taxonomy of Classifiers ............................................................ 7
2. Static Neural Network Speech Classification ............................................. 10
3. Time-Dynamic Neural Network Speech Classification ............................. 13
4. Structure of a Time Delay Neural Network .............................................. 30
5. Place of Articulation vs. Voicing for Stop Consonants ............................. 33
6. Klatt's Eight Problem Areas ................................................................. 34
7. Speech Preprocessing Steps and Parameters ......................................... 37
8. Kewley-Port Place of Articulation Features ....................................... 46
10. Classifying Stops With Spectral Moments ............................................. 51
11. TDNN Study Descriptions and Results ............................................... 56
12. Statistical Discriminant Error Rates .................................................... 57
13. CV and VC Statistical Discriminant Error Rates ................................. 59
14. Confusion Matrices for Quadratic and KNN ....................................... 60
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Kohonen's Feature Map</td>
<td>11</td>
</tr>
<tr>
<td>2</td>
<td>Advantages of Additional Network Layers</td>
<td>21</td>
</tr>
<tr>
<td>3</td>
<td>Learning Rate Tests</td>
<td>26</td>
</tr>
<tr>
<td>4</td>
<td>Backpropagation Fails a Shift Invariance Test</td>
<td>27</td>
</tr>
<tr>
<td>5</td>
<td>A Sample Spectrogram of &quot;The auctioneer accepted the bid&quot;</td>
<td>39</td>
</tr>
<tr>
<td>6</td>
<td>Comparison of LPC vs. FFT output</td>
<td>41</td>
</tr>
<tr>
<td>7</td>
<td>Sample Spectra Which Fit the Blumstein and Stevens Templates</td>
<td>43</td>
</tr>
<tr>
<td>8</td>
<td>Modified Raised-Cosine Time Window.</td>
<td>44</td>
</tr>
<tr>
<td>9</td>
<td>Lahiri et al. Time Dynamic Features for Labial vs. Dental and Alveolar</td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>Consonants</td>
<td></td>
</tr>
</tbody>
</table>
CHAPTER I

INTRODUCTION

The goal of speech recognition is to enable machines to consistently and correctly identify the sounds produced in speech. The long range applications of this technology include enabling physically impaired individuals to control their environment through the use of speech and robotics. Verbal interaction with machines could extend to control tower computers talking to pilots for routine situations, or to a pilot verbally controlling a jet which is undergoing a high gravitational turn. Automatic dictation systems for doctors, lawyers or other professionals could avoid repeated typing and proofing steps by generating the correct document the first time. Verbal control systems could be a great benefit for lab technicians or other personnel with occupied hands to control equipment or record results. Multi-media applications could help integrate the use of speech recognition in everyday activities such as home shopping or requesting a weather report through an interactive home television.

Speech recognition is currently a difficult problem which is much more of a research topic than a feature commonly available in commercial products. Pattern recognition is the analysis of data in known groups to develop a means of classifying new data into the existing groups. Pattern recognition depends heavily on preprocessing the data into a form which closely correlates with the correct classes. One difficulty is that the current understanding of the physiology of speech production and perception is not complete and detailed enough to implement a completely effective and general system. The task of speech recognition is difficult to perform accurately when time is not a constraint, and should ideally be performed in real-time. While there is a large body of knowledge on speech, and many specific classes of examples are understood, speech recognition could greatly benefit from progress in the
continued development of general models of speech production and perception, as well as further research into pattern recognition techniques.

This thesis contains two chapters of literature review in the areas of speech processing, pattern recognition (PR) and neural networks (NN). The review chapters start out covering a general background and narrow down to the focus of this paper, neural networks and speech recognition. Following the review chapters I describe the study reproducing one of the seminal projects in speech recognition with the addition of a Time Delay Neural Network and statistical classification methods.
CHAPTER II

A REVIEW OF PATTERN RECOGNITION AND NEURAL NETWORKS

2.1 Pattern Recognition

Pattern recognition (PR) is the study of classification or grouping of information based on previous examples of each group. PR may be applied to character recognition, airport bomb detection, quality control in assembly lines, automatic blood cell diagnosis, drug interaction, process control, electrocardiograms and a long list of other applications (Patrick 1972, 1). PR methods typically process data in numeric form to produce a symbolic representation. The details of various pattern recognition models are covered in a variety of texts (Patrick 1972; Duda and Hart 1973; Chien 1978; James 1985; Niemann 1989 and Pao 1989).

Two types of PR algorithms are unsupervised and supervised learning methods. Unsupervised methods are used for exploratory data analysis, to determine the inherent groupings in the data. Unsupervised PR methods are sometimes referred to as self-organizing; they analyze and automatically group the data. Supervised methods are used when the classes of data are known, and require a training set of pre-labeled data. In general, a supervised method has a training and testing phase. In the training phase supervised algorithms learn to discriminate between predefined categories of data while unsupervised methods find groups of similar data items. In addition to supervised and unsupervised methods, there are also syntactical techniques based on grammars. A pattern recognition system may incorporate multiple approaches, using an unsupervised method to group the data and a supervised method to further classify the data. The number of features of the data
may also be referred to as the dimensionality of the data. For example, if three features contain the data which are grouped into five classes, the data points could be plotted on a three-dimensional grid, and in a simple case the class boundaries could be represented by five spheres, each one containing all the data points for a given class.

2.2 Supervised Learning Methods

Fisher's linear discriminant function (LDF) is the first published use of a LDF, and the Iris data from this plant study is often used to test new classification methods (Fisher 1936; Cooley and Lohnes 1971; and Pinkowski 1987). Fisher's LDF is based on a linear combination of variables, and uses a variance-covariance matrix to abstract the data. The LDF discriminates by maximizing group differences between variance-covariance regions representing the labeled groups. Fisher's LDF is most accurate when the variance-covariance matrices of each group are similar. If and when the variance-covariance matrices differ significantly, a quadratic discriminate function becomes more appropriate (Cooley and Lohnes 1971).

Bayes' Theorem classifies a data item to the group with the largest posterior probability, \( P(G_i \mid x) \), an objective probability, which signifies frequency of occurrence in a random experiment (Chien 1978; James 1985; Pao 1989). The posterior probabilities are based on prior probabilities, \( P(G_i) \) and likelihoods, \( P(x \mid G_j) \). The prior probabilities are found from the training data set and the known probabilities \( x \) in group \( G \), from the likelihoods.

\[
P(G_i \mid x) = \frac{P(x \mid G_i) P(G_i)}{\sum_{j=1}^{g} P(x \mid G_j) P(G_j)}
\]

The Bayes' decision rule, which is used for classification, is distinct from Bayes' theorem in that the decision rule creates "risk functions from such conditional-probability functions, and
we can then make decisions on the basis of minimum risk or maximum gain" (Pao 1989, 28).

The decision rule is based on a covariance matrix representation of the groups. The problem
is linearly separable if for each class the corresponding components of the covariance matrix
are equal. Otherwise the discriminant function is a quadratic curve. There are weaknesses
of Bayes' decision rule; one is that the membership groups are mutually exclusive, so the
probabilities add up to exactly one, which contrasts with other methods which do not make
this requirement (Pao 1989, 48), such as fuzzy sets (Kandel 1986).

The K-nearest neighbor (KNN) algorithm is not based on statistical measures such as
variance, covariance, or standard deviation. The KNN algorithm (Patrick and Fischer 1970;
Patrick 1972; Duda and Hart 1973) classifies new data elements into the closest existing group.
The closest group is computed by using a distance measure such as the Euclidian or
Mahlanobis distance, and averaging the distance to the $K$ closest data elements in each
existing group. This algorithm is fine-tuned by changing the value of $K$, the distance measure,
data preprocessing or the number of features in the data set. The KNN boundary may be
smooth between distant classes or convoluted for overlapping classes which may cause
complex concave and convex boundaries. This variable type of boundary contrasts with the
smooth linear or polynomial boundaries computed by Fisher's LDF or Bayes' Decision rule.
The statistically-based methods, Fisher's and Bayes', tend to generalize better on small training
sets without outliers, and the KNN algorithm does well with complex boundaries based on
a large, representative data set. Both Fisher's and Bayes' methods are parametric and make
certain assumptions about the distribution of the underlying data. Methods such as these two
create convex discrimination regions. Non-parametric methods, such as KNN, do not make
any assumptions about the underlying data, and can form convoluted, convex and concave
decision boundaries.

When classification is viewed as the development of multidimensional boundaries
between classes, methods not thought of as PR algorithms can be used for PR. From
computational geometry, K-D trees are multidimensional binary trees with \( O(\log n) \) retrieval
time, and are relevant for associative memory and K nearest neighbor pattern matching. At
each node of the K-D tree, the remaining data points in that region are divided in half. The
dimension to be divided can be based on a heuristic such as the dimension of greatest data
spread. The application of K-D trees (Bentley 1975; Friedman, Bentley and Finkel 1977; and
Bentley 1979) to classification has been shown theoretically to have significant speed
advantages over neural networks by over a million-fold (Omohundro 1987). In Burr's (1988)
paper on NN recognition of spoken and written text, he concluded that NN were a good
choice for parallel hardware, but for sequential machines, the use of a nearest neighbor
method with K-D trees may be preferred. Lee and Lippmann (1990, 175) made a comparison
of the KNN with K-D trees with the conclusion: "the K-nearest neighbor classifier is one of
the slowest in classification when implemented serially without complex search techniques
such as K-D trees. These techniques greatly reduce classification time but make adaptation
to new training data more difficult and increase complexity." Pao (1989, 54) has suggested
the use of fractals and non-Euclidean space representations for classification.

2.3 Comparisons of Neural Nets With Traditional Classification Methods

Lippmann (1987) gives a very useful taxonomy of neural networks and classical
pattern recognition algorithms (see Table 1, from Lippmann 1987, 9). The classifiers are
divided into several categories: neural net vs. conventional, supervised vs. unsupervised and
continuous vs. binary input. The classifiers listed in the two columns in Table 1 are different
algorithms which process comparable types of data sets.

For some classification problems, the traditional classification methods and neural
network methods perform with similar accuracy, but neural networks have advantages in
more complex classification problems, such as those with outliers. Recent studies have made
a comparison between neural networks and traditional classifiers (Huang and Lippmann 1987,
Lippmann 1987, Burr 1988, Lippmann 1988, Lee and Lippmann 1990). The Bayesian classifier is considered optimal under certain assumptions, such as when the underlying distribution is Gaussian with few outliers. Backpropagation can be more robust because it does not require any assumption regarding the underlying distribution of the data, and because the sigmoid squashing function helps to minimize the confusing effect of outliers (Huang and Lippmann 1987). The classification ability of NN was compared with other classifiers in a paper by Shadmehr and D'Argenio (1990). This paper reported that the NN classification performance was better than that of the nearest neighbor classifier or a Maximum Likelihood Estimator (MLE) (Symons 1981), and approached the recognition ability of a Bayesian estimator.

Other comparisons have shown NN to perform generally comparable to the other types of classifiers. In a study by Lee and Lippmann (1990), eight classical and neural network classifiers were compared using two small artificial problems and two speech problems. The artificial problems were: the bullseye, two concentric circles with the inside
circle one class and the outside circle another; and the disjoint, two non-adjacent squares containing points belonging to the same class. The speech problems studied were vowel classification and spoken digit classification. The accuracy of the classification methods was reported to be generally within one standard deviation of the average accuracy of all methods. The accuracy rate of the most difficult problem, vowel classification, averaged 80% and deviated by 3% among the different methods. The methods compared included: (a) Backpropagation (BP), (b) multiple-stepszie BP, (c) Learning Vector Quantizer (LVQ), (d) Kohonen’s feature-map network, (e) hypersphere, (f) KNN, (g) Bayesian and (h) binary tree. This paper compared classification and training time as well as memory requirements for the different methods. The results showed no overall winners. The conclusion of this paper was there is no best method for all cases, but the tradeoffs were emphasized that must be make when choosing among methods.

2.4 Neural Networks and Speech Recognition

There have been several extensive reviews of neural networks for speech recognition (Burr 1988; Lippmann 1988, 1989; and Waibel and Hampshire 1989b) which include a wide variety of neural network models. While there have been uses of neural networks in other areas of speech processing, such as noise suppression, or the conversion of text to speech (Sejnowski and Rosenberg 1986), the primary focus of this section is speech recognition.

The best existing speech recognizers perform well only in artificially constrained tasks. Performance is generally better when training data is provided for each talker, when words are spoken in isolation, when the vocabulary size is small, and when restrictive language models are used to constrain allowable word sequences (Lippmann 1989, 2).

All recognition projects can be divided into two temporal categories, static and dynamic. Static classification requires an additional pre-segmentation algorithm, or hand-segmentation, to segment the speech into a logical unit called a template. The template-sized speech segment is presented to the classification algorithm. Temporally dynamic classification
algorithms process the speech on a continuous basis without requiring segmentation. Part of the difficulty of temporally dynamic classification is to find the best segmentation.

2.4.1 Static Classification of Speech

Static classification methods rely on pre-segmentation of the data into phonemes or words, and hence are limited by the accuracy of segmentation. The resulting segment is processed all at once, without use of any temporal-relationship analysis. The extra segmentation step slows the speed of the static classification methods, and requires more memory to buffer the incoming speech data in a real-time recognition system. For a summary of representative static classification studies, see Table 2 (Lippmann 1989, 9). In Table 2, treat Multilayer Perceptrons (MLP) as synonymous with Backpropagation (BP), discussed in more detail in this paper.

The Elman and Zipser (1988) BP study hand-centered the fixed length time windows on the consonant voicing onset as part of the speech preprocessing in order to present the data in a time-independent fashion. An interesting preprocessing step performed in this study was the application of uniformly distributed white noise to the training tokens, which decreased the error rate by 2% to 6%. The following quote regards another Elman and Zipser (1987) experiment, with more details of the experiment listed in Table 2.

An analysis indicated that hidden nodes often become feature detectors and differentiate between important subsets of sound types such as consonants versus vowels. This study demonstrated the importance of choosing a good data representation for speech and of normalizing speech inputs. It also raised the important question of training time because many experiments on this small data base required more than 100,000 training trials. (Lippmann 1989, 10).

Lippmann and Gold (1987) compared single and multilayer perceptrons with conventional classifiers and found that KNN performed the best, with BP better than Gaussian classifiers. The single-layer perceptron sometimes did not converge, and in general provided poor performance. The results of this study demonstrated that non-linear classifiers
<table>
<thead>
<tr>
<th>Study</th>
<th>Network</th>
<th>Speech Materials</th>
<th>Error Rate</th>
<th>Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elman and Zipser (1988)</td>
<td>Multilayer Perceptron</td>
<td>1 Talker, CV's</td>
<td>5.0%</td>
<td>Consonants</td>
</tr>
<tr>
<td></td>
<td>16x20 inputs</td>
<td>/b,d,g/ /i,a,u/</td>
<td>0.5%</td>
<td>Vowels</td>
</tr>
<tr>
<td></td>
<td></td>
<td>505 Tokens</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Huang and Lippmann (1988)</td>
<td>MLP, Feature Map Classifier (FMC)</td>
<td>67 Talkers</td>
<td>~20%</td>
<td>MLP,FMC,Gauss.</td>
</tr>
<tr>
<td></td>
<td>2 Inputs</td>
<td>10 Vowels</td>
<td></td>
<td>FMC Trains Fastest</td>
</tr>
<tr>
<td></td>
<td></td>
<td>671 Tokens</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kammerer and Kupper (1988)</td>
<td>MLP</td>
<td>11 Talkers</td>
<td>0.4%</td>
<td>Talker Dependant</td>
</tr>
<tr>
<td></td>
<td>16x16 Inputs</td>
<td>20 Words</td>
<td>2.7%</td>
<td>Talker Independent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5720 Tokens</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kohonen et al. (1988)</td>
<td>Learning Vector Quantizer (LVQ)</td>
<td>Labeled</td>
<td>12.9%</td>
<td>Gaussian</td>
</tr>
<tr>
<td></td>
<td>15 Inputs</td>
<td>Finnish Speech</td>
<td>12.0%</td>
<td>KNN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3010 Tokens</td>
<td>10.9%</td>
<td>LVQ</td>
</tr>
<tr>
<td>Lippmann and Gold (1987)</td>
<td>MLP</td>
<td>16 Talkers</td>
<td>8.7%</td>
<td>Gaussian</td>
</tr>
<tr>
<td></td>
<td>11x2 Inputs</td>
<td>7 Digits</td>
<td>6.0%</td>
<td>KNN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2,912 Tokens</td>
<td>7.6%</td>
<td>MLP</td>
</tr>
<tr>
<td>Peeling and Moore (1987)</td>
<td>MLP</td>
<td>40 Talkers</td>
<td>0.3%</td>
<td>Talker Dependant</td>
</tr>
<tr>
<td></td>
<td>19x60 Inputs</td>
<td>10 Digits</td>
<td>1.9%</td>
<td>Multi-Talker</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16,000 Tokens</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

performed better than the linear perceptron or Gaussian classifiers, indicating the complexity of the problem. The data classified were spoken digits from the Texas Instruments (TI) 20-Word Speech Data Base (Doddington and Schalk 1981).

The concept behind Kohonen's feature-map (FM) classifier (Kohonen 1984, 1988a, 1988b) is based on the topographical organization of the mammalian brain, where close optic areas in the brain coincide with close areas in the visual system. The feature map is
organized as a two dimensional matrix with connections between neighboring nodes and distant nodes. All nodes collect the same input pattern; however, neighboring nodes reinforce each other and distant nodes inhibit each other. The map self-organizes similar features to adjacent locations (Anderson, Pellionisz and Rosenfeld 1990, 653). The FM classifier network has been used on speech recognition projects to perform vector quantization as a dimensional reduction step similar to principal components analysis. Huang and Lippmann (1988) used a two-layer hybrid network composed of a FM layer feeding into a perceptron layer (Table 2). In the first layer, the FM self-organized the data to perform as a vector quantizer. In the second layer, the supervised perceptron is trained with a version of the least mean square (LMS) algorithm. This hybrid network converged, or learned, in fewer than 50 trials. A non-hybrid two-layer perceptron required 50,000 training trials, three orders of magnitude more than the hybrid one (Lippmann 1989, 12-13). Kohonen's network is insensitive to time warp and has been able to learn a primitive grammar (Tattersall, Linford and Linggard 1988). In generating a complete phonetic feature-map (see Figure 1, from Kohonen 1988a, 19), it was reported that the distinction of /p,t,k/ was not reliable, although no other difficulties were described (Kohonen 1988a).

The learning vector quantizer (LVQ) by Kohonen and colleagues (1988) listed in Table

![Figure 1. Kohonen's Feature Map](image-url)
2 was similar to the FM with the addition of a supervised training stage which adjusts a codebook of vectors when an error occurs. In general, a vector quantizer (VQ) algorithm uses a vector to represent each class, or each disjoint member of a class. Some applications of the VQ method use several vectors for a similar contiguous class instead of one vector, which could be considered a fine-tuning aspect of the algorithm. Each vector is normalized to a length of one and is considered to represent the center, or average, of that class. The set of all vectors is called a codebook. A test sample to be classified is converted to a vector and compared with each vector in the codebook by perpendicular projection. The codebook vector with the largest projection is the class in which the test vector is grouped. For example, if the vectors were two dimensional, and there were three codebook vectors pointing in the direction of two o’clock, seven o’clock and ten o’clock, a test vector pointing in the direction of five o’clock would be grouped with the seven o’clock codebook vector. The LVQ implements a VQ by adjusting the codebook values when errors occur. The LVQ (Kohonen et al. 1988) was classifying 18 phoneme classes using 117 codebook nodes, which had a better error rate than the Bayesian and KNN classifiers, 10.9% vs. 12.9% and 12.0% respectively.

2.4.2 Dynamic Classification of Speech

In general, the error rates for the time-dynamic projects in Table 3 (Lippmann 1989, 15) are lower than the error rates for the static projects listed in Table 2. Several types of time-dynamic projects will be discussed in this section, though the Time Delay Neural Networks (TDNN), will be discussed in greater detail in a later section. It is common in these projects to use careful hand-segmentation of the training data, and in some cases hand-segmentation of the test data. Because these classifiers are dynamic in time, they do not require presegmentation, which generally makes the processing time quicker by removing an extra step. Some of the following studies may not require presegmentation, but they use it anyway, probably for better results. 'Performance for small vocabularies often slightly
Table 3  
Time-Dynamic Neural Network Speech Classification  

<table>
<thead>
<tr>
<th>Study</th>
<th>Network</th>
<th>Speech Materials</th>
<th>Error Rate</th>
<th>Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lang and Hinton (1988)</td>
<td>Time Delay MLP</td>
<td>100 Talkers &quot;B,D,E,V&quot; 768 Tokens</td>
<td>7.8%</td>
<td>Multi-Talker</td>
</tr>
<tr>
<td>Unnikrishnan, Hopfield and Tank (1988)</td>
<td>Time Concentration Net</td>
<td>1 Talker Digits 432 Tokens</td>
<td>0.7%</td>
<td></td>
</tr>
<tr>
<td>Waibel et al. (1989)</td>
<td>Time Delay MLP</td>
<td>3 Japanese Talkers /b,d,g/ 4,000 Tokens</td>
<td>1.5%</td>
<td>/b,d,g/</td>
</tr>
<tr>
<td>Waibel, Sawai and Shikano (1988)</td>
<td>Time Delay MLP</td>
<td>1 Japanese Talker 18 Conson, 5 Vow &gt; 10,000 Tokens</td>
<td>1.4%</td>
<td>/b,d,g, p,t,k/</td>
</tr>
<tr>
<td>Watrous (1988)</td>
<td>Temporal Flow Struct. MLP</td>
<td>1 Talker Phonemes, Words &gt; 2,000 Tokens</td>
<td>0.8%</td>
<td>/b,d,g/</td>
</tr>
<tr>
<td>McDermott and Katagiri (1988)</td>
<td>Time Delay LVQ</td>
<td>3 Japanese Talkers /b,d,g/ 4,000 Tokens</td>
<td>1.7%</td>
<td>/b,d,g/</td>
</tr>
</tbody>
</table>

Lang and Hinton (1988) investigated time-dynamic recognition with a multi-resolution training technique. The network would first be trained using fewer hidden nodes. Each node would be duplicated, creating a much larger number of hidden nodes for additional network training. However, this network required presegmentation. A second experiment avoided presegmentation by using an automatic energy-based method. Another aspect of this second
experiment involved the use of counter-example inputs to suppress false classifications. The data set used in both experiments was part of what is called the E-set, which consists of the letters of the alphabet with names ending in a long "E" sound. The letters they were using were "B", "D", "E" and "V". The E-set is considered one of the most difficult sets of sounds to distinguish (Lippmann 1989, 17). The only difference is the short burst at the beginning of each sample.

Watrous (1988) used recurrent connections (Rumelhart and McClelland 1988, 354-360) in a structure called the Temporal Flow Model. Recurrent connections are a type of backpropagation network for which the outputs feed into the inputs. This forms a type of finite state machine which computes the network output based on its input and its previous output. The network structure was carefully hand-adjusted to extract specific speech features for each classification, and the target outputs were Gaussian-shaped pulses instead of binary values. The test-data were hand-segmented for this study.

Rossen and colleagues used a brain-state-in-a-box (BSB) network model (Anderson 1977; Rumelhart and McClelland 1988, 66-68), which is similar to a simple linear associator but has limits of [-1,+1] on the activation values of each unit. A BSB using two features could be represented by a square with each axis ranging from [-1,+1], three features could be represented by a cube, etc.. The BSB can be represented by an additional dimension for each unit, and uses positive feedback to change from an initial state somewhere in the middle to an answer state in one of the corners. Auto-association is used in the training phase of a BSB. In this project, smaller modules were trained and then combined into larger modules. Counter-examples were used in training to reject noisy inputs.

Unnikrishnan, Hopfield and Tank (1988) used a time-concentration net (Tank and Hopfield 1987) with variable-length delay lines. "Outputs of delay lines are multiplied by weights and summed to form separate matched filters for each word. These matched filters concentrate energy in time and produce a large output pulse at the end of the correct word"
Variable-length delay lines over time gradually decrease the strength of earlier pulses.


Two NN projects which process continuous-stream patterns with the use of slope detectors were developed by Smythe (1987) and Makowski (1989). Smythe observed that the vertebrate visual system contains cells which respond to motion in a specific direction in their receptive field, and applied that idea to slope detection for use in formant tracking. Individual slope detector subnets were developed to activate in the presence of a variety of slopes. Makowski abstracted the continuous pattern of a frequency-domain signal by segmenting the pattern into linear segments. The linear segments were described in terms of starting position, length and slope. The descriptions of the segments were then presented to the input layer of a BP network.

The Boltzmann machine (Rumelhart and McClelland 1988, 318-362) has been compared to BP (Bengio and DeMori 1988) in a project investigating place of articulation. The project reported a lower error rate for the Boltzmann machine by about 2%, but the execution time was nine times longer. "It has been argued for a long time that linear combinations of formants can produce effective talker normalization. As formants are difficult to extract, spectral lines are used and suitable parameter combinations are learned by connectionist networks" (Bengio and DeMori 1988, 103).
2.5 The Backpropagation Neural Network Model

2.5.1 Historical Development of Backpropagation

Neural networks have had a long history with respect to the field of computer science. The area of neural networks has enjoyed a multidisciplinary interest from researchers in fields such as computer science, electrical engineering, applied mathematics, physics, optics, biology, psychology and cognitive science. There have been many applications of BP in addition to classification, such as data compression, control systems, learning models and prediction. Many concepts used in the area of neural networks come from basic neurology (McCulloch and Pitts 1943; Hebb 1949; Rosenblatt 1961). Neural networks are not intended to be a strict simulation of the biochemical events between neurons, but the area of study frequently makes observations of neurological information processing in nature seeking for hints or guidelines for topics of investigation. Researchers are normally challenged to keep abreast of their own field, but those involved in neural nets need to follow several disciplines.

The perceptron, an early neural network model, was developed and published more than once, by Rosenblatt (1958a, 1958b). In 1957-1958 Rosenblatt, Wightman and others were part of a team which built the Mark I Perceptron to recognize the characters "A", "B" and "C" (Hecht-Nielsen 1989, 4-6). Steinbuch (1961, 1963) developed a "learnmatrix" which was a binary associative network. In 1969 Minsky and Papert published Perceptrons, which demonstrated the lack of ability of the perceptron to solve the non-linear XOR problem. This text not only pointed out this lack of ability, but also dried up research funding into neural networks, which started going into artificial intelligence (AI). From 1967 to 1982 there was little sponsored or published in neural nets, until Hopfield published two papers (Hopfield 1982, 1984) and DARPA started funding NN research in 1983. A multilayer version of the perceptron, backpropagation (BP), was developed which solved the XOR problem and renewed interest and research funding into neural networks. The BP model has been one of

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
the most commonly published and applied neural net models. According to Wasserman (1989) and Hecht-Nielsen (1989), BP was invented three times: (1) Werbos (1974) in his dissertation, (2) Parker (1982) and (3) in 1985 by the Parallel Distributed Processing (PDP) group, including David Rumelhart, Ronald Williams and James McClelland, (1986). The PDP books were not the first, but they were among the few influences which greatly stimulated work in neural nets and BP. Other parallel developments have been described by White (1989), who has given a historical review comparing neural nets and statistical methods, and has found similarities with learning law and the Robbins/Monro (1951) technique. Hecht-Nielsen (1989, 124) has also found "a mathematically similar recursive control algorithm" presented by Bryson and Ho (1969).

2.5.2 Backpropagation Algorithm Overview

The reader may find many discussions in the literature regarding the BP algorithm and its derivation (Lippmann 1987; Rumelhart and McClelland 1988, 1989; Wasserman 1989; Hecht-Nielsen 1989; and Pao 1989). Public domain software is available from Rumelhart and McClelland (1989), which includes C source code and was designed to be portable across systems. A brief top-down overview of backpropagation (BP) will be discussed before elaborating on the details. The framework of classification will be used, as that is the framework appropriate for speech recognition.

BP consists of an input layer of nodes, typically one or two hidden layers and an output layer. The nodes might be thought of as operating in the way biological systems process information, with the input layer representing sensory input such as sight or hearing, and the output layer representing classification of the sensory input. The nodes between each successive layer are completely connected, with modifiable weights on the connections. The modification of these connection weights is the process of learning. BP is a supervised learning method because in the training mode a pattern is presented to the input layer, the
data pass forward through the network, and the final output is compared with the expected output. The difference between the actual output and the expected, or goal, output is computed as the error. The error is then swept back through the network in the training mode, adjusting the weights in such a way as to minimize the overall error. The comparison between the actual output and the goal output is the characteristic that makes this a supervised learning method. Backpropagation is named for the propagation of error back through the network. The network has finished the training phase when the overall error for all the training samples has reached a user-defined level. When all training patterns have been presented to the network for a forward and backwards sweep, one epoch has occurred. Over hundreds or thousands of epochs of training the network learns, and the average error between the desired and actual output decreases.

After the BP weights are trained, the system may be used in a test mode. In this mode new patterns may be presented to the input layer, and the forward pass computed to produce the classification answer indicated by the output node with the largest value. In the test mode the goal output is neither used nor known, and no weights are modified. The percentage of error can be computed when all the samples in the test set have been presented to the trained network.

2.5.3 Forward Pass

In the forward pass a set of inputs from the previous layer is used to compute the next layer. The outputs, $O_{i}$ from the previous layer are then multiplied by a connection weight and the resulting products are summed. Every pair of nodes in adjacent layers is connected by a connection weight, $W_{ij}$ which is initially random. The resulting sum, $NET_{i}$, is then passed through the sigmoid function, $f(NET_{i})$, which results in the output for the current node. This process is repeated for each node in each layer until the values of the output nodes are computed. The sigmoid function is significant primarily because it is non-
linear, differentiable and non-decreasing, and secondarily because it converts the output to a zero to one range. If the sigmoid was linear, there would be no advantage in having multiple layers because BP would only be capable of linear discrimination.

2.5.4 Backward Pass

In the backward pass, the weight of each connection, \( w_{pq} \), is modified proportional to the product of the error signal, \( \delta_p \), and the output, \( a_p \). The product is then scaled by the learning rate \( \eta \) (see equation 4). The error signal \( \delta \) is calculated recursively starting with the output layer (equation 5), and then back through the other layers of the network (equation 6).

\[
\Delta_p w_{ij} = \eta \delta_p o_p \\
\delta_p = (t_p - o_p) f'_j (net_{pj}) \\
\delta_p = f'_j (net_{pj}) \sum_k \delta_p w_{kj}
\]  

The theory assumes infinitely small stepsize with infinite steps, which is obviously not practical to implement. Both the learning rate and momentum terms can be used to approximate the continuous gradient descent with a finite number of steps. The learning rate term represents a finite stepsize, with larger steps representing faster progress. The disadvantage of larger step sizes is the potential oscillation skipping over the desired minima.
The momentum term helps to prevent oscillation by adding a tendency of the vector direction to be similar to the previous direction. This can be accomplished by:

$$\Delta w_{ij}^{(N+1)} = n(\delta_p a_p) + \alpha \Delta w_{ij}^{(N)}$$  \hspace{1cm} (7)$$

Where $n$ is the learning rate, $\alpha$ is the constant of proportionality which indicates the influence of the previous weight changes and $N$ indexes the presentation number (Rumelhart and McClelland 1988).

2.6 Setting Backpropagation Parameters

2.6.1 Advantages of Network Layers

There are many parameters of BP which may be fine-tuned to balance sometimes conflicting trade-offs. One of the most important is the number of network layers, which in practice may vary from one to four. Figure 2 (adapted from Lippmann 1988, 115) emphasizes the advantages of increasing the discrimination power from additional network layers. The figure assumes that one, two and three layer networks have been trained and are tested with data points from two classes which are represented as "a" and "b". The decision regions from the three trained networks are represented as shaded for class "a" and unshaded for class "b". A data point not in the correct region is considered an error. A single layer network is similar to the linear perceptron algorithm and to other linear discriminant methods such as Bayes' Theorem, Fisher's LDF or the Maximum Likelihood Estimator (MLE). It has been shown that a single layer is only capable of linear discrimination (Lippmann 1987, 1988; Huang and Lippmann 1987). The discrimination power of a two layer network can be characterized as a non-linear discrimination region composed of multiple intersecting linear discrimination regions, limited to a convex shape.

Non-linear discrimination is also possible with non-neural net methods such as
<table>
<thead>
<tr>
<th>Number Of Layers</th>
<th>Types of Decision Regions</th>
<th>Sample Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>One</td>
<td>Half-plane, linear discriminant.</td>
<td></td>
</tr>
<tr>
<td>Two</td>
<td>Intersecting half-planes, typically convex, non-linear.</td>
<td></td>
</tr>
<tr>
<td>Three</td>
<td>Convex and Concave, can form disjoint regions.</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2. Advantages of Additional Network Layers.

quadratic discriminant functions, KNN and hypersphere classifiers. A three-layer network is capable of discrimination involving convex, concave and disjoint regions, and is theoretically capable of computing any arbitrary function. In general, adding layers beyond three does not add any new capability not available with three layers, although on a case-by-
case basis, there may be some speed or classification advantage (Kolmogorov 1957; Cybenko 1988). Two and three-layer networks were developed by Rumelhart and McClelland (1986) and shown to have non-linear properties which could solve the XOR problem as well as more complex non-linear or disjoint problems. After the development of the multilayer capabilities, in practice, typical projects start with three-layer networks. Once the network's classification has been tested, experiments may be conducted to test speed increases by reducing the number of nodes in a layer or changing the number of layers.

The theoretical basis which proves that a three-layer network can compute any function is Kolmogorov's Theorem (Kolmogorov 1957; Sprecher 1965; Lorentz 1976; Hecht-Nielsen 1987b, 1989). This theorem could be summarized by stating that for an arbitrary continuous function from an n-dimensional cube, there exists a mapping from [0,1]^n to the space of real numbers, R, using a three-layer network. This theorem states that such a mapping always exists, but it does not indicate how to find such a mapping.

Kolmogorov's Mapping Neural Network Existence Theorem: Given any continuous function \( f: [0,1]^n \rightarrow R^m, f(x) = y, f \) can be implemented exactly by a three-layer feedforward neural network having \( n \) fanout processing elements in the first (x - input) layer, \((2n+1)\) processing elements in the middle layer, and \( m \) processing elements in the top (y - output) layer. (Hecht-Nielsen 1989, 122)

The network consists of \( n \) nodes, \( x_{i,m} \) at the input layer connected to \((2n+1)\) nodes, \( z_{j,(2n+1)} \) in the hidden layer, which are connected to \( m \) nodes, \( y_{l,m} \) in the output layer. Equation (8) represents the computation between the first and second layers. The function \( \psi \) is independent of \( f \), real and continuously monotonically increasing. \( \delta \) is an arbitrary positive constant which bounds the rational constant \( \varepsilon \), \( 0 < \varepsilon \leq \delta \). Equation (9) shows the relationship between the hidden and output layer. The functions \( g_{l,m} \) depend on \( f \) and \( \varepsilon \) and are real and continuous. The theorem only states that the functions \( \psi \) and \( \varepsilon \) exist, no other
clue is given.

\[ y_d = \sum_{k=1}^{2n+1} g_d(z_k) \]  

(9)

2.6.2 Number of Nodes Per Layer

Determining the number of nodes in each layer is more guesswork than science. If there are too few nodes the classification may not be learned. A general lower limit is \( \log_2 n \), where \( n \) is the number of inputs. Near that limit, input data may be ignored or lost before it propagates to higher layers. A general upper limit is the number of data samples. Too many nodes slow down the network, and may cause the network to "memorize" the data resulting in a procedure similar to table lookup (Neuralyst). Small data sets can be a problem. An empirical rule has been reported that the number of samples should be 5 to 10 times the number of weights. The small data set problem can, in some cases, be avoided by adding noise to existing samples to create new samples, or modeling the data set using Bayesian statistics (Stubbs 1991, 4; Neuralyst). A small data set can cause memorization and poor generalization, as indicated when the training data are classified with high accuracy and the test data are classified with low accuracy. Within the upper and lower bounds, more nodes tend to cause quicker learning, or fewer presentations of the training data, while fewer nodes decrease the time needed for each training pass.

2.6.3 Backpropagation-Specific Preprocessing

In general it is beneficial to use these heuristics when using backpropagation or a similar NN for classification. These heuristics in general have an empirical rather than a theoretical basis, and may not necessarily be beneficial for every applicable problem. Some of the preprocessing heuristics discussed in this section may apply to other pattern recognition
techniques as well as to NN methods. Many of these processing steps could also be applied to the goal output values as well as to the input values.

In general, all features should be scaled to a zero to one range to prevent small relative changes caused by over-influencing classification in a field with a large magnitude. An example of this might be the annual national budget increasing by 100 billion dollars, which might be small considering the total budget is 1.4 trillion and the previous increase was 500 billion dollars. The relative change used for classification could be the percentage change in budget, or the percentage change between the most recent two changes in budget. A general equation that could be used for scaling features is as follows:

\[
    \text{Feature}_{\text{Scaled}} = \frac{(\text{Feature} - \text{Min}(\text{Feature}))}{(\text{Max}(\text{Feature}) - \text{Min}(\text{Feature}))}
\]

(10)

Features with wide variation such as those which vary in the number of digits could be scaled after taking the logarithm of the feature (Neuralyst). The sigmoid function of BP does partially handle this problem without previous scaling; however, the network may require more learning, more training cycles, to overcome the differences in field values.

Reference point preprocessing can be used to show the change relative to a significant reference point (Braincel). An example would be judging the effectiveness of a medication to reduce temperature of a fever. The significant information is not the body temperature, but the change in body temperature relative to the normal temperature. If no obvious reference point is inherent in the problem, either the mean or the median could be tried as the reference point. The resulting figure could then be scaled using equation 10.

A method called thermometer scale or histogram preprocessing can be used to represent one field containing a data range as several fields with 1 or 0 values used to represent successive ranges (Braincel; Neuralyst). An example could be a field representing the number of dependents, which ranges from zero to five. Six fields could be substituted for the dependent field, with the fields defined as: >1, >2, >3, >4, >5 and >6. One dependent
would be represented as 110000, three dependents would be represented as 111100. Histogram preprocessing is used because NN tend to not be accurate for many significant digits. A network could learn to distinguish one field with values of 0, 2, 4, .6, .8 and 1, but it might learn a histogram representation quicker or with less classification error. BP networks should not be devised with outputs that must make many fine distinctions (Neuralyst, 41).

2.6.4 Momentum and Learning Rate

Implementation is closest to theory when momentum is zero and stepsize is small, [.005, .05]. In practice, learning heuristically occurs faster if the learning rate is set to [.05, .25] and is still stable. Larger learning rates, [.25, 1.0], may cause unstability, classes may never be learned and the system may become stuck in a local minima (Neuralyst; Rumelhart and McClelland 1988). Figure 3 shows the error over time, or number of training epochs, generally decreasing for a variety of learning rates. The line representing a learning rate of .02 shows the slowest gradual decrease in error. The learning rate of .30 gets stuck in a local minima in error space, and never improves. The learning rate of .20 is stuck for several hundred epochs before decreasing in error. The problem used for Figure 3 is described in more detail in the next section and in Figure 4. A network could be trained conservatively by starting with a low learning rate and increasing it if training is taking too long, or aggressively by starting with a high learning rate and decreasing it if unstability is observed (Neuralyst).

The momentum parameter should be set higher for larger learning rates. Zero momentum uses none of the previous error, only the current error; a momentum of one uses only the previous error. The PDP software defaults to a momentum of .9, Braincel defaults to .5, there is no one best initial value for momentum for all problems. Some packages do not use momentum, but change the learning rate during the course of training.
LEARNING RATE TESTS

Total Error

Figure 3. Learning Rate Tests.

2.7 Testing Backpropagation for Shift Invariance

A comparison between Tables 2 and 3 reveals that the time-dynamic classification methods resulted in lower error rates than the static methods. The table refers to MLP, which is synonymous with BP. There have been several modifications to BP which result in shift invariance for speech recognition (Waibel 1989) or character recognition (Le Cun 1990), which suggests that the unmodified BP does not perform well when shift invariance is required. To investigate the static vs. dynamic capabilities of BP further, the author designed a shift invariance test of BP. The test used a series of four pattern classes, A-D, which were designed on an eight by eight matrix to check horizontal shift invariance. The patterns were designed so that each one would resemble a single predominant frequency in speech, called a formant. The patterns for each class are symbolically represented in Figure 4. The actual values for each matrix element ranged from zero to .99, with a "*" representing the largest value in each
|| Class A | Class B | Class C | Class D |
|--------|--------|--------|--------|
| Training Sample 1 |
| Training Sample 2 |
| TARGET OUTPUT |
| $(A) B C D; (A) =$ target class  
$B =$ Max value indicates classifier result |

| 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 |

Test Pattern  
Shifted Horizontally  
1 Unit  
Actual Output, CORRECT 3/4

| 3 4 7 2 | 4 5 3 2 | 5 1 1 2 |

| 0 1 3 2 | 4 6 3 2 | 1 9 3 2 |

| 0 1 3 2 | 7 1 5 2 | 5 6 2 3 |

Figure 4. Backpropagation Fails a Shift Invariance Test.

horizontal row. On either side of the predominant frequency were intermediate values to smooth the transition from the peak to the background. The remaining elements in a row
were random values from .00 to .10, representing background noise. The values in each row added up to one. No patterns within a class were the same, both in the overall shape of the maximum values and the specific matrix values. The pattern variety and low-level randomness was designed into the patterns to prevent the network from correlating the patterns based on less significant specific details. Ideally the network would learn the non-varying characteristics of the overall pattern.

BP was trained and tested on several non-shifted samples as a control experiment. For the test experiment one original pattern for each class was fabricated. The first pattern was used to create additional patterns which were horizontally shifted copies of the original. The results of the test are listed in Figure 4, below each actual output. The target answer is parenthesized, the result is underlined. When the patterns were shifted one column, the network classified the data correctly 3 out of 4 times. When the patterns were shifted two or three columns, the network completely failed. When the patterns are shifted three columns, the target output is the lowest activation in all four classes, which indicates that no shape information is used. The only reason why any patterns were classified correctly when shifted once was because of the intermediate values on each side of the peaks intersected with the shifted pattern.

2.8 Time Delay Neural Networks

2.8.1 Time Delay Neural Network Background

Time Delay Neural Networks (TDNN's) have been developed as a modification of BP specializing in time-invariant speech recognition. The significance of time-invariance is that it avoids the difficulties of automatic segmentation of the speech to be recognized by the use of layers of shifting time-windows. The majority of the development of TDNN's has come from a research group at Carnegie Mellon University (Haffner, Waibel and Sawai 1988;
An example TDNN model is shown in Table 4 for illustration purposes. The nodes in the standard BP model are completely connected between each successive layer, which is not the case in the TDNN model. The TDNN model connects a time window from a lower layer to a single column in the next layer. The specific pattern of connections between the layers is the core of the time-invariant feature of the network. Waibel et al. (1989b) compared recognition of /b,d,g/ with TDNN and HMM and reported an error rate of 1.5 vs. 6.3, indicating that the TDNN error rate was one-fourth that of the HMM in this particular experiment.

### 2.8.2 TDNN Details

The example in Table 4 consists of a four-layer BP network, with an input matrix of 15 time slices by 16 Melscale filterbank coefficients representing frequency, and an output layer of three nodes classifying /b,d,g/. Each column of the input matrix is a successive time slice and each row represents a different frequency band. The input layer is connected to the first hidden layer with a time window three columns wide, and the first hidden layer and the second hidden layer are connected with a time window five columns wide. Specifically, the first three time slices of the input layer, nodes 0-47, are connected to the first column in the first hidden layer, nodes 240-247. This same arrangement shifts one column at a time. In the example nodes 16-63 are connected to nodes 248-255. Nodes 240-279 in the first hidden layer are connected to nodes 344-346 in the second hidden layer. Nodes 344,347,...368 are connected to the output node 371. In a BP network with the same sized layers, nodes 0-239 would be connected to nodes 240-343. When the weights are updated in a TDNN, they are updated by the average of all time-delayed weight changes. For this example (Table 4), the BP network
### Table 4
Structure of a Time Delay Neural Network

<table>
<thead>
<tr>
<th>Output Layer</th>
<th>1</th>
<th>371</th>
<th>372</th>
<th>373</th>
<th>/b/</th>
<th>/d/</th>
<th>/g/</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hidden Layer 2</td>
<td>1</td>
<td>344</td>
<td>347</td>
<td>350</td>
<td>353</td>
<td>356</td>
<td>359</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>345</td>
<td>348</td>
<td>351</td>
<td>354</td>
<td>357</td>
<td>360</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>346</td>
<td>349</td>
<td>352</td>
<td>355</td>
<td>358</td>
<td>361</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hidden Layer 1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>247</td>
<td>255</td>
<td>263</td>
<td>271</td>
<td>279</td>
<td>287</td>
<td>295</td>
<td>303</td>
<td>311</td>
<td>319</td>
<td>327</td>
<td>335</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>246</td>
<td>254</td>
<td>262</td>
<td>270</td>
<td>278</td>
<td>286</td>
<td>294</td>
<td>302</td>
<td>310</td>
<td>318</td>
<td>326</td>
<td>334</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>245</td>
<td>253</td>
<td>261</td>
<td>269</td>
<td>277</td>
<td>285</td>
<td>293</td>
<td>301</td>
<td>309</td>
<td>317</td>
<td>325</td>
<td>333</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>244</td>
<td>252</td>
<td>260</td>
<td>268</td>
<td>276</td>
<td>284</td>
<td>292</td>
<td>300</td>
<td>308</td>
<td>316</td>
<td>324</td>
<td>332</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>243</td>
<td>251</td>
<td>259</td>
<td>267</td>
<td>275</td>
<td>283</td>
<td>291</td>
<td>299</td>
<td>307</td>
<td>315</td>
<td>323</td>
<td>331</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>242</td>
<td>250</td>
<td>258</td>
<td>266</td>
<td>274</td>
<td>282</td>
<td>290</td>
<td>298</td>
<td>306</td>
<td>314</td>
<td>322</td>
<td>330</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>241</td>
<td>249</td>
<td>257</td>
<td>265</td>
<td>273</td>
<td>281</td>
<td>289</td>
<td>297</td>
<td>305</td>
<td>313</td>
<td>321</td>
<td>329</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>240</td>
<td>248</td>
<td>256</td>
<td>264</td>
<td>272</td>
<td>280</td>
<td>288</td>
<td>296</td>
<td>304</td>
<td>312</td>
<td>320</td>
<td>328</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input Layer</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15 (Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>16</td>
<td>15</td>
<td>31</td>
<td>47</td>
<td>63</td>
<td>79</td>
<td>95</td>
<td>111</td>
<td>127</td>
<td>143</td>
<td>159</td>
<td>175</td>
<td>191</td>
<td>207</td>
<td>223</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>14</td>
<td>30</td>
<td>46</td>
<td>62</td>
<td>78</td>
<td>94</td>
<td>110</td>
<td>126</td>
<td>142</td>
<td>158</td>
<td>174</td>
<td>190</td>
<td>206</td>
<td>222</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>13</td>
<td>29</td>
<td>45</td>
<td>61</td>
<td>77</td>
<td>93</td>
<td>109</td>
<td>125</td>
<td>141</td>
<td>157</td>
<td>173</td>
<td>189</td>
<td>205</td>
<td>221</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>12</td>
<td>28</td>
<td>44</td>
<td>60</td>
<td>76</td>
<td>92</td>
<td>108</td>
<td>124</td>
<td>140</td>
<td>156</td>
<td>172</td>
<td>188</td>
<td>204</td>
<td>220</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>11</td>
<td>27</td>
<td>43</td>
<td>59</td>
<td>75</td>
<td>91</td>
<td>107</td>
<td>123</td>
<td>139</td>
<td>155</td>
<td>171</td>
<td>187</td>
<td>203</td>
<td>219</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>10</td>
<td>26</td>
<td>42</td>
<td>58</td>
<td>74</td>
<td>90</td>
<td>106</td>
<td>122</td>
<td>138</td>
<td>154</td>
<td>170</td>
<td>186</td>
<td>202</td>
<td>218</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>09</td>
<td>25</td>
<td>41</td>
<td>57</td>
<td>73</td>
<td>89</td>
<td>105</td>
<td>121</td>
<td>137</td>
<td>153</td>
<td>169</td>
<td>185</td>
<td>201</td>
<td>217</td>
</tr>
<tr>
<td></td>
<td>09</td>
<td>08</td>
<td>24</td>
<td>40</td>
<td>56</td>
<td>72</td>
<td>88</td>
<td>104</td>
<td>120</td>
<td>136</td>
<td>152</td>
<td>168</td>
<td>184</td>
<td>200</td>
<td>216</td>
</tr>
<tr>
<td></td>
<td>08</td>
<td>07</td>
<td>23</td>
<td>39</td>
<td>55</td>
<td>71</td>
<td>87</td>
<td>103</td>
<td>119</td>
<td>135</td>
<td>151</td>
<td>167</td>
<td>183</td>
<td>199</td>
<td>215</td>
</tr>
<tr>
<td></td>
<td>07</td>
<td>06</td>
<td>22</td>
<td>38</td>
<td>54</td>
<td>70</td>
<td>86</td>
<td>102</td>
<td>118</td>
<td>134</td>
<td>150</td>
<td>166</td>
<td>182</td>
<td>198</td>
<td>214</td>
</tr>
<tr>
<td></td>
<td>06</td>
<td>05</td>
<td>21</td>
<td>37</td>
<td>53</td>
<td>69</td>
<td>85</td>
<td>101</td>
<td>117</td>
<td>133</td>
<td>149</td>
<td>165</td>
<td>181</td>
<td>197</td>
<td>213</td>
</tr>
<tr>
<td></td>
<td>05</td>
<td>04</td>
<td>20</td>
<td>36</td>
<td>52</td>
<td>68</td>
<td>84</td>
<td>100</td>
<td>116</td>
<td>132</td>
<td>148</td>
<td>164</td>
<td>180</td>
<td>196</td>
<td>212</td>
</tr>
<tr>
<td></td>
<td>04</td>
<td>03</td>
<td>19</td>
<td>35</td>
<td>51</td>
<td>67</td>
<td>83</td>
<td>099</td>
<td>115</td>
<td>131</td>
<td>147</td>
<td>163</td>
<td>179</td>
<td>195</td>
<td>211</td>
</tr>
<tr>
<td></td>
<td>03</td>
<td>02</td>
<td>18</td>
<td>34</td>
<td>50</td>
<td>66</td>
<td>82</td>
<td>098</td>
<td>114</td>
<td>130</td>
<td>146</td>
<td>162</td>
<td>178</td>
<td>194</td>
<td>210</td>
</tr>
<tr>
<td></td>
<td>02</td>
<td>01</td>
<td>17</td>
<td>33</td>
<td>49</td>
<td>65</td>
<td>81</td>
<td>097</td>
<td>113</td>
<td>129</td>
<td>145</td>
<td>161</td>
<td>177</td>
<td>193</td>
<td>209</td>
</tr>
<tr>
<td></td>
<td>01</td>
<td></td>
<td>16</td>
<td>32</td>
<td>48</td>
<td>64</td>
<td>80</td>
<td>096</td>
<td>112</td>
<td>128</td>
<td>144</td>
<td>160</td>
<td>176</td>
<td>192</td>
<td>208</td>
</tr>
</tbody>
</table>

(Horizontal Axis = Time; Vertical Axis = Frequency)
(15 frames at 10 ms frame rate)
has about five times more connections than the corresponding TDNN, making the TDNN about five times faster. While TDNN's may be significantly faster than BP, they are still significantly slower than other pattern recognition methods, and they are not very close to the speed needed for real-time speech recognition.

One of the methods investigated by Waibel (Waibel et al. 1989b) to speed up the training time of the networks was staged learning. Staged learning trains on data subsets to speed up learning time. The first subsets are small and quickly learned, with each successive subset rapidly increasing in size until the full training data set is used. The smaller data sets are learned quickly. The larger sets fine tune the networks to minimize error and to generalize better. An example of this method is discussed in Waibel et al. (1989b), where the network was trained to recognize /b,d,g/, using sets of size 3, 6, 9, 24, 99, 249 and 780 over a total of 35,000 iterations.

2.8.3 TDNN Extensions

One of the difficulties of the TDNN method is scaling it up to networks large enough to recognize the full set of phonemes. When the task size of recognizing three phonemes is doubled, the number of connections to be trained is tripled. One method to address this problem is to train a modular TDNN (Waibel 1989). The modular TDNN was created by training a /b,d,g/ network and a /p,t,k/ network, then saving the trained first hidden layer network from each network. The first hidden layer of the modular TDNN was composed of unmodifiable /b,d,g/ and /p,t,k/ modules and another modifiable module. The modifiable module was referred to as "connectionist glue" and develops the ability to switch the entire network between the output of the two fixed modules.
3.1 Speech Recognition Background and Challenges

3.1.1 Background and Terminology

Phonemes are typically grouped in articulatory or acoustic classes. Articulatory phonetics is based on the anatomical details of speech production, while acoustic phonetics has its basis in measurable waveform characteristics which discriminate the basic phonemes. Articulatory features may involve: (a) place of articulation, the anatomical location of the primary constriction in the vocal tract, (b) manner of articulation, the degree of constriction and (c) voicing, the presence of phonation. When whispering, all phonemes are unvoiced, whereas in normal speech some phonemes are voiced and others are unvoiced. The articulatory classes of phonemes analyzed in this project are stops, which include /b, d, g, p, t, k/. Slashes, "/", enclose phonemes in phonetic transcription. Stops are produced when the nasal passages are closed and the vocal tract is shut off at the point of articulation. Stops have the articulatory characteristics listed in Table 5.

Articulatory features have developed from research such as the theory of distinctive features by Jakobson, Fant and Halle (1952, 1956). In the ideal case, these features are independent of each other and may be represented by a binary string to indicate the presence or absence of the feature (Parsons 1987, 95). Features described below that were used in this project include compact/diffuse and voiced/voiceless. The attributes used in the theory of distinctive features, as quoted from Parsons (1987), are as follows:

1. Vocalic/nonvocalic. Refers to presence or absence of a well-defined
2. Consonantal/nonconsonantal. Consonantal implies a relatively small amount of total energy.
3. Compact/diffuse. Refers to distribution of spectral energy.
4. Tense/lax. Tense implies larger total energy with wider bandwidth and longer duration.
5. Voiced/voiceless. Voicing indicates the presence of low-frequency components due to vibration of the vocal cords.
7. Discontinuous/continuous. Discontinuous phonemes show abrupt changes in spectral energy spread.
8. Strident/mellow. Strident phonemes have stronger and more random noise components.
9. Checked/unchecked. Energy in checked phonemes appears as a burst, as in plosives (stops).
10. Grave/acute. Grave sounds are dominated by low-frequency resonances, acute ones by high-frequency resonances.
11. Flat/plain. Difference is one of relative energy of high-frequency resonances: flat weaker, plain stronger.
12. Sharp/plain. Sharp phonemes show a raising in the relative frequency of higher-frequency resonances. (Parsons 1987, 95)

Table 5
Place of Articulation vs. Voicing for Stop Consonants

<table>
<thead>
<tr>
<th>Front</th>
<th>Middle</th>
<th>Back</th>
</tr>
</thead>
<tbody>
<tr>
<td>Labial</td>
<td>Alveolar</td>
<td>Velar</td>
</tr>
<tr>
<td>Voiced</td>
<td>/b/</td>
<td>/d/</td>
</tr>
<tr>
<td>Unvoiced</td>
<td>/p/</td>
<td>/t/</td>
</tr>
</tbody>
</table>

One approach to speech recognition research consists of finding acoustic cues which may correlate with articulatory classes. One example of a grouping of phonemes by manner of articulation is the nasals /m, n, n/. A second example of an acoustic cue is a formant, which is a primary indicator of a vowel. Formants are vocal-tract resonances which show up as a concentration of energy in the speech spectrum. Peterson and Barney (1952) conducted a seminal study which successfully plotted the first formant against the second formant to
classify ten vowel sounds.

3.1.2 Difficulties in Speech Recognition

Once the acoustic cues are extracted, many difficulties remain in speech recognition. Klatt (1980) in his review of several complete speech recognition systems which start with waveform analysis and end with text, discusses many of the basic problem areas that must be dealt with in any system of bottom-up lexical analysis. These problem areas are listed in Table 6. Phonemes are heavily context-dependent and may be influenced by other phonemes before and after the phoneme to be recognized. Jusczyk (1986) gives a review of the research verifying that human listeners use context-dependent cues to identify speech. Many comprehensive speech recognition systems are based on a context-sensitive model. The models which are context-sensitive must build a representation of the possible permutations of contexts for every phoneme, which grows exponentially. Many of the most widely published context-sensitive systems use the Hidden Markov Model (HMM), which is based on two layers of probability (Rabiner and Juang 1986; and Rabiner 1989). One of the most successful projects in speech recognition, the Carnegie Mellon Sphinx project, uses HMM (Lee 1989). For HMM models, "the first-order Markov assumption makes it difficult to model
coarticulation directly and HMM training algorithms can not currently learn the topological structure of word and sub-word models” (Lippmann 1989b, 2).

Segmentation of speech into phonetic units and time normalization are interdependent and rarely error free. The transition from one phoneme to another is gradual and fluid, and the duration of specific phonemes varies widely. HMM’s use a form of time-alignment, the Viterbi algorithm (Forney 1973). Other statistical or template-based methods have used Dynamic Time Warping (DTW), an algorithm which employs a search strategy to try and optimize the best fit between a given template and the current input data (Rabiner 1978; and Vidal et al. 1988). A wide variety of neural net methods exist for time alignment, including the Viterbi net and Time Delay Neural Networks (TDNN) by Waibel (1989).

Talker normalization adjusts for the difference in the shape and length of a person’s vocal tract, in the amount of coarticulation (effects of local phonemes on the production of a phoneme), stress, speaking rate and sex or age-linked traits of a voice. Normalization methods may preprocess the data to try to subtract the differences which cause misclassification. Recognition methods may normalize by generalizing enough to ignore insignificant feature changes and use the meaningful features.

Lexical analysis involves labelling a given speech segment based on the analysis results. The size of the unit of recognition is a design parameter which may be diphone, phoneme, morpheme, word, or other unit of speech. The features of a given segment are compared with features in a lexicon, or dictionary, of known speech segments. The closest feature match identifies the speech segment. There are tradeoffs involved in using a larger atomic unit of recognition instead of a smaller one; a smaller unit of speech may be simpler to classify, but there may be more difficulty at higher levels of the recognition system involving searching and parsing to combine the atomic units.

The need for phonological recoding stems from the fact that spoken English differs from written English in many ways. For a given word, there may be many understandable
but different phoneme combinations. This point is illustrated with the example of a talker quickly or sloppily saying "would you," which is spoken with the "d y" pronounced as "j," as in "wouja." The number of recodings is quite large and attempts to reliably include all possible recodings are difficult at best.

Dealing with phonetic errors requires the analysis at higher levels of speech recognition not only to identify errors, but to form hypotheses which suggest other interpretations that should be re-investigated at a lower level. This can happen at many level boundaries, such as phoneme-to-syllable or syllable-to-word. The number of levels varies from system to system. Klatt (1977) reviews several major ARPA speech recognition projects which are multi-layer (e.g. HARPY, HEARSAY II and HWIM). Phonetic errors can be caused from problems such as environmental noise, talker variability, feature extraction error, segmentation error and classification error. Small errors early in the process tend to magnify as they pass through the system.

An ideal system would be able to interpret prosodic cues based on stressed syllables and intonation to find the talker's emphasis or attitude from a speech segment. Prosodic characteristics, such as stress and intonation, include the features of speech which range over syllables or words. In English, stress and intonation can change a sentence from a statement to a question or make it sarcastic. Interpretation of the stress pattern is depends on correct segmentation and knowledge of a given speaker's use of stress. This is rarely a component of current speech recognition systems because of the difficult analysis.

3.2 Speech Preprocessing

3.2.1 Speech Acquisition and Manipulation

Speech preprocessing issues should be addressed before a detailed discussion of speech recognition. The purpose of speech preprocessing is to develop the most meaningful
representation of the data to be analyzed. In speech research, the term "speaker" could be used to indicate the person talking or to refer to the electronic speaker. To avoid confusion, the term "talker" is often used to refer to the person producing the speech sample. Many experiments involve testing new combinations of preprocessing steps with classification algorithms to find the combination with the best overall classification. A brief listing of preprocessing steps that the typical speech recognition researcher would address is given in Table 7. When audio tapes are digitized to computer files, the data size is determined by the sample rate and the number of bits per sample, with more detail resulting in more storage. The low-pass filtering helps to prevent aliasing (Nyquist 1928; Parsons 1987) by removing the highest frequency sounds. Low-pass filtering also minimizes the effects of formants on pitch estimation using Linear Predictive Coding (LPC) (Witten 1982, 88). The pitch estimation is useful for separating voiced and unvoiced phonemes (Parsons 1987). A general rule developed by Sondhi (1968) was to choose the low-pass filtering level at 30% of the maximum frequency.

The speech manipulation stages are used in a variety of possible combinations. One

Table 7
Speech Preprocessing Steps and Parameters

<table>
<thead>
<tr>
<th>Data Acquisition</th>
<th>Speech Manipulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample Rate</td>
<td>Pre-Emphasis</td>
</tr>
<tr>
<td>8-20 kHz</td>
<td>Mel or Bark-Scale</td>
</tr>
<tr>
<td>Low-Pass Filtering</td>
<td>Hand-Segmentation</td>
</tr>
<tr>
<td>4-5 kHz</td>
<td>FFT or LPC</td>
</tr>
<tr>
<td>Sample Overlap</td>
<td>phoneme onset</td>
</tr>
<tr>
<td>0-5</td>
<td>To train method how to segment automatically</td>
</tr>
<tr>
<td></td>
<td>algorithm</td>
</tr>
<tr>
<td></td>
<td>Linear, then curve</td>
</tr>
<tr>
<td></td>
<td>algorithm</td>
</tr>
<tr>
<td></td>
<td>Logarithmic scale based on cochlea</td>
</tr>
<tr>
<td></td>
<td>phoneme onset</td>
</tr>
<tr>
<td></td>
<td>Time to frequency domain, giving a spectrogram</td>
</tr>
</tbody>
</table>

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
optional step is preemphasis, which enhances the high-frequency component. There are a
variety of scaling algorithms which are based on the knowledge of the approximately
logarithmic scale of sound discrimination of human hearing. One simpler method is to
emphasize the high frequencies, leaving the lower frequencies on a linear scale. This is
sometimes referred to as computing the first difference of the waveform. More recent studies
have used a Bark Scale (Klatt 1976; Zwicker and Terhardt 1980; and Syrdal and Gopal 1986)
or Mel-Scale, both of which have been empirically derived. The training data may be hand-
segmented or automatically segmented. Automatic segmentation, a subtask of speech
recognition, is not an error-free problem. Some researchers hand-segment their training data.
If the input to a classification algorithm is not a segmented template, the algorithm must try
to automatically segment the test data based on the examples given in the training data.

3.2.2 Time Domain to Frequency Domain Transforms

When audio signals are digitized, they are in the time domain; however, the vast
majority of speech analysis works with the data in the frequency domain (Witten 1982;
Parsons 1987; and Lieberman and Blumstein 1988). The conversion from time domain to
frequency domain is typically made using the Discrete Fourier Transform, computed with the
Fast Fourier Transform (FFT). Linear Predictive Coding (LPC) is a time domain method
which may be used instead of FFT (McCandless 1974; Itakura 1975; Makhoul 1975, 1977, 1978;
Witten 1982; and Parsons 1987). Data in the frequency domain can be used to plot a
spectrogram, a graph with frequency on a vertical axis vs. time on a horizontal axis, with
energy represented by darker shades for higher energies. Figure 5 shows a spectrogram of
"The auctioneer accepted the bid." A spectrogram shows vowel formant tracks as darker
streaks which are in general horizontal, with some rising and falling over time. The image
is composed of narrow vertical lines which blur together. The vertical lines, or striations, are
caused by glottal pulses. Glottal pulses are the vibration of sound caused by air passing
Figure 5. A Sample Spectrogram of "The auctioneer accepted the bid."

through the epiglottis creating a voiced phoneme. The shape of spectrogram striations changes over different speaking conditions, talker and pitch. The bandwidth of a spectrogram may range from narrow or wide. An example of a frequency that would be considered narrow is 50 Hz, a wide frequency is 300 Hz. Wide-bandwidth filters respond more rapidly to energy changes in the signal, and are better for identifying formant frequencies (Lieberman and Blumstein 1988).

An FFT (Cooley and Tukey 1965; Bergland 1969; Bracewell 1986; Press et al. 1988; and Embree and Kinble 1991) is used to analyze any complex waveform and to break the waveform down to a set of simple sinusoidal waveforms. One of the discoveries of Fourier analysis was harmonics, the relation of frequencies which are multiples of the smallest frequency, i.e., 500 Hz, 1000 Hz, 1500 Hz. An underlying assumption of the Fourier Transform is that the input sample is periodic, in other words, the sample segment is a pattern which repeats infinitely and the end of the signal wraps around to the front. To
prevent a sharp transition from joining the two ends of the signal, a window function, such as the Hamming or Hanning (Rabiner and Schafer 1979), is applied to scale both ends of the waveform to zero. Speech waveforms are quasi-periodic, but can be broken down into periodic-like segments of varying length. The FFT is an approximation because of this periodicity assumption. FFTs can be used to modify speech by performing the FFT, manipulating the frequency domain data, then performing an inverse FFT to convert the data back to the time domain. A faster variant of the FFT is the Hartley Transform (Hartley 1942; Bracewell 1984; Bracewell 1986; Akginya 1987; O'Neill 1988). The Hartley transform is roughly four times faster because it directly computes the power spectrum, and not the real and imaginary components. In contrast, the FFT directly computes the real and imaginary components, which need to be converted to the power spectrum. The power spectrum, $p$, is related to the real, $r$, and imaginary, $i$, components by the following equation:

$$p = \sqrt{r^2 + i^2}$$  \hspace{1cm} (11)

The LPC method generates smoother curves than the FFT (see Figure 6), but it is a time-domain method which also generates several frequency domain features and does not require the assumption of periodicity. The concept behind Linear Predictive Coding is to predict the current signal based on a linear combination of changes in the past signal (Atal and Hanauer 1971; McCandless 1974; Itakura 1975; Makhoul 1975, 1977, 1978; Written 1982; and Parsons 1987). The most commonly used LPC model is the all-pole, or autoregressive model (Parsons 1987, 137). The all-pole LPC is the closest to the transfer function of the vocal tract, which is also generally all-pole. The exceptions to the all-pole assumption are some fricatives and nasals, which may require a mixture of poles and zeros. The number of poles in an LPC algorithm should be over twice the maximum possible number of formants generated by the supralaryngeal vocal tract (Parsons 1987, 164). The LPC algorithm is heavily
used in research, but it does have minor drawbacks. The LPC method may have difficulty with noisy speech because white noise across the spectrum tends to flatten the formant peaks and valleys and can be confused by periodic background noise (Parsons 1987). This problem can be minimized with additional preprocessing.

3.3 Acoustic Invariance in Stop Consonants

The theory of acoustic invariance proposes that features in speech correlate with the method of acoustic production, and these features can be used to recognize phonemes independent of the phonetic environment. There is an active body of research using place of
articulation as a feature in speech recognition (Fant 1956; Stevens and Blumstein 1978, 1981; Blumstein and Stevens 1979, 1980, 1981; Searle, Jacobson and Rayment 1979; Kewley-Port 1981, 1983; Lahiri, Gewirth and Blumstein 1984; Forrest, Weismer, Milenkovic and Dougall 1988; and Lieberman and Blumstein 1988). In the seminal speech recognition project in this series, Blumstein and Stevens (1979) extracted a set of features to recognize place of articulation in stop consonants, /b,d,g,p,t,k/, obtaining 85% accuracy. The stops were found to be largely invariant of vowel environment, with the speech samples consisting of consonant-vowel (CV) and vowel-consonant (VC) pairs. The data for the study consisted of six talkers, with 300 samples per talker for a total of 1800 samples (Blumstein and Stevens 1979). The invariance hypothesis was tested by recognizing the stop regardless of the vowel, and regardless of the stop location in the CV or VC pair of phonemes. This series of studies involved both speech recognition and listening tests with synthetic speech signals. The speech generation experiments tested the hypothesis that the invariant features allowed human subjects to correctly classify simulated speech samples. Lahiri et al. (1984) did not expect the invariant features to be the only features used in speech recognition, they wanted to test the theory of acoustic invariance.

Using context-sensitive information in speech recognition generally gives better results than not using the context because of the changes to a phoneme by coarticulation with preceding or following phonemes. This is demonstrated by the success of models such as HMM's which heavily rely on contextual analysis, and the probabilities of different combinations of context. If speech recognition is to be enhanced, several areas for improvement include: context-invariant recognition, context analysis and better error analysis between the different levels of a complete speech recognition system. All of the higher layers of speech recognition must deal with errors at the lower levels, which tend to increase as they propagate through a system. Recognizing phonemes independently of context is a keystone of speech recognition because errors at this level are magnified later. If a small decrease in
error at the lowest level occurs, it could potentially reduce a disproportionally large degree
of error in the overall system. Context-invariant recognition rates are not comparable with
context-sensitive recognition rates because the additional information of context is
intentionally omitted when investigating invariance. One approach to enhancing speech
recognition is to improve context-invariant features and methods. The improved invariant
features could then be incorporated into context-based recognition systems in order to increase
overall speech recognition rates.

The acoustically invariant features investigated in this series of studies (Blumstein and
Stevens 1979, 1980, 1981; and Stevens and Blumstein 1978, 1981) were implemented using one
template for each place of articulation; front or labial, middle or alveolar and back or velar.
The templates were characterized as diffuse-falling, diffuse-rising and compact (see Figure 7).
To understand the templates, it is necessary to understand how the speech samples were
preprocessed before the templates were applied.

In the Blumstein and Stevens (1979) study, the speech samples were low-pass filtered
at 4800 Hz and sampled at 10 kHz. Then each speech sample was hand-marked to identify
the onset time, which is the beginning of the stop consonant. The waveform in the first 26

![Figure 7](image.png)

Figure 7. Sample Spectra Which Fit the Blumstein and Stevens Templates.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
milliseconds (ms) from the onset was multiplied by a modified raised-cosine time window (see Figure 8) to emphasize the stop consonant and to gradually de-emphasize any following phonemes. The time window duration of 26 ms was found empirically as the best way to show the hypothesized features. The first 13 ms of the waveform were unmodified, the second 13 ms are de-emphasized by multiplying by a decreasing factor. The factor decreased from one to zero over time in the shape of a cosine curve, which gradually phased out the significance of the second 13 ms. With a sampling frequency of 10 kHz, a sample was taken at burst or onset with a duration of 26 ms. The final step of the Blumstein and Stevens study (1979) was to transform the waveform from the time domain to the frequency domain using a 14-pole Linear Predictive Coding (LPC) algorithm.

The diffuse-falling, diffuse-rising and compact templates (Blumstein and Stevens 1979; 1004-1005) were applied to the result of the above transformations at the point of consonant onset. The LPC output for labials, /b, p/, was not concentrated at any frequency, hence diffuse, and consisted predominantly of high frequencies, therefore falling from left to right, from low frequency to high. Alveolars, /d, t/, were in the same manner diffuse-rising and

![Modified Raised-Cosine Window](image)

Figure 8. Modified Raised-Cosine Time Window.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
velars, /g, k/, were concentrated at some central frequency, therefore compact (see Figure 7). The theory of acoustic invariance proposes that phonemes produced in a similar articulatory manner have similar acoustic characteristics, which the study Blumstein and Stevens study (1979) was able to confirm with an 85% recognition accuracy.

3.4 Time-Varying Feature Extraction

Kewley-Port (Kewley-Port 1981, 1983; and Kewley-Port and Luce 1984) investigated time-varying features describing the stops /b,d,g/ in CV position only. The specific features were based on Blumstein and Stevens (1979) project. The classification in this study was not performed by computer, but by people experienced in reading spectrograms who were following specific guidelines. The three judges were trained in this task and then classified the data independently and collaboratively. For each speech sample, the judges used running spectra and the three features, which were reduced to binary categories:

Feature 1: Tilt of the spectrum at burst onset: Tilt was estimated by visually fitting a straight line to the first frame between 0 and 3500 Hz. The feature categories were R=rising and F=flat or falling.

Feature 2: Late onset of low-frequency energy. Late onset was defined as the occurrence of high amplitude, low frequency peaks (FI peaks) starting in the fourth frame of the display or later. Feature categories were L=late onset and N=no late onset.

Feature 3: Mid-frequency peaks extending over time. This feature was defined as the presence of a single, prominent peak between 1000 and 3500 Hz occurring for three or more frames, although not necessarily consecutive frames. The feature categories were Y=yes, peaks exist and N=no, no such peaks are present. (Kewley-Port 1983, 325)

The use of the three features is listed in Table 8 (Kewley-Port 1983, 325). In Table 8, the "?" entry indicates either feature could occur for that place of articulation, and the "L*" entry indicates late onset is sufficient to classify the sample in the velar category. The late onset feature is also referred to as the voice onset time (VOT), the delay in the onset of voicing relative to the burst. Time-varying features reported which were not present in analysis using one sample in time, such as the Blumstein and Stevens (1979) study, included the burst
Table 8
Kewley-Port Place of Articulation Features

<table>
<thead>
<tr>
<th>Tilt of Burst</th>
<th>Late Onset</th>
<th>Mid-Frequency Peaks</th>
<th>Assigned Consonant</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>N</td>
<td>N</td>
<td>/b/</td>
</tr>
<tr>
<td>R</td>
<td>?</td>
<td>N</td>
<td>/d/</td>
</tr>
<tr>
<td>?</td>
<td>L*</td>
<td>Y</td>
<td>/g/</td>
</tr>
</tbody>
</table>

duration and change in burst. Velar bursts were 20-30 ms in duration, in contrast to the 5-10 ms burst length of labials and alveolars. The compact spectrum of velar bursts varied slowly over the burst duration. From the Kewley-Port (1983) study, 88% of the stops were classified correctly, 92% for the four male talkers and 78% for the two female talkers.

The project by Lahiri et al. (1984) investigated time-varying features of alveolars and dentals in contrast with labials in English, Malayalam and French. Part of the data for this study were the data collected in the Blumstein and Stevens (1979) study. This project included a pilot study which found that the Blumstein and Stevens (1979) features at a static point in time failed to classify together dental and alveolar stops and failed to differentiate dentals and labials. The Blumstein and Stevens (1979) study did not include analysis of dentals. The pilot study motivated Lahiri et al. to investigate dynamic spectral changes over time. The features developed used temporal relationships of the changes in gross spectral shape.

Changes in distribution of energy from burst release to the onset of voicing were distinctively different for these two classes of stops. For labial stop consonants, either the difference in energy between the burst release and the onset of voicing was less at low frequencies than at high frequencies, or the difference in energy was about the same at low and high frequencies. For dental and alveolar consonants, the differences in energy between the stimulus onset and onset of voicing was less at high frequencies than at low
frequencies, whether the gross shape of the onset spectrum was diffuse-flat or diffuse-rising. (Lahiri et al. 1984, 393)

See Figure 9 for a visual description of how the features were extracted from plots of LPC over time (Lahiri et al. 394). They tracked the relative change in energy level of the second and fourth formant peaks (F2 and F4), consistently sampled at 1500 and 3500 Hz. The classification method could be defined using the points a,b,c,d in Figure 9 to specify the signed line segments ca, (c - a) and db, (d - b), which are used in the pseudo-code segment in Table 9. The two features were represented as ratios to show the change in energy over a specific time period. The first feature was the ratio of the change in energy at 3500 Hz. The second feature was the ratio of the change in energy at 3500 Hz to the change in energy at 1500 Hz. The second feature tries to capture the change in time of the F2 and F4 peaks to discriminate between labial and alveolar stops. Over time, the energy level for labials evenly decreases. The energy level for alveolars decreases at a faster rate in the higher frequencies. Using these time-varying features, this method was able to differentiate 91% alveolars and dentals from labials, which re-affirmed the existence of invariant properties of place of

![LPC Over 74 ms of [ba]](image)

![LPC Over 74 ms of [do]](image)

Figure 9. Lahiri et al. Time Dynamic Features for Labial vs. Dental and Alveolar Consonants.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
Table 9

Pseudo-Code Description of the Lahiri et al. Classification

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF (db &lt; 0) OR (0 &lt; db/ac &lt; .5)</td>
</tr>
<tr>
<td>class = alveolar</td>
</tr>
<tr>
<td>ELSE</td>
</tr>
<tr>
<td>class = labial-dental</td>
</tr>
<tr>
<td>ENDIF</td>
</tr>
</tbody>
</table>

articulation when time-dynamic features are used.

A relatively recent project in this area was done by Forrest, Weismer, Milenkovic and Dougall (1988). Their contribution was a method of abstracting the Blumstein and Stevens templates statistically. Five male and five female talkers were used to read a list of 31 test words in a sentence. The use of test words differs from the Blumstein and Stevens (1979) approach of having the talker read a list of CV and VC pairs, yielding all possible combinations of vowel context. It does not appear that the Forrest et al. project thoroughly tested consonant recognition in all permutations of possible vowel environments. In this project, Forrest et al. computed the mean, variance, skewness and kurtosis of the FFT at each time slice on both linear and Bark scale transformed data. These features were picked as a way of abstracting the overall center, tilt, concentration and peakedness of the energy distribution. Negative skewness indicates the spectrum has predominantly high frequencies, and positive skewness indicates predominant low frequencies. High kurtosis and low variance can indicate a compact predominant midfrequency peak. The four spectral moments are computed from the FFT power spectrum $p(k)$ and frequency $f_k$ at the $k^{th}$ sample. $K$ ranges from 1 to 256 because of the size of an FFT.

\[ 0 \leq p(k) \leq 1; \quad \sum_{k=1}^{k=256} p(k) = 1 \]  \hspace{1cm} (12)
These four statistical features are also referred to as the four moments \(M_1, M_2, M_3\) and \(M_4\):

\[
M_1 = \sum_{j=1}^{j=256} f_j \Delta(j)
\]

\[
M_2 = \sum_{j=1}^{j=256} (f_j - M_1)^2 \Delta(j)
\]

\[
M_3 = \sum_{j=1}^{j=256} (f_j - M_1)^3 \Delta(j)
\]

\[
M_4 = \sum_{j=1}^{j=256} (f_j - M_1)^4 \Delta(j)
\]

\[
\text{Dimensionless } M_3 = \frac{M_3}{(M_2)^{3/2}}
\]

\[
\text{Dimensionless } M_4 = \frac{M_4}{(M_2)^{2}} - 3
\]

According to Newell and Hancock (1984), dimensionless versions of the moments \(M_3\) and \(M_4\) were computed and then "normalized with respect to shifts in center frequency and frequency scale that can occur between subjects producing the same sound" (Forrest et al. 1988, 118). The four moments have also been applied not only to voiceless stops, but also to voiceless fricatives, \(/f, \emptyset/\) and sibilants, \(/s, \emptyset/\) by (Forrest et al. 1988) and to voiceless fricatives by Tomiak (1990) in his doctoral dissertation. The classification method used in Forrest et al. was stepwise discriminant analysis (BMDP7M). Using the first 40 ms of the VOT, 92% of the voiceless stops were classified correctly. The primary contribution of this project was the use of the four moments as a way of automatically abstracting the Blumstein and Stevens (1979) place of articulation characterizing the diffuse-falling, diffuse-rising and compact templates.
The Kewley-Port method was not tested as an automated method; it used human judges.

Investigations into time dynamic feature extraction were continued by Sawusch and Dutton (1991). They developed three criteria for evaluating metrics to be used for classification of voiced stops and vowels based on a preliminary study. The three criteria were: (1) robustness, the acoustic features need to be reliably classified correctly across a wide variety of samples, (2) graceful degradation, the metric should not fail catastrophically in the presence of noisy data and (3) computability, the metric should be expressible in algorithmic or formula form and should not rely on human judges. Based on these criteria Sawusch and Dutton developed two metrics, or feature extraction methods. These metrics were the peak difference and the spectral moments metrics.

The peak difference metric used frequency differences between adjacent formant peaks to categorize vowels, which omits details of the formant frequency location. The Bark scale was used for preprocessing before computing the differences between the formant peaks (F1-F0), (F2-F1), (F3-F2), (F4-F3) and (F4-F2). The relationship between formant peaks for classifying vowels is well established and has been used since 1952 (Peterson and Barney). The peak difference metric was also employed over time to classify voiced stops, using (F1-F0), (F2-F1) and (F3-F2). The peak difference method was robust, but it failed catastrophically when formant peaks were missing.

The second metric was the spectral moments metric, which was used for classification of voiced stops. It is based on the place of articulation features determined by Blumstein and Stevens (1979) and moment abstraction of Forrest et al. (1988). The use of the moments for classification by Sawusch and Dutton (1991) is listed in Table 10. This metric is not as robust or accurate as the peak difference metric, but it has the advantage of graceful degradation. The metrics were combined because the weakness of one metric is the strength of another for classification of voiced stops and vowels. It was suggested that the metrics be combined with a fuzzy-logical or neural network method which could process continuous speech, although
Table 10
Classifying Stops With Spectral Moments

<table>
<thead>
<tr>
<th>Moment</th>
<th>Labial, /b/</th>
<th>Alveolar, /d/</th>
<th>Velar, /g/</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1$, Mean</td>
<td>Rising (Increasing)</td>
<td>Falling (Decreasing)</td>
<td>Steady or Steady then Falling</td>
</tr>
<tr>
<td>$M_2$, Variance</td>
<td>Increasing</td>
<td>Decreasing</td>
<td>Increasing then Falling</td>
</tr>
<tr>
<td>$M_3$, Skewness</td>
<td>Positive then Changes to Vowel</td>
<td>Negative then Changes to Vowel</td>
<td>Variable (Near Zero)</td>
</tr>
<tr>
<td>$M_4$, Kurtosis</td>
<td>Low and Variable</td>
<td>Low and Variable</td>
<td>Positive then Decreasing</td>
</tr>
</tbody>
</table>

specifics were not discussed. This was a preliminary study to develop the metrics. The metrics were not integrated and used for classification.

The application of the moments metric to five voiceless fricatives was the subject of Tomiak’s dissertation, resulting in error rates of 22% to 26% for linear and Bark scale (1990). When the two nonsibilant fricatives were excluded, an error rate of 8% was achieved. Further studies were conducted investigating the effects of sample duration on classification. It was found that nonsibilant information is contained in the onset and/or offset, and 100 ms was needed to best classify sustained frication. It was also concluded that systematic time window placement as a function of the phonetic category was a difficulty which needs to be addressed.
CHAPTER IV

PROJECT DESIGN AND DISCUSSION

4.1 Project and Model Selection

The purpose of this thesis was to explore neural networks and pattern recognition. Speech recognition was chosen because it is a significant challenge, with many potential benefits. The first stage of this project investigated context invariant speech recognition based on a seminal project in speech recognition (Blumstein and Stevens 1979). The second stage was expanded on the first stage with a series of investigations into classification techniques, both neural network and statistical, with emphasis on time-dynamic recognition techniques. Speech scientists tend to spend more time researching features of speech recognition and generation, while computer scientists tend to focus on the development of classification methods; my goal was to combine the best feature extraction with the best classification techniques. The Blumstein and Stevens (1979) investigation into context-invariant recognition was chosen as the seminal speech recognition project, and copies of their original audio recordings were obtained from the authors. The first stage was designed to reproduce that study, specifically the preprocessing and the feature extraction.

In designing the second stage of this project many other successful time-dynamic techniques were considered and eliminated for use on this specific speech recognition problem. The time-dynamic method selected was Time Delay Neural Networks. In a project design, the reasons for excluding certain methods can be as important as reasons for including others. The other recognition methods considered included Hidden Markov Models (HMM) and Dynamic Time Warping (DTW). HMM systems, such as SPHINX, have been shown to
be successful for large vocabulary, continuous, sentence-level speech recognition (Lee 1988). HMMs can be weaker at the lowest level of phoneme recognition and at the highest level involving unusual sentence context. HMM are thoroughly context-sensitive, and do not seem to be an appropriate tool for investigating the concept of context invariant phoneme recognition because of their heavy reliance on contextual information. If an accurate context invariant method is developed, it would be a complementary addition to a HMM for phoneme level recognition. Learning Vector Quantizer (LVQ) neural networks and HMM have been combined for speech recognition (McDermott, Iwamida, Katargiri and Tohkura 1990). DTW is an established time-dynamic method which is not inappropiate for this investigation, although it has not been as heavily investigated in the last few years as other NN models. Time-dynamic NN models have been reaching the accuracy of HMM systems, with added benefits of easy parallel implementation for real-time operation, and the potential of real-time adaption to new examples. Among the several time-dynamic NN models, Time Delay Neural Nets (TDNN) were chosen because of the published success of this model. Another advantage of the TDNN model is that it was implementable in the portable PDP public domain backpropagation software (Rumelhart and McClelland 1989).

4.2 Methods

In the Blumstein and Stevens (1979) study the templates were hand-developed by trial and error, and they were tested by hand with "all-or-nothing" classification criteria. This exact labor-intensive procedure was not followed, but the closest automated approximation was used. Following Forrest et al. (1988), an attempt was made to capture the Blumstein and Stevens templates using spectral moments. Forrest et al. did not test all vowel environments because the speech samples they used did not consist of consonant-vowel (CV) and vowel-consonant (VC) pairs. In my study, the original Blumstein and Stevens data were used, which contained multiple samples of each combination of CV and VC pairs. In addition to the
moments, overall energy was added to the feature set. To approximate the classification method used by Blumstein and Stevens, linear discriminant functions were used.

The data consisted of 300 samples from each talker, comprising four lists of 75 samples, with each list using three consonants and five vowels. There were four male and two female talkers in the Blumstein and Stevens data set. The lists incorporated voiced, /b,d,g/ and unvoiced, /p,t,k/, stop consonants in CV and VC pairs. The lists were: (a) voiced CV, (b) voiced VC, (c) unvoiced CV and (d) unvoiced VC. There were three different samples of each unique combination of consonant and vowel per list of 75 samples.

The data preprocessing involved sampling at 10 kHz with a resolution of 12 bits, followed by low-pass filtering at 4.8 kHz. The samples were taken every 12.8 ms for a duration of 25.6 ms to blend the data. The most time consuming step was digitizing and hand-labeling the onset time, which is the beginning of the stop consonant. A modified raised-cosine time window was used before processing the data with a 14 pole LPC. From the LPC output, the four moments and energy level for each sample were computed using the moment equations listed earlier in section 3.4. The moments and energy level were linearly scaled to a zero to one range using the maximum and minimum values for each feature in the total set of speech samples.

\[
Feature_{\text{Scaled}} = \frac{\text{Feature} - \text{Min(Feature)}}{\text{Max(Feature)} - \text{Min(Feature)}}
\]

(19)

The processing of the speech data began with hooking up a reel-to-reel tape player to a low-band-pass filter and digitizing using a PDP-11 running the RT11 real-time operating system. For convenience of having all the data online, the data was transferred to an Advanced Logic Research (ALR) 486/33 microcomputer. An existing C library of about 150 speech processing routines was ported to Microsoft C on the ALR. The C library was originally developed by Dr. Jim Hillenbrand, other associates and students on a variety of
platforms. The onset times were found using a digital signal processing package (DSP) donated by Hyperception. The user interface in the Hyperception package greatly facilitated hand-processing of each speech sample to find the onset location. The moment and onset data were loaded into a dBase file for creating neural network input templates, selecting random samples and other manipulations. The statistical classification studies were conducted using SAS (version 6) PROC DISCRIM and STEPDISC on a VAX cluster. The Parallel Distributed Processing backpropagation (BP) public domain neural network software (Rumelhart and McClelland 1989) was used to implement the Time Delay Neural Network.

4.3 Classification Tests

The classification tools used for this investigation were the PDP BP, SAS and a Maximum Likelihood Estimator (MLE) program. Fewer TDNN studies were conducted in comparison with SAS studies because the typical TDNN training time on a data set of four speakers was significantly longer. The full data set was split into four talkers for training and two for testing. The training set was 2/3 of the total data set, but it could have been any size from 1/2 to 99/100 of the total data set. The training set size is based on the quantity of data: studies with less data tend to use a larger percentage of the total data set to build a more accurate representation of the data. All multi-talker experiments used the same training and test data. Single talker studies split the data set in the same proportion, using a random selection to split the full data set into training and test sets. After preliminary classification tests, the data were re-checked for errors in hand-processing. Some data samples were re-digitized because the consonant release feature was partially missing. Onset times in some samples were changed to match auditory cues that were not apparent looking at the waveform. In one list, voiced VC for talker 4, it was particularly difficult to find the onset times because the release was minimal. Many samples in this list sounded like a vowel with an abrupt end, and it was difficult even for a listener to classify the stop.
Prototype TDNN studies were developed from small, single talker two-class test studies to the full training set size of 1,200 samples using three classes (Table 11). Final TDNN single talker tests were trained starting with the full training set. Multiple talker tests used a staged learning approach discussed earlier as a method to speed up learning. The number of samples in each successive set was 3, 9, 27, 99, 402 and 1200. The samples in each learning set contained an equal number of samples of each of the three output classes. Each

<table>
<thead>
<tr>
<th>Type</th>
<th>Talkers Used in:</th>
<th>Number of Training Error</th>
<th>Test Set</th>
<th>Test Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Talker</td>
<td>T2 (75%)</td>
<td>T2 (25%)</td>
<td>8,000</td>
<td>16.2%</td>
</tr>
<tr>
<td>(15x5 template)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Multi-Talker</td>
<td>T1,T2,T5,T6</td>
<td>T3,T4</td>
<td>30,000</td>
<td>19.0%</td>
</tr>
<tr>
<td>(15x5 template)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Tn, n=Talker number

sample for a particular learning class was randomly selected from any speaker and list in the training set. In developing the staged learning TDNN, multiple experiments were conducted, using both exclusive and inclusive sets. Exclusive sets used training sets containing samples exclusive of all other sets except for the final, complete, set of training data. The staged learning description (Waibel 1989) did not specify if the sets were exclusive or inclusive. The exclusive set training was investigated as a method of faster network training, but it was found that this did not work well, possibly because the initial sets were too small to be
representative of the class to be learned. Later studies used inclusive staged learning, that is, the larger sets contained all samples used in the smaller sets. The inclusive staged learning was used in the final TDNN model.

The traditional statistical methods included a linear discriminant function and MLE for purposes of reproducing the Blumstein and Stevens (1979) study (Table 12). Both the linear and quadratic discriminant function were used so a comparison between them could

Table 12

Statistical Discriminant Error Rates

<table>
<thead>
<tr>
<th>CV AND VC COMBINED</th>
<th>Resubstitute</th>
<th>Cross-validate</th>
<th>Classify New Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>15 TIME SLICES</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linear Discrim</td>
<td>19.1%</td>
<td>23.5%</td>
<td>36.0%</td>
</tr>
<tr>
<td>Quadratic Disc</td>
<td>6.1%</td>
<td>21.6%</td>
<td>33.2%</td>
</tr>
<tr>
<td>KNN, K=4</td>
<td>19.9%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KNN, K=7</td>
<td>20.1%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KNN, K=11</td>
<td>23.6%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KNN, K=23</td>
<td>26.2%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLE</td>
<td>18.0%</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>5 TIME SLICES</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linear Discrim</td>
<td>27.7%</td>
<td>29.9%</td>
<td>36.3%</td>
</tr>
<tr>
<td>Quadratic Disc</td>
<td>16.1%</td>
<td>22.6%</td>
<td>34.8%</td>
</tr>
<tr>
<td>KNN, K=4</td>
<td>18.4%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KNN, K=7</td>
<td>17.3%</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>3 TIME SLICES</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linear Discrim</td>
<td>32.4%</td>
<td>33.9%</td>
<td>41.8%</td>
</tr>
<tr>
<td>Quadratic Disc</td>
<td>8.9%</td>
<td>21.6%</td>
<td>34.0%</td>
</tr>
<tr>
<td>KNN, K=4</td>
<td>15.8%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KNN, K=7</td>
<td>15.6%</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>1 TIME SLICE</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linear Discrim</td>
<td>33.1%</td>
<td>33.5%</td>
<td>35.9%</td>
</tr>
<tr>
<td>Quadratic Disc</td>
<td>27.4%</td>
<td>28.5%</td>
<td>38.3%</td>
</tr>
<tr>
<td>KNN, K=4</td>
<td>18.2%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KNN, K=7</td>
<td>18.9%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
be made, and the KNN method was chosen because it has been a successful non-parametric
PR method used in pattern recognition (Patrick and Fischer 1970). Table 12 summarizes the
discriminant studies with three different columns of error rates. The first column,
resubstitution, shows the misclassification rate of the training data. This column reflects the
best classification, which should not be considered to be the average classification capability
for data outside of the training set. Crossvalidation is the process of training the PR
algorithm on all data samples except for one, and testing classification on the sample that was
left out. Crossvalidation is sometimes referred to as "jackknifing" or the leave-one-out rule
(Lachenbruch and Mickey 1968). The process of leaving one sample out is repeated for every
data sample, with the total error rate based on all tests. The classify new data column lists
results from training on four talkers and testing on two. The final results are those listed in
the cross-validate and in the classify new data columns. For the first series of tests, the input
templates consisted of 15 time slices centered on the onset time, which is similar to the 15
time slices used in the TDNN studies, with the four moments and energy level replacing the
15 cepstral coefficients. A second series of tests used five time slices with the VC time slices
chronologically reversed to make them more similar to the CV samples as another way of
testing the place of articulation feature. For the five time slice samples, the onset was placed
at the beginning of the sample. A third type of template consisted of three time slices, and
a fourth type uses only one time slice, designed to be similar to the single time slice used in
the Blumstein and Stevens (1979) templates. The tests in Table 12 classified place of
articulation in both CV and VC order, and the tests in Table 13 classified place of articulation
for CV separately for the 5x5, 3x5 and 1x5 templates, showing more detail for the studies in
Table 12.

Table 14 shows the confusion matrices for the quadratic discriminant and KNN (K=4).
The confusion matrices show details which reveal how the data are misclassified. The 5x5
templates reflect fewer time slices, and are closer to the Blumstein and Stevens (1979) study,
Table 13
CV and VC Statistical Discriminant Error Rates

<table>
<thead>
<tr>
<th>Time Slices</th>
<th>CV ONLY</th>
<th>VC ONLY</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Resubstitue</td>
<td>Cross-validate</td>
</tr>
<tr>
<td>5 TIME SLICES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linear Discrim</td>
<td>28.3%</td>
<td>30.9%</td>
</tr>
<tr>
<td>Quadratic Disc</td>
<td>18.5%</td>
<td>22.0%</td>
</tr>
<tr>
<td>KNN, K=4</td>
<td>20.3%</td>
<td></td>
</tr>
<tr>
<td>KNN, K=7</td>
<td>20.8%</td>
<td></td>
</tr>
</tbody>
</table>

| 3 TIME SLICES |         |         |         | 19.1% | 20.6% | 40.3% |
| Linear Discrim | 27.2% | 30.0% | 39.7% | 6.9% | 14.6% | 32.6% |
| Quadratic Disc | 14.4% | 24.3% | 33.5% | 17.8% |         |         |
| KNN, K=4 | 20.4% |         |         | 12.9% |         |         |
| KNN, K=7 | 21.1% |         |         |         |         |         |

| 1 TIME SLICE |         |         |         | 18.4% | 20.1% | 34.4% |
| Linear Discrim | 35.9% | 37.2% | 41.1% | 28.1% | 28.8% | 31.0% |
| Quadratic Disc | 30.5% | 33.0% | 41.8% | 19.9% |         |         |
| KNN, K=4 | 18.4% |         |         | 16.8% |         |         |

the 15x5 templates comparable with the 15 time slice data samples used in the TDNN studies.

4.4 Analysis of Results

In the first stage of this project the original Blumstein and Stevens (1979) data and the place of articulation features were statistically abstracted by the use of the four moments (Forrest et al. 1988). These speech data were classified using a linear discriminant function yielding a best error rate of 19.1%, which is comparable to the 15% error rate reported by Blumstein and Stevens (1979). Forrest et al. (1988) developed the use of moments for place of articulation, and reported an error rate of 8% for vowel-consonant (VC) classification; however, their data set did not include all combinations of VC pairs, or both VC and
Table 14
Confusion Matrices for Quadratic and KNN

Resubstitution Results Using Quadratic Discriminant Function for 15x5 Features

<table>
<thead>
<tr>
<th>Group Classified to:</th>
<th>Back</th>
<th>Front</th>
<th>Middle</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Back</td>
<td>361</td>
<td>23</td>
<td>14</td>
<td>398</td>
</tr>
<tr>
<td></td>
<td>90.70</td>
<td>5.78</td>
<td>3.52</td>
<td>100.00</td>
</tr>
<tr>
<td>Test Samples</td>
<td>5</td>
<td>382</td>
<td>10</td>
<td>397</td>
</tr>
<tr>
<td></td>
<td>1.26</td>
<td>96.22</td>
<td>2.52</td>
<td>100.00</td>
</tr>
<tr>
<td>Middle</td>
<td>3</td>
<td>18</td>
<td>377</td>
<td>398</td>
</tr>
<tr>
<td></td>
<td>0.75</td>
<td>4.52</td>
<td>94.72</td>
<td>100.00</td>
</tr>
<tr>
<td>Total</td>
<td>369</td>
<td>423</td>
<td>401</td>
<td>1193</td>
</tr>
<tr>
<td>Percent</td>
<td>30.93</td>
<td>35.46</td>
<td>33.61</td>
<td>100.00</td>
</tr>
<tr>
<td>OVERALL % ERROR</td>
<td>6.12</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Resubstitution Results Using K=4 K Nearest Neighbors for 5x5 Features

<table>
<thead>
<tr>
<th>Group Classified to:</th>
<th>Back</th>
<th>Front</th>
<th>Middle</th>
<th>Other</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Back</td>
<td>258</td>
<td>92</td>
<td>26</td>
<td>22</td>
<td>398</td>
</tr>
<tr>
<td></td>
<td>64.82</td>
<td>23.12</td>
<td>6.53</td>
<td>5.53</td>
<td>100.00</td>
</tr>
<tr>
<td>Test Samples</td>
<td>14</td>
<td>377</td>
<td>6</td>
<td>0</td>
<td>397</td>
</tr>
<tr>
<td></td>
<td>3.53</td>
<td>94.96</td>
<td>1.51</td>
<td>0.00</td>
<td>100.00</td>
</tr>
<tr>
<td>Middle</td>
<td>11</td>
<td>23</td>
<td>351</td>
<td>13</td>
<td>398</td>
</tr>
<tr>
<td></td>
<td>2.76</td>
<td>5.78</td>
<td>88.19</td>
<td>3.27</td>
<td>100.00</td>
</tr>
<tr>
<td>Total</td>
<td>283</td>
<td>492</td>
<td>383</td>
<td>35</td>
<td>1193</td>
</tr>
<tr>
<td>Percent</td>
<td>23.72</td>
<td>41.24</td>
<td>32.10</td>
<td>2.93</td>
<td>100.00</td>
</tr>
<tr>
<td>OVERALL % ERROR</td>
<td>17.34</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
consonant-vowel (CV) pairs.

In the second stage of this project both static statistical methods and the time-dynamic TDNN method were investigated. Among the statistical methods, the lowest error rate on Table 14 was 6.1% for the quadratic discriminant function on the larger feature set containing 15 time slices. The lower error rate for the 15x5 vs. 5x5 input matrix for the linear and quadratic discriminant functions indicate that they used information in the additional time slices. The higher error rate for the 5x5 templates with chronologically reversed VC pairs reflects that chronological reversal does not improve classification when using the linear or quadratic discriminant. The KNN method resulted in higher error rates for the additional time slices, which indicates that the extra time slices may have added noise, or variance, to the system to make classification more difficult. Fewer time slices yielded less accuracy with the discriminants and more accuracy with KNN, which illustrates the point that assumptions cannot easily be made regarding increasing or decreasing the number of features presented to a PR algorithm to minimize the error rate. The VC classification studies were more accurate than the CV, which I would not have guessed because of the difficulty in marking some of the VC samples.

When comparing my results listed on Tables 11 and 12 with the results of the other classification projects on Tables 2 and 3, the studies most similar in design are the ones that should be contrasted. In general, projects classifying fewer consonants or using data from a single talker should have a lower error rate than studies with more consonants or talkers. My project used six talkers and six consonants in both CV and VC positions, and tested unlabelled speech. In Table 2, the most similar project to this study is by Elman and Zisper (1988), which achieves 5.0% error on the consonants /b,d,g/ for one talker using backpropagation. The 6.1% that I was able to achieve with six talkers vs. one indicates that my results are comparable with other studies.

The time-dynamic projects listed in Table 3 have very low error rates, 1.5% to 1.7%
for multiple talker stops, which indicates that my multi-talker TDNN error rate of 19.0% can be improved. Improvements may include investigating my implementation of the TDNN, or using preprocessing steps in the TDNN study such as cepstral or melscale coefficients. The PDP configuration files were able to correctly connect the layers of the TDNN, but the PDP software did not allow a subtle modification to the error update rule. The TDNN method specifies that a given node should be updated once with the average of the updates from each sliding time-window to which the node belongs.

4.5 Directions for Future Research

Future studies could compare the spectrogram input data with the features based on the four moments and energy. A principal components analysis on the four moments and energy could also prove interesting. A study on preprocessing to maximize the difference between the diffuse-rising and diffuse-falling templates could include using the Mel-Scale or Bark scale along with forms of various frequency pre-emphasis. Continued study in the area of feature extraction could include more work in abstracting the change over time with features such as those reported by Lahiri, Gewirth and Blumstein (1984). The moment and energy features should be classified to a more traditional spectrographic or cepstral representation.

Investigations into further PR modifications could include the combination of single-pass learning methods with a time-window structure such as the one used in TDNN. The single-pass learning methods under consideration include the Probabilistic Neural Net (PNN) by Specht (Maloney and Specht 1989, Specht 1988, Specht 1990, Specht and Shapiro 1990) and the real-time net by Malkoff (1990). Both of these single-pass learning methods are based on a multi-variate Bayes model and report a speed-up potential of 10,000 to 1,000,000 over backpropagation, with the trade-off of speed for increased memory usage. A significant contribution could involve the development of a single-pass TDNN model. The value of
single-pass learning can be illustrated with the fact that TDNN networks in this project iterated for 30,000 epochs over three days, whereas the single-pass learning would require 1 epoch, representing a 30,000:1 improvement.
Appendix A

Human Subjects Institutional Review Board Acceptance Letter
Date: August 9, 1990
To: Greg Makowski
From: Mary Anne Bunda, Chair
Re: HSIRB Project Number: 90-08-06

This letter will serve as confirmation that your research protocol, "Context-Insensitive Phoneme Recognition Using Neural Networks," has been approved under the exempt category of review by the HSIRB. The conditions and duration of this approval are specified in the Policies of Western Michigan University. You may now begin to implement the research as described in the approval application.

You must seek reapproval for any changes in this design. You must also seek reapproval if the project extends beyond the termination date.

The Board wishes you success in the pursuit of your research goals.

xc: Drs. Pinkowski, Trenary, Hillenbrand - Computer Science

Approval Termination: August 9, 1991
Appendix B

Parallel Distributed Processing Configuration Files for
Time Delay Neural Networks
The public domain Parallel Distributed Processing (PDP) software uses several configuration files to specify the network structure (*.net), screen layout (*.tem) and batch commands for controlling the system (*.str). Detailed information on the structure of these files is included in an appendix in Rumelhart and McClelland, 1988. The input matrix to this network is 15 time slices by 5 features ($M_1$, $M_2$, $M_3$, $M_4$ and energy). The hidden layers are 13 by 4 and 9 by 3. The output layer is 3 by 1, with each output node corresponding to a different place of articulation.

[file: TD-M.NET]

definitions:
nunits 157
ninputs 75
noutputs 3
end
	network:
% r 75 4 0 15
% r 79 4 5 15
% r 83 4 10 15
% r 87 4 15 15
% r 91 4 20 15
% r 95 4 25 15
% r 99 4 30 15
% r 103 4 35 15
% r 107 4 40 15
% r 111 4 45 15
% r 115 4 50 15
% r 119 4 55 15
% r 123 4 60 15
% r 127 3 75 20
% r 130 3 79 20
% r 133 3 83 20
% r 136 3 87 20
% r 139 3 91 20
% r 142 3 95 20
% r 145 3 99 20
% r 148 3 103 20
% r 151 3 107 20
% 154 3 127 27
end
biases:
%r 75 81
end


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.


Braincel. Promised Land Technologies, Inc., New Haven, CT.


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.


Hampshire, John and Alex Waibel. "A Novel Objective Function for Improved Phoneme Recognition Using Time-Delay Neural Networks." IEEE Transactions on Neural Networks 1, no. 2 (June 1990): 216-228.


<table>
<thead>
<tr>
<th>Index Term</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>acoustic</td>
<td>1, 32-34, 41, 42, 45, 50, 70-74, 79</td>
</tr>
<tr>
<td>acoustic invariance</td>
<td>41, 42, 45, 70, 74</td>
</tr>
<tr>
<td>acute</td>
<td>33</td>
</tr>
<tr>
<td>Adaptive Resonance</td>
<td>7</td>
</tr>
<tr>
<td>Agbinya</td>
<td>40, 69</td>
</tr>
<tr>
<td>AI</td>
<td>16</td>
</tr>
<tr>
<td>AIP</td>
<td>70, 80</td>
</tr>
<tr>
<td>Aleksander</td>
<td>69</td>
</tr>
<tr>
<td>ALR</td>
<td>54</td>
</tr>
<tr>
<td>alveolar</td>
<td>33, 43, 46-48, 47, 51</td>
</tr>
<tr>
<td>American Institute of Physics</td>
<td>69, 70, 76, 80</td>
</tr>
<tr>
<td>Anderson</td>
<td>11, 14, 69, 74, 77, 80</td>
</tr>
<tr>
<td>articulatory</td>
<td>32, 33, 45</td>
</tr>
<tr>
<td>artificial intelligence</td>
<td>16, 71</td>
</tr>
<tr>
<td>Atal</td>
<td>40, 69</td>
</tr>
<tr>
<td>backpropagation</td>
<td>1, 7-9, 14, 16-18, 20, 23, 26, 27, 53, 55, 61, 62, 75</td>
</tr>
<tr>
<td>Ball</td>
<td>69</td>
</tr>
<tr>
<td>bandwidth</td>
<td>33, 39, 81</td>
</tr>
<tr>
<td>Bark</td>
<td>37, 38, 48, 50, 51, 62</td>
</tr>
<tr>
<td>Bayes</td>
<td>4, 5, 20, 62</td>
</tr>
<tr>
<td>Bayesian</td>
<td>7, 8, 12, 23</td>
</tr>
<tr>
<td>Bengio</td>
<td>15, 69</td>
</tr>
<tr>
<td>Bentley</td>
<td>6, 69, 71</td>
</tr>
<tr>
<td>Bergland</td>
<td>39, 70</td>
</tr>
<tr>
<td>Blumstein</td>
<td>1, ii, 38, 39, 42-46, 48-50, 52, 53, 54, 57-59, 62, 70, 74, 79</td>
</tr>
<tr>
<td>Blumstein and Stevens</td>
<td>1, 42-46, 48-50, 52-54, 57-59</td>
</tr>
<tr>
<td>Boltzmann</td>
<td>15</td>
</tr>
<tr>
<td>BP</td>
<td>8, 9, 15-20, 24-26, 28, 29, 31, 55</td>
</tr>
<tr>
<td>Bracewell</td>
<td>39, 40, 70</td>
</tr>
<tr>
<td>Brain-state-in-a-box</td>
<td>14</td>
</tr>
<tr>
<td>Braincel</td>
<td>24, 25, 70</td>
</tr>
<tr>
<td>Bryson</td>
<td>17, 70</td>
</tr>
<tr>
<td>BSB</td>
<td>14</td>
</tr>
<tr>
<td>Burr</td>
<td>6-8, 70</td>
</tr>
<tr>
<td>burst</td>
<td>14, 33, 44-46, 45, 46</td>
</tr>
<tr>
<td>burst onset</td>
<td>45</td>
</tr>
<tr>
<td>Cepstrum</td>
<td>76</td>
</tr>
<tr>
<td>Charles-Luce</td>
<td>78</td>
</tr>
<tr>
<td>Chien</td>
<td>3, 4, 70</td>
</tr>
<tr>
<td>coarticulation</td>
<td>35, 42</td>
</tr>
<tr>
<td>codebook</td>
<td>12</td>
</tr>
<tr>
<td>Cohen</td>
<td>70</td>
</tr>
<tr>
<td>compact</td>
<td>32, 33, 43-46, 48, 49</td>
</tr>
<tr>
<td>conference</td>
<td>69, 70, 72, 73, 75-80</td>
</tr>
<tr>
<td>connectionist</td>
<td>15, 31, 69, 71, 72, 78-80</td>
</tr>
<tr>
<td>consonant</td>
<td>9, 42-44, 46, 48, 53-55, 59, 61</td>
</tr>
<tr>
<td>consonantal</td>
<td>33</td>
</tr>
<tr>
<td>continuous</td>
<td>7, 6, 9, 15, 19, 22, 33, 50, 53, 70, 73, 74, 79</td>
</tr>
<tr>
<td>Croley</td>
<td>4, 39, 70</td>
</tr>
<tr>
<td>cosine</td>
<td>44, 54</td>
</tr>
<tr>
<td>Counterpropagation</td>
<td>72</td>
</tr>
<tr>
<td>crossvalidation</td>
<td>58</td>
</tr>
<tr>
<td>CV</td>
<td>10, 42, 45, 48, 53, 54, 57-59, 58, 61, 78</td>
</tr>
<tr>
<td>Cybenko</td>
<td>22, 70</td>
</tr>
<tr>
<td>D'Argenio</td>
<td>7, 78</td>
</tr>
<tr>
<td>DARPA</td>
<td>16</td>
</tr>
<tr>
<td>De Mori</td>
<td>69</td>
</tr>
<tr>
<td>Denker</td>
<td>70, 74, 76</td>
</tr>
<tr>
<td>dental</td>
<td>46-48</td>
</tr>
<tr>
<td>Devijver</td>
<td>70</td>
</tr>
<tr>
<td>diffuse</td>
<td>32, 33, 43, 44, 47, 49, 62, 74</td>
</tr>
<tr>
<td>diffuse-falling</td>
<td>43, 44, 49, 62</td>
</tr>
<tr>
<td>diffuse-rising</td>
<td>43, 44, 47, 49, 62</td>
</tr>
<tr>
<td>Doddington</td>
<td>10, 70</td>
</tr>
<tr>
<td>Dougall</td>
<td>42, 48, 71</td>
</tr>
<tr>
<td>DTW</td>
<td>35, 52, 53, 80</td>
</tr>
<tr>
<td>Duda</td>
<td>3, 5, 71</td>
</tr>
<tr>
<td>Dutton</td>
<td>50, 78</td>
</tr>
<tr>
<td>Dynamic Time Warping</td>
<td>35, 52, 77</td>
</tr>
<tr>
<td>E-set</td>
<td>14</td>
</tr>
<tr>
<td>Elman</td>
<td>10, 9, 61, 71</td>
</tr>
<tr>
<td>Embree</td>
<td>39, 71</td>
</tr>
<tr>
<td>English</td>
<td>35, 36, 46, 78, 79</td>
</tr>
<tr>
<td>epoch</td>
<td>18, 63</td>
</tr>
<tr>
<td>ExplorNet</td>
<td>3000, 71</td>
</tr>
<tr>
<td>F1</td>
<td>45, 50</td>
</tr>
<tr>
<td>F2</td>
<td>47, 50</td>
</tr>
<tr>
<td>F3</td>
<td>50</td>
</tr>
<tr>
<td>F4</td>
<td>47, 50</td>
</tr>
<tr>
<td>Fant</td>
<td>32, 42, 71, 72, 79</td>
</tr>
<tr>
<td>Feature/Filter</td>
<td>Name</td>
</tr>
<tr>
<td>----------------</td>
<td>------</td>
</tr>
<tr>
<td>feature-map</td>
<td>8, 10, 11</td>
</tr>
<tr>
<td>FFT</td>
<td>37-41, 40, 48</td>
</tr>
<tr>
<td>filter</td>
<td>37, 54, 72, 73</td>
</tr>
<tr>
<td>Finkel</td>
<td>6, 71</td>
</tr>
<tr>
<td>first difference</td>
<td>38</td>
</tr>
<tr>
<td>Fischer</td>
<td>5, 58, 77</td>
</tr>
<tr>
<td>Fisher</td>
<td>4, 5, 20, 71</td>
</tr>
<tr>
<td>flat</td>
<td>33, 45, 47</td>
</tr>
<tr>
<td>FMC</td>
<td>10</td>
</tr>
<tr>
<td>formant</td>
<td>15, 26, 33, 38, 39, 41, 47, 50, 71, 75, 78</td>
</tr>
<tr>
<td>Forney</td>
<td>15, 35, 71</td>
</tr>
<tr>
<td>Forrest</td>
<td>42, 48-50, 53, 59, 71</td>
</tr>
<tr>
<td>four moments</td>
<td>1, 49, 54, 58, 59, 62</td>
</tr>
<tr>
<td>Franzini</td>
<td>71</td>
</tr>
<tr>
<td>French</td>
<td>46</td>
</tr>
<tr>
<td>fricatives</td>
<td>40, 49, 51</td>
</tr>
<tr>
<td>Friedman</td>
<td>6, 71</td>
</tr>
<tr>
<td>fuzzy</td>
<td>5, 50, 73</td>
</tr>
<tr>
<td>Gale</td>
<td>71</td>
</tr>
<tr>
<td>Gaussian</td>
<td>7, 10, 9, 10, 14</td>
</tr>
<tr>
<td>Gewirth</td>
<td>42, 62</td>
</tr>
<tr>
<td>Gold</td>
<td>10, 9, 15, 75</td>
</tr>
<tr>
<td>Gonzalez</td>
<td>71</td>
</tr>
<tr>
<td>Gopal</td>
<td>38, 79</td>
</tr>
<tr>
<td>graceful degradation</td>
<td>50</td>
</tr>
<tr>
<td>grave</td>
<td>33</td>
</tr>
<tr>
<td>Grossberg</td>
<td>70</td>
</tr>
<tr>
<td>Haffner</td>
<td>28, 29, 71, 78</td>
</tr>
<tr>
<td>Halbert</td>
<td>80</td>
</tr>
<tr>
<td>Hall</td>
<td>69, 71, 77</td>
</tr>
<tr>
<td>Halle</td>
<td>32, 71, 72</td>
</tr>
<tr>
<td>Hamming</td>
<td>7, 40</td>
</tr>
<tr>
<td>Hampshire</td>
<td>8, 29, 72, 80</td>
</tr>
<tr>
<td>Hanauer</td>
<td>40, 69</td>
</tr>
<tr>
<td>Hanazawa</td>
<td>29, 80</td>
</tr>
<tr>
<td>Hancock</td>
<td>49, 76</td>
</tr>
<tr>
<td>Hanning</td>
<td>40</td>
</tr>
<tr>
<td>Hart</td>
<td>3, 5, 71</td>
</tr>
<tr>
<td>Hartigan</td>
<td>72</td>
</tr>
<tr>
<td>Hartley</td>
<td>40, 69, 70</td>
</tr>
<tr>
<td>Hebb</td>
<td>16, 72</td>
</tr>
<tr>
<td>Hecht-Nielsen</td>
<td>16, 17, 22, 71, 72</td>
</tr>
<tr>
<td>Hidden Markov Model</td>
<td>34</td>
</tr>
<tr>
<td>Hilbert</td>
<td>75</td>
</tr>
<tr>
<td>Hinton</td>
<td>13, 29, 69, 71, 74, 79, 80</td>
</tr>
<tr>
<td>HMM</td>
<td>13, 29, 34, 35, 42, 52, 53, 75</td>
</tr>
<tr>
<td>Ho</td>
<td>17, 70</td>
</tr>
<tr>
<td>Huang</td>
<td>6, 7, 10, 11, 20, 72</td>
</tr>
<tr>
<td>Hyperception</td>
<td>ii, 55</td>
</tr>
<tr>
<td>hypersphere</td>
<td>8, 21</td>
</tr>
<tr>
<td>invariance</td>
<td>26, 27, 26, 28, 41-43, 45, 70, 74</td>
</tr>
<tr>
<td>invariant</td>
<td>1, 28, 29, 42, 43, 47, 52, 53, 79</td>
</tr>
<tr>
<td>Itakura</td>
<td>38, 40, 72</td>
</tr>
<tr>
<td>Jakobson</td>
<td>32, 71, 72</td>
</tr>
<tr>
<td>James</td>
<td>3, 4, 17, 69, 72, 74, 78</td>
</tr>
<tr>
<td>Juang</td>
<td>15, 34, 77</td>
</tr>
<tr>
<td>Jusczyk</td>
<td>34, 72</td>
</tr>
<tr>
<td>K-D trees</td>
<td>6</td>
</tr>
<tr>
<td>k-Means</td>
<td>7, 72</td>
</tr>
<tr>
<td>k-Nearest Neighbor</td>
<td>5-7, 77</td>
</tr>
<tr>
<td>Kai-Fu Lee</td>
<td>75, 80</td>
</tr>
<tr>
<td>Kammerer</td>
<td>10, 73</td>
</tr>
<tr>
<td>Kandel</td>
<td>5, 73</td>
</tr>
<tr>
<td>Katagiri</td>
<td>13, 75</td>
</tr>
<tr>
<td>Kewley-Port</td>
<td>42, 45, 46, 45, 46, 50, 73</td>
</tr>
<tr>
<td>Kimble</td>
<td>71</td>
</tr>
<tr>
<td>Klatt</td>
<td>34, 36, 38, 73</td>
</tr>
<tr>
<td>KNN</td>
<td>1, 5, 6, 8, 10, 9, 12, 21, 57-59, 58, 61</td>
</tr>
<tr>
<td>Koberle</td>
<td>79</td>
</tr>
<tr>
<td>Kohonen</td>
<td>7, 8, 10-12, 73-75</td>
</tr>
<tr>
<td>Kolmogorov</td>
<td>22, 74</td>
</tr>
<tr>
<td>Kupper</td>
<td>10, 73</td>
</tr>
<tr>
<td>kurtosis</td>
<td>48, 51, 76</td>
</tr>
<tr>
<td>labial</td>
<td>33, 43, 46-48, 47, 51</td>
</tr>
<tr>
<td>Lachenbruch</td>
<td>58, 74</td>
</tr>
<tr>
<td>Lahiri</td>
<td>42, 46-48, 47, 62, 74</td>
</tr>
<tr>
<td>Lang</td>
<td>13, 29, 74, 80</td>
</tr>
<tr>
<td>lax</td>
<td>33</td>
</tr>
<tr>
<td>LDF</td>
<td>4, 5, 20</td>
</tr>
<tr>
<td>Le Berge</td>
<td>74</td>
</tr>
<tr>
<td>Le Cun</td>
<td>26, 74</td>
</tr>
<tr>
<td>Learning Vector Quantizer</td>
<td>8, 11, 53</td>
</tr>
<tr>
<td>least mean square</td>
<td>11</td>
</tr>
<tr>
<td>leave-one-out</td>
<td>58</td>
</tr>
<tr>
<td>Lee</td>
<td>6, 7, 34, 53, 74, 75, 80</td>
</tr>
<tr>
<td>lexical</td>
<td>34, 35, 73</td>
</tr>
</tbody>
</table>
Lieberman 38, 39, 42, 74
Linear Prediction 75
Linford 11, 79
Linggard 11, 79
Lippmann 6, 7, 6-8, 10, 9, 11-15, 17, 20, 35, 72, 74, 75
LMS 11
Lohnes 4, 70
Lorentz 22, 75
LPC 37, 38, 41, 40, 41, 44, 47, 54
Luce 45, 73, 78
LVQ 8, 10-13, 53, 75
Ml 49, 51, 67
M2 49, 51, 67
M3 49, 51, 67
M4 49, 51, 67
Makhoul 38, 40, 75
Makowski 1-3, ii, 15, 65, 75
Malayalam 46
Malkoff 62, 75
Maloney 62, 75
Martin 15, 75
Maximum Likelihood Estimator 1, 7, 20, 55
McCandless 38, 40, 75
McClelland 14, 15, 17, 20, 22, 25, 53, 55, 67, 78
McCulloch 16, 75
McDermott 13, 53, 75
mean 11, 24, 48, 51
Mel-Scale 38, 62
Mickey 58, 74
Microsoft 54, 75
Milenkovic 42, 48, 71
Minsky 16, 76
MLE 1, 7, 20, 55, 57
MLP 10, 9, 13, 26
moment 51, 50, 54, 55, 62
Moore 10, 77
Multilayer Perceptrons 9
multiple-stepszie BP 8

Nyquist 37, 76
O’Neill 40, 76
Omhundro 6, 76
onset 9, 37, 43-46, 45-47, 51, 54, 55, 58, 70
oral 33
Parabolic Distributed Processing 17, 55, 66, 67, 75, 78
parametric 5, 58
Parker 17, 76
Parsons 32, 33, 37, 38, 40, 41, 76
Parzen 76
Patrick 3, 5, 58, 77
Paul 15, 71, 75
PDP 17, 25, 53-55, 62, 67
peak 27, 45, 48, 50
Peeling 10, 77
Pellionisz 11, 69, 74
perceptron 7, 10, 9-11, 16, 20, 77
Peterson 33, 50, 77
phoneme 12, 34-37, 39, 42, 53, 65, 72, 75, 78, 80
Pinkowski ii, 4, 65, 77
pitch 37, 39, 71, 76, 78, 80
Pitts 16, 75
place of articulation 1, 15, 33, 32, 42, 43, 46, 45, 48-50, 58, 59, 67, 69, 73, 74, 78, 79
plosives 33
PNN 62
power spectrum 40, 48
PR 2, 3, 5, 58, 61, 62
pre-emphasis 37, 62
preprocessing 1, 5, 9, 23-25, 36, 37, 41, 50, 52, 54, 62
Press 39, 69, 70, 72, 74-80
principal components analysis 11, 62
PROC DISCRIM 55
Rabiner 15, 34, 35, 40, 77
release 46, 55
research funding 16
resonance 7
resubstitution 1, 58, 59
Robbins 17, 77

nasal 32, 33
Neuralyst 23-25, 76
Newell 49, 76
NeXT 18, 25, 29
Niemann 3, 76
Noll 76

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
robust 7, 50, 75
Rosenberg 8, 77, 78
Rosenblatt 16, 77
Rosenfeld 11, 69, 74
Rossen 14, 77
Rumelhart 14, 15, 17, 20, 22, 25, 53, 55, 67, 78

sample overlap 37
sample rate 37
sampling resolution 37
SAS 55, 78
Sawai 13, 28, 29, 71, 78, 80
Sawusch 50, 78
Schafer 40, 77
Schalk 10, 70
Searle 42, 78
segmentation 8, 9, 12, 28, 34-38
Sejnowski 8, 69, 71, 78, 79
Shadmeirh 7, 78
Shapiro 62, 78
sharp 33, 40
Shikano 13, 29, 80
sigmoid 7, 18, 19, 24
skewness 48, 51, 76
Smythe 15, 78
Sondhi 37, 78
Specht 62, 75, 78
spectra 43, 45, 70, 73, 75
spectrogram 37, 39, 38, 39, 62
Sphinx 34, 54, 72
Sprecher 22, 79
statistics 23, 71, 72, 76, 77
Steinbuch 16, 79
Stevens 1, ii, 42-46, 48-50, 52-54, 57, 58, 59, 70, 79
stop 1, 33, 41-44, 46, 54, 55, 70, 72-74, 79
Stork 70
strident 33
Stubbs ii, 23, 79
supervised 3, 4, 7, 6, 11, 12, 17, 18
Symons 7, 79
Syrdal 38, 79

Tank 13, 14, 79, 80
Tattersall 11, 79
TDNN 1, 12, 28, 29, 31, 35, 53, 55-59, 61-63
tense 33
Terhardt 38, 81
Texas Instruments 10
Theumann 79
Thomason 71
TI 10
tilt 45, 46, 48
Time Delay Neural Networks 1, 3, 12, 28, 35, 52, 66, 80
time warping 35, 52, 77
time-alignment 35
time-varying 45-47, 73
Tomiak 49, 51, 79
Tourretzky 69, 71, 72, 74, 75, 79
Tukey 39

Unnikrishnan 13, 14, 80
unsupervised 3, 7, 6

variance 4, 5, 48, 51, 61
VC 42, 48, 53-55, 57-59, 61
Vector Quantizer 8, 11, 12, 53
velar 33, 43, 45, 46, 51
Vidal 35, 80
Viterbi 15, 35, 71
vocalic 32
voice onset time 45
voiced 33, 32, 33, 37, 39, 50, 54, 55
voiceless 32, 33, 49, 51, 71, 79
VOT 45, 49
vowel 1, 3, 8, 33, 34, 38, 42, 48, 51, 53, 54, 55, 59, 61, 70, 79
VQ 12

Waibel 1, 8, 13, 26, 28, 29, 31, 35, 56, 71, 72, 74, 75, 78, 80
Wasserman 17, 80
Watrous 13, 14, 80
Weismer 42, 48, 71
Werbos 17, 80
White 9, 17, 41, 80
Wightman 16
window 29, 40, 44, 51, 54, 62
Wise 75, 80
Witten 37, 38, 80

XOR 16, 22

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.