The Use of Artificial Neural Networks for Phonetic Recognition

by

Hong Chung Leung

S.M., Massachusetts Institute of Technology
(1985)

B.E.E.E., City College of New York
(1981)

Submitted in Partial Fulfillment
of the Requirements for the
Degree of

Doctor of Philosophy

at the

Massachusetts Institute of Technology

May 1989

Signature of Author .....
Department of Electrical Engineering and Computer Science
May 16, 1989

Certified by .....
Victor W. Zue
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Departmental Committee on Graduate Students

The Use of Artificial Neural Networks
for Phonetic Recognition
by
Hong Chung Leung

Submitted to the Department of Electrical Engineering
and Computer Science on May 16, 1989 in partial fulfillment
of the requirements for the degree of Doctor of Philosophy

Abstract

Speech recognition is difficult mainly because of the high degree of variability in the encoding of phonetic information in the speech signal. The lack of success in accounting for this variability is a direct reflection of our incomplete understanding of the processes of human speech production and perception. Although great improvements have been made in the understanding of the acoustic properties of speech sounds, our knowledge in this area is still far from perfect. While more researchers are embracing solutions based on self-organizing pattern recognition techniques, there is increasing evidence that these techniques can be made more powerful by appropriately utilizing acoustic-phonetic knowledge. It is possible that a well-balanced use of our knowledge and such self-organizing techniques can lead to better speech recognition performance. However, problems with current recognition systems are either that the recognition framework is very powerful but too rigid for incorporating more specific speech knowledge or that there is a significant amount of human knowledge in the system but the control strategy is not powerful enough.

Recently, there has been a resurgence of interest in artificial neural networks (ANN's). Due to their flexible self-organizing framework, ANN's can potentially bridge the gap between our knowledge in speech and powerful self-organizing mechanisms. This thesis is concerned with the use of ANN's for phonetic recognition. There are three major objectives. First, by investigating ANN's in order to gain a better understanding of their basic characteristics and capabilities, we may be able to exploit them more fully as pattern classifiers. Second, by properly applying our acoustic-phonetic knowledge, we can potentially enhance the flexible framework of ANN's for phonetic recognition. Third, by comparing them with traditional pattern classification techniques, we can better understand the merits and shortcomings of the different approaches.

The multi-layer perceptron (MLP) was selected for our investigation, which centered around a set of vowel recognition experiments. In order to isolate different sources of variability in the speech signal, four different databases were used for our study. The largest database consists of 22,000 vowel tokens extracted from continuous sentences in the TIMIT database, spoken by 550 male and female speakers. The performance of the network was evaluated in several ways. Evaluation in terms of

average agreement with the phonetic transcription suggests that the performance of the network compares favorably to human performance in perceptual experiments. Evaluation along the phonological dimension suggests that most of the confusions between the network and transcription labels are quite reasonable.

Next, the characteristics and representations of the MLP were explored. Specifically, we examined the performance of the network as a function of the number of training iterations, amount of training data, number of hidden units, number of hidden layers, and use of the nonlinear sigmoid function. We also discuss the structure and self-organization of the internal representations, choices for output representations, and the use of heterogeneous input representations. Other issues that we discuss include error metrics for training the network, initializations of the network, and rapid adaptation of the network to a new speaker.

Finally, the performance of the network was compared with that of two traditional classification techniques. For our vowel classification task, experiments demonstrate that the MLP can yield higher performance than k-nearest neighbor and Gaussian classifiers. The results suggest that the MLP can provide an effective alternative for pattern classification, especially if the classification problem is not well understood.

Thesis Supervisor: Dr. Victor W. Zue
Title: Principal Research Scientist

Acknowledgments

I express my deepest gratitude and appreciation to my thesis advisor, Victor Zue, for creating a marvelous research environment, which brought me to MIT, and enabled me to pursue this research. I thank him for his continuous support, and stimulating questions, as well as for his constant guidance and insightful feedback. Over the past few years, he has also offered me valuable advice on various occasions, for which I will always be grateful.

There are many more people who have contributed to this research. I would like to extend my appreciation to:

The members of my thesis committee, Richard Lippmann, Campbell Searle, and Kenneth Stevens, for their interest in this work, and for their many helpful insights and suggestions.

Nancy Daly, Jim Glass, Mike Phillips, John Pitrelli, and Stephanie Seneff, for reading carefully drafts of this thesis, and for their comments and suggestions. Special thanks are due Stephanie, who spent considerable time spotting bugs in my early drafts.

Scott Cyphers, Katy Isaacs, Rob Kassel, David Kaufman, and Keith North, for keeping things running smoothly. I would also like to thank Scott, Katy, Rob, and David for developing powerful tools on the Lisp Machines.

Dave Whitney, and particularly Rob Kassel, for creating and maintaining an exceptional environment for making documents and figures, and for answering virtually infinite numbers of questions.

Arlene Wint and Vicky Palay for their assistance.

All members of the Speech Communication Group and Spoken Language Systems Group for their support, and understanding of how much machine time I needed. In particular, I would also like to thank Jim Glass, Lori Lamel, Jeff Marcus, Mike Phillips, John Pitrelli, Mark Randolph, and Stephanie Seneff for discussions that helped clarify issues related to this work.

Christina, Kenneth, and Suk for putting up with me, giving me lots of encouragement, and making delicious meals.

My parents for their love, and for instilling within me the value of education and hard work since I was too young to understand. Wing, my sister, and my brother, for their encouragement, and believing that I have actually finished my thesis.

Suk for her endless love, patience, and support through our ups and downs in the past years.

This research was supported by DARPA.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Automatic speech recognition by computer has been a topic that many researchers from diverse areas have been studying for a few decades. Over these many years of active research, different approaches have been suggested. However, it is still not clear what approach will successfully lead to a computer that can achieve performance comparable to human listeners. During the past few years, there has been increasing evidence that different approaches can be made more powerful by properly utilizing specific speech knowledge. It is possible that an approach that has a well-balanced use of speech knowledge and self-organizing techniques or algorithms can result in a robust speech recognition system. While speech knowledge enables the algorithms to function more intelligently, the algorithms can make efficient use of our knowledge and model our ignorance.

This thesis reports an investigation into the use of artificial neural networks in phonetic recognition. Specifically, it explores their basic characteristics and examines how the self-organizing frameworks can be exploited and applied to phonetic recognition when they are augmented with our acoustic-phonetic knowledge.

1.1 Motivation

Past approaches to speech recognition fall into two major extremes. Both have their own merits as well as shortcomings. In this section, we will describe these two early approaches and discuss some attempts to combine the advantages offered by these two extremes. We will also point out some problems and motivate the investigation of artificial neural networks.

1.1.1 Past Approaches to Speech Recognition

Over the past few decades, many researchers have been drawn to the problem of developing heuristically-based or rule-based speech recognition systems [17,31,34,84, 99,136,137,138]. Such an approach has the intuitive appeal that it focuses on the linguistic information in the speech signal and exploits our specific speech knowledge, such as the processes of human speech production and perception, inherent characteristics of different speech sounds, coarticulatory effects, and phonotactic constraints. As a result, speech recognition systems designed and developed this way can potentially discard extra-linguistic information in the speech signal and be less sensitive to changes in talker and environmental characteristics than approaches that do not explicitly extract relevant phonetic information from the speech signal.

This approach of utilizing specific speech knowledge gained further momentum and popularity after a series of spectrogram-reading experiments in the late 1970's [24, 25,140]. The experiments demonstrate that by proper extraction and integration of multiple acoustic cues, a great deal of phonetic information can be extracted from the acoustic speech signal. However, what the experiments have not suggested is *how* the acoustic cues should be reliably extracted by a computer or *what* control strategy is the most appropriate for integrating the acoustic cues. Spectrogram reading is used only as a paradigm to demonstrate the importance of utilizing acoustic-phonetic knowledge in speech recognition independent of the specific approach, as well as in other areas of

speech research, such as synthesis. Nevertheless the results of the experiments have led to the speculation that perhaps high performance phonetic recognition systems can be achieved by capturing our acoustic-phonetic knowledge in the form of a set of heuristic rules.

However, the performance of such rule-based recognition systems has not been very encouraging. Due to our incomplete understanding of the process through which phonetic information is encoded in the speech signal, the production rules intended to describe the variations of different speech sounds still cannot deal with the complicated, highly variable speech signal. Furthermore, while great strides have been made in the discovery and quantification of relevant acoustic attributes for phonetic contrasts, relatively little is known about how they should interact in reaching a unified final decision. As an example, information concerning the underlying feature of voicing for an intervocalic stop in English may be encoded in the duration of the release, the intensity of the burst, the fundamental frequency contour, the presence of low-frequency energy, and the duration of the preceding vowel [73,93]. While these acoustic attributes have been identified and quantified, relatively little is known about how they should be utilized collectively. In other words, our improved knowledge in the acoustic-phonetic characteristics of the speech signal is overshadowed by the ignorance in control strategy.

In contrast to such an approach, many speech recognition systems take the other extreme and rely primarily on self-organizing pattern recognition algorithms. The strong appeal is that these algorithms can be trained automatically, thus bypassing human intervention which could be subjective, inconsistent and very time-consuming. Furthermore, these algorithms can provide a mechanism to model our ignorance in control strategy, or other speech-related knowledge that we have not obtained. With enough training data and computational power, it is possible that these techniques can extract enough statistical regularities from their relatively primitive input representations. This approach has been successfully applied to various tasks, including recognition of phonemes, isolated words and continuous speech [1,2,109,110,118].

Despite the successes in applying this approach, it is still questionable whether it can be extended to achieve performance comparable to human listeners when dealing with continuous speech, multiple speakers and very large vocabularies. While great improvements have been made in the understanding and training of the self-organizing techniques, relatively little is understood about how specific speech knowledge accumulated over the past few decades can be incorporated into such systems. In other words, it is possible that the well-defined self-organizing techniques could be made even more intelligent by proper utilization of our speech knowledge.

1.1.2 Systematic Application of Speech Knowledge

Conceivably, each of these two extreme approaches in speech recognition can be improved by adopting features offered by the other. Therefore, it is not unreasonable to expect that a middle ground somewhere along the continuum could be more effective. On the one hand, our acquired speech knowledge can provide guidance to the structure and design of a self-organizing mechanism. On the other hand, a self-organizing mechanism can provide a control strategy for utilizing our speech knowledge and help us achieve a better understanding of speech. As Makhoul [118] and Zue [139] predict, future successes in speech recognition will rely on appropriate incorporation of our speech knowledge into some robust framework.

In fact, recent attempts have suggested that recognition systems can be improved by making intelligent use of our knowledge as well as modeling our ignorance using self-organizing algorithms. For example, the use of statistical phoneme models in continuous speech recognition explicitly applies our knowledge about the inventory of speech sounds of a language [83,118]. The concatenation of phoneme models to form word models relies on our understanding that these basic speech sound units are produced in sequence. The use of context-dependent models acknowledges the fact that the acoustic realization of a phoneme can be affected by its adjacent phonemes. Other attempts in improving self-organizing techniques by the judicious application

of speech knowledge include the FEATURE system developed at CMU to recognize isolated letters of the alphabet [23]. As a final example, the CASPAR system developed at MIT utilizes speech knowledge, as well as self-organizing pattern classification and path-finding techniques, to automatically align the speech signal with its corresponding phonetic transcription [88].

Although previous work has demonstrated the capability of using speech knowledge in self-organizing frameworks, progress in this direction has been relatively slow. One of the possible stumbling blocks has been in finding a suitable framework where specific speech knowledge can be utilized naturally and effectively. Current problems are either that the framework is very powerful but can sometimes be too rigid for incorporating more speech knowledge or that there is a significant amount of intelligent human knowledge in the system but the control strategy needs to be more powerful. For example, despite the power of hidden Markov phoneme models, incorporating segmental information in such a framework is difficult. Despite the judicious selection of acoustic attributes, the application of specific parametric statistical models to form a final decision may make invalid assumptions about their underlying probability distributions. Despite the significant amount of human knowledge incorporated, the use of a large set of production rules requires tremendous human intervention, resulting in a non-robust recognition system. In other words, a powerful, self-organizing, and flexible framework is needed so that our acoustic-phonetic knowledge can be utilized effectively.

1.1.3 Artificial Neural Networks

Recently, there has been a resurgence of interest in artificial neural networks (ANN's) [3,18,32,40,41,56,58,60,61,68,76,78,79,90,92,111,116,135]. ANN's offer an alternative self-organizing mechanism for pattern classification. Instead of examining constraints sequentially, they can solve problems by considering multiple simultaneous constraints. They can explore different alternatives in parallel without committing to a decision until all of the multiple constraints have been considered.

Besides parallel computation, ANN's have an appealing characteristic that can potentially make them well-suited for phonetic recognition: they do not make assumptions about their inputs. This is an important property for phonetic recognition since the multitude of information sources about the speech signal are results of complex interactions of many linguistic and extra-linguistic factors, ranging from inherent characteristics of the speech sounds and coarticulations to speaker characteristics and the physiological state of the speaker. Some information sources are in continuous and numerical form while others can be in discrete and symbolic form. Until we have a clear understanding of all these factors, making assumptions about the input data such as specific distribution models may sometimes be acceptable, but it may sometimes be invalid, resulting in an unreliable phonetic classifier. The fact that ANN's do not make assumptions about their inputs can potentially allow them to provide flexible frameworks for incorporating our acoustic-phonetic knowledge. On the one hand, they may allow us the freedom to make use of our speech knowledge to choose relevant information sources. On the other hand, they may provide a self-organizing mechanism to integrate the relevant sources of information. Thus while we can keep learning and studying the different characteristics of speech, ANN's may provide a framework to make use of what we have learned, and model what we have not learned. In a later chapter, we will discuss in more detail the appealing characteristics of the network that has been chosen for study, and its potential applications to phonetic recognition.

1.2 Review of Artificial Neural Networks

A great many valuable research efforts have been drawn to the area of ANN since its resurgence [3,18,32,40,41,56,58,60,61,68,76,78,79,92,90,91,111,116,132,135]. In the following sections, the basic architecture of ANN's and two examples will be reviewed briefly. We will also discuss previous attempts to apply ANN's to phonetic recognition, and point out some of the problems that remain to be answered.

ANN's consist of many inter-connected basic computational units, an architecture that is inspired by biological neural networks. The simple units and connections are reminiscent of biological neurons and synapses, respectively. Most ANN's can be specified by the topology of the network, the characteristics of the basic elements, and the algorithm used for training [90,111].

1.2.1 Topology

The arrangement of the basic computational units and their inter-connections determines the topology of a network. There are three types of units. First, the input units allow the network to receive a stimulus from its environment. They can also send or receive signals from other units in the network. Second, the output units receive signals from the rest of the network and can generate signals to the outside world. Depending on the training algorithm and the local characteristics of the basic units, the input and output signals can take on continuous or binary values. The third type of units, hidden units, are internal. They can be connected to the input and/or output units but do not interface with the environment directly. As a result, the hidden units are not mandatory but can, in general, increase the computational power of a network.

In some networks, the input and output units can be the same set of units, resulting in an *auto-associative* network. In such a network, if part of an original pattern or possibly a degraded original pattern is presented, the network can retrieve the original pattern. Thus, a degraded version of the original pattern can act as a retrieval cue. In other networks, the input and output units are disjoint, resulting in a *hetero-associative* network. Such a network can learn to associate pairs of patterns. Once the association or mapping is learned, the presentation of one member of the pair will produce the other. In general, a hetero-associative network can also be used as an auto-associative network.

Inter-connections of the basic units control the information flow in the network,

which can be uni-directional or bi-directional. The connections allow each basic unit to excite or inhibit other units. The weights on the connections determine the extent to which a unit can influence other units. As a result, each basic unit can constrain other units and can be constrained by units to which it is connected.

1.2.2 Basic Units

As shown in Figure 1.1, each basic unit receives signals from other units to which it is connected. It forms a weighted sum of its inputs, x_j, subtracts a threshold, and often passes the result through a nonlinearity, a sigmoid function, to produce its output,

$$y_i = S(z_i) = \frac{1}{1 + e^{-(z_i/T)}}, \tag{1.1}$$

where

$$z_i = (\sum_{j=1}^{N-1} w_{ij} x_j) - t_i, \tag{1.2}$$

t_i is a variable threshold for unit i, T is a constant often called the temperature, and w_{ij} is the connection weight associated with the connection from the j^{th} unit to the i^{th} unit. For simplicity, t_i can be treated as a connection weight tied to a truth unit, x_N, whose output is always 1. Thus,

$$z_i = \sum_{j=1}^{N} w_{ij} x_j \tag{1.3}$$

where $w_{iN} = -t_i$.

Figure 1.2a shows the input-output characteristic of the sigmoid function in Equation 1.1 when $T = 1$. Note that the temperature controls the width of the transition region of the sigmoid function. If the temperature is zero, the width of the transition function approaches zero, resulting in an abrupt non-continuous step function,

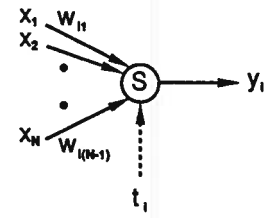


Figure 1.1: A basic computational unit forms a weighted sum of its inputs, subtracts a threshold, and passes the result through a nonlinearity, the sigmoid function.

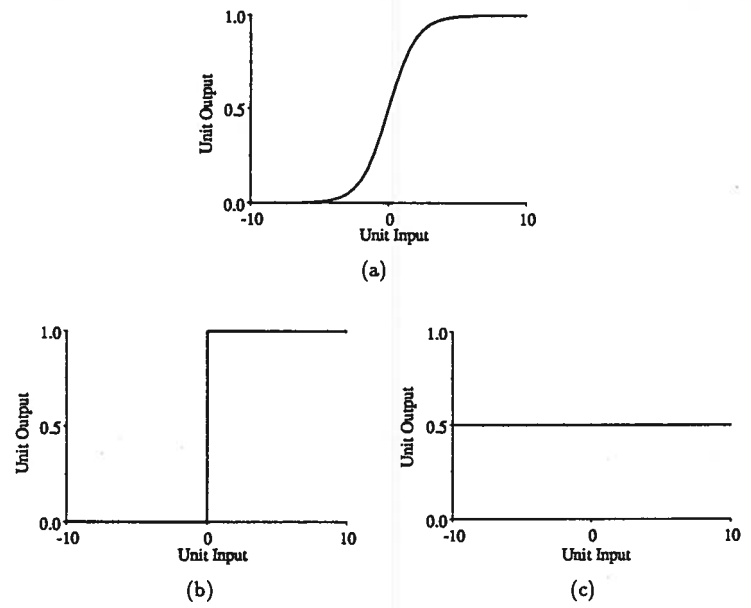


Figure 1.2: Input-output characteristic of the sigmoid function for (a) $T = 1$, (b) $T = 0$, (c) T very large.

as shown in Figure 1.2b. If the temperature becomes very large, the sigmoid function approaches a constant, as shown in Figure 1.2c.

The nonlinearity of the sigmoid function enables the network to make decisions. When the temperature is zero, a basic unit essentially becomes a linear threshold unit with binary scores. That is, the output is high if the input, z, is greater than zero, and is low otherwise. A sigmoid function with non-zero temperature allows decisions to be made with varying levels of confidence. In addition, the flat regions of the function are relatively insensitive to changes in the input and can therefore provide noise suppression and saturation characteristics to the network. The transition region is relatively sensitive and could be approximated by a straight line.

1.2.3 Training Algorithm

The weight pattern associated with the connections determines the processing or knowledge structure in the network. In order to train a network to perform classification, these weight patterns and thresholds must be adapted by an effective algorithm. Many current training algorithms for ANN's use the gradient descent technique. Given a criterion function, the connection weights are modified incrementally to minimize the function. For example, the training algorithm for a multi-layer perceptron often minimizes the mean squared error between the desired and actual outputs of the network [111].

The training algorithm can be supervised or unsupervised. Supervised learning requires a teacher to inform the network of the correct answers during training. As a result, the teacher has the flexibility of determining different ways to guide the self-organizing mechanism of the network. Unsupervised learning has the advantage that it does not require a teacher or human intervention. However, incorporating specific knowledge into the network also becomes more difficult.

1.2.4 Artificial Neural Network Examples

In this section, we will discuss two different examples of ANN's, the Hopfield network and the multi-layer perceptron. The Hopfield network is an auto-associative network that can perform unsupervised learning, whereas the multi-layer perceptron is a hetero-associative network that learns with supervision. We will describe their topology, the characteristics of their basic elements, and their training algorithms. Discussions of their applications to phonetic recognition will be given in the next section.

1.2.4.1 Hopfield Network

The Hopfield network is an auto-associator with no hidden units as shown in Figure 1.3 [60,61]. Each unit may be connected to any other units. The unsupervised training method uses a global energy measure to represent the amount of total constraint violation among the units. In many tasks, the individual constraints or information may be ambiguous, weak or imperfectly specified. However, all the constraints put together can play a decisive role in determining the outcome of the processing. Thus, minimizing the global energy measure in the network corresponds to simultaneously minimizing the amount of violated constraint. The minimization procedure converges to the nearest local minimum in the energy landscape. Each local minimum offers the least amount of constraint violation relative to its immediate neighborhood.

The energy measure, E, is inspired from statistical mechanics where it is used to measure the energy stored in some atoms:

$$E = -0.5 \sum_i \sum_j w_{ij} y_i y_j. \tag{1.4}$$

where w_{ij} and y_i are defined in Equations 1.1 and 1.3. If the sigmoid function in Equation 1.1 is shifted so that

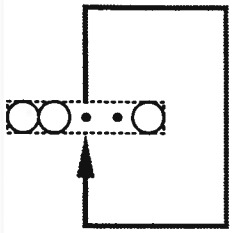


Figure 1.3: The topology of the Hopfield network.

$$y_i = \frac{2}{1 + e^{-(z_i/T)}} - 1, \tag{1.5}$$

and $T = 0$, then the output of each unit changes according to the following rule:

$$y_i = \begin{cases} 1 & z_i > 0 \\ -1 & z_i \leq 0, \end{cases} \tag{1.6}$$

where z_i is defined in Equation 1.3. It can be proved that if the weight matrix, W, is symmetric with zero diagonal elements,

$$w_{ij} = w_{ji}; \quad w_{ii} = 0, \tag{1.7}$$

and if the units change states according to Equation 1.6, then the energy measure defined in Equation 1.4 decreases monotonically.

Thus when the Hopfield network is used to perform computation, one needs first to create an energy function. The function should be created in such a way that the smaller the energy measure is, the better the unit outputs represent the answer. By constraining and competing with each other, the units change state until a local minimum is reached.

For example, when the Hopfield network is used to store m N-dimensional speech patterns, the connection weights, w_{ij}, can be computed as:

$$w_{ij} = \sum_{k=1}^{m} p_i^k p_j^k, \tag{1.8}$$

where p_i^k stands for the i^{th} element of the k^{th} pattern. From Equations 1.4 and 1.8,

$$\begin{aligned} E &= -0.5 \sum_i \sum_j (\sum_k^m p_i^k p_j^k) y_i y_j \\ &= -0.5 \sum_k (\sum_i p_i^k y_i)(\sum_j p_j^k y_j) \\ &= -0.5 \sum_k (P^k \cdot Y)^2. \end{aligned} \tag{1.9}$$

where $P^k = (p_1^k, p_2^k, ..., p_N^k)^t$ and $Y = (y_1, y_2, ..., y_N)^t$, and $\cdot$ denotes the inner product. Thus if the input speech pattern, Y, is a random vector, each parenthesized term is very small. But if Y is one of the stored patterns, P^k, then one term in the sum dominates and is equal to N^2. As a result, the energy landscape has minima of depth approximately $-0.5N^2$ at each of the stored patterns. If a stored pattern is distorted and presented to the input units, then it is possible that the constraints embedded in the connection matrix can retrieve the original stored pattern. As in any gradient descent methods, the energy minimization procedure may get stuck at a undesirable local minimum which has no significant physical meaning. Furthermore, when the stored patterns are not orthogonal, crosstalk between the stored patterns may shift the desired local minima from their intended locations. The precise relationship between the local minima and the energy function is still not well understood. In Appendix A, we will discuss in more detail the crosstalk between the stored patterns, and present a procedure that can eliminate some of the undesirable shifted and spurious local minima in the energy landscape [86].

1.2.4.2 Multi-Layer Perceptron

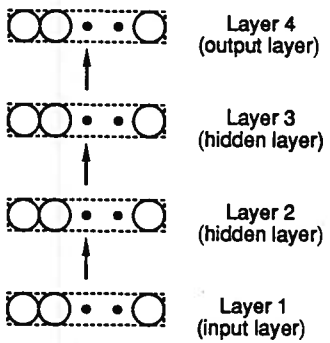


Figure 1.4: The topology of the multi-layer perceptron. It can have variable number of layers. Each layer can have different number of units.

The architecture of the multi-layer perceptrons (MLP's) is quite simple [116]. They are feed-forward networks with one or more hidden layers between the input and output layers. Figure 1.4 shows an example with two hidden layers. Signals applied to the input units are processed to produce intermediate signals, which are then allowed to flow "forward" through the hidden layers toward the output layer. Depending on the representation of the output units, MLP can be used as an auto-associative or hetero-associative network. The basic units are often characterized by Equation 1.1. If the training algorithm requires a differentiable sigmoid function, the temperature, T, is often not allowed to be zero, resulting in a graded response. The training algorithm is supervised. The mean squared error, E, between the actual output values and the desired output target values is often used to measure the performance of the network:

$$E = 0.5 \sum_j (t_j - y_j)^2 \tag{1.10}$$

where t_j is the target value for unit j, and $j \in O$, the set of output units. By minimizing E and implementing gradient descent, the amount of update for each connection weight can be obtained:

$$\Delta w_{ji} = \eta \delta_j y_i, \tag{1.11}$$

where the error signal for the jth unit, δ_j, is defined as

$$\delta_j = \begin{cases} \frac{dy_j}{dz_j}(t_j - y_j) & j \in O \\ \frac{dy_j}{dz_j} \sum_k \delta_k w_{kj} & j \in H, \end{cases} \tag{1.12}$$

H is the set of hidden units, and η is a small constant. From Equation 1.1, if $T = 1$,

$$\frac{dy_j}{dz_j} = y_j(1 - y_j). \tag{1.13}$$

Then, Equation 1.12 becomes

$$\delta_j = \begin{cases} (t_j - y_j) y_j (1 - y_j) & j \in O \\ y_j(1 - y_j) \sum_k \delta_k w_{kj} & j \in H. \end{cases} \tag{1.14}$$

In order to increase the learning speed without oscillations, a momentum term is often added to Equation 1.11 so that

$$\Delta w_{ji}(n+1) = \eta \delta_j y_i + \alpha \Delta w_{ji}(n). \tag{1.15}$$

where α is a small constant.

Interesting insights can be obtained by assuming that the temperature in Equation 1.1 is zero so that the input-output characteristic is as shown in Figure 1.2b. In this case, what a unit does is simply to make half-plane decisions, where the plane is defined by its connection vector, v, as shown in Figure 1.5a. The output is high when the input vector is on one side of the hyperplane and is low when it is on the other side. If the input vector is binary, then the hyperplanes can form boolean operations such as AND/OR, as shown in Figure 1.5b.

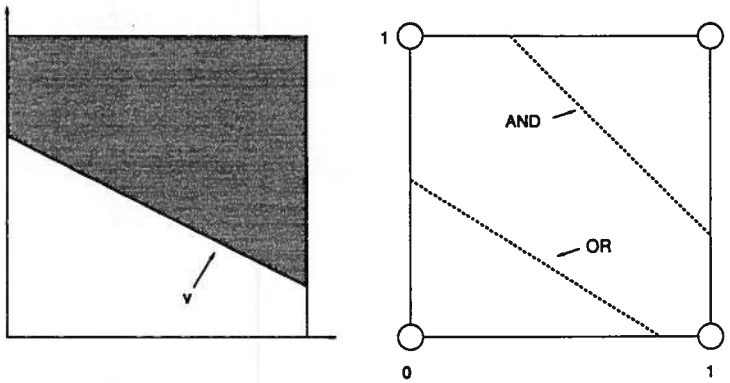


Figure 1.5: (a) Half-plane decision of a unit in two dimensions when the input is continuous. The unit produces a high value if the input vector is on one side of its connection vector, v, and a low value if on the other side. (b) Half-plane decisions corresponding to boolean operations such as AND/OR if the input vector is binary.

Units in Layer 1 are used to store the input signal, which can be continuous or binary. Thus no nonlinear sigmoid function is needed, resulting in their outputs being the same as the original input vector. Each unit in Layer 2 then decides to which

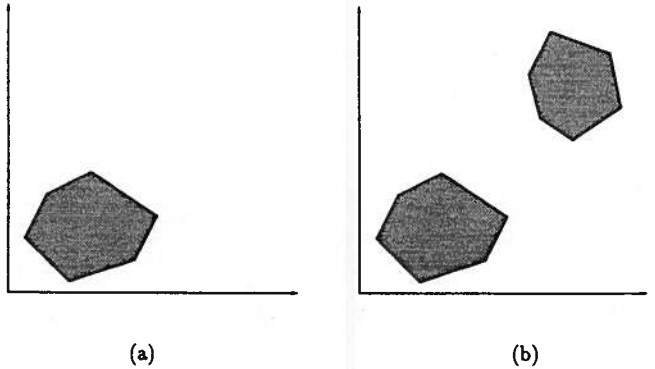


Figure 1.6: Possible decision regions formed by the multi-layer perceptron with (a) 1 hidden layer and (b) 2 hidden layers.

side of its hyperplane the input vector should belong and passes its binary decision to higher layers.

Each unit in Layer 3 then performs boolean operations on half-plane decisions made by Layer 2. For example, convex decisions regions can be formed by performing the AND operation as shown in Figure 1.6a. Furthermore, if a unit in Layer 4 accepts the convex regions from Layer 3 and performs the OR operation, disjoint decision regions[1] can also be formed as shown in Figure 1.6b.

The training algorithm tries to find a set of connection weights in the network to minimize the error criterion, E. This amounts to finding appropriate decision regions defined by the hyperplanes. If a graded sigmoid function is used, decision regions become continuous. Instead of producing high output value of 1 on one side of a hyperplane and low output value of 0 on the other side, the response drops slowly from 1 to 0. Since the error signal, δ_j, in Equation 1.12 is proportional to a quadratic

[1]Cybenko, Huang and Lippmann have found that MLP's with 1 hidden layer can also form disjoint decision regions [29,62].

Task	Network	Input	Training/Test Tokens	Speakers	Percent Correct	Author
3 vowels	BM/MLP	FFT	72/72	28	96/93	Bengio *et al.* [5]
alphabet	MLP	FFT	104/104	1	85	Burr [13]
9 syllables	MLP	FFT	250/250	1	90	Elman *et al.* [37]
8 vowels	Hopfield	FFT	8	1	-	Gold [53]
10 vowels	MLP	formants	338/333	67	80	Huang *et al.* [62]
3 stops	MLP	FFT	2620/2620	1	98	Waibel *et al.* [133]

Table 1.1: Summary of recent phonetic classification work using ANN's. "-" means the information is not available in the references.

function, $\frac{dy_j}{dz_j}$, connection weights are changed the most when the output values are close to 0.5. In other words, the training algorithm pays more attention to regions near the decision surfaces and less attention to regions far away. This is a desirable property for pattern classification, since errors are mostly committed near the decision surfaces.

1.3 Previous Applications to Speech Recognition

Previous applications of ANN's to different speech recognition tasks can be readily found in the literature [5,12,13,37,38,53,54,62,63,64,69,77,78,90,91,94,108,112,114, 133]. Most of the work has been in classification: given a specific time region and knowing the possible classes, the network determines to which one of the possible classes the input signal should belong. Table 1.1 summarizes some of the classification work over the past few years. The list is not intended to be exhaustive but to illustrate some of the problems that have been examined. The lack of thorough coverage is also indicative of the diversity and rapid growth in this area. Following is a list of some observations.

Networks: Different networks have been studied. For example, the multi-layer perceptron (MLP) and its variants, the Hopfield network (HN), and the Boltzmann's machine (BM). All these networks have well-defined training algorithms but are different in the way the connection weights are incrementally updated. Both MLP and BM adopt supervised learning while HN adopts unsupervised learning.

Speaker-dependence: Most of the work concentrates on speaker-dependent recognition of phonemes or words, with the exception of that by Bengio, who uses speech data recorded from 28 speakers, and that by Huang and Lippmann, who use speech data recorded from 67 speakers.

Tasks: Tasks of different complexities have been studied. For example, Bengio studies the classification of three places of articulation of the vowels, Waibel studies the classification of /b, d, g/, and Burr studies the classification of the English alphabet.

Data: Different amounts of data have been used for study. For example, Gold uses eight tokens to study the basic characteristics of the Hopfield network, Waibel uses 5,000 tokens to classify three stop consonants. As another example, Elman added small random noise to the original tokens to increase the size of the training set.

Input Representations: Most of the input representations are FFT or mel-scale coefficients [30], with the exception of Huang, who uses hand-labeled formant data provided by Peterson and Barney [106].

Network Characteristics: Some training characteristics of the networks have been studied. For example, Burr has found that there is a critical number of hidden units that would yield the best performance. Increasing or decreasing the number of hidden units could only decrease the performance. It is also found that the learning time improves if the average value of the inputs is subtracted from the inputs, resulting in the operating point of the nonlinear sigmoid function to be near the transition region. As another example, Huang and Lippmann have shown experimentally that a MLP with only one hidden layer can form disjoint decision regions.

Acoustic-Phonetic Features: Close examination of the connections of networks

suggests that the network can discover some acoustic-phonetic features. For example, Waibel has reported that the network can learn to detect formant transitions of the vowels as well as locate segment boundaries of the voiced stops. Specifically, one hidden unit can signify the presence of the vowel onset while another can signify transition of the second formant.

Comparisons with Traditional Techniques: Comparisons with k-nearest neighbor (KNN) classifiers have shown different results. Some experiments have shown that the MLP yields higher performance [62] while others have shown the opposite [94].

Comparisons with hidden Markov models (HMM): Specific comparisons have shown that MLP and BM can yield slightly higher performance than HMM. For example, in Bengio's study, "spectral lines" are used as inputs to BM, MLP, and HMM. It is found that the performance of the two neural networks is slightly higher than that of the HMM. As another example, Waibel develops a Time Delay Neural Network (TDNN) to achieve time-invariance by constraining time delayed connection weights to be the same. The mel-scale spectral coefficients are directly used as inputs to the network. When compared with HMM which receives its inputs from a vector quantizer, it is found that the performance of the network is slightly higher than that of the HMM.

Local Minima: The study of the Hopfield network by Gold indicates that the auto-associative network cannot always recall the stored patterns, due to local minima in the overall energy landscape. Later it was also found that the Hamming distance is able to yield better performance [92].

Input Format: Inputs to the networks are either binary (in HN and BM) or continuous (in MLP).

1.3.1 Discussion

Previous work has provided important insights and has induced great interest in applying ANN's to phonetic recognition. However, more careful study and understanding of the networks are needed before we can more fully exploit ANN's for phonetic recognition. For example, we need to understand more about the inherent characteristics, capabilities and limitations of the networks. When applied to phonetic recognition, such understanding will enable us to utilize the networks more effectively. Second, performance of a network can be affected by different factors such as across-speaker differences, contextual effects, and speaking styles. In order to study the extents of these effects, speech data under different conditions need to be collected. Third, to investigate how well a network can generalize and how much data are needed before robust classification can be achieved, a large amount of data needs to be collected. Fourth, a multitude of acoustic-phonetic cues can be extracted from the speech signal. In order to exploit more extensively the flexible framework of ANN's, different input representations or sources of information need to be presented to the network. Fifth, in order to gain some understanding about how a network performs classification, the internal representation of a network needs to be examined. Finally, in order to understand how well ANN's can perform classification relative to traditional classification techniques, systematic comparisons need to be made.

The rest of this thesis reports an investigation into the use of ANN for phonetic recognition. Based on what has been learned from the previous studies, the work described in this thesis is an attempt to understand more about some of the basic problems that have not been addressed fully.

1.4 Thesis Overview

This work has three primary objectives. First, by investigating and gaining a better understanding of the basic nature and properties of ANN's, we may be able to exploit

them more fully as pattern classifiers. Second, by properly applying our acoustic-phonetic knowledge, we can potentially enhance the flexible framework of ANN's for phonetic recognition. Third, by comparing the networks with traditional pattern classification techniques, we can better understand the merits and shortcomings of the different approaches.

In the next chapter, the particular task chosen for study in this thesis will be described. As an initial step, the study is limited to the task of recognizing the 16 vowels in American English independent of speaker. Databases and signal representations will also be described.

In Chapter 3, the selection and structure of the network will be described. Specifically, the MLP is used, supplied with heterogeneous sources of information. The fact that the MLP does not assume any specific probability distributions or distance metrics may make it well-suited for integrating heterogeneous sources of information in the speech signal. Performance results will be given and errors will be analyzed.

Chapter 4 examines different characteristics and representations of the network. Specifically, it discusses the performance of the network as a function of the number of training iterations, amount of training data, number of hidden units, number of hidden layers, and use of the nonlinear sigmoid function. It also discusses issues about representations such as structure of the internal representation, alternative choices for output representations, and the use of heterogeneous input representations.

Chapter 5 discusses potential problems with using traditional classification techniques for phonetic recognition. It also compares the performance and complexity of the network with those of two traditional pattern classification techniques: the k-nearest neighbor and Gaussian classifiers.

In Chapter 6, some further refinements are described, including determination of a more appropriate error metric for pattern classification, initialization of the network, and rapid adaptation of the network to a new speaker.

The final chapter summarizes the results of this work, and discusses future directions in utilizing ANN's for phonetic recognition.

Chapter 2

Task, Databases, and Signal Representation

2.1 Task Description

As an initial step toward understanding the basic characteristics of artificial neural networks (ANN's) and their potential applications to phonetic recognition, the work described in this thesis is constrained to the task of recognizing the vowels in American English. There are several major reasons for selecting this task. First, restricting to only vowels makes the task more manageable. Thus we can better focus on the basic issues of utilizing the network for phonetic classification and leave out other less related but important issues such as proper input representation for different sounds. Second, as we will see, the restricted task is still interesting and non-trivial. The acoustic realizations of the vowels can be drastically affected by contextual variations. Finally, recognition of the vowels has been studied relatively extensively in the past few decades. Table 2.1 shows the IPA symbols for the 16 vowels that are used for this task, including monophthongs and diphthongs, and their corresponding examples.

The work described in this thesis is constrained to classification. Given a time region, the network determines which one of the 16 vowels was spoken. In all the experiments, the time regions are obtained from the time-aligned phonetic transcriptions [88,145]. However, in a practical speech recognition system, these time regions

Vowel	Example	Vowel	Example
i	beet	ɔ	bought
ɪ	bit	u	boot
e	bait	ʊ	book
ɛ	bet	ũ	Tuesday
æ	bat	ɝ	bird
ɑ	body	ɑʸ	bite
o	boat	ɔʸ	boy
ʌ	but	ɑʷ	about

Table 2.1: Sixteen vowels in American English, with their corresponding examples.

must first be determined or hypothesized [52,88]. Furthermore, the overall recognition system is not restricted to representing the lexical items in terms of phonemes. Other phonological units such as distinctive features, diphones, or syllables can potentially be employed.

2.2 Database Description

In order to study the effects due to different variabilities in the speech signal, databases with different characteristics were used, ranging from single-speaker, with restricted phonetic context in isolated utterances, to multiple-speakers, with unrestricted phonetic context in continuous speech. There are altogether four different databases, as shown in Table 2.2. In all these databases, the speech data have been phonetically transcribed and aligned using CASPAR, a semi-automatic time-alignment system developed at MIT [87,88,145]. The transcription and alignment process involves three major steps. First, an acoustic-phonetic sequence is entered manually by a transcriber, who can listen and examine various visual displays of the speech signal. Second, the speech signal is aligned automatically with the acoustic-phonetic sequence. Third, the results obtained from the second step are checked, and corrected if necessary, by experienced acoustic phoneticians.

Database	Speakers(M/F)	Context	Training Tokens	Test Tokens	Speaking Mode	Remark
I	1(1/0)	b __ t	80	80	isolated	rotational
II	17(8/9)	b __ t	272	272	isolated	rotational
III	1(1/0)	* __ *	3,200	800	continuous	independent
IV	550(383/167)	* __ *	20,000	2,000	continuous	independent

Table 2.2: Four different databases used to study effects due to different conditions. "*" stands for any phonetic contexts.

In Database I, the vowel tokens were extracted from a /b/-vowel-/t/ environment (i.e. the phonemes before and after the vowel are restricted to /b/ and /t/, respectively) spoken in isolation by one male speaker. Each of the 16 vowels was spoken 5 times, resulting in a total of 80 vowel tokens. Due to the very limited amount of data, a rotational procedure was adopted for training and testing. In each step of the rotational procedure, four tokens of each vowel are used for training, resulting in a total of 64 training tokens. The remaining token of each vowel is then used for testing, resulting in 16 test tokens. The same procedure is repeated 5 times, each time using different sets of training and test tokens. Restricting to single speaker and a limited phonetic context suppresses many sources of variations and helps to establish some baseline results for a relatively straightforward task.

In Database II, the vowel tokens were extracted from the same isolated /b/-vowel-/t/ environment, but spoken by 8 male and 9 female speakers. Each speaker uttered the 16 vowels once, resulting in a total of 272 vowel tokens. Again, a rotational procedure was adopted for training and testing. In each of the 17 steps, 256 vowel tokens from 16 speakers are used for training. The 16 tokens from the remaining speaker are then used for testing. Using speech data from multiple speakers enables the study of across-speaker effects.

In Database III, the vowel tokens were extracted from 600 continuous sentences spoken by one male talker. The sentences were randomly chosen from the Harvard

Vowel	Tokens	Vowel	Tokens
i	554	o	262
e	400	ʌ	254
ɛ	365	ɝ	143
æ	348	ü	128
ɑ	348	ɑʷ	117
ɪ	324	ʊ	73
ɔ	321	u	57
ɑʸ	308	ɔʸ	27

Table 2.3: Distribution of the 16 vowels in Database III.

"Glue the sheet to the dark blue background."
"Four hours of steady work faced us."
"The salt breeze came across the sea."
"The young girl gave no clear response."
"The play seems dull and quite stupid."
"Bring your problems to the wise chief."
"The third play was dull and tired the players."
"The brown house was on fire to the attic."
"He took the lead and kept it the whole distance."
"Help the weak to preserve their strength."

Table 2.4: Typical sentences for Database III.

Lists of phonetically-balanced sentences [36]. There are altogether 4,000 extracted vowel tokens with no restrictions on the phonetic contexts, i.e. the phoneme before and after the vowel can be any phonemes in the database. The 3,200 training tokens are extracted from 480 sentences, and the 800 test tokens are extracted from the remaining 120 sentences. Having no restrictions on the phonetic environment enables the study of contextual effects. Furthermore, this database can be used to study rapid adaptation to a new speaker. Frequency of occurrence of the 16 vowels and typical examples of the sentences may be found in Tables 2.3 and 2.4, respectively.

Finally, Database IV was constructed from the TIMIT database, which was recorded

at Texas Instruments, and phonetically transcribed and time-aligned at MIT [43,81]. The entire TIMIT database consists of 10 sentences recorded from each of 630 male and female speakers, representing a wide range of dialectical variations. Two of the 10 sentences were calibration sentences used for dialectical studies of American English [21]. These two sentences were spoken by all 630 speakers. Five of the 10 sentences were drawn from a set of 450 phonetically-compact sentences hand-designed at MIT with emphasis on thorough coverage of phonetic pairs [81]. The remaining three sentences were randomly chosen from the Brown corpus, and were intended to provide examples of typical American English sentences [80]. The vowel tokens in Database IV were extracted from the phonetically-compact sentences of 550 speakers, representing a total of 2,750 sentences. There are altogether 20,000 training tokens extracted from sentences spoken by 500 speakers (350 male and 150 female). The 2,000 test tokens were extracted from sentences spoken by an independent set of 50 different speakers (33 male and 17 female). For both the training and test sets, the ratio of male to female speakers is about 2 to 1. Although the 50 test speakers are randomly chosen, the distribution of the different dialects in the database is maintained. Having a large number of speakers and no restrictions on the phonetic context permits the study of effects due to different phonetic contexts and other sources of variations such as across-speaker and dialectical differences. Distribution of the 16 vowels and typical examples of the sentences may be found in Tables 2.5 and 2.6, respectively.

The work in this thesis is based mainly on Database IV. Recognition of the vowels in this database is expected to be more difficult than for the other three databases. In the next chapter, comparisons of performance results on the four databases will be given. In Chapter 6, the study of rapid speaker adaptation using Databases III and IV will be discussed.

Vowel	Tokens	Vowel	Tokens
i	3228	α^{y}	1420
ɪ	2741	ɝ	1282
ɛ	2097	o	1118
ɑ	1722	ü	992
æ	1627	α^{w}	480
e	1556	u	440
ɔ	1474	ʊ	352
ʌ	1435	$ɔ^{y}$	289

Table 2.5: Distribution of the 16 vowels in Database IV.

"Bright sunshine shimmers on the ocean."
"Are your grades higher or lower than Nancy's?"
"Aluminum silverware can often be flimsy."
"Only the most accomplished artists obtain popularity."
"The eastern coast is a place for pure pleasure and excitement."
"Last year's gas shortage caused steep price increases."
"Cooperation along with understanding alleviate dispute."
"Kindergarten children decorate their classrooms for all holidays."
"The mango and the papaya are in a bowl."
"In developing film, many toxic chemicals are used."

Table 2.6: Typical sentences for Database IV.

2.3 Signal Representation of Speech

Previous studies have shown that the evolution of speech can be influenced by the constraints of both the production and the perception mechanisms [125]. Although many questions about the human auditory system remains unanswered, developing phonetic recognizers would probably be well served by paying attention to the constraints of the auditory system and focusing on information that is perceptually important [52,66,122].

The spectral representations used in this thesis are obtained from the auditory model proposed by Seneff [122]. This model incorporates several known characteristics of the human auditory system, such as critical-band filtering, half-wave rectification, adaptation, saturation, forward masking, and spontaneous response. Specifically, outputs of two different stages of the auditory model are used: the mean rate response and the synchrony envelopes. The mean rate response corresponds to the short-term average or the mean probability of firing on the auditory nerve. It has been shown to enhance the temporal aspects of the speech signal. The synchrony envelopes measure the extent of dominance of information at the critical band filters' characteristic frequency. It has been shown to enhance the formant information in the speech signal.

2.3.1 Normalization

2.3.1.1 Speaker Normalization

There are at least two major types of across-speaker differences [72]. First, systematic acoustic variations can arise from differences in sex and in vocal-tract size and shape. Second, occasional variations can result from differences in sociolinguistic background and dialect. Although there are still many aspects of the human perceptual normalization mechanism that are as yet unexplained, scientists have been

developing speaker normalization procedures to deal with the first class of across-speaker variations [8,102,123,120,123,130,134]. Specifically, the procedure adopted in this thesis is the same as the one proposed by Seneff [123]. In this procedure, the spectrum, which can be the synchrony envelopes or the mean rate response, is shifted down on the the bark-frequency scale. The amount of shift is determined by the median pitch, F_0, computed over the entire vowel region, using the pitch detector proposed by Gold and Rabiner [55]. Seneff has found that shifting the spectrum according to F_0 can reduce the variations of the formant locations across speakers [123]. Specifically, the spectral coefficient of the normalized spectrum at bark-frequency b is:

$$S_N(b) = S(b + B_o) \tag{2.1}$$

where $S(b)$ stands for the original spectral coefficient at bark-frequency b, and B_o is the median pitch on the bark-frequency scale. Rapid speaker adaptation will be discussed in Chapter 6 to further deal with across-speaker differences.

2.3.1.2 Amplitude Normalization

After applying the speaker normalization procedure described above, each spectrum is normalized such that

$$\sum_b S_N(b) = 0 \tag{2.2}$$

There are two reasons for doing this. First, removing variations in the overall magnitudes of the spectral coefficients may enable the network to focus on relevant linguistic information in the speech signal such as the formant frequencies. Second, previous work has found that performance typically improves when the mean of the input

vectors is close to zero [13]. This results in initial outputs of the units *near* the transition regions of the sigmoid function, where learning is faster than in the saturation regions.

Chapter 3

Network Structure and Performance Evaluations

In this chapter we will discuss the motivation for studying the multi-layer perceptron (MLP) for phonetic recognition. We will describe the basic structure of the network and study the effects on the performance due to different variabilities in the speech signal, using the databases summarized in Table 2.2. We will also evaluate the performance of the network in terms of average agreement with the phonetic transcription, an information theoretic measure, and distinctive features.

3.1 Network Selection: Multi-Layer Perceptron

During the past few years, several ANN training algorithms and architectures have been suggested [3,18,32,40,41,56,58,60,61,68,76,78,79,90,92,111,116,135]. Although they all offer parallel and self-organizing mechanisms, the multi-layer perceptron (MLP) has several characteristics that make it particularly well-suited for phonetic classification. First, like some other ANN's, it can make decisions with no assumptions about the underlying probability distribution of the input data. This is an important characteristic for phonetic recognition, since the probability distributions are often not known. Previous attempts in phonetic classification often employ specific distribution models such as Gaussian. Such approaches can be successful when

the model matches the true underlying distribution, but may lead to inferior results if the model is invalid. Until the valid models are found, MLP's can potentially provide an effective mechanism to model our ignorance in the underlying distributions of the data.

Second, MLP's do not assume any distance metrics. Traditional classifiers which do not assume any probability distributions often need to use specific distance metrics such as Euclidean or Itakura distance, to measure degree of similarity. However, as Klatt pointed out, current metrics fall far short of the goals needed to achieve a robust recognition system [72]. He also suggested that speech recognition algorithms can be improved by finding appropriate perceptually-based metrics to measure phonetic similarity. Until such powerful distance metrics are found, MLP can potentially provide an automatic procedure to find some reasonable metrics through training. Instead of assuming any specific distance metric, connection weights in MLP are trained using data to form decision regions.

Third, inputs for MLP can be continuous and/or binary. As a result, inputs can be any combination of continuous acoustic attributes and/or discrete linguistic features in numerical or symbolic form. This property together with the fact that MLP does not assume any specific probability distributions or distance metrics can potentially enable MLP to integrate heterogeneous sources of information in the speech signal. As a result, researchers can have the flexibility to select different sources of information as inputs to the network.

Fourth, MLP is a discriminator that maximizes the contrasts between classes. During training, the network is taught both positive and negative examples. Not only is the network given examples of a class, but it is also given examples that do not belong to that class. For instance, let the i^{th} output unit, u_i, correspond to the i^{th} decision class, ω_i. If a training example belongs to ω_i, then the target value for u_i, t_i, is often set to a high value, indicating that u_i should respond strongly to such an example. However, if a training example belongs to ω_j, not only is t_j set to a high value, but t_i is also set to a low value, indicating to u_i that such an example

does not belong to ω_i. In addition, as mentioned in the previous chapter, the training algorithm pays most attention to errors made near the decision surfaces. As a result, the network can potentially be more effective in discriminating different classes than approaches that model individual classes independently of others.

Fifth, MLP can form disjoint decision regions in the multi-dimensional input space for the same classification category without supervision [28,29,62,90]. This characteristic makes MLP particularly suitable for the discovery of subtle regularities such as different allophones. For example, Zue and Laferriere found that the phoneme /t/ can have many different acoustic realizations, depending on the adjacent speech sounds and prosodic patterns [143]. If the realizations are sufficiently different, they may occupy disjoint regions in the input space. Thus the use of MLP can potentially bypass the need for subjective decisions in constructing different models for different allophones.

Finally, MLP can be used as a hetero-associator to associate pairs of patterns. Once the association has been learned, the presentation of an input pattern will produce the output pattern. Thus MLP can potentially learn to map the complex speech signal to different levels of phonetic and/or phonological representations. As a result, researchers can have the flexibility to design and organize the structure of the network in different ways.

3.2 Network Structure

The network structure that has been studied the most in this thesis is the MLP with one hidden layer as shown in Figure 3.1.[1] The number of output units, N_O, depends on the number of classes to be recognized. For example, when the network is used to recognize the 16 vowels, $N_O = 16$. Given a set of input and output units,

[1] As Cybenko, Huang and Lippmann have shown, MLP with one or two hidden layers can approximate any continuous functions and form complex surfaces such as non-convex and disjoint decision regions [28,29,62]. In Section 4.1.4, comparisons with networks that have different numbers of hidden layers will be presented.

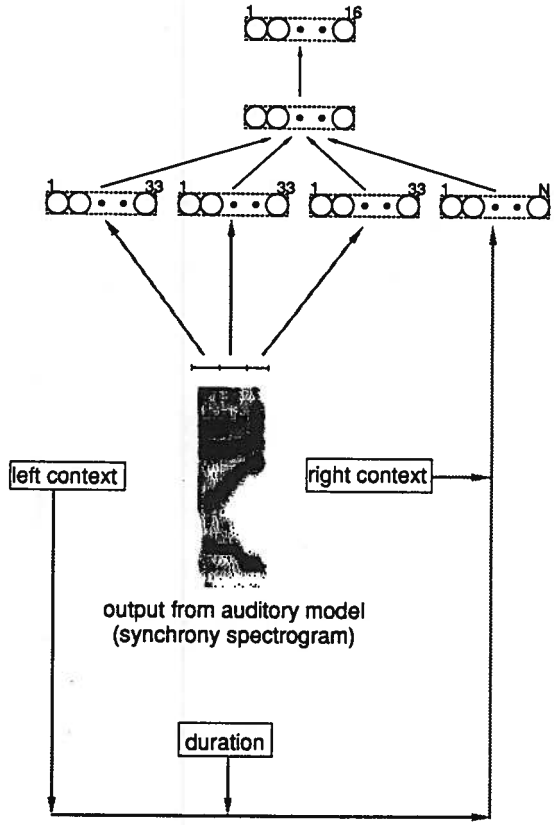


Figure 3.1: Basic structure of the network. Input to the network can be heterogeneous sources of information.

the number of hidden units, N_H, determines the size of the network. As we will see in the next chapter, characteristics and performance of the network change as N_H is varied. The input units are used to receive information from the outside world. The number of input units, N_I, depends on the amount of information available. In our network, the input can be different spectral representations, phonetic contexts, and/or durational information. While spectral and durational inputs are continuous and numerical, the contextual inputs are discrete and symbolic. Recognition accuracy is expected to improve as more independent sources of information are made available to the network.

In order to capture dynamic spectral information and to reduce the amount of computation, the vowel region is first divided into three regions of equal duration. Three averaged spectra are then computed from the left, middle, and right regions of each vowel token.[2] After applying the speaker normalization procedure described in the previous chapter, 33 spectral coefficients on the Bark scale are obtained for each spectrum, thus retaining information approximately from 0 to 3,500 Hz. These three spectra are then normalized in amplitude, and applied to the first three sets of input units. As mentioned earlier, the spectral representations can be the mean rate response and/or the synchrony envelopes obtained from an auditory model [122].

In all the experiments described in this chapter and in most experiments in this thesis, only the synchrony envelopes are made available to the network.[3] After random initialization, the connection weights in the network are updated for each training token using the procedures outlined in Chapter 1. The gain constant, η, and the momentum constant, α, in Equation 1.15 are both chosen to be 0.05.

[2] A more detailed discussion on the acoustic representation will be presented in Section 7.2.1.
[3] The use of other sources of information will be discussed in Section 4.4.

3.3 Performance Evaluations

Besides the amount of information available, there are at least four major factors that can affect the performance of a phonetic recognizer. First, is the system supposed to recognize speech independent of speaker? Since every speaker has his/her own speaking characteristics, a speaker-independent speech recognizer has to deal with the across-speaker variability and capture the relevant linguistic information from the speech signal. As a result, recognizing speech independent of speakers can be significantly more difficult than recognizing speech from one speaker. Second, what are the phonetic contexts? The acoustic realization of a phonetic unit can sometimes be significantly affected by its local context, depending on the identities of its adjacent phonetic units. For example, the acoustic realization of a vowel can be changed quite significantly if the preceding or the following phoneme is a liquid or glide (/l/, /w/, /r/, or /y/). As a result, recognizing the vowels in some contexts can be more difficult than in other contexts. Third, is the speech spoken continuously? In continuous speech, a phonological unit is subject to a greater number of variations including stress, speaking rate, and across-word boundary effects. As a result, recognizing phonetic units in continuous speech can be more difficult than in isolated words. Fourth, how much training data is available? In general, the performance increases as more training data is available. But how much data is needed before robust performance can be achieved?

The influence of these factors on the performance of the network is examined, using the different databases described in Table 2.2. In these experiments, only the synchrony envelopes are used. There are 99 spectral coefficients, since each of the three averaged spectra has 33 coefficients. Including the truth unit, there are altogether 100 input units. The number of hidden units is 32. Since consecutive layers are fully connected, there are 3,712 connections in the network.

In order to measure the performance of the network, statistics of how often the network agrees with the transcriptions are obtained. However, some of the transcription labels may be subjective and biased towards the underlying phonemic form of the orthography. The transcribers could also use contextual information during the labeling process. Nevertheless, the agreement with the transcription gives some indication of how well the network can classify the vowels.

Figure 3.2 shows the performance for the different tasks. Since vowels in Database I are extracted from a restricted phonetic environment spoken by only one speaker, recognizing the vowels is relatively straightforward and perfect agreement between the transcription and the network labels is achieved. For vowel tokens from Database II, the average agreement with the transcription decreases to 89%, presumably due to across-speaker variability or lack of adequate speaker normalization procedure. In Database III, the vowel tokens are extracted from continuous sentences spoken by one male speaker. Since the acoustic realizations of the vowels can be affected quite significantly due to contextual variations, the average agreement decreases to 74%. Finally, vowel tokens from Database IV, spoken by multiple speakers and extracted from unrestricted contexts, are used. Due to across-speaker and contextual variations, the average agreement decreases to 60%.

These results collectively indicate that a substantial difference in performance can be expected under different conditions, depending on whether the task is speaker-independent, what is the restriction on the phonetic contexts, and whether the speech material is spoken continuously. For a restricted task, relatively high performance can be achieved using relatively few training tokens. However, even with significantly more training data, the performance can decrease substantially as the task becomes more difficult. For example, only 64 training tokens are needed to achieve perfect performance when the network is tested on Database I. For Database IV, the performance decreases by 40%, although 20,000 training tokens are used.[4]

As mentioned before, the performance is only a measure of how well the network agrees with the transcription. For comparison, one may ask how well human listeners

[4] A more detailed study of the performance of the network as a function of the amount of training data will be discussed in Section 4.1.2.

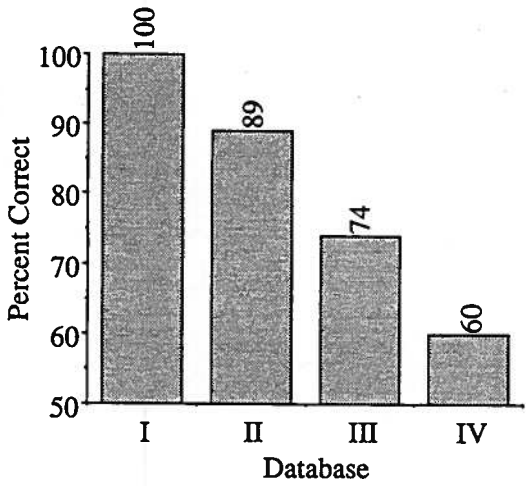


Figure 3.2: Performance results for the four databases described in Table 2.2, using only the synchrony envelopes.

can agree with the transcription, or how well human listeners can agree with each other. Perceptual experiments have been performed with human subjects who can listen to phoneme triplets, i.e. the phoneme before the vowel, the vowel itself, and the phoneme after the vowel [107]. There are no restrictions on the phonemes before and after the vowels. The sequences are extracted from the continuous sentences spoken by multiple speakers in the TIMIT database. Results indicate that the average agreement among three listeners and the transcription on the identities of the vowels is between 60% and 65%. Furthermore, the average agreement among the three listeners is between 65% and 70%.

These results indicate that when only the synchrony envelopes in the vowel regions are available to the network, its performance is comparable to that of the human listeners who can listen to the phoneme triplets. As we will see in Section 4.4, performance of the network can be improved if more sources of information about the speech signal are available. However, it should be noted that although the three listeners are experienced phoneticians, their performance can potentially be improved if they are well-trained for this particular task.

3.4 Error Analyses

3.4.1 Confusions

In order to gain some insight about the kinds of disagreements on the vowel labels provided by the network and the transcription, confusion statistics between the two sets of labels are obtained. Tables 3.1 to 3.3 show the confusion matrices of the 16 vowels for Databases II-IV. The rows correspond to the transcription labels, whereas the columns correspond to the labels produced by the network. An entry in the i^{th} row and j^{th} column stands for the percentage of the tokens transcribed as the i^{th} vowel and classified by the network as the j^{th} vowel. For example, in Table 3.1, 94% of the vowel tokens transcribed as /æ/ are classified by the network correctly, while

	ü	i	ɪ	e	ɛ	æ	ɔʸ	ɑʸ	ɑʷ	ɑ	ɔ	ʌ	o	ɝ	u	ʊ	Total
ü	94		6														17
i		100															17
ɪ	6		82		12												17
e			6	88				6									17
ɛ			12		71	12								6			17
æ					6	94											17
ɔʸ							94									6	17
ɑʸ							6	94									17
ɑʷ									94				6				17
ɑ										76	18	6					17
ɔ										12	82					6	17
ʌ										6		88	6				17
o													100				17
ɝ												6		88	6		17
u		6											6		82	6	17
ʊ												6				94	17

Table 3.1: Percent confusion table for Database II when only the synchrony envelopes are available.

6% are classified as /ɛ/. An entry in the i^{th} row and the last column corresponds to the total number of test tokens transcribed as the i^{th} vowel. For example, there are 17 tokens for each of the vowels in Database II.

From these tables, we can see that most of the disagreements, or errors, are quite reasonable. For example, Table 3.1 shows that the vowel /æ/ is the most confusable with /ɛ/ and that the vowel /ɑ/ is the most confusable with /ɔ/. Examination of Table 3.2 suggests that some of the disagreements may be due to contextual variations. For example, the accuracy for the vowel /i/ decreases from 100% for Database II to 88% for Database III. As another example, the diphthong /ɑʸ/ is more often classified as /ɑ/, possibly due to coarticulation with the following semivowels such as /l/ or /w/. Table 3.3 indicates that the vowel tokens in Database IV are more confusable than those in Databases II and III, possibly due to compounded effects of contextual variations and across-speaker differences. However, a front vowel is still mostly confused with other front vowels, while a back vowel is mostly confused with other back vowels. Furthermore, we can also see that the performance tends to be

	ü	i	ɪ	e	ɛ	æ	ɔʸ	ɑʸ	ɑʷ	ɑ	ɔ	ʌ	o	ɝ	u	ʊ	Total
ü	69	8	12		4										4	4	26
i	3	88	3	3		1										3	118
ɪ	6	9	62	6	9	1							1	1		4	68
e		1	9	78	5	3		2								1	93
ɛ			11	5	71	7		3				4					75
æ			2	3	7	83		2		2						2	60
ɔʸ				14	14		57	14									7
ɑʸ		1				4		90		3	1	1					78
ɑʷ	3					17			66	10		3					29
ɑ				2	2	3		3	3	63	8	12	2		2		60
ɔ			2		2	2				20	69	3	2		2		61
ʌ	2				11	4			2	9	7	57	6		2		54
o	2		4		2				2		8	4	73		2	4	52
ɝ			3		3									88	3	3	33
u	17		8							8					67		12
ʊ	8		46		8								8		8	23	13

Table 3.2: Percent confusion table for Database III when only the synchrony envelopes are available.

higher on vowels that are more frequently represented in the database.

3.4.2 Entropy

When the average agreement is adopted to measure the performance of the network, only statistics along the diagonal of the confusion matrix are used. Thus a measure that can account for the statistics in the entire confusion matrix can potentially be more informative. In this section, the performance of the network is quantified using an information theoretic measure [48]. Such a measure utilizes the entire confusion matrix and quantifies how much information the classifier can provide, or how much uncertainty it can remove.

Specifically, let X stand for the random variable of the transcription labels, and Y stand for the random variable of the vowel labels produced by the network. The average mutual information between X and Y is the difference between the entropy of X and the conditional entropy of X given Y:

	ü	i	ɪ	e	ɛ	æ	ɔʸ	ɑʸ	ɑʷ	ɑ	ɔ	ʌ	o	ɝ	u	ʊ	Total
ü	52	17	16		1								1	5	6	1	77
i	3	90	2	4													267
ɪ	5	14	56	7	6	3				1		2	2	3		1	216
e		11	4	73	4	2	1	1			1	1	1				134
ɛ	1	2	13	10	30	20		1	1	4		8	4	6			158
æ		1	6	1	15	63		2	1	5	2	3	1				136
ɔʸ							53	21		16		5		5			19
ɑʸ			1	6	3	3	1	60		18	1	7		1			137
ɑʷ					4	6		2	38	23	4	15	8				52
ɑ						2		7	1	70	8	7	1	4			165
ɔ			1				3	2	1	23	61	2	4	1	1		139
ʌ			6	1	12	6		6	1	12	1	48	6	2		2	126
o			4	1	1		1	1	1	3	13	15	56	3	1		100
ɝ		1	6			1		1		1	1	2		85			82
u	12	3	12								12	9		9	33	9	33
ʊ	3		41		3						3	13	9	6	9	13	32

Table 3.3: Percent confusion table for Database IV when only the synchrony envelopes are available.

$$I(X;Y) = H(X) - H(X \mid Y) \tag{3.1}$$

where $I(X;Y)$ is a measure of the average amount of uncertainty in X resolved by the observation of the output labels of the network, $H(X)$ is the entropy or average uncertainty of X,

$$H(X) = -\sum_{x} P_X(x) log P_X(x), \tag{3.2}$$

$H(X \mid Y)$ is the average remaining uncertainty in the transcription label of a vowel token after observing the output of the network,

$$H(X \mid Y) = -\sum_{xy} P_{XY}(x,y) log P_{XY}(x \mid y), \tag{3.3}$$

$P_X(x)$ is the probability distribution of X, $P_{XY}(x \mid y)$ is the probability distribution of X given Y, and $P_{XY}(xy)$ is the joint probability of X and Y. All these probabilities are readily available from the confusion matrices in Tables 3.1 to 3.3.

Such an information theoretic measure provides another way to quantify the performance of a classifier. It becomes particularly important when the distribution of the input data is highly skewed. As an extreme example, if it is known that all the test tokens have the same label, then a perfect classification rate can be trivially achieved by simply using only *a priori* statistics. Thus measuring the percent correct of a classifier does not always give a good indication of how useful the classifier is. On the other hand, when the labels are all the same, $H(X) = 0$ and $H(X \mid Y) = 0$, indicating that there is no uncertainty in X before and after classification. In other words, the entropy measure indicates that no information is provided by the classifier.

Figure 3.3 shows the entropy or the uncertainty of the vowels in Databases I-IV before and after the observation of the labels produced by the network. It can be seen that the initial entropy, $H(X)$, for Databases I and II is 4.0, since the vowels are uniformly distributed. Furthermore, we can see that the network can remove all the initial uncertainty in the vowel labels of Database I, since the conditional entropy, $H(X \mid Y)$, is equal to 0. For Database IV, the initial entropy is 3.8 and the network can remove 50% of the initial uncertainty, resulting in $H(X \mid Y) = 1.9$.

As discussed earlier, Tables 3.1 to 3.3 seem to indicate that a front vowel tends to be more confusable with other front vowels, while a back vowel tends to be more confusable with other back vowels. In order to analyze these tendencies more objectively, a binary clustering procedure was adopted, using the entropy measure. With the 16 vowels, there are a total of $\sum_{j=1}^{8} \binom{16}{j}$ possible ways to categorize the vowels into two classes. Associated with each possible categorization, a corresponding confusion matrix with 2 classes can be obtained from, say, Table 3.3. Out of these $\sum_{j=1}^{8} \binom{16}{j}$ possible categorizations, the one that yields the highest mutual information is retained. The retained classes are then further divided and the same process repeats

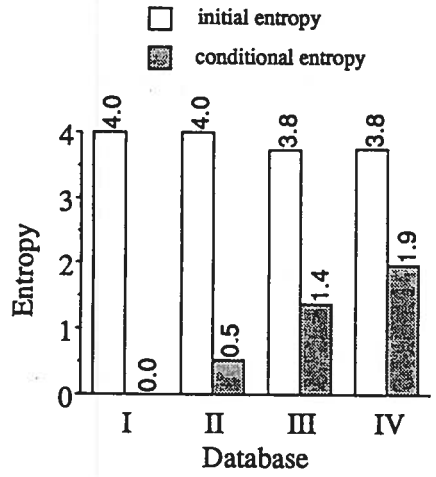


Figure 3.3: Initial and conditional entropy for Databases I-IV.

until there is only one vowel label in each class. By selecting the categorization with the highest mutual information, vowels that are confusable with each other will tend to fall into the same class. Specifically, let

$$\left\{C^l_{(i+1)(2k-1)}, C^l_{(i+1)(2k)}\right\} = P_l\left[\hat{C}_{i,k}\right] \quad for \quad \begin{cases} 1 \le l \le \sum_{j=1}^{\frac{|\hat{C}_{i,k}|}{2}} \binom{|\hat{C}_{i,k}|}{j} \\ 1 \le i \le I \\ 1 \le k \le 2^{(i-1)} \end{cases} \tag{3.4}$$

and

$$\left\{\hat{C}^l_{(i+1)(2k-1)}, \hat{C}^l_{(i+1)(2k)}\right\} = \max_{I_l(X;Y)} \left\{C^l_{(i+1)(2k-1)}, C^l_{(i+1)(2k)}\right\}, \tag{3.5}$$

where $P_l\left[\hat{C}_{i,k}\right]$ stands for the l^{th} possible way to categorize the vowels in $\hat{C}_{i,k}$ into 2 classes, $\left\{C^l_{(i+1)(2k-1)}, C^l_{(i+1)(2k)}\right\}$, $C^l_{i,k}$ stands for the l^{th} possible class in the k^{th} entry of the i^{th} level of the binary tree, $I_l(X;Y)$ stands for the mutual information for $\left\{C^l_{(i+1)(2k-1)}, C^l_{(i+1)(2k)}\right\}$, $|\hat{C}_{i,k}|$ stands for the number of vowel labels in $\hat{C}_{i,k}$, I stands for the depth of the resulting binary tree, and $\hat{C}_{1,1}$ stands for the initial class with all the 16 vowels.

The binary tree obtained using the above iterative procedure on Database IV is shown in Figure 3.4. It can be seen that the 16 vowels are first split into two classes. One class corresponds mostly to the front or high vowels (the vowels with high second formant frequencies or low first formant frequency), while the other corresponds mostly to the back vowels and diphthongs. The terminal nodes also give some indication of the confusions between individual vowels. For example, /ɑ/ is the most confusable with /ɔ/, /æ/ with /ɛ/, and the two diphthongs, /ɑʸ/ with /ɔʸ/. These confusions seem to be quite reasonable, since the acoustic realizations of these pairs of vowels are quite similar. As another example, the two allophones, /u/ and /ü/ are the most confusable with each other.

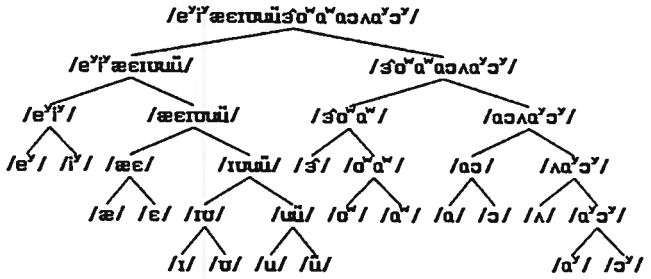


Figure 3.4: Binary tree obtained by clustering, using the entropy measure.

The two vowel labels, /u/ and /ü/, are used to describe idealized extremes on a continuum for the acoustic realizations of the same underlying phoneme. Specifically, the label, /ü/, is used to describe the vowel when the second formant frequency is relatively high, due to its coarticulation with the adjacent phonemes. The label, /u/, is used to describe the vowel when the second formant frequency is relatively low. However, when the formant frequency is highly transitional, it is difficult to assign these two labels consistently. Therefore, it also becomes difficult for the network to distinguish these two allophones reliably, resulting in these two allophones highly confusable with each other. A case can certainly be made for not making the distinction between these two allophones.

As an example, Figure 3.5 compares the acoustic realizations of /u/ and /ü/. Parts (a) and (b) show the spectrograms of two typical /u/ and /ü/, respectively. It can be seen that the acoustic realizations are quite different, one with higher second formant frequency and the other with lower second formant frequency. For comparison, parts (c) and (d) show two more examples. It can be seen that the acoustic realizations in parts (c) and (d) are actually quite similar, with the second formant frequency lowering rapidly.

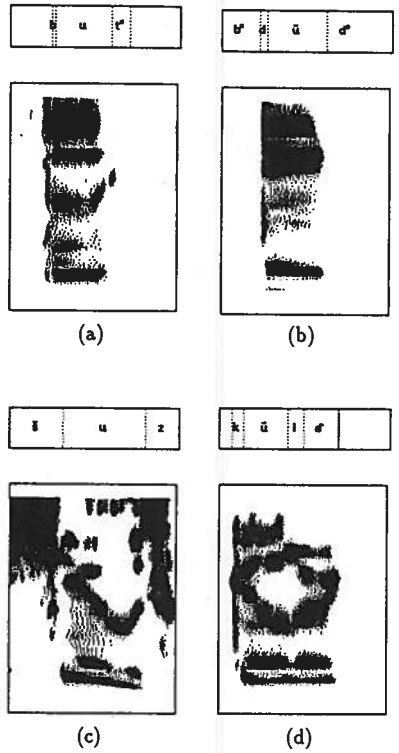


Figure 3.5: Comparisons of the acoustic realizations of /u/ in parts (a) and (c), and /ü/ in parts (b) and (d). When the second formant frequency is highly transitional, it becomes difficult to assign phonetic labels consistently.

3.4.3 Distinctive Features

While the classification rate indicates how often the network labels agree with the transcription labels, it does not show whether the disagreements are reasonable. Although the binary tree in Figure 3.4 and the confusion matrices in Tables 3.1 to 3.3 suggest that most of the confusions are quite reasonable, they do not give an objective measure of *how* reasonable the confusions are. In this section, the network is evaluated in a phonological dimension which can provide an objective way to measure the reasonableness of the confusions.

Specifically, the network is evaluated using the distinctive features. A distinctive feature is a minimum unit in the phonological dimension that can be used to characterize speech sounds [20,67]. Different phonemes take on different distinctive feature values. If the feature values are binary, about 20 features are needed to represent the phonemes in American English [129]. Assuming the distinctive features are independent and equally important, the differences in feature values can then be used as a measure of the phonetic difference or distance between two phonemes. Table 3.4 shows the distinctive features for some vowels [126]. The features are: high, low, back, round, retroflex, and tense. For example, the vowels /ɑ/ and /ɔ/ are different by only the rounded feature.

In this evaluation, the network trained on Database IV was used. The distinctive features of the vowel labels provided by the transcription and the network were looked up from Table 3.4 and compared. Since feature values might be ambiguous for diphthongs, only the vowels listed in Table 3.4 were evaluated. Figure 3.6 shows the performance of the network in terms of the number of binary features different between the two sets of labels. It can be seen that 63% of the test tokens have labels that agree perfectly and that almost 95% of the labels differ by two or fewer distinctive features, suggesting that most of the confused vowels are quite similar.

Vowel	High	Low	Back	Round	Retroflex	Tense
/e/	-	-	-	-	-	+
/æ/	-	+	-	-	-	-
/i/	+	-	-	-	-	+
/ɛ/	-	-	-	-	-	-
/ɪ/	+	-	-	-	-	-
/ü/	+	-	-	+	-	+
/o/	-	-	+	+	-	+
/u/	+	-	+	+	-	+
/ɑ/	-	+	+	-	-	-
/ɔ/	-	+	+	+	-	-
/ɝ/	-	-	+	+	+	-
/ʌ/	-	-	+	-	-	-
/ʊ/	+	-	+	+	-	-

Table 3.4: Distinctive features for some vowels. "+" stands for the presence of the feature, whereas "-" stands for the absence.

3.4.4 Performance on Individual Speakers

Due to across-speaker variations, the performance of the network on different speakers could be quite different. In Database IV, there are about 2,000 test tokens extracted from continuous sentences spoken by 50 speakers. On the average, there are about 40 vowel tokens from each speaker. Although the number of tokens from each speaker is very limited, we nevertheless tested the performance of the network using data from the 50 individual speakers.

The network trained on the 500 speakers in Database IV is used for our study. Figure 3.7 shows the distribution of the performance of the network on the 50 speakers. Although the average performance is about 60%, the performance on a speaker can be as high as 90%, but can also be as low as 40%. The fact that the performance results can vary so much suggests that the network and the speaker normalization procedure discussed in Chapter 2 cannot adequately deal with across-speaker variations. In Chapter 6, rapid adaptation of the network to improve the performance on a new

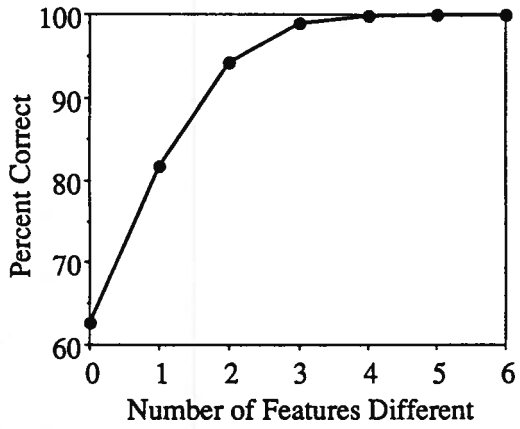


Figure 3.6: Performance of the network in terms of the number of features different from those in the transcription labels.

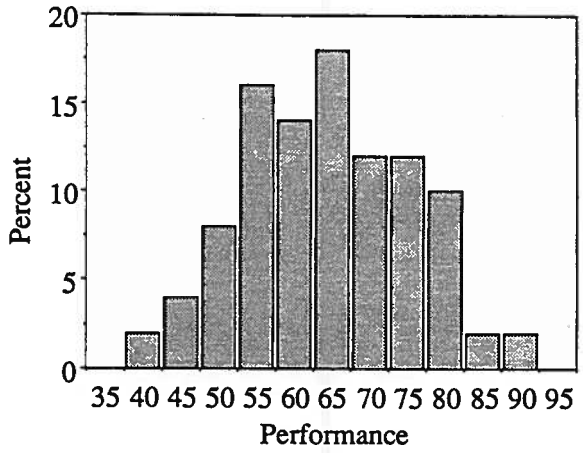


Figure 3.7: Distribution of the performance of the network on 50 new speakers. The network was trained on Database IV.

speaker will be discussed.

3.5 Chapter Summary

In summary, this chapter discusses the motivations for selecting the multi-layer perceptron (MLP). The fact that MLP does not need to assume specific probability distributions or distance metrics may enable it to model our ignorance in the detailed acoustic-phonetic characteristics of speech. The fact that they can take on continuous or discrete inputs may provide us the flexibility for selecting appropriate acoustic and/or linguistic attributes. Its discriminatory characteristics may also enable it to distinguish effectively among different classes.

The structure of the network is quite simple, with one hidden layer. Heterogeneous sources of information can be made available to the network. Performance is measured in terms of how often the labels produced by the network agree with the transcription labels.

The network has been evaluated on the four databases described in the previous chapter, using only the synchrony envelopes. Experimental results indicate that a substantial difference in performance can be expected over a wide range of recognition tasks, depending on whether the task is speaker-independent, what is the restriction on the phonetic contexts, and whether the speech material is spoken continuously. As the task becomes more difficult, significantly more training data may be needed.

Different ways to evaluate the network have also been presented. The performance of the network has been found to be comparable to human performance. However, as we will see in Chapter 4, with the availability of more sources of information, the performance of the network can be improved. Visual inspection of the confusion tables, results of a clustering procedure based on the entropy measure, and performance in terms of the distinctive features indicate that most of the disagreements between the transcription and network labels are quite reasonable.

Chapter 4

Network Characteristics and Representations

This chapter describes a set of experiments that were designed to help us gain a better understanding of different *characteristics* and *representations* of the network. Specifically, it discusses the performance of the network as a function of the number of training iterations, amount of training data, number of hidden units, number of hidden layers, and use of the nonlinear sigmoid function. It also discusses the structure and self-organization of the internal representations, alternative choices for output representations, and the use of heterogeneous input representations. Unless otherwise specified, our study uses all the vowel tokens from Database IV. Furthermore, only the synchrony envelopes are used and the network has one hidden layer of 32 units.

4.1 Network Characteristics

4.1.1 Training Characteristics

The network must be trained before it can perform reasonable classification. The training procedure proceeds by iterating error back-propagation through the training set [111]. One may keep on training the network until the mean squared error is close to zero or below a pre-determined threshold. However, the mean error may

never approach zero if the task is complicated, and an appropriate threshold may depend on the specific task. Furthermore, due to the finite amount of training data, performance on the training data may keep improving as the network is trained using the same data repeatedly. If too much training is allowed, the network may even memorize irrelevant information in the training data. Thus a reasonable terminating criterion must be determined.

The performance of the network is repeatedly examined while it is being trained. Figure 4.1 shows the performance of the network and the weighted mean square error (WMSE) as a function of the number of training iterations. In this figure, one iteration corresponds to training the network using all the training tokens once. Since there are 20,000 training tokens, one iteration corresponds to performing back-propagation 20,000 times. The motivation for using the weighted mean square error metric will be described in Chapter 6.

Several interesting characteristics can be seen. First, most of the learning occurs in the first few iterations, since the performance increases most rapidly during the early stages of training. After a few iterations, the performance increases progressively slower. Second, as the number of iterations increases, the performance on the training data increases very slowly while that on the test data approaches an asymptote. At this point, it is possible that the training algorithm is forcing the network to memorize the detailed and irrelevant characteristics of the patterns in the training set. Since such improvement of performance on the training set cannot generalize to the test set, significant improvement on new test data cannot be expected by simply iterating through the training tokens repeatedly. Third, the incremental performance improvement on the training data may be used as a terminating criterion. The use of such an incremental measure of performance may be less sensitive to the specific task than an absolute measure. In this particular example, the "knee" of the performance curve on the training data is at about 10 iterations.[1] Fourth, the WMSE remains quite high for this particular task. In fact, the asymptote is above 0.4.

[1]To ensure convergence, 50 iterations were allowed in all experiments.

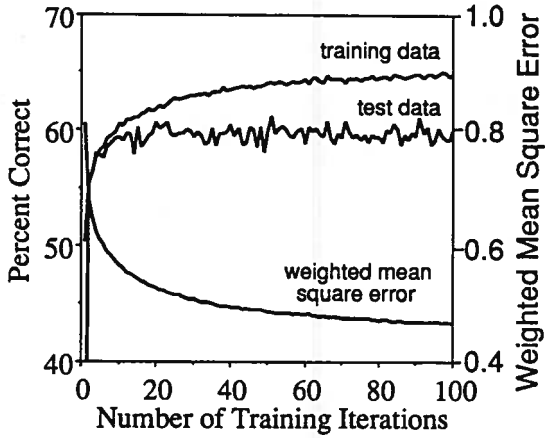


Figure 4.1: Performance of the network for the training and test data, as a function of the number of iterations through the training set. Also shown is the weighted mean square error (WMSE) on the training data.

4.1.2 Data and Robustness

Figure 4.1 suggests that for a fixed amount of training data, there exists an asymptote for the performance of the network. However, such an asymptotic performance can potentially be a function of the amount of training data. Although performance of pattern classifiers typically improves as the amount of training data increases, different classifiers may have different characteristics. First, the performance may have different levels of sensitivity to the amount of training data. While the performance for some classifiers may improve very rapidly as the amount of training data is increased, others may improve rather slowly. Second, the robustness of the classifiers may be different. While some classifiers have comparable performance on the training or test data, others may require very much more data before such convergence occurs.

The characteristics of the network are examined using different amount of training data. Figure 4.2 illustrates the robustness and the performance of the network as the number of training tokens increases. First, we can see that the performance on the test data is quite linear with the logarithm of the number of training tokens. At 200 tokens, the performance is approximately 40%. However, the performance increases to 60% at 20,000 tokens. On the average, increasing the amount of training data ten-fold improves the performance on the test data by approximately 10%. Second, the performance on the training data decreases as the number of training tokens increases. When the number of training tokens is small, it is possible that the network can memorize the training tokens individually, resulting in high performance on the training set. As the number of training tokens increases, the network can no longer memorize all the patterns. As a result, the performance decreases and the network is forced to pay attention only to the relevant information. Third, the two curves eventually converge as the number of training tokens increases. At about 20,000 tokens, the difference between the two curves is only about 3%. This suggests that the network can now generalize quite well to data that it has never seen. Furthermore, the average performance on new test data should not exceed the average performance on training data. All other conditions being equal, substantial improvement in performance on the test data cannot be expected by simply increasing the number of the training tokens to over 20,000. Fourth, if robustness is defined to be inversely proportional to the performance difference between the training and test sets, then the robustness of the network consistently improves as the number of training tokens increase. Finally, this experiment also points out the importance of having sufficient training data. Whether one has enough training data can be inferred by examining the robustness of the network. For example, the performance difference is about 20% at 2,000 training tokens, indicating that performance on the test data can be substantially improved by using more training data.

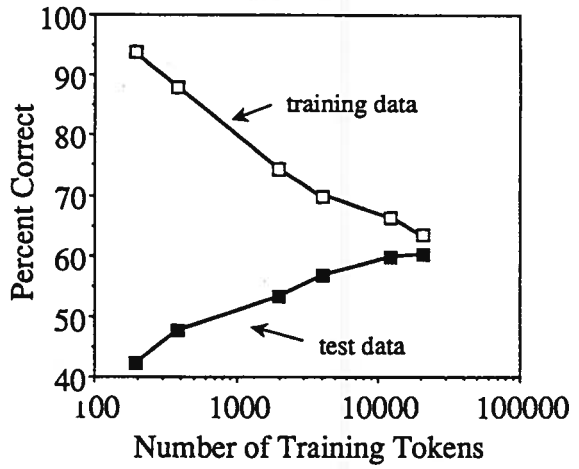


Figure 4.2: Performance for the training and test data, as a function of the number of training tokens.

4.1.3 Number of Hidden Units

Figure 4.2 suggests that for a fixed number of hidden units, the performance results on the training and test data eventually converge. However, the amount of training data needed to reach convergence, and the resulting performance, can potentially be a function of the number of hidden units. The number of hidden units can affect the network's capability of capturing the underlying characteristics of the input data. If there are too many hidden units, the network may simply memorize the irrelevant information in the training tokens. Thus increasing the number of hidden units may not improve the performance on new test data, although it usually enhances the flexibility of the network. On the other hand, if there are not enough hidden units, the network may not be able to capture the subtle but important differences between various classes of data. Thus decreasing the number of hidden units may also decrease the performance of the network. As Burr points out, there is a *critical* number of hidden units that would yield the best performance [13]. Increasing or decreasing the number of hidden units from this critical value could only decrease the performance.

The performance of the network was examined as the size of the network was varied. Figure 4.3 shows the performance of the network on the training and test data as a function of the number of hidden units. First, we can see that the performance on the training data consistently improves as the number of hidden units increases. Second, the performance on the test data is initially very similar to that on the training data, but improves progressively slower as the number of hidden units increases. At 256 hidden units, the performance difference between the two sets of data is about 12%, suggesting that there is enough flexibility in the network to capture irrelevant information in the training data. Third, the robustness of the network consistently degrades as the number of hidden units increases. For example, the performance difference increases from 3% for 32 hidden units to 12% for 256 hidden units. Thus although Figure 4.2 suggests that significant improvement in performance on the test data cannot be expected by simply having more than 20,000 training tokens when there are only 32 hidden units, some further improvement is possible by increasing

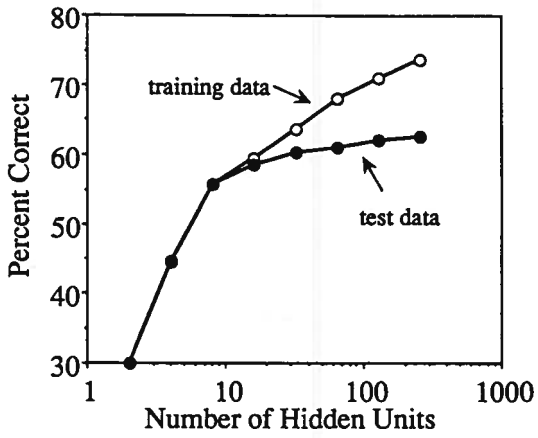


Figure 4.3: Performance as a function of the number of hidden units in the network.

both the number of hidden units and training tokens. In other words, there is a family of curves, each one of which is as shown in Figure 4.2. Depending on the number of hidden units, the performance on the training and test data may converge at a different number of training tokens, resulting in different performance.

Thus the performance of the network can depend on the amount of training data available as well as the number of hidden units. Comparing Figures 4.2 and 4.3, one may also ask the *relative* importance of the number of hidden units and the amount of training data. Figure 4.4 shows the performance of the network as a function of the number of hidden units and the number of training tokens. Like Figure 4.3, we can see that the performance typically improves as the number of hidden units increases. Second, we can see that the performance may decrease when there are too many hidden units, suggesting that there may be too many parameters in the

network to estimate using the limited amount of training data. Third, as long as the number of hidden units is reasonably chosen, increasing the size of the training set typically improves the performance of the network. Finally, the "right" number of hidden units also seems to depend on the number of training tokens. For example, the performance for 200 training tokens is the best when there are 32 hidden units, while that for 2,000 training tokens is the best when there are 128 hidden units.

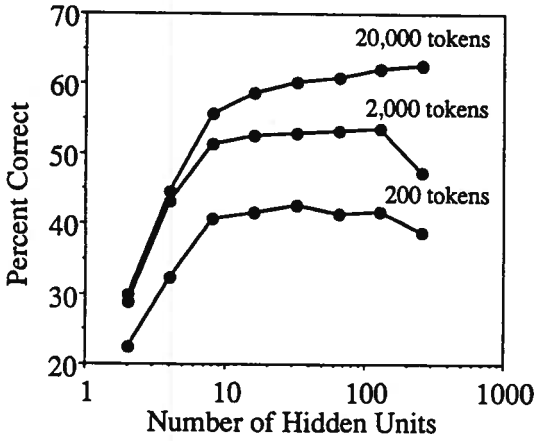


Figure 4.4: Performance on the test data as a function of the number of hidden units and the number of training tokens.

4.1.4 Number of Hidden Layers

Besides the number of units in a hidden layer, the number of hidden layers can also influence the capability of a network. Specifically, while single-layer perceptron (SLP) can only make half-plane decisions, MLP's can form decision regions of arbitrary

shapes [90]. Furthermore, it has been shown recently that an MLP with 1 or 2 hidden layers can approximate any continuous function as well as form nonconvex and disjoint decision regions [28,29,62,96]. This section compares networks with 0, 1, or 2 hidden layers. Network characteristics and potential limitations will also be discussed.

4.1.4.1 Two Hidden Layers

A careful study performed by Huang and Lippmann demonstrated that the error rates for networks with 1 or 2 hidden layers are quite similar, indicating that problems that are difficult for a network with 1 hidden layer are also difficult for a network with 2 hidden layers [62]. To gain further insights, networks with 2 hidden layers were used to recognize the 16 vowels. Figure 4.5 shows the performance of the network as a function of the number of hidden units in the first hidden layer, $H1$, and the number of hidden units in the second hidden layer, $H2$. It can be seen that increasing the number of hidden units in the first or second layer, in general, increases the performance of the network. Comparing Figures 4.3 and 4.5, we can see that the best performance results using 1 or 2 hidden layers are actually quite close, both at about 60%. Furthermore, close examination of Figure 4.5 reveals that the performance improves quite abruptly as $H2$ is increased to 4, and that a further increase in $H2$ results in a relatively small improvement. This seems to agree with our intuition. If the temperature is assumed to be zero so that the resulting sigmoid function is as shown in Figure 1.2b, then the output of a basic unit can only be 0 or 1. Since there are altogether 16 output classes, a minimum of 4 hidden units in $H2$ is required to encode the vowels. These results suggest a criterion for choosing the number of units in the hidden layer immediately before the output layer, N_{H_L}. Specifically,

$$N_{H_L} > log_2 N_O, \tag{4.1}$$

where N_O stands for the number of units in the output layer.

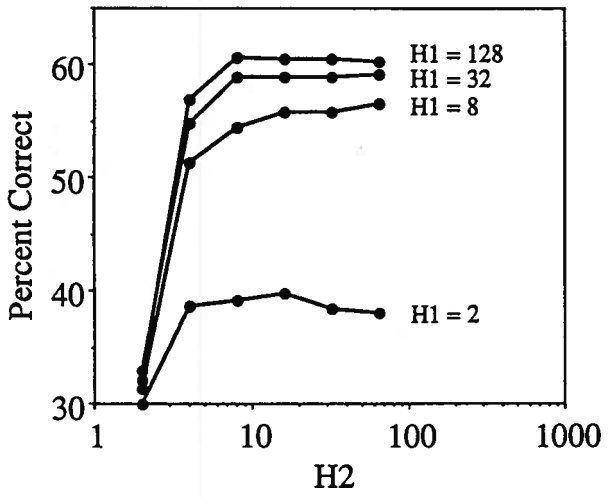


Figure 4.5: Performance of a network with two hidden layers, as a function of the numbers of hidden units. H1: number of hidden units in the first hidden layer. H2: number of hidden units in the second hidden layer.

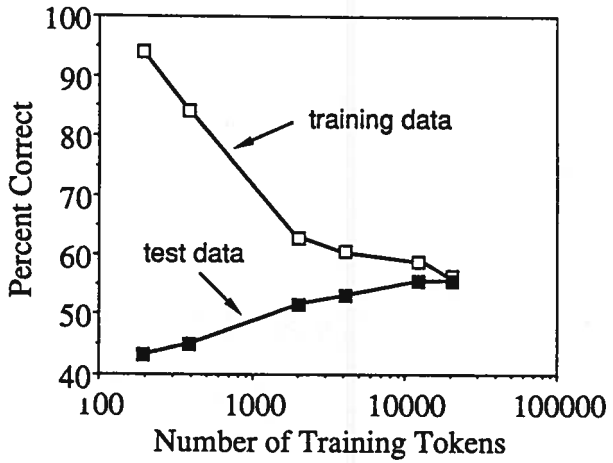


Figure 4.6: Performance of SLP on the training and test data.

4.1.4.2 No Hidden Layers

In order to compare the performance and characteristics of SLP with MLP, a network with no hidden layer is used to recognize the 16 vowels. Figure 4.6 shows the performance of the SLP as a function of the number of training tokens. Like Figure 4.2, we can see that the performance on the training data decreases while that on the test data improves quite linearly with the log of the number of training tokens. When 20,000 tokens are used, the performance on the training and test sets are both at about 55%. This result suggests that all other factors being equal, the performance using SLP cannot be improved by having more training data, in contrast to MLP whose performance can be improved by increasing *both* the number of hidden units and the amount of training data.

4.1.5 Importance of the Nonlinear Sigmoid Functions

As discussed in Chapter 1, the sigmoid function allows the training algorithm to pay more attention to regions near the decision surfaces and less attention to regions farther away. Thus using the sigmoid function in the output and hidden layers may improve the discriminating capability of the network. Furthermore, the sigmoid function allows complex decision regions to be formed. If the nonlinear sigmoid function is not used in the hidden layers, each layer of units simply performs a linear operation. The sequence of linear operations in the feedforward network can then be combined to form one linear operation. In other words, MLP would become SLP if the sigmoid function were not used in the hidden layers.

The importance of the sigmoid function in the output layer was examined. Figure 4.7 compares the performance of the network using two different techniques. First, the network is trained in the usual way, i.e. the sigmoid function is used in each of its basic units. Second, the sigmoid function is used only in the hidden units. In other words, each hidden unit applies the sigmoid function to the weighted sum of its inputs while each output unit simply performs a weighted sum, i.e. Equation 1.1 becomes

$$y_i = z_i \quad for \quad i \epsilon O, \tag{4.2}$$

where O is the set of output units. From Figure 4.7, we can see that the second network needs to have more hidden units in order to achieve the same performance as that of the first network. For example, in order to achieve a performance of 55%, the network that does not use the sigmoid function in its output layer requires twice as many hidden units as the one that uses the sigmoid function. This seems to suggest that the sigmoid function in the output layer can indeed improve the discriminating power of the network.

As a further step, similar comparison using the SLP was made. The performance

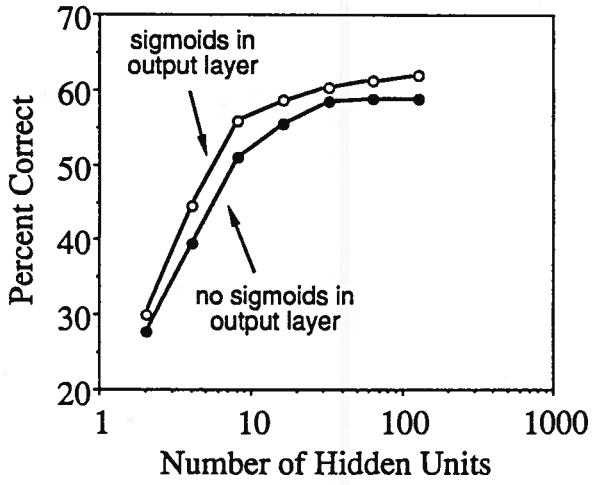


Figure 4.7: Comparison of two networks: one uses the sigmoid function in its output layer and the other does not use the sigmoid function in its output layer.

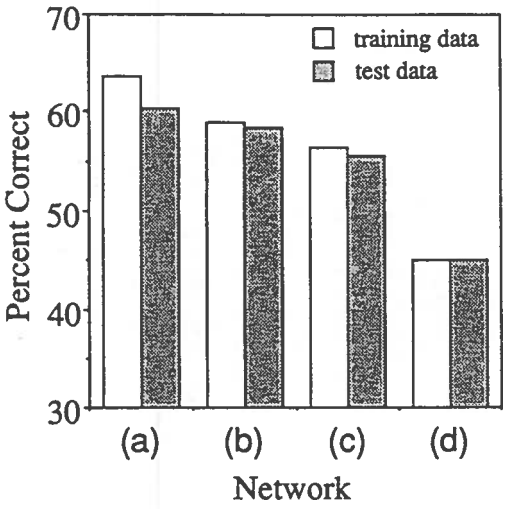


Figure 4.8: Network performance on the training and test data when (a) sigmoid function is used in the hidden and output units of MLP, (b) sigmoid function is used only in the hidden units of MLP, (c) sigmoid function is used in the output units of SLP, and (d) no sigmoid function is used in SLP.

decreases from 55% to 45% when the sigmoid function is not used. Figure 4.8 summarizes the above results.

4.2 Internal Representations

After the training procedure is completed, knowledge about the vowels is embedded in the connections of the network. Thus studying the internal representations of the network can potentially help us acquire a better understanding of how the network

uses its input information to perform classification or how it learns to pay attention to the relevant linguistic information in the speech signal. In the following sections, we will examine the internal knowledge representations of the SLP and MLP.

4.2.1 Extraction of Linguistic Information

The connection weight pattern of a network with no hidden units is first studied. In this network, there are 16 output units and 100 input units. To make the study more manageable, the network is trained with the vowel tokens from Database I, thus leaving out contextual and across-speaker variations. After the network is trained, the weights connected from all the input units to one particular output unit are extracted as shown in Figure 4.9a. These weights are then displayed in a spectrographic form. Figure 4.9b shows the connection weight pattern to an output unit that corresponds to the vowel /α^y/. For comparison, the synchrony spectrogram and input signal of an /α^y/ token are also shown. In these displays, the larger weights are shown in black, while the smaller weights are shown in white. We can see the three distinct average input spectra to the network. We can also see that the connection weights are the greatest at the formant locations and gradually decrease as the connections depart away from the formant locations. Weights for the third formant also seem to be lower than those for the first two formants. This seems to suggest that the network can learn to pay attention to spectral information near the formant frequencies. It also agrees with the perceptual results that the first two formant frequencies contain most of the linguistic information [44].

As a further step, a different network is constructed to extract the distinctive feature: back. The network has 2 output units, one for the presence of the feature and the other for the absence. For example, the target value for the unit of presence is set to high and that for the unit of absence is set to low when the input vowel token has the back feature. Figure 4.10 shows the connection weight patterns to the back and -back units after the network is trained using Database II. For comparison, the input signals of two examples are also shown. We can see that the weights for the

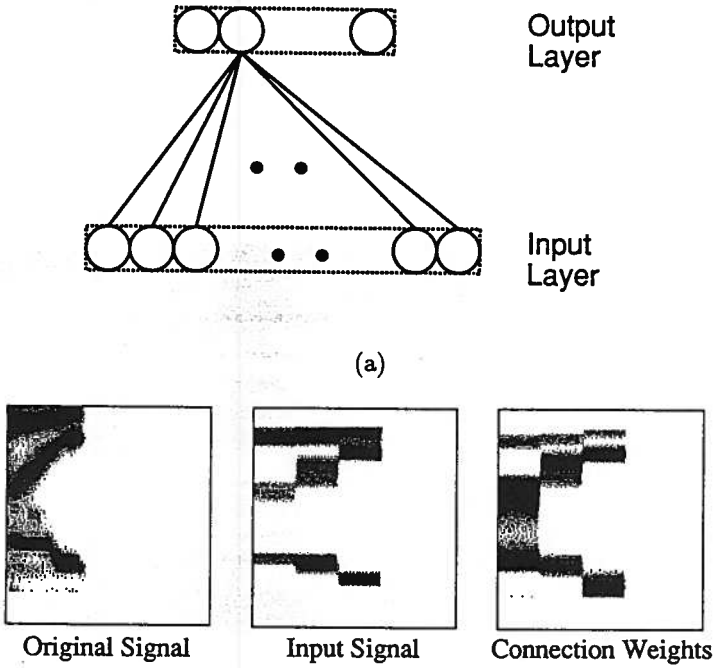


Figure 4.9: Internal representation with no hidden layers: (a) Extraction of all connection weights to one output unit. (b) Spectrographic displays for the original signal, input signal, and the connection weights to an output unit that corresponds to the vowel /ɑʸ/.

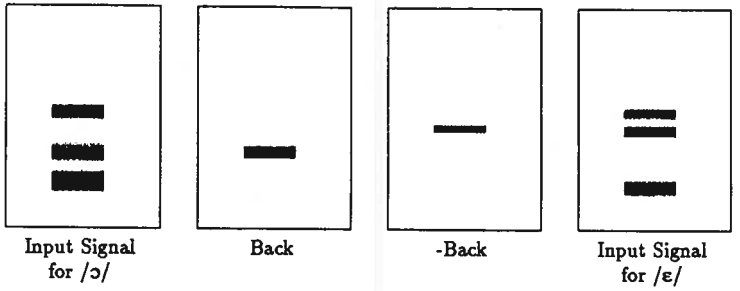


Figure 4.10: Internal representation with no hidden layers: spectrographic displays for the connection weights to output units that correspond to the back and -back features, and the input signals of two examples, /ɔ/, a back vowel, and /ɛ/, a -back vowel.

back unit are the greatest at the frequency where the back vowels typically have their second formant and that those for the -back unit are the greatest at the frequency where the -back vowels typically have their second formant. This example suggests that the network can learn to extract the relevant acoustic properties from the speech signal that correspond to the distinctive features. Appendix B shows more examples of networks that extract other distinctive features.

4.2.2 Orthogonality

The internal representation of the network with one hidden layer is also examined, using speech data from Database IV. Let V_j^o stand for the connection vector to the j^{th} output unit, u_j^o, as shown in Figure 4.11a. Y^H stands for the vector formed by the outputs of all the hidden units. The input to u_j^o, z_j^o, is simply the dot product of V_j^o and Y^H:

$$z_j^o = V_j^o \cdot Y^H, \tag{4.3}$$

where $1 \leq j \leq N_O$. z_j^o has a high value when Y^H is highly correlated with V_j^o, and has a low value when they are uncorrelated. Therefore, if the network is well-trained, V_j^o can be considered as a prototype vector for u_j^o. If the input vector belongs to the j^{th} class, ω_j, then its corresponding Y^H should be highly correlated with V_j^o, and not so much correlated with $V_i^o \; \forall i \neq j$.

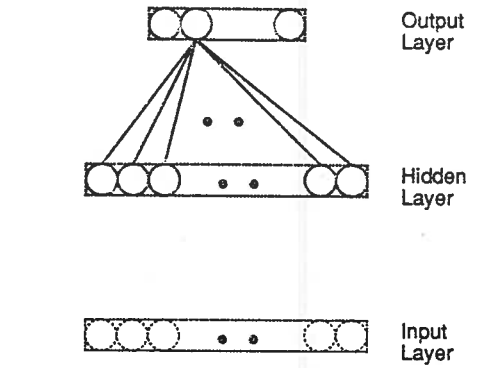
As a result, one may suspect the N_O connection vectors, V_j^o *for* $1 \leq j \leq N_O$, to be highly uncorrelated after the network is trained. To examine their correlations, Figure 4.11b shows the distribution of the angles between these connection vectors as the number of hidden units increases. The circles represent the mean of the distribution and the vertical bars stand for one standard deviation away from the mean. We can see that as the number of hidden units is increased, the vectors become increasingly orthogonal to each other, and the distributions become progressively more concentrated. This seems to suggest that the training procedure tries to find a set of orthogonal prototype vectors in the hidden layer space. In other words, when the number of hidden units, N_H, is large,

$$\frac{V_i^o \cdot V_j^o}{\mid V_i^o \mid\mid V_j^o \mid} \approx 0, \qquad \forall i \neq j, \tag{4.4}$$

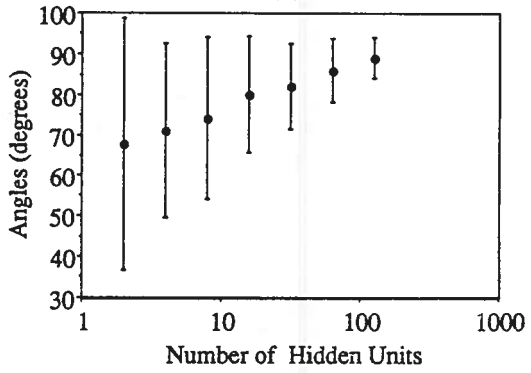
where $\mid V_j^o \mid$ stands for the magnitude of the connection vector, V_j^o.

4.2.3 Random Connections

Figure 4.11b suggests that the N_O connection vectors in the network *after* training are quite orthogonal when N_H is large. However, when $N_H >> N_O$, the probability of obtaining N_O random vectors that are orthogonal to each other in a N_H-dimensional



(a)



(b)

Figure 4.11: Internal representation with one hidden layer. (a) Extraction of connection weights from hidden to output layer. (b) Distribution of angles as a function of the number of hidden units after training.

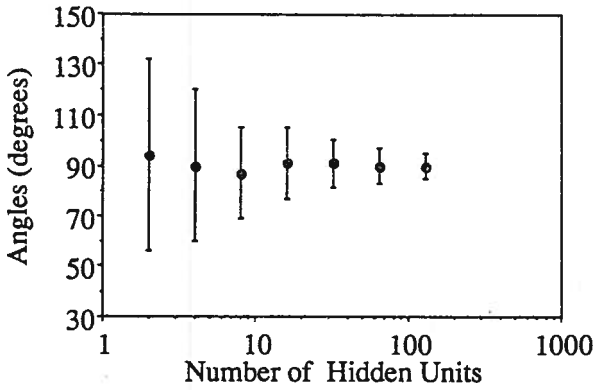


Figure 4.12: Distribution of correlations of connection vectors as a function of the number of hidden units before training.

space is quite high. Thus the distribution of the angles between N_O random connection vectors can potentially be quite similar to those in Figure 4.11b when N_H is large.

The random connection vectors of a network are examined. The corresponding correlations *after* random initialization but *before* training are shown in Figure 4.12. It can be seen that although the means of the distributions are quite constant, the standard deviations can be quite different for different number of hidden units. Comparing Figures 4.11b and 4.12, we can see that the two distributions become increasingly similar as the number of hidden units increases. For example, with 128 hidden units, the two distributions are almost the same.

This observation leads to the speculation that perhaps the connections between the output and hidden layers, V_j^o, need not be trained when there is a sufficient number of hidden units in the network. Perhaps these connections can be fixed after random

initialization of the network. In other words, it may be sufficient to train only the connections between the input and the hidden layers. As a result, the computational requirement for training can be reduced. Although the connections between the hidden and output layers are not trained, they can be used to back-propagate errors to the hidden layer.

Figure 4.13 compares the performance of the network using three different methods. In Method A, the network is trained in the usual way, i.e. all the connection weights are trained. In Method B, only the connections between the input and hidden layers are trained, i.e. the connections between the hidden and output layers are fixed after random initialization. In Method C, only the connections between the hidden and output layers are trained, i.e. the connections between the input and hidden layers are fixed after random initialization. Method C has been proposed and studied previously in the literature [47,62,89,101,113].

We can see from Figure 4.13 that with a sufficient number of hidden units, it may not be necessary to train all the connections in the network. For example, with 128 hidden units, the performance difference between training all the connections and training only the connections between the input and hidden layers is only a fraction of a percent. We can also see that when the number of hidden units is small, training only the connections between the hidden and output layers is more effective than training only the connections between the input and hidden layers. However, when only the connections between the input and hidden layers are trained, the performance increases rapidly as the number of hidden units increases. With a sufficient number of hidden units, such a training method becomes more effective, suggesting that training the connections between the input and hidden layers is more important than training those between the hidden and output layers.

Although the relative importance of the layers of connection weights is not as yet fully understood, the observations from Figure 4.13 could be explained by assuming the temperature of the sigmoid function to be zero so that the output of each basic unit is binary. As we have discussed in Chapter 1, a hidden unit makes half-plane

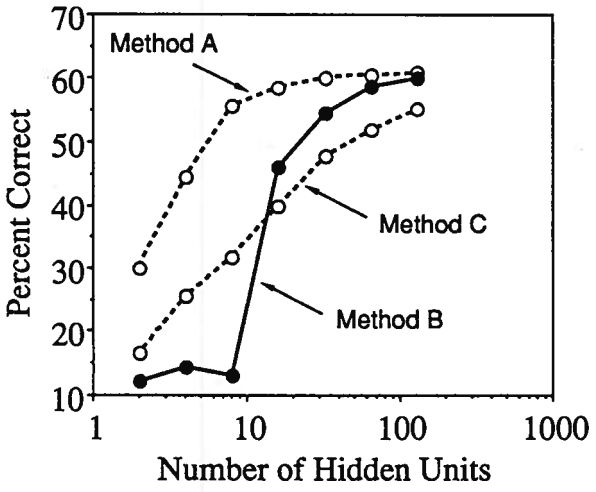


Figure 4.13: Performance of recognizing the 16 vowels using Method A, in which all the connections in the network are trained, Method B, in which only the connections between the input and hidden layers are trained, and Method C, in which only the connections between the hidden and output layers are trained.

decision while an output unit forms regions by performing boolean operations on the binary half-plane decisions made by the hidden units. Fixing the random connections between the input and hidden layers amounts to choosing random hyperplanes in the input signal space. Furthermore, fixing the random connections between the hidden and output layers simply freezes the boolean operations, which in turn determine the "boolean relationship" between the hyperplanes.

In Method B, the boolean operations are random. However, the "shape" or "location" of a decision region can still vary since the hyperplanes can be changed during training. Although the connections between the hidden and output layers are not allowed to change, using these connections to back-propagate errors to the hidden layer allows the hyperplanes to change and cooperate with the boolean operations to form effective decision regions. For example, Figure 4.14 shows two different decision regions formed by the same boolean operation but different sets of hyperplanes.

In Method C, the hyperplanes and the possible decision regions are random. Training the connections between the hidden and output layer amounts to finding the most effective boolean operations or selecting some of these random decision regions. However, the performance of the network would depend on the locations and/or orientations of the random hyperplanes, which are not likely to form effective decision regions. Nevertheless, the performance can be improved by using more hidden units, since more random hyperplanes are more likely to better approximate the effective decision regions.

Figure 4.13 also shows that when only the connections between the input and hidden layers are trained, the performance improves most rapidly when the number of hidden units, N_H, is the same as the number of output units, N_O, i.e. 16. Although the most appropriate number of hidden units may very well depend on the specific task, this result suggests that perhaps the number of hidden units should be chosen to be greater than or equal to the number of output units:

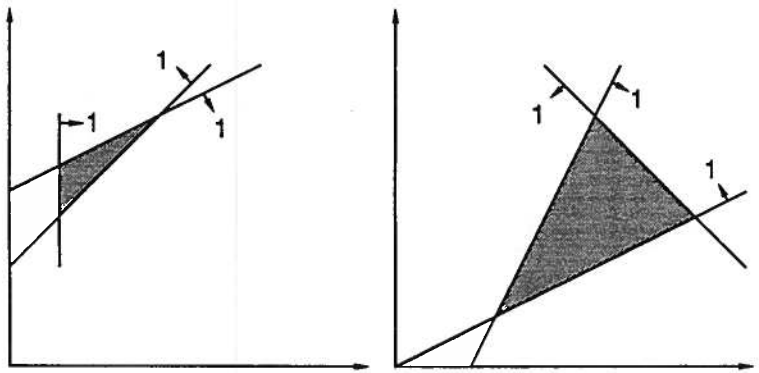


Figure 4.14: Different decision regions can be formed for the same connection weights or boolean operation between the hidden and output layers. The boolean equation for the decision regions in both (a) and (b) is **111**.

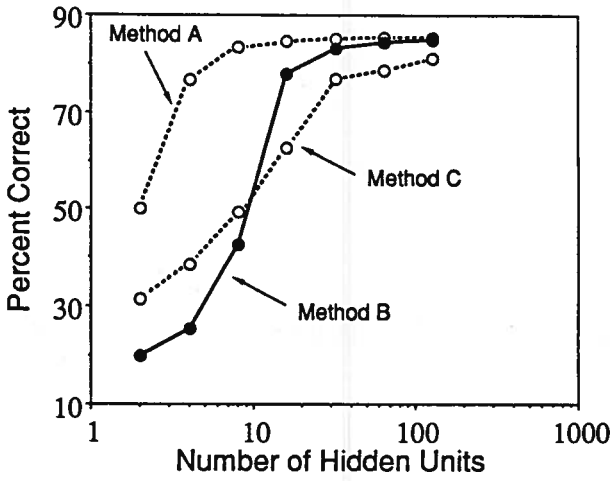


Figure 4.15: Performance of recognizing the 8 vowels using Method A, in which all the connections in the network are trained, Method B, in which only the connections between the input and hidden layers are trained, and Method C, in which only the connections between the hidden and output layers are trained.

$$N_H > N_O. \tag{4.5}$$

As a result, there is, on the average, at least one hyperplane for each class.

For comparison, Figure 4.15 shows the performance results of a different network used to recognize 8 of the vowels: /e, æ, i, ɑʸ, u, ɔ, ɑʷ, ɝ/. In other words, $N_O = 8$. Again, we can see that when only the connections between the input and hidden layers are trained, the performance rises most rapidly when $N_H = N_O$.

4.2.4 Self-Organization

The connection weights capture knowledge about the input signal and determine how information should be processed inside the network. In MLP, an input vector is first transformed into an intermediate vector in the N_H-dimensional hidden space, which in turn, is transformed into a final vector in the N_O-dimensional output space. Thus the different sources of input information are internally organized in a different way before classification is completed.

The internal organization of a network is studied by examining the activations or outputs of the hidden units after training. The network has one hidden layer and is trained to recognize the vowels listed in Table 3.4. For the j^{th} training token, X_j, the vector formed by the outputs of the hidden units, Y_j^H, is obtained. All the vectors of the same class are then averaged to form one prototype vector,

$$\bar{Y}_i^H = average(Y_j^H) \quad for \quad X_j \, \epsilon \, \omega_i \quad 1 \leq j \leq N, \tag{4.6}$$

where N is the number of training tokens. These average vectors, $\bar{Y}_i^H$, are then grouped together using hierarchical clustering, with a Euclidean distance metric [35]. Figure 4.16 shows the dendrogram as a result of the clustering procedure [35,52]. It can be seen that vowels of similar phonetic dimensions are grouped together. For example, the low and back vowels are grouped together into one category while the high and front vowels are grouped into another. As another example, the vowel /ɝ/, which requires a retroflexed tongue position and is quite distinct in its articulation from all other vowels, stays by itself in the dendrogram for the longest. These results suggest that the network can automatically organize its internal structure in a way that seems to agree with our knowledge about how the vowels should be organized in the phonological dimension.

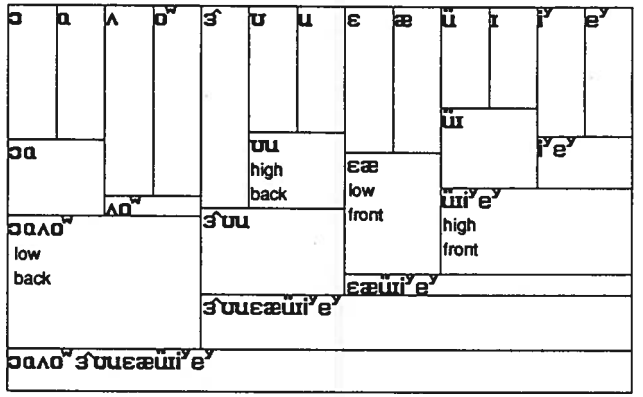


Figure 4.16: Dendrogram obtained by hierarchically clustering the outputs of the hidden layer.

4.3 Output Representations: Alternatives and Expandability

The fact that the network can organize the vowels into natural phonological classes seems to agree with the belief among phonologists that phonemes can be represented underlyingly as a collection of distinctive features.[2] Such an approach has the appeal that it decomposes the problem of phoneme recognition by focusing on acoustic attributes motivated by theory. As a further step in this direction, a network is used to extract the six distinctive features from the vowels listed in Table 3.4. The features are: high, low, back, round, retroflex, and tense. Instead of using 16 output units, we have in this situation only six output units, one for each feature. Figure 4.17 shows the performance, or the average agreement with the transcription, for the extraction of the features. We can see that the accuracy ranges from about 86% for the tense feature to 98% for the retroflex feature, suggesting that the network can extract the features quite reliably. Furthermore, it is found that the network can extract all the features correctly for 57% of the vowel tokens.

After the distinctive features are extracted, they can be used directly for lexical access or for further transformation into different phonological representations. For our particular task, these features can potentially be used for recognition of the vowels. Specifically, recognition of the vowels can be performed in two successive stages. In the first stage, a network is trained to extract the distinctive features. This network's connections are then fixed and its outputs are used as inputs to a second network. In other words, the second network is trained to map the set of distinctive features to the vowels. Since only 57% of the vowel tokens have all their features extracted correctly by the first network, some of the errors can potentially be corrected by the second network. It has been found that the overall performance is about 63%. When evaluated using the distinctive features, the result is quite similar to that shown in Figure 3.6. Furthermore, this experiment also suggests that if there exists a technique

[2]Distinctive features and their use for performance evaluation are discussed in Section 3.4.3.

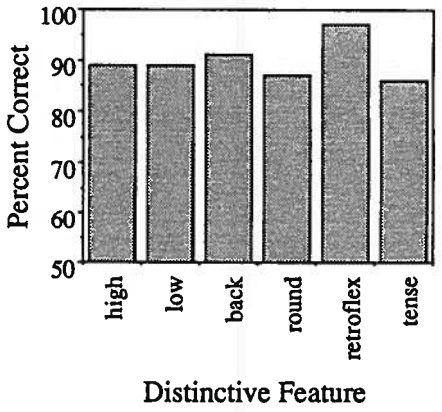


Figure 4.17: Performance for extracting 6 different distinctive features of the vowels.

that can extract the distinctive features reliably, then the network can potentially be used to map the features to other phonological units.

4.4 Input Representations: Integration of Heterogeneous Information

As Figure 3.1 illustrates, the input representations of the network can be heterogeneous, including spectral, durational, and contextual information, and they can be other acoustic/linguistic attributes. In this section, the use of the network to integrate heterogeneous sources of information is discussed. There is one layer of 64 hidden units in each network.

Figure 4.18 shows the performance when different amounts of information are available. As we have seen in Chapter 3, when only the synchrony envelopes are available, the average agreement between the labels produced by the network and those provided by the transcription is about 60%. Since there are 100 input units and consecutive layers of units are fully connected, there are altogether 7,424 connections in the network.

Next, the mean rate response is added to the input units, resulting in a total of 199 input units. While the number of connections in the network increases to 13,760, the average agreement with the transcription labels improves to 64%. Apparently the mean rate response contains some information that is not present in the synchrony envelopes.[3]

Third, durational information is also made available to the network. The vowel durations are first quantized into 20 equally spaced intervals between 0 and 200 *msec*, since durations of the vowels are typically less than 200 *msec*. An additional set of 20 durational units, one for each interval, is then appended to the spectral units,

[3]When only the mean rate response is available to the network, the average agreement with the transcription labels is also about 60%. The improvement of only 4% by using both the synchrony envelopes and mean rate response suggests that the two sources of information are highly correlated.

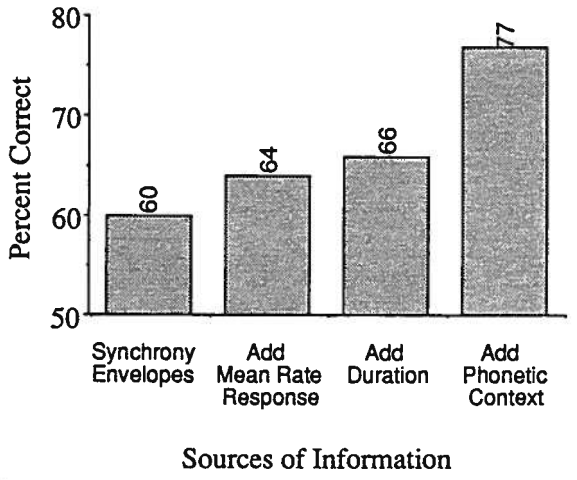


Figure 4.18: Integration of heterogeneous sources of information.

resulting in a total of 219 input units. A durational input unit is set to 1 if the vowel duration falls within its interval, but is set to 0 otherwise. Quantizing the vowel duration and using the unary representation creates redundancy and can help the network to converge [111]. There are altogether 15,040 connections in the network. As Figure 4.18 shows, the average agreement increases to 66%.

Researchers in the past have employed context-dependent phoneme models to account for the effects of coarticulation [83,118]. In this approach, multiple models are created for each phoneme. Each model corresponds to the same underlying phoneme but realized in different phonetic contexts. As a result, contextual effects can be better accounted for with context-dependent models than with context-independent ones. However, such context-dependent models can result in a severe training problem if there are not enough training tokens. For example, if there are 40 phonemes, there can be as many as 64,000 triphone models. As a result, researchers have been investigating methods to reduce the number of context-dependent models [83].

Since phonetic context affects the realization of a phoneme, the identity of an adjacent phonetic unit can provide useful information for recognition. For example, the second formant frequency of a front vowel is expected to be lower if it is adjacent to /l/. Instead of constructing different context-dependent networks, the identities of the adjacent phonetic units can be used as additional sources of input information for the network. The following two criteria are adopted to incorporate the contextual information. If the phonetic context can be learned by the network as a separate and additional source of information for distinguishing among different speech sounds, then the recognition accuracy should improve. By the same token, when contextual information becomes less and less certain, recognition performance should degrade gracefully towards that of a network which does not have contextual information.

The following procedure is used to train the network, taking into account potential variations in the certainty of the phonetic context. Let y_j^c denote the value of the j^{th} context input unit, and unit i correspond to the actual adjacent phonetic unit. Then

$$y_j^c = \begin{cases} \frac{1}{M} + R(1 - \frac{1}{M}) & if \quad j = i \\ \frac{1}{M-1}(1 - y_i^c) & otherwise, \end{cases} \tag{4.7}$$

where R is constrained between 0 and 1, and M stands for the number of possible adjacent phones (in our case, M = 61). Note that $\sum_j y_j^c = 1$, and y_j^c is uniformly distributed except for a peak at $j = i$. When $R = 1$, the phonetic context is known with certainty, and $y_i^c = 1$. When $R = 0$, no context information is provided, and $y_j^c = \frac{1}{M}$ for all j. The value of R is randomly chosen for each training token to account for different levels of certainty associated with the context during actual recognition.

By adding $2M$ context units to the input layer, the total number of input units increases to 341, resulting in a total of 22,848 connections. As Figure 4.18 shows, the average agreement with the transcription labels improves to 77% when the context information is known with certainty, i.e. $R = 1$. When tested with $R = 0$, i.e. no context information, the average agreement drops back to 66%, the percentage when the network is trained and tested without contextual information.

It should be noted that Equation 4.7 provides a very crude model for the phonetic contexts. It assumes that the context has been partially specified. During actual phonetic recognition, contextual information may not be available initially, i.e. $R = 0$. However, as the utterance is processed, the values of *some* contextual units may increase while those of others may decrease. As a result, the phonetic context becomes less and less uncertain and can be used as an additional source of information. Furthermore, adjacent phonetic units can exchange contextual information to aid recognition. Thus subjective decisions for constructing different models for different allophones can potentially be bypassed.

Figure 4.19 shows the performance results of the above four tasks in more detail. It can be seen that the performance, in terms of rank-order statistics, consistently improves as more sources of information are made available to the network. For exam-

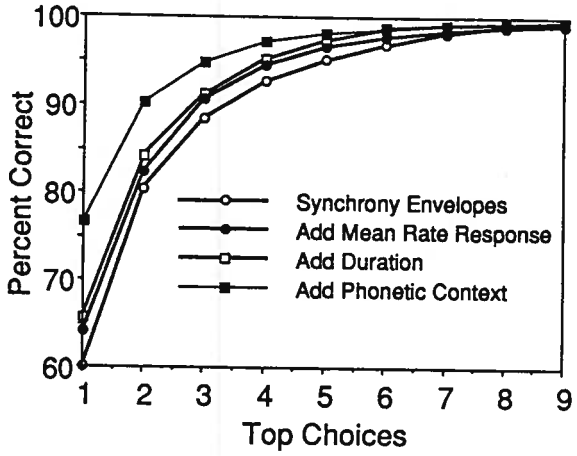


Figure 4.19: Performance in terms of rank-order statistics.

ple, the agreement with the transcription labels within the top three choices improves from below 90% when only the synchrony envelopes are available to approximately 95% when all sources of information are available. All these experiments suggest that the network can utilize different sources of acoustic and linguistic information.

4.4.1 Error Analyses

4.4.1.1 Confusions

Table 3.3 shows the confusion statistics when only the synchrony envelopes are available. As more sources of information are available, the confusion statistics should improve. For comparison, Tables 4.1 to Tables 4.3 show the confusion matrices of the 16 vowels when additional amounts of information are available to the network.

	ü	i	ɪ	e	ɛ	æ	ɔʸ	ɑʸ	ɑʷ	ɑ	ɔ	ʌ	o	ɝ	u	ʊ	Total
ü	39	22	22			1							1	4	10		77
i	1	91	4	2		1											267
ɪ	1	11	62	6	13	3		1						2		1	216
e		8	6	72	9	2	1	2									134
ɛ		3	13	3	44	17		1	3	2	2	2	3	5			158
æ		1	5		24	57		5	2	4		2					136
ɔʸ					5		63	16			11			5			19
ɑʸ		1	1	3	2	4	1	71	2	7	4	4		1			137
ɑʷ									67	10	6	8	8	2			52
ɑ					1	2		12	1	55	19	6	1	2			165
ɔ			1				2	1	1	12	75	1	5	1		1	139
ʌ			8		15	8		8	3	5	2	41	6	2		2	126
o			1	1	3				1		14	12	65	2	1		100
ɝ	1	1	4			1				1		2	1	88			82
u	9		6								6	3	3	3	64	6	33
ʊ			41		6						6	9	9		13	16	32

Table 4.1: Percent confusion table for Database IV when the synchrony envelopes and mean rate response are available.

Comparing Tables 4.1 and 4.2, we can see, for example, that not only does the percent correct for the vowel /ɪ/ improve from 62% to 65%, but the other vowels are also less often misclassified as /ɪ/. This may reflect the contribution of the durational information, since the vowel /ɪ/ is often relatively short.

Comparing Tables 4.2 and 4.3, we can see that due to the availability of the contextual information, the performance result on each of the vowels (along the diagonal of the confusion tables) consistently improves. To illustrate how contextual information can aid recognition, Figure 4.20 shows two examples of the vowel /ɑʸ/ in different phonetic contexts. We can see that the two realizations are quite different. The second formant frequency in the later part of the vowel /ɑʸ/ in part (b) is lowered quite significantly due to coarticulation with the following /l/. In fact, when no contextual information is available, the network produces /ɑ/ as the top choice, and /ɑʸ/ as the second choice. However, with the inclusion of contextual information, the network

	ü	i	ɪ	e	ɛ	æ	ɔʸ	ɑʸ	ɑʷ	ɑ	ɔ	ʌ	o	ɝ	u	ʊ	Total
ü	60	10	14	1	1	1								3	8	1	77
i	7	83	6	3													267
ɪ	5	9	65	4	13	1						2		1			216
e		7	5	75	5	3		2				1		1			134
ɛ	1	1	12	12	37	16				1		11	3	6			158
æ	1	1	1	1	7	76		6	2	1		3					136
ɔʸ					5		68	16			5			5			19
ɑʸ			1	4	1	6	1	64	1	9	2	9	1	1			137
ɑʷ					2	10			58	12	4	6	10				52
ɑ						3		8	2	55	15	13	2	2			165
ɔ				1	1	1	2	1		17	64	4	7	1		1	139
ʌ			6		9	7		6		4		61	3	2		2	126
o			1	2					2	1	9	12	71	1	1		100
ɝ	4		1		1			1		1		1		90			82
u	27				3						9	3	9	9	39		33
ʊ	3	3	41		6						3	22	6		9	6	32

Table 4.2: Percent confusion table for Database IV when the synchrony envelopes, mean rate response and duration are available.

can account for coarticulation and produce /ɑʸ/ as the top choice.

4.4.1.2 Entropy

Figure 4.21 shows the entropy or the uncertainty of the vowel labels before and after the observation of the outputs of the network. As more and more sources of information are provided to the network, the entropy decreases and the transcription labels become less and less uncertain. For example, while the synchrony envelopes remove 50% of the initial entropy, all sources of information combined together remove about 65%.

4.5 Chapter Summary

In summary, this chapter describes a set of experiments that were designed to help us gain a better understanding of the different characteristics and representations of

	ü	i	ɪ	e	ɛ	æ	ɔʸ	ɑʸ	ɑʷ	ɑ	ɔ	ʌ	o	ɝ	u	ʊ	Total
ü	70	9	10	1								1		3	5		77
i	1	92	4	1	1												267
ɪ	1	5	78		10	1						1		1	1		216
e	1	9	4	76	5	1		1				1	1	1			134
ɛ		1	16	2	60	12	1	1				4	2	2			158
æ			4		7	79		4	1	2		3					136
ɔʸ			5				79	11						5			19
ɑʸ				2	1	1	1	84		6	1	3		1			137
ɑʷ						4		2	73	10		4	6	2			52
ɑ					1		1	5	1	79	9	3	1				165
ɔ	1					1	1	1	1	13	71	4	7				139
ʌ			7		11	3		6		5	2	63	2			1	126
o			1	1	4			2	1		6	7	75	2		1	100
ɝ	2		4			1		1				1		90			82
u	15	3									3	6	9	9	55		33
ʊ			16		13			3			3	16	3	3	3	41	32

Table 4.3: Percent confusion table for Database IV when all the information is available.

the network. It has been found that most of the learning can occur within the first several iterations through the training set. Increasing the size of the network and/or the number of training tokens may improve the performance of the network. The performance difference between the training data and test data can provide some indications. When the difference is relatively large, increasing the amount of training data may improve the performance. When the difference is relatively small, increasing the size of the network may improve the performance. Furthermore, as long as the number of hidden units is reasonably chosen, having more training data typically improves the performance.

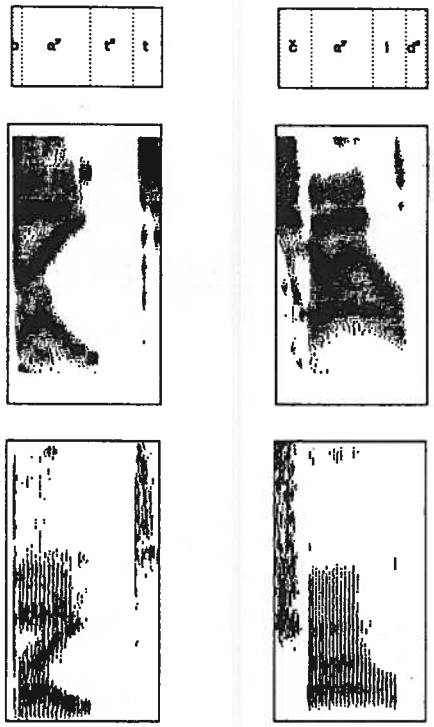
Networks with 0, 1, and 2 hidden layers have been examined. It has been found that performance results using 1 or 2 hidden layers are quite similar. Furthermore, experiments seem to suggest that the number of units in the hidden layer immediately before the output layer should be chosen to be at least $log_2 N_O$, where N_O is the number of units in the output layer. Due to restrictions on the decision regions formed, a SLP has been found to yield lower performance. Furthermore, it has been

found that using the nonlinear sigmoid function in the hidden and output layers can both improve the discriminating capability of the network.

Examination of the internal representations of the network suggests that the network can learn to pay attention to relevant linguistic information in the speech signal. It has also been found that with a sufficient number of hidden units, the connection vectors between the hidden and output layers are quite orthogonal to each other and need not be updated during training. Furthermore, experimental results suggest that perhaps the number of units in the hidden layer immediately after the input layer should be chosen to be at least the number of output classes, resulting, on the average, in one hyperplane for each class. Hierarchical clustering of the internal activations also suggests that the network can organize its internal representation in a way that seems to agree with our knowledge.

Exploration of alternative output representations shows that the network can extract the distinctive features quite reliably. Experiments also suggest that the network can be trained in successive stages and that the network can potentially use the outputs of feature detectors that may have been found to perform well.

Examination of input representations suggests that the network can integrate heterogeneous sources of acoustic and linguistic information. As more sources of information are available, the average agreement with the transcription labels improves while the entropy of the vowels after observing the network outputs decreases monotonically.



(a) (b)

Figure 4.20: Comparisons of two acoustic realizations of the vowel /ɑʸ/ in different phonetic contexts. The top panels correspond to the time-aligned transcriptions. The middle panels display the spectrograms obtained from the synchrony envelopes. The bottom panels display the spectrograms obtained using the FFT.

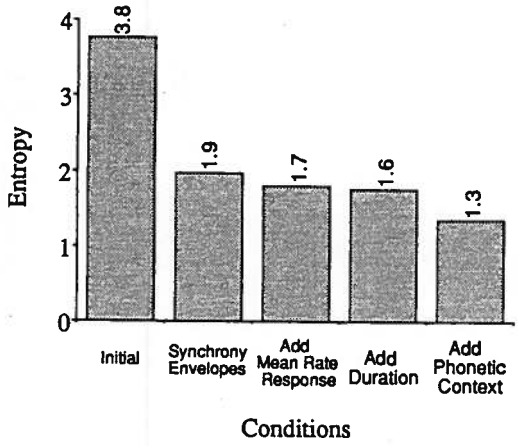


Figure 4.21: Entropy of the vowel ensemble when different amounts of information are available.

Chapter 5

Comparisons with Traditional Techniques

When applied to phonetic recognition, a pattern classifier can be evaluated in different ways. For example, the error rate or the average probability of error provides an indication of how well the classifier performs. For our particular task, it corresponds to the average agreement between the vowel labels provided by the transcription and the network. However, the error rate may actually depend on the amount of training data. Since different classifiers may have different levels of sensitivity to the amount of training data, evaluation or comparison of different classifiers needs to be made over a relatively wide range of amounts of training data. Furthermore, the practical use of a phonetic classifier could be limited by its complexity. If training is slow, a considerable amount of training time must be allowed before the classifier can be used in practice. If the actual classification process is too slow, the classifier may not be applicable to some tasks. Even with new advances in computer technology, a classification technique that requires a huge amount of memory may still be infeasible in many situations.

This chapter discusses and compares the relative merits and shortcomings of traditional classification techniques and the multi-layer perceptron (MLP). We will first discuss the use of traditional techniques in phonetic recognition and point out some

of the potential problems, which arise mostly due to our incomplete knowledge about the speech signal. We will then describe two experiments that compare the accuracy of the MLP with those of the traditional techniques. Complexity of the different techniques will also be discussed.

5.1 Traditional Techniques

One of the primary objectives in using traditional statistical pattern classifiers is to minimize the average probability of error [35,131]. From Bayes decision theory, a classifier determines that an input vector, $\mathbf{x}$, belongs to ω_j if

$$P(\omega_j \mid \mathbf{x}) > P(\omega_i \mid \mathbf{x}) \quad \forall i \neq j, \tag{5.1}$$

If the true underlying multi-dimensional probability functions in Equation 5.1 were known, then the resulting probability of error, $P(E)$, would be the minimum possible and is called the Bayes rate, P^*. However, the underlying probability functions are often not known for many practical problems. Until some valid functions or models are discovered, these underlying functions need to be estimated using some training data. Traditional techniques in estimating the underlying probability functions fall into two major categories: parametric and nonparametric. In parametric techniques, a specific form of the probability distribution is assumed, e.g. Gaussian. Thus the problem of estimating the entire probability function can be reduced to that of estimating relatively few parameters such as the mean or the covariance matrix. In nonparametric techniques, such as the Parzen window or k-nearest neighbor (KNN) classifier, assumptions about the form of the underlying distribution can be bypassed. By using a distance metric, the underlying distribution could be estimated directly from the training data.

5.1.1 Potential Problems

When these traditional techniques are applied to phonetic recognition, some specific knowledge about the characteristics of the speech signal is needed. For example, when a parametric technique is adopted, the form of the parametric model needs to be specified. Depending on the task, the model may yield high performance if it matches well with the true underlying distributions, but may lead to inferior results if the model is invalid. The use of a nonparametric technique has the advantage that such models do not need to be specified explicitly. Given a sufficient amount of training data, the generality and flexibility of such a technique can result in a reliable estimation of the underlying distribution. However, when the amount of data is limited, the capability of the nonparametric techniques is also limited and may depend on the specific choices of some variables, such as the distance metric, the local geometry in the feature space, or the size and shape of the window [45]. Unfortunately, the appropriate choices for these variables are not as yet well understood in phonetic recognition, due to our incomplete understanding of the speech communication process [72]. In other words, until we have a clearer understanding of phonetic encoding in the speech signal, using traditional techniques can potentially be problematic.

Even if the distance metric or the form of the probability model were valid, estimation of the parameters or the true probability functions can still be difficult, since the number of dimensions involved in phonetic recognition is usually quite large. For example, a feature space of d dimensions would require the estimation of a $d \times d$ covariance matrix when the Gaussian model is used.[1] The number of training tokens required for KNN to function reliably grows exponentially with the number of dimensions in the feature space [35]. In other words, a very large amount of training data is needed before the underlying probability functions can be estimated reliably.

[1] Due to symmetry, there are only $d(d+1)/2$ distinct parameters in the $d \times d$ covariance matrix.

5.1.2 Application of Speech Knowledge

Previous attempts have suggested that some of these potential problems can be alleviated by proper application of speech knowledge. By judiciously selecting relevant acoustic attributes based on our acoustic-phonetic knowledge, the classification procedures can potentially focus on the relevant linguistic information in the speech signal. As a result, the amount of training data needed to achieve robust estimation can be reduced. For example, the use of cepstral analysis or linear predictive coding reduces significantly the number of dimensions for vector quantization [15,16,109,118]. The careful selection of fifty attributes in the FEATURE system leads to successful recognition of the English alphabet [23]. Other examples include the vowel recognition approach proposed by Seneff [123]. In this approach, a relatively small set of "line-formants" are first extracted from the synchrony envelopes to represent the formant locations. In the SUMMIT system under development, a set of generic property detectors is first determined based on acoustic-phonetic knowledge. An optimization procedure is then used to select a subset of the attributes that can maximize the performance of the resulting classifier [146].

Although specific application of speech knowledge can reduce the number of dimensions and simplify the problem of not having sufficient training data, the classification procedure itself still often needs to resort to the traditional techniques. For example, a distance metric is needed to determine the codebook for vector quantization. A multivariate Gaussian distribution is used in the FEATURE system to model the unknown underlying probability distributions. A Parzen window associated with some speech heuristics is used to score the line-formants. In the SUMMIT system, an orthogonal feature space is first obtained by using principal component analysis. The Gaussian distribution is then used to model the distributions in the transformed feature space.

5.2 Comparisons: Accuracy

Conceivably, the classification result can be improved by applying techniques that do not make specific assumptions about the speech signal. As discussed in Chapter 3, some of the appealing characteristics of the MLP are that no probability distributions or distance metrics are assumed. Thus the MLP can potentially provide an effective mechanism for classification, until some appropriate distance metrics or probability models are discovered.

The following sections discuss the use of two traditional pattern classification techniques: (1) the k-nearest neighbor, a nonparametric technique that uses a distance metric to measure degree of similarity between two observations, and (2) the multivariate Gaussian density, a specific parametric probability model. Comparisons with MLP will also be presented.

5.2.1 K-Nearest Neighbor (KNN) Classification

The k-nearest neighbor rule classifies a test token, x, by assigning it the label most frequently represented among the k nearest tokens in the training set [33,35]. Thus the classification procedure involves three major steps. First, the distances between x and all the training tokens are computed. Second, the computed distances are then sorted to find the k nearest neighbors. Third, the labeling decision is made by a majority rule. Specifically,

$$p(\mathbf{x} \mid \omega_i) = \frac{k_i}{n_i V}, \tag{5.2}$$

and

$$P(\omega_i) = \frac{n_i}{n}, \tag{5.3}$$

where ω_i stands for the i^{th} decision class, $p(\mathbf{x} \mid \omega_i)$ stands for the conditional probability density of $\mathbf{x}$ given ω_i, V stands for the volume in the input feature space that captures the k nearest tokens from all possible classes, k_i stands for the number of training tokens in V from ω_i, n_i stands for the number of training tokens from ω_i, and n stands for the total number of training tokens. Combining Equations 5.2 and 5.3,

$$p(\mathbf{x}, \omega_i) = \frac{k_i}{nV}, \tag{5.4}$$

and

$$p(\mathbf{x}) = \frac{k}{nV}. \tag{5.5}$$

Therefore,

$$P(\omega_i \mid \mathbf{x}) = \frac{k_i}{k}. \tag{5.6}$$

Since k is a constant, Equation 5.6 provides the k-nearest neighbor rule and assigns $\mathbf{x}$ to ω_j if

$$k_j > k_i \quad \forall\, i \neq j. \tag{5.7}$$

5.2.1.1 Infinite Data

Asymptotic characteristics of KNN are relatively well understood. Specifically, when the total number of training samples, n, approaches infinity, upper bounds on the error rate, $\hat{P}(E)$, can be obtained. For the two-class case [27,35],

$$\hat{P}(E) = \sum_{i=0}^{\frac{(k-1)}{2}} \binom{k}{i} \left[(P^*)^{i+1}(1-P^*)^{k-i} + (P^*)^{k-i}(1-P^*)^{i+1} \right]. \tag{5.8}$$

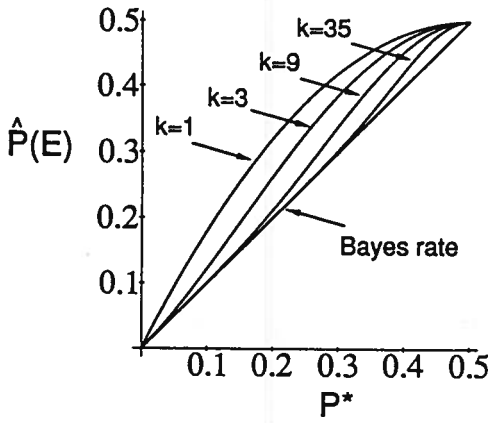


Figure 5.1: Upper bounds on the error rate for the k-nearest neighbor classifier when there are two classes.

Figure 5.1 shows the upper bounds, $\hat{P}(E)$, as a function of the Bayes rate, P^*. It can be seen that the upper bound gets progressively closer to the Bayes rate as k increases. In the limit as k goes to infinity, the upper bound approaches the lower bound. In other words, k-nearest neighbor can achieve the best performance possible when there is an infinite amount of training data.

5.2.1.2 Limited Data

In practice, it is very difficult, if not impossible, to have a very large amount of training data. In continuous speech recognition, for example, it may involve collecting many hours, or even days and months, of speech material before such asymptotic behavior can be approached. For a speaker-dependent recognition system, such a long training

period would severely limit the functionality of the system. Until a sufficiently large amount of data can be collected to reach the asymptotic performance, the problems due to a limited amount of data must be considered.

Unfortunately, characteristics of KNN with a limited amount of training data are still not well understood. For example, the choice of the distance metric can determine which k training samples should be used by the majority rule, thus affecting the overall performance of the classifier. Furthermore, a distance metric that is appropriate in one region in the feature space may not be appropriate in a different region. Figure 5.2 shows an example in which two different distance metrics in the same feature space may be needed. The circles and ellipses stand for the contours of constant probabilities. It can be seen that while the Euclidean distance can suffice to measure degree of similarity from Point A, a weighted Euclidean distance may be more appropriate at Point B.

The selection of k can also be an important factor. On the one hand, k should be chosen large enough to obtain a reliable estimation of the underlying probability. On the other hand, k should be chosen small enough to ensure that relevant variations of the underlying probability would not be smoothed out too severely. Nevertheless, theoretical analysis of asymptotic characteristics suggests that we can choose

$$k = \alpha\sqrt{n}, \tag{5.9}$$

where α is a positive constant [35]. While Equation 5.9 provides some insight, it does not uniquely determine how k should be chosen. Other questions, such as how close the performance can approach the Bayes rate or how well the performance improves as a function of the amount of training data, are still not completely answered. Nevertheless, Cover [26] speculated that the nearest neighbor rule

> *is probably a very good estimate of the best that any nonparametric decision rule may do in terms of the small sample. In other words, we feel that*

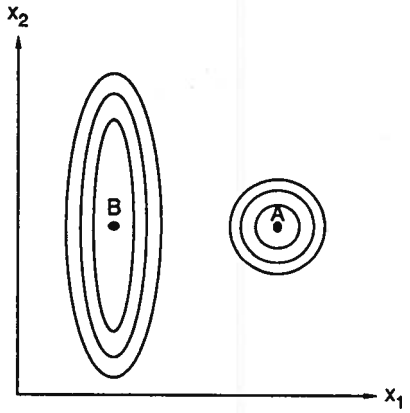


Figure 5.2: Different distance metrics may be needed at different local regions of the same feature space (see text). x_1 and x_2 stand for the dimensions of the feature space.

the failure of the NN [nearest neighbor] rule score to be near its limit is a good indication that every other decision rule based on the n samples will also be doomed to poor behavior. A small sample with respect to the NN rule is probably a smaller sample with respect to more complicated data processing rules.

Although the k-nearest neighbor rule suggests using the distances from x to the k nearest tokens from all the possible classes, previous experience has suggested that the performance can potentially be improved by explicitly specifying k_i in Equation 5.2 when only a limited amount of training data is available [117]. As a result, a different volume in the multi-dimensional feature space is used for each class, depending on the distribution of the training data. Specifically, Equation 5.2 becomes

$$p(x \mid \omega_i) = \frac{k_i}{n_i V_i}, \tag{5.10}$$

From Equations 5.3 and 5.10, we have

$$p(\omega_i \mid x) = \frac{k_i}{V_i} \frac{1}{np(x)}. \tag{5.11}$$

Thus the input vector, x, is assigned to ω_j if

$$\frac{k_j}{V_j} > \frac{k_i}{V_i} \quad \forall\, i \neq j. \tag{5.12}$$

Equation 5.12 provides a decision rule that assigns x to ω_j if the corresponding *a posteriori* probability is the maximum. By explicitly specifying k_i, the decision rule can potentially be more effective in estimating the a posteriori probability, $P(\omega_i \mid x)$. As we have discussed before, it is reasonable to choose k_i to be proportional to the square root of the number of training tokens from ω_i. Similarly to Equation 5.9, we have

$$k_i = \beta\sqrt{n_i}. \tag{5.13}$$

where β is a constant. V_i can be chosen to be the volume that captures the k_i training tokens from ω_i. If the Euclidean distance is used, V_i is the volume of a hypersphere:

$$V_i = \gamma(R_i)^d, \tag{5.14}$$

where

$$R_i = \max_l [r_l] \quad for \quad 1 \leq l \leq k_i, \tag{5.15}$$

γ is a constant, d is the number of dimensions in the feature space, r_l stands for the distance to the l^{th} neighbor, and R_i stands for the radius of the hypersphere and is the maximum distance from x to the k_i neighbors. Combining Equations 5.12 and 5.14, the decision rule in Equation 5.12 becomes

$$\frac{k_j}{(R_j)^d} > \frac{k_i}{(R_i)^d}. \tag{5.16}$$

In practice, when the number of training tokens is relatively small, R_i as specified by Equation 5.15 may become sensitive to the particular training tokens. The median of the distances to the k_i tokens has been found to be more effective [117]. Specifically,

$$R_i = median\,[r_l] \quad for \quad 1 \leq l \leq k_i. \tag{5.17}$$

5.2.2 Gaussian Classification

The multivariate Gaussian probability density function is often used to represent the true underlying probability density function. Thus the problem of estimating the

entire probability function can be reduced to that of estimating the mean vector and covariance matrix. Specifically, the conditional probability density is:

$$p_G(\mathbf{x} \mid \omega_i) = \frac{1}{2\pi^{\frac{d}{2}} \mid \Sigma_i \mid^{\frac{1}{2}}} exp\left[-\frac{1}{2}(\mathbf{x}-\mu_i)^t \Sigma_i^{-1}(\mathbf{x}-\mu_i)\right] \tag{5.18}$$

where $\mathbf{x}$ is a column vector with d dimensions, μ_i is the mean vector, and Σ_i is the $d \times d$ covariance matrix. The maximum likelihood estimates for μ_i and Σ_i are:

$$\hat{\mu}_i = \frac{1}{n_i}\sum_{k=1}^{n_i} \mathbf{x}_{ik} \tag{5.19}$$

and

$$\hat{\Sigma}_i = \frac{1}{n_i}\sum_{k=1}^{n_i} (\mathbf{x}_{ik}-\hat{\mu}_i)(\mathbf{x}_{ik}-\hat{\mu}_i)^t. \tag{5.20}$$

where $\mathbf{x}_{ik}$ stands for the k^{th} training token that belongs to ω_i, and n_i stands for the total number of training tokens in ω_i. A test token, $\mathbf{x}$, is assigned to ω_j if

$$p_G(\mathbf{x} \mid \omega_j)P(\omega_j) > p_G(\mathbf{x} \mid \omega_i)P(\omega_i) \quad \forall\, i \neq j, \tag{5.21}$$

where $P(\omega_i)$ is the *a priori* probability for ω_i.

5.2.3 Multi-Layer Perceptron: Infinite Data

While the characteristics of MLP's are not fully understood, recent study has demonstrated that with a sufficient number of hidden units, MLP's with one or two layers can approximate any continuous functions arbitrarily well [28,29]. Thus the network can potentially be used to represent the true underlying probability distribution functions of the data, resulting in optimal performance. However, many hidden units

may be needed to approximate the probability function arbitrarily well, resulting in very many connection weights in the network. As we have discussed in Chapter 4, a sufficient amount of training data is needed to estimate a set of connection weights robustly. Therefore, as the number of hidden units approaches infinity, an infinite amount of training data may be needed before the optimal performance can be achieved. Furthermore, even with an infinite amount of training data, it remains to be shown whether the training algorithm can indeed converge to a set of connection weights that approximates the true probability functions.

5.2.4 Experiments

This section compares the performance results of the KNN and Gaussian classifiers with that of the MLP. Unless otherwise specified, only the synchrony envelopes are used, resulting in input vectors of 100 dimensions. All the input vectors are obtained from the vowel tokens in Database IV. The network has one hidden layer of 32 units.

5.2.4.1 Comparisons with KNN Classification

The performance of the KNN was compared with that of the MLP using various amounts of training data. The decision rule and the radius of the hypersphere specified by Equations 5.16 and 5.17, respectively, were adopted. Not knowing what the most appropriate distance is, we used the Euclidean distance with the KNN. Since the most appropriate k_i is unknown, six different values were attempted. Specifically, for each training size,

$$k_i = \{1, 0.5\sqrt{n_i}, \sqrt{n_i}, 2\sqrt{n_i}, 5\sqrt{n_i}, 10\sqrt{n_i}\}. \tag{5.22}$$

Furthermore, since the performance of MLP may fluctuate for different random initializations, performance results of ten different networks were obtained for each training size, with each network randomly initialized.

Figure 5.3 compares the performance as a function of the number of training tokens. For simplicity, only 3 different values of k_i are shown: (*i*) $k_i = \sqrt{n_i}$, (*ii*) $k_i = 10\sqrt{n_i}$, and (*iii*) $k_i = 1$. Among the six different values of k_i used in this experiment, the performance result for the KNN was found to be the best when $k_i = \sqrt{n_i}$ and the lowest when $k_i = 1$. Each cluster of ten crosses in Figure 5.3 corresponds to the performance results of ten randomly initialized networks. Due to different initialization, a fluctuation of about 2% is observed even for the same training size. It can also be seen that up to 20,000 training tokens, the network consistently compares favorably to the KNN. It is possible that the network is able to effectively find more appropriate distance metrics than the Euclidean distance. However, it should be noted that while the performance of the KNN can potentially be improved with a more appropriate distance, that of the network can also be improved by using more hidden units, as suggested by Figure 4.3.

To further illustrate the importance of the choice of the distance metric for KNN, the network is compared with the KNN when all the heterogeneous sources of information shown in Figure 3.1 are available: Again, six different values for k_i are used according to Equation 5.22 and the Euclidean distance is used to measure the degree of similarity. Since the information sources are heterogeneous, the Euclidean distance is not expected to work well. As Figure 5.4 shows, although the performance results for both the MLP and KNN improve relative to those in Figure 5.3, the difference in performance between the two classifiers also becomes larger, again suggesting that the performance of KNN can indeed be affected by the choice of the distance metric.

These experiments indicate that when only a limited amount of training data is available, the MLP could yield higher performance than KNN. The higher performance of MLP is potentially due to the fact that no distance metrics need to be specified. As a result, the network has more flexibility to adapt to the data.

To illustrate the effectiveness of using different values of k_i for different classes, Figure 5.5 shows the performance results of the KNN using the decision rules in (i) Equation 5.7 with $k = \sqrt{n}$, and (ii) Equation 5.16 with $k_i = \sqrt{n_i}$. Only the synchrony

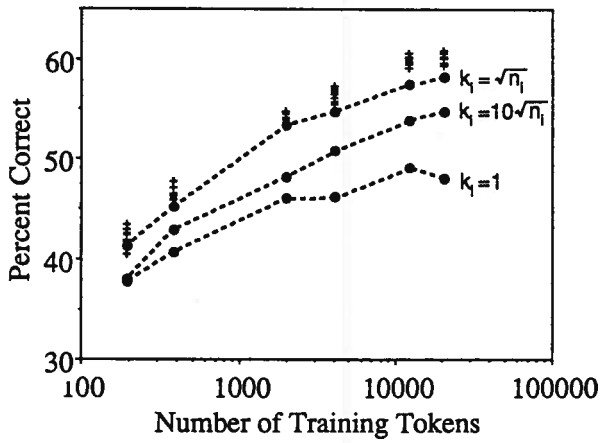


Figure 5.3: Comparison of the MLP with KNN using only the synchrony envelopes. The network has 1 hidden layer of 32 units. Each cluster of 10 crosses corresponds to the performance results of 10 randomly initialized networks.

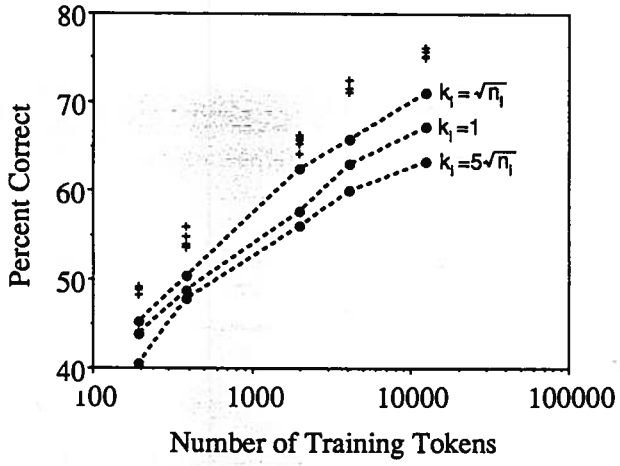


Figure 5.4: Comparison of the MLP with KNN using the synchrony envelopes, mean rate response, duration, and the phonetic contexts. The network has 1 hidden layer of 64 units.

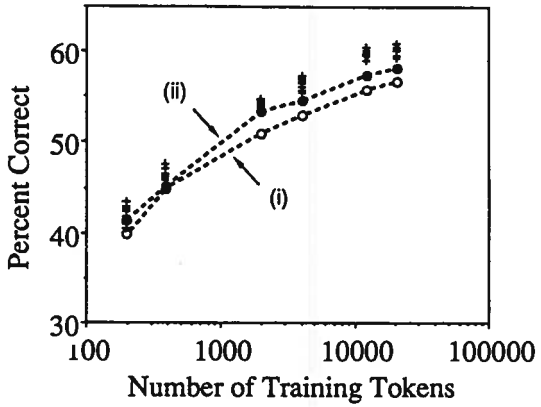


Figure 5.5: Comparison of two decision rules for KNN by (i) using only one value of k, and (ii) specifying different values of k_i for different classes. The performance results of MLP are also shown.

envelopes are available. We can see that the performance results can be improved by 1-3% when k_i is explicitly specified. For comparison, the performance results of MLP are also shown.

5.2.4.2 Comparisons with Gaussian Classification

The performance of the Gaussian classifier was also compared with that of the MLP using various amounts of training data. Figure 5.6 shows the performance as a function of the number of training tokens. Due to problems with singularity, the full covariance matrix was not used with less than 2,000 training tokens. For comparison, results using the diagonal matrix, which has non-zero elements only along the diagonal, are also shown. It can be seen that the network compares favorably to the Gaussian classifiers, suggesting that either the Gaussian assumption is invalid or that

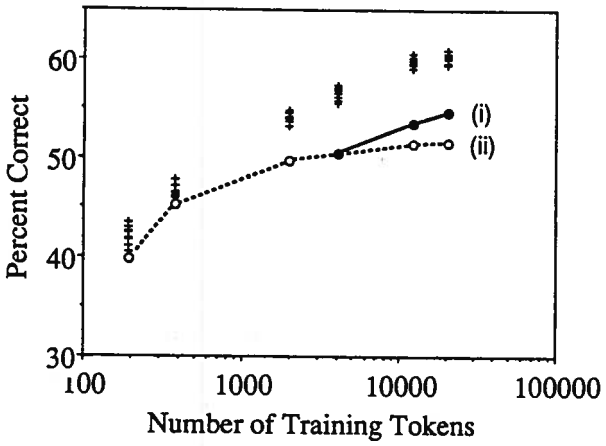


Figure 5.6: Comparison with the Gaussian classification using (i) the full covariance matrix, and (ii) the diagonal covariance matrix. Only the synchrony envelopes are available. The network has 1 hidden layer of 32 units.

significantly more training data are needed before the parameters in the Gaussian model can be robustly estimated. However, it is possible that the performance of the parametric approach can be improved if a more valid probability model is found.

5.3 Comparisons: Complexity

This section compares the complexity of the two traditional techniques with that of the MLP. Specifically, the required amount of computation and memory for training and recognition are compared. All calculations assume that only the synchrony envelopes are available and the task is to recognize the 16 vowels in Database IV. In other words, the input space has 100 dimensions and the training set has 20,000

tokens. The MLP has 1 hidden layer of 32 hidden units, whereas the KNN classifier performs a linear search over the entire set of training tokens.[2] Comparisons of computation and memory requirements for the three different techniques are summarized in Table 5.1.

The amounts of computation required for training are quite different for the three classification techniques. Back-propagation adopts a gradient descent approach and requires a significant amount of time for training. As we have seen in Figure 4.1, approximately 10 iterations through the training set of 20,000 tokens are needed before the performance reaches an asymptote, requiring back-propagation about 200,000 times. From the procedures outlined in Chapter 1, each training token requires approximately 12,000 multiplications and 48 nonlinear sigmoid operations in each iteration, for a total of about 2×10^9 multiplications and 10^7 nonlinear operations. For comparison, estimation of the covariance matrices in the multivariate Gaussian distributions requires approximately 10^8 multiplications. Additional computation is then needed to obtain the determinants and inverses of the covariance matrices. Thus training the MLP requires about 20 times as much computation as training the Gaussian classifier. The KNN classifier, on the other hand, requires no training, and thus no computation is needed.

The amounts of computation required for recognition are also quite different for the three different techniques. For each test token, KNN needs to compute the Euclidean distances to all 20,000 training tokens, requiring about 2×10^6 multiplications and 2×10^6 additions. The 20,000 computed distances are then sorted and the majority rule is applied. Each Gaussian model requires approximately 10^4 multiplications for each test token, thus representing a total of 1.6×10^5 multiplications for classification. The MLP, on the other hand, requires about 4×10^3 multiplications and 48 nonlinear operations. Thus if KNN is used with a linear search, it requires at least three orders of magnitude more computation than MLP does, while the Gaussian classifiers requires about 40 times more computation than MLP.

[2] Algorithms for reducing the amount of computation for KNN have been suggested [6,7,46].

	Multiplications for training	Multiplications for testing	Numbers for storage
KNN	0	2×10^6	2×10^6
Gaussian	10^6	1.6×10^5	8×10^4
MLP	2×10^9	4×10^3	4×10^3

Table 5.1: Complexity comparison of the Gaussian, KNN, and MLP classifiers.

Memory requirement for the KNN is the most significant, since all the 2×10^6 vector components in the training set need to be stored. While the Gaussian models have about 8×10^4 distinct parameters in the 16 covariance matrices and mean vectors, the MLP has about 4×10^3 connection weights. Thus space requirement for the KNN is about 500 times as much as that for the MLP, and that for the Gaussian classifiers is about 20 times more than that for the MLP.

5.3.1 Specific Implementation

Both MLP and KNN were simulated on the Symbolics Lisp Machine with an FPS array processor. The back-propagation algorithm and computation of the Euclidean distances for KNN were both performed on the FPS. Since the memory of the FPS is limited, data and results need to be transferred between the two machines. For our specific simulation, while training the network with 20,000 tokens for 10 iterations requires about 2 hours, testing the network with 2,000 tokens takes less than 30 seconds. On the other hand, while KNN does not need any training, testing with the 2,000 tokens requires almost one week of computation time. The Gaussian classifier was simulated on the Lisp Machine. While training the classifier requires about 10 hours, testing takes only a few minutes.

5.4 Discussion

Since the recent resurgence of interest in ANN's, a great deal of research efforts has been directed to the area of applying MLP to pattern classification. Its capability to form complex decision regions, to approximate any continuous functions, and to generalize to new test data, along with its relatively high computational speed led to the speculation that the MLP can potentially be a powerful computational paradigm for pattern classification. However, due to the lack of a thorough understanding of the characteristics and capability of the network to perform classification, it has remained unclear whether or why such a paradigm can yield respectable performance when dealing with a relatively difficult task. In Chapter 3, we have shown that the task of recognizing vowels excised from continuous speech independent of speaker is quite difficult. In this chapter, we speculate that the flexibility of the framework can potentially enable the network to be more effective than traditional techniques in adapting to the data. Our experimental evidence suggests that over a relatively wide range of amounts of training data, it is possible for the performance of the network to surpass that of traditional techniques. Its improved performance over that of KNN is particularly interesting since the KNN is thought to be the traditional algorithm that is most similar to the MLP [90], and its asymptotic performance is known to be optimal [35]. Comparison of complexity shows that although MLP requires substantial computational power for training, its requirement for computational power and memory space for actual classification is the least among the three pattern classification techniques. Although the choice of one technique over the other may very well depend on the specific application, the need for high computational power and space during classification can often be prohibitive.

While the experiments suggest that the MLP can yield higher performance than the k-nearest neighbor and Gaussian classifiers for our specific task of vowel recognition, they have not demonstrated that the same result will apply to any other tasks. Furthermore, performance of all techniques can potentially be improved by gaining

a better understanding of the specific problem. Discovering and quantifying the circumstances under which one technique can achieve better performance than the other is an area that deserves a great deal of research effort.

5.5 Chapter Summary

In summary, this chapter discusses and compares the use of the MLP with traditional parametric and nonparametric techniques for phonetic recognition. Parametric techniques assume specific forms for the underlying probability functions. Depending on the specific task, such techniques can yield high performance, but may lead to inferior results if the model is invalid. Nonparametric techniques can estimate the underlying functions directly without any specifications about the forms for the underlying functions. However, when the amount of training data is limited, its performance can depend on a number of factors. Until we have a clearer understanding of the speech signal, making assumptions about the form of the underlying functions or the distance metrics can potentially lead to problems.

The fact that the MLP does not need to make any assumptions about the underlying distribution functions or distance metrics can potentially enable the network to adapt more effectively to the data. Experiments demonstrate that when the amount of training data is limited, it is possible that the network can yield better performance than the parametric and nonparametric techniques such as the Gaussian density and the KNN. Complexity of the three classification techniques is quite different. Although the MLP requires the most amount of computation for training, its requirements during actual classification are quite low. By developing and applying efficient algorithms, the complexity of the classification techniques can be reduced.

Chapter 6

Refinements

This chapter discusses some investigations into improving the characteristics of the multi-layer perceptron (MLP). We will first examine the error criterion for training the network and suggest a specific error measure that can be more effective for pattern classification. We will also discuss the importance of the initial state of the network and suggest initialization procedures that can improve the performance, as well as the training and adaptation characteristics of the network. Experimental evaluations of the different procedures will also be presented.

6.1 Weighted Mean Squared Error

Although the MLP does not need to make assumptions about a distance metric or the form of an underlying probability function, the error criterion for training the network needs to be specified. Thus the characteristics and capability of the network can be affected by the choice of the error criterion. Furthermore, different criteria may be needed for different tasks.

As mentioned in Chapter 1, the supervised training procedure for the MLP involves the presentation of pairs of input and target output vectors. An error signal is then generated by comparing the difference between the target output vector and

the actual output vector of the network. A gradient descent approach is adopted and the connection weights are updated to minimize the error signal. The mean squared error (MSE) between the actual and target output vectors, E, is often adopted as the error signal [111]. Thus

$$E = \frac{1}{2}\sum_j (t_j - y_j)^2 \tag{6.1}$$

where t_j stands for the target value of the j^{th} output unit, and y_j stands for the corresponding actual output value.

When the network is used as a hetero-associator to associate pairs of vectors, adopting such an error measure corresponds to searching for a set of connection weights that minimizes the mean squared error between the vectors. Once the network is well-trained, the presentation of an input vector will produce an output vector that is similar to the target output vector. When the network is used as a pattern classifier, if a training token belongs to the i^{th} class, t_i can be set to a high value while t_j can be set to a low value $\forall j \neq i$. Thus the network is trained to associate the input vectors with a set of binary output vectors. However, Equation 6.1 does not explicitly consider the classification rank-order statistics. The same numerical error can be obtained if the output value of the i^{th} output unit is the second highest or the tenth highest among all the output units. Therefore when the network is used as a classifier, a different error measure may be needed to more explicitly account for the rank-order statistics.

To improve the classification characteristics of the network, a weighted mean squared error (WMSE) measure, $\tilde{E}$, can be used, where the weights are directly determined by the classification performance. Specifically, let

$$\tilde{E} = \frac{1}{2}\sum_j W_j (t_j - y_j)^2, \tag{6.2}$$

where

$$W_j = \begin{cases} (1 + r\epsilon)^2 & j = i \\ 1 & j \neq i, \end{cases} \tag{6.3}$$

i stands for the class to which the training token belongs, r stands for the rank of the actual output of unit i among all other output units, and ϵ is a small non-negative constant. If ϵ is set to zero, all the weights in Equation 6.3 are equal to one, resulting in MSE. Assuming W_j independent of w_{ji}, Equation 1.12 becomes

$$\delta_j = \begin{cases} \frac{dy_j}{dz_j} W_j (t_j - y_j) & j \in O \\ \frac{dy_j}{dz_j} \sum_k \delta_k w_{kj} & j \in H. \end{cases} \tag{6.4}$$

It can be seen from Equation 6.4 that the error signal for the j^{th} output unit is proportional to the weighting factor, W_j. If the output value of the unit associated with the correct answer is low relative to other units, then a large weighting factor will be given to that unit. Thus using WMSE can potentially enable the network to pay more attention to classification errors during training. Evaluations of the WMSE will be presented in Section 6.3.

6.2 Initialization

6.2.1 Potential Problems with Random Initialization

Besides the error measure, the initial connection weights can also affect the capability of the network. For example, if the set of initial connection weights happens to be at a local minimum of the high-dimensional error surface, then the gradient descent training procedure that tries to minimize the error will be stuck, resulting in the connection weights to remain unchanged. Unfortunately, the precise relationship

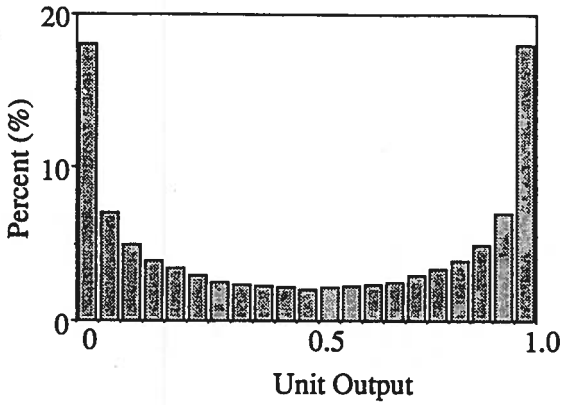


Figure 6.1: Distribution of the outputs of the basic units in the network after random initialization.

between the error measure, the connection weights, and the location of the local minima is not yet fully understood. Without any *a priori* knowledge, the connection weights are often randomly initialized.

One of the potential problems with random initialization of a network is related to the nonlinear sigmoid function.[1] Since the transition region of the sigmoid function is relatively narrow while the saturation regions are relatively wide, randomly initializing the network can potentially cause most of the basic units to operate in the saturation regions of the sigmoid function. For example, Figure 6.1 shows a histogram of the output values of the basic units after the connection weights are randomly initialized.[2] Comparing Figures 1.2 and 6.1, we can see that when the connections are random, a basic unit is more likely to operate in the saturation regions of the sigmoid function.

However, having the basic units initially operate in the saturation regions is undesirable. As Equations 1.12, 1.13, and 6.4 show, the error signal is relatively large when $y_j = 0.5$, but becomes progressively smaller as y_j departs away from 0.5. From these equations and Equation 1.15, we can see that a basic unit can learn faster when it is operating in the transition region than when it is operating in the saturation regions. Furthermore, if the magnitude of the initial input to a basic unit, z_j, is very large, the basic unit may stay operating in the saturation region for many training iterations, thus reducing the learning capability of the network. Therefore, random initialization of the connection weights can potentially decrease the learning speed and the performance of the network.

[1] Symmetry in the connection weights is another potential problem [111].
[2] The magnitudes of the initial random connections are all less than 2.0.

To gain a better understanding of the important factors for initialization, let each of the initial connection weights be randomly generated such that the expected value is zero and the variance is a constant:

$$E[w_{ij}] = 0 \qquad and \qquad \sigma^2_{w_{ij}} = \sigma^2_w. \tag{6.5}$$

Furthermore, assume the random weights are uncorrelated so that

$$E[w_{ij}w_{ik}] = E[w_{ij}]E[w_{ik}]. \tag{6.6}$$

From Equation 1.3,

$$\begin{aligned} E[z_i] &= E[\sum_j w_{ij}x_j] \\ &= \sum_j x_j E[w_{ij}] \\ &= 0. \end{aligned} \tag{6.7}$$

Therefore,

$$
\begin{aligned}
\sigma_{z_i}^2 &= E[z_i^2] \\
&= E[\sum_j \sum_k w_{ij} w_{ik} x_j x_k] \\
&= \sum_j x_j^2 E[w_{ij}^2] + \sum_{k, j \neq k} x_j x_k E[w_{ij}] E[w_{ik}] \\
&= \sigma_w^2 \sum_j x_j^2.
\end{aligned} \tag{6.8}
$$

From Equations 6.7 and 6.8, it can be seen that although the expected value of the input to the sigmoid of a basic unit, $E[z_i]$, is zero, the variance, $\sigma_{z_i}^2$ depends on the variance of the the random weights, σ_w^2, as well as the magnitudes and the number of dimensions of the input vectors. If $\sigma_{z_i}^2$ is large, many basic units may operate in the saturation regions.

Conceivably, the learning speed and performance of the network can be improved by reducing $\sigma_{z_i}^2$. In fact, previous applications of the MLP often initialize the network with random weights of small magnitudes, i.e. small σ_w^2 [111,116]. If the input values are not too large, $\sigma_{z_i}^2$ will be small, thus resulting in most of the basic units operating in or *near* the transition region. Furthermore, Burr suggests biasing the inputs to the network [13]. By subtracting the average value of the inputs over the entire training set, $\sigma_{z_i}^2$ can be reduced.

6.2.2 Center Initialization

Although initializing with small random weights or biasing the inputs can reduce $\sigma_{z_i}^2$, there is no guarantee that all the basic units are indeed operating in the transition region. The following initialization procedure is adopted to ensure that all the basic units operate initially at the center of the sigmoid function, i.e.

$$
z_i = 0 \quad or \quad y_i = 0.5 \quad \forall\, i \in (H \cup O), \tag{6.9}
$$

where H and O are the sets of hidden and output units, respectively.

First, the connections between the input and the first hidden layer are initialized with zero weights. Thus inputs to the sigmoids of the units in the first hidden layer are all zero, resulting in the corresponding outputs being 0.5. Second, the connection weights between the remaining consecutive layers are initialized with pairs of random numbers that differ only in their signs as shown in Figure 6.2. For simplicity, this figure shows a network with only one hidden layer. Specifically,

$$
w_{ij} = \begin{cases} 0 & j \in I \\ r_{ij} & j \in H \quad \& \quad j = 2k \\ -r_{i(j-1)} & j \in H \quad \& \quad j = 2k-1 \end{cases} \tag{6.10}
$$

where

$$
0 \leq k \leq \frac{N_H}{2}, \tag{6.11}
$$

I is the set of input units, and r_{ij} is a random number. If the number of units in each hidden layer is even, then all the hidden and output units in the network operate right at the center of the sigmoid function, where learning is the fastest. We call this procedure center initialization (CI).[3] Such an initialization procedure ensures that each basic unit can learn quickly as soon as the training procedure begins. Therefore, both the training time and performance of the network should be improved. Evaluations of CI will be presented in Section 6.3.

6.2.3 Speaker Adaptation by Transfer Initialization

Although the use of center initialization can potentially improve the performance of the network, the initial connection weights are nevertheless random and do not con-

[3] If the sigmoid function is shifted so that $S(0) = 0$, center initialization can be achieved by simply setting the connections between the input and first hidden layers to zero.

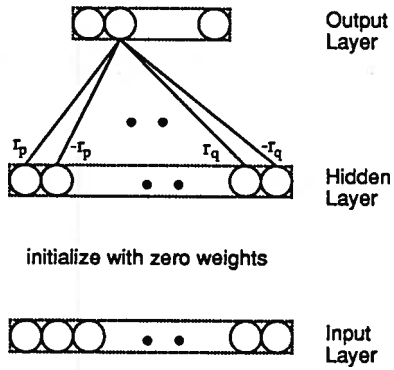


Figure 6.2: Center Initialization: initialization of the network to ensure each basic unit is initially at the center of the sigmoid function.

tain any linguistic knowledge. If there is some *a priori* knowledge about what the connections should be, then incorporating such knowledge into the initialization or training procedure may further improve the performance of the network. In this section, an initialization procedure is proposed to adapt a speaker-independent network to a new speaker, using some *a priori* knowledge about the connection weights.

Since a speaker-dependent phonetic recognizer does not need to deal with across-speaker variations, its performance on the speaker that it is well-trained is typically higher than that of a speaker-independent one. However, to achieve reliable performance, a speaker-dependent recognizer often needs to be trained with a large amount of data from the speaker, which can severely limit the practical use of the recognizer. When only a very limited amount of data from the speaker is available, its performance degrades and can be lower than that of a speaker-independent recognizer.

Conceivably, when the amount of training data from a new speaker is very limited, the performance could be improved by adapting a speaker-independent recognizer to the new speaker. Thus if a speaker-independent network can indeed capture and store relevant linguistic knowledge about the speech signal in its connection weights, then such knowledge can be transferred to a speaker-adaptation network by simply initializing the network with the connection weights of a well-trained speaker-independent network as shown in Figure 6.3. With some a priori knowledge about what the connection weights should be when the amount of training data is limited, training such a speaker-adaptation network can potentially achieve higher performance than training a randomly initialized speaker-dependent network. Evaluation of this transfer initialization procedure will be presented in the next section.

6.3 Evaluations

This section discusses several experiments conducted to evaluate the effectiveness of the different techniques suggested in this chapter. Only the synchrony envelopes were used, and the network has one hidden layer of 32 units.

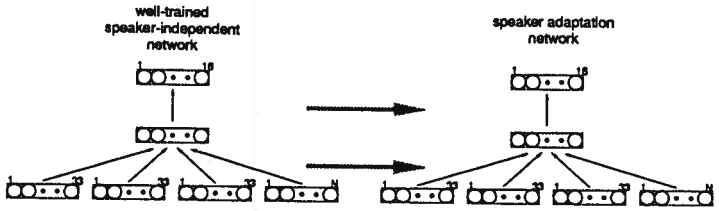


Figure 6.3: Initialization of a speaker-adaptation network by transferring connection weights from a well-trained speaker-independent network.

6.3.1 Weighted Mean Squared Error and Center Initialization

The WMSE and CI were evaluated using the speech material in Database IV.[4] First, the training characteristics of the network were examined. Figure 6.4 shows the performance of the network as a function of the number of training iterations, using (i) MSE, (ii) WMSE, and (iii) CI with WMSE. We can see that both WMSE and CI can improve the training time of the network.[5] For example, to achieve 55% accuracy, MSE requires about 11 training iterations, while WMSE requires 4 iterations, and CI with WMSE requires only 1 iteration. Furthermore, the asymptotic performance using CI and WMSE is slightly higher than using only WMSE, which is also slightly higher than using MSE.

As Figure 4.2 suggests, the asymptotic performance is a function of the number of training tokens available. Figure 6.5 compares the performance as a function of the number of training tokens. Each point in this figure corresponds to the average result of ten networks, each one randomly initialized. We can see that using CI and WMSE can improve the performance of the network. For example, with 200 training tokens,

[4]Unless otherwise specified, all the 20,000 training tokens were used to train the network.
[5]ϵ in Equation 6.3 is chosen to be 0.2.

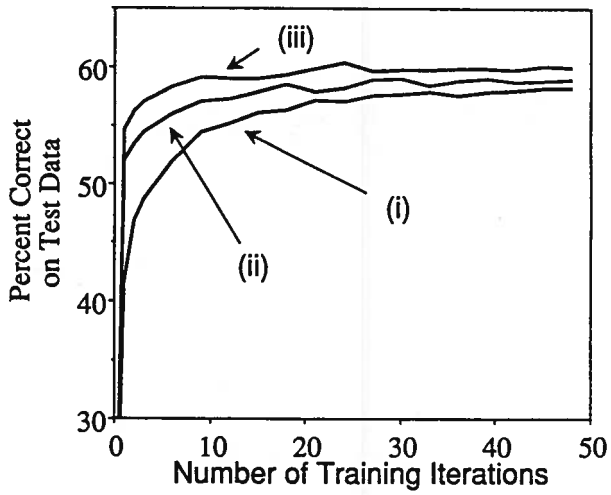


Figure 6.4: Training characteristics of the network using (i) mean squared error (MSE), (ii) weighted mean squared error (WMSE), and (iii) center initialization (CI) with WMSE.

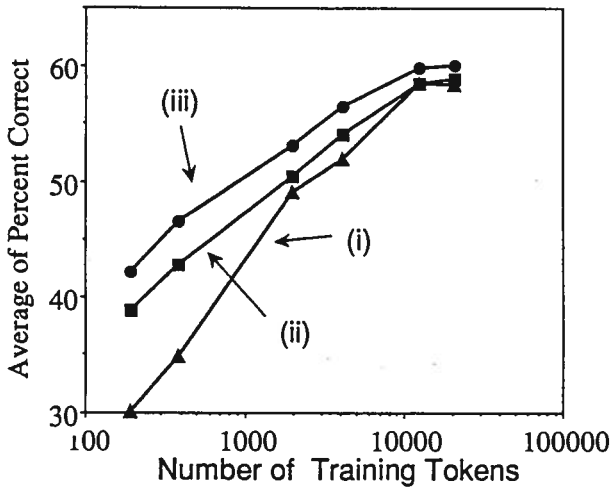


Figure 6.5: Performance of the network using (i) mean squared error (MSE), (ii) weighted mean squared error (WMSE), and (iii) center initialization (CI) with WMSE. Each point is the average performance of ten networks, each one randomly initialized.

the performance using CI and WMSE is about 42%, while that using only WMSE is about 3% lower, and that using MSE is about 12% lower. We can also see that as the number of training tokens increases, the performance differences also decrease.

Since the connection weights are randomly initialized, the performance result of the network after training may fluctuate. In order to examine the stability of the network performance, the standard deviations of the performance of the networks using the three techniques were measured, with the results shown in Figure 6.6. It can be seen that independent of the number of training tokens, the performance results using CI and the WMSE are quite stable, with the standard deviation at or

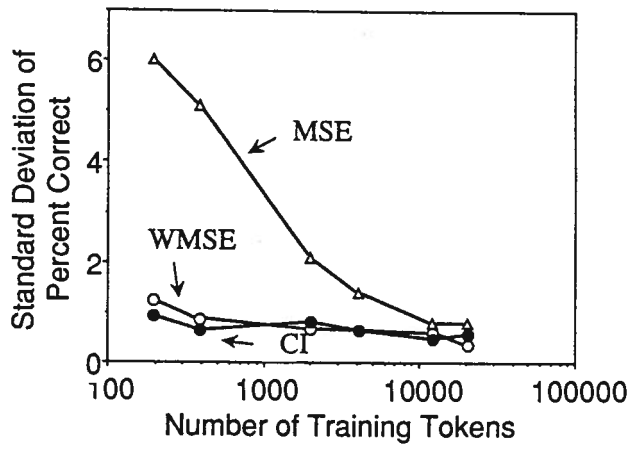


Figure 6.6: Reliability of the network performance using (i) MSE, (ii) WMSE, and (iii) CI with WMSE. Each point is the standard deviation of the performance results of ten networks, each one randomly initialized.

below 1%. However, the performance results using MSE fluctuates quite significantly when the number of training tokens is limited. For example, the standard deviation is about 6% when only 200 training tokens are available. However, as the number of training tokens increases, the performance fluctuation decreases quite rapidly. With over 10,000 training tokens, the reliability of the three different techniques is quite similar.

These results collectively suggest that both CI and WMSE can improve the training time, performance, and the reliability of the performance of the network. As a consequence, all the experiments described in all chapters in this thesis have adopted CI and WMSE.

6.3.2 Transfer Initialization

The application of transfer initialization to speaker adaptation was evaluated using the speech material in Databases III and IV. Note that the speech data in Database III were recorded using an omni-directional microphone suspended approximately 10 inches from the speaker's mouth, whereas those in Database IV were recorded using a close-talking noise canceling microphone. Furthermore, the speaker in Database III is not a speaker for Database IV. Thus the speaker adaptation network needs to deal with variations in recording conditions and speaking characteristics. Figure 6.7 compares the performance results on Database III using random initialization, center initialization, and transfer initialization from a speaker-independent network well-trained on Database IV. The WMSE is adopted in all cases. We can see that the performance on the new speaker in Database III using the speaker-independent network is initially about 47% (Point A in Figure 6.7). In this figure, this result is connected with a dashed line to the performance result of the speaker-adaptation network. At about 100 training tokens or approximately 6 training tokens for each vowel, the performance using transfer initialization is about 53%, which is 6% higher than using center initialization, and about 11% higher than using random initialization. However, as the number of training tokens is increased, the performance differences also decrease.

6.4 Chapter Summary

In summary, this chapter has suggested some procedures that can improve the performance and training time of the network. Specifically, the use of a weighted mean squared error can more explicitly account for the classification performance of the network. The use of center initialization ensures that all the basic units will learn quickly once the training procedure starts. Examination of the training characteristics and performance results shows that both the weighted mean squared error and

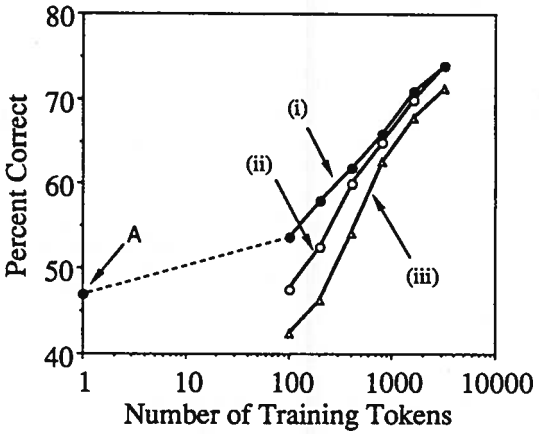


Figure 6.7: Performance of the network on Database III using (i) transfer initialization, (ii) center initialization, and (iii) random initialization. Point A corresponds to the performance of the speaker-independent network.

center initialization are effective in improving the training time, performance, and reliability of the network.

Rapid speaker adaptation has been studied in light of incorporating some *a priori* knowledge into the network. By initializing a speaker-adaptation network with the connection weights of a well-trained speaker-independent network, the performance of the speaker-adaptation network can be improved, especially when the amount of training data is limited.

Chapter 7

Discussion

In this thesis, an investigation into the use of artificial neural networks (ANN's) for phonetic recognition has been presented. The motivation for our work stems from the observation that although a great deal of acoustic-phonetic knowledge has been gained over the past few decades, our understanding of the detailed encoding process of linguistic information in the speech signal is still quite limited. One of the current major problems in phonetic recognition is in finding a suitable framework in which our acoustic-phonetic knowledge can be utilized effectively and naturally, and efficient control strategies for decoding the speech signal can be generated automatically. Due to their flexible frameworks, ANN's can potentially provide effective mechanisms to make use of what we have learned, and model what we have not.

With this motivation, our work was pursued in different dimensions, often guided by our acoustic-phonetic knowledge. First, a network that may be well-suited for phonetic recognition was selected. An appropriate network needs to be flexible for incorporating our speech knowledge, and capable for developing powerful control strategies. Second, basic network characteristics and capabilities, as well as different representations, were investigated. A better understanding of these issues can potentially enable us to exploit the network more fully as a pattern classifier. Third, the network and traditional pattern classification techniques were evaluated and compared. Such comparisons can enable us to gain a better understanding of the relative

advantages and constraints of the different techniques. Finally, the basic limitations of the network were examined, and procedures for improving the characteristics of the network were suggested. The following sections summarize our explorations and discuss some of the issues that have been raised.

7.1 Network Selection and Performance Evaluations

Although most ANN's offer parallel and self-organizing mechanisms, the multi-layer perceptron (MLP) has a number of characteristics that can potentially make it well-suited for phonetic recognition. Since the MLP does not need to assume any specific probability distributions or distance metrics, it may be more effective in adapting to the characteristics of the speech signal. This property, together with the fact that it can take on continuous and/or discrete inputs, may enable it to integrate heterogeneous sources of information in the speech signal. Other appealing characteristics for phonetic recognition include its effectiveness for class discrimination, flexibility in forming disjoint decision regions without supervision, and capability in performing hetero-associative transformations.

As an initial step toward understanding the basic issues in applying MLP to phonetic recognition, our work is constrained to the task of classifying the vowels in American English independent of speaker. As perceptual experiments suggest, this restricted task is quite interesting and non-trivial. Most of the work described in this thesis is based on the TIMIT acoustic-phonetic database, which covers a wide range of dialectical variations. In order to gain a better understanding of the potential capability of the network, the performance of the network on this database was evaluated in different ways. Evaluation in terms of average agreement with the phonetic transcription suggests that the performance of the network compares favorably to human performance. Evaluation along the information theoretic dimension indicates that the network can remove about two-thirds of the uncertainty in the vowel labels.

When evaluated along the phonological dimension, it was found that most of the confusions between the network and transcription labels are quite reasonable, with most of the confused vowel labels differing by two or fewer distinctive features.

In order to study the influence on the performance of the network due to different sources of variability in the speech signal, different databases were employed for our study. As expected, a substantial difference in performance is observed under different conditions, depending on whether the task is speaker-independent, whether the phonetic context is constrained, and whether the speech material is spoken continuously.

7.2 Input Representations

Motivated by the belief that it is important to incorporate the constraints provided by the human auditory system, we used the outputs of the auditory model proposed by Seneff for our signal representations [122]. Specifically, we use the synchrony envelopes and the mean rate response, which have been shown to enhance the formant information and the temporal aspects of the speech signal, respectively. In order to capture dynamic spectral information and reduce the amount of computation, the vowel tokens are represented by three averaged spectra, computed from the left, middle, and right regions of the vowel token. Additional sources of information such as duration and phonetic contexts can also be made available to the network. Our results suggest that the network can effectively integrate heterogeneous sources of information, which can be in continuous and/or discrete forms. Furthermore, performance consistently improves as more sources of information are available.

7.2.1 Acoustic Representations

The use of the three averaged spectra is motivated by the observation that some vowels such as the diphthongs would require a representation that captures the time-

varying information in the speech signal. However, the acoustic realization of a vowel can be drastically affected by contextual variations, especially near the beginning or the end of the vowel. Since the middle part of a vowel token is more likely to be able to reach its articulatory target position, it may be reasonable to suspect that effects due to contextual variations can be reduced by making measurements only in the middle of the vowel.

However, although measurements made near the beginning or the end of a vowel token are subject to more variations, they may nevertheless provide *some* information. If all three averaged spectra are available, a graceful pattern classifier should be able to extract the relevant information provided by the middle spectrum, as well as the additional information that the other spectra may provide. Therefore, it is possible that the performance of the network using the three spectra can be higher than using only the middle spectrum.

In order to gain a better understanding, two experiments were performed with only the middle averaged spectrum available to the network, using the synchrony envelope representation and the speech material in Database IV. In the first experiment, a network was trained to recognize the 16 vowels. Since time-varying information of the diphthongs is not captured, the performance is about 7% lower than using all three averaged spectra. In the second experiment, a network was trained to recognize only the monophthongs. The resulting performance is 58%, which is 5% lower than that when using all three averaged spectra. Thus these experiments suggest that the additional information in the left and right regions can indeed improve the overall performance.

Although it is reasonable to use signal representations that incorporate human auditory constraints, its effectiveness for machine recognition of speech needs to be justified with experimental evidence. Previous comparisons have shown that acoustic segmentation of the speech signal can be achieved more reliably using the mean rate response than more standard representations such as DFT or LPC-based spectral representations [51]. Results have also suggested that the auditory representation is

relatively robust in the presence of noise [49,66]. However, it is still not very clear under what circumstances or in what recognition paradigms the auditory representation can lead to higher performance than standard signal representations. Future work on the use of the auditory representations for phonetic recognition should include systematic comparisons with other signal representations.

In addition to examining the appropriateness of the signal representations, another interesting dimension for future pursuit is investigating the use of other forms of acoustic attributes. Although the choice of raw spectra as inputs is reasonable, using more sophisticated acoustic cues can potentially permit further acoustic-phonetic knowledge to be incorporated. For example, voicing information for an intervocalic stop may be encoded in six different acoustic cues such as the intensity of the burst, fundamental frequency contour, and the duration of the preceding vowel [93]. While these acoustic attributes have been identified, relatively little is known about how they should be integrated to form a final decision. It would be of great interest to study how the network can be used to extract these acoustic cues or to provide a control strategy for integrating these cues.

7.2.2 Contextual Representations

As discussed in Chapter 4, the contextual information is represented in terms of the adjacent phonetic labels. Although the use of such a representation can improve the performance of the network, representations in terms of the distinctive features can potentially allow the contextual variability to be accounted for in a more natural manner [129]. In many situations, contextual variations can be explained by means of distinctive features. For example, a coronal obstruent is often pronounced with a more palatal place of articulation when it is followed by a palatal phoneme. An initial stop often takes on the retroflex feature when it is followed by a retroflex phoneme. Thus the use of the distinctive features can potentially make the acoustic-phonetic constraints more explicit.

Another possible advantage of representing the contextual information in terms of distinctive features is that the number of connection weights can be reduced. In the experiment described in Chapter 4, the contextual information was represented in a unary form. Since there are 61 possible phonetic labels in our databases, incorporating left or right phonetic context requires addition of 61 input units to the network. Such a representation is arbitrary and results in a substantial increase in the number of connection weights. By employing distinctive features, it is possible that a relatively efficient coding scheme for the phonetic contexts can be obtained. If the feature values are binary, less than 20 features or input units are needed to encode the left or right phonetic context.

An experiment was performed with the left and right phonetic contexts represented in terms of 17 binary distinctive features, using a network with 64 hidden units. The resulting performance remains about the same as using the unary representation. However, the number of connections for the contextual information reduces from 7808 to 2176, suggesting that distinctive features can provide an explicit and more efficient code for the network to deal with contextual variations.

However, having all the features available during actual recognition can sometimes be difficult. Within a given time region, it is possible that only some of the features can be extracted reliably. Thus a noisy or incomplete description of the features must be accounted for. As an initial step to study the effects of having only an incomplete description of the phonetic contexts, a network was trained and tested when the phonetic contexts were specified with only 10 categories of features. They are high, low, back, labial, alveolar, velar, nasal, retroflex, and liquid. The resulting performance is 74%, which is 3% lower than having a complete description of the phonetic contexts. The decrease of only 3% suggests that an incomplete description of the phonetic context can still provide a great deal of contextual information to improve the performance. In addition, the number of connections for the contextual information reduces to 1280. Table 7.1 summarizes the results.

	Percent Correct	Relative Number of Connections
61 Phonetic Labels	77	1.0
17 Distinctive Features	77	0.28
10 Broad Features	74	0.16

Table 7.1: Comparisons of the performance, and the relative number of connections for contextual information, by representing the contextual information in three different ways.

7.3 Output Representations

In Chapter 4, alternative output representations were explored. Experiments show that the network can extract the distinctive features from the vowels quite reliably, ranging from 86% for the tense feature to 98% for the retroflex feature. Training the network in successive stages was also examined. We found that when the outputs of the first network that extracted the distinctive features were used as inputs to another network, the overall performance was quite similar to that obtained using one network to recognize the vowels directly.

The results of these experiments suggest two possible future directions for using MLP for phonetic recognition. First, the features extracted by the network can be used directly for lexical access or for formation of larger phonological units. These features together with the features in adjacent phonemes can potentially provide a rich description of the acoustic-phonetic information in the speech signal. Second, if there exists a reliable technique for extracting the distinctive features, then the network can be used to map the features to other phonological units. One of the potential difficulties in using the distinctive features is that when the extracted features are error-prone, mapping the features to other phonological units may be non-trivial. Since MLP is a supervised hetero-associator, it may provide an effective mechanism to correct some of the errors introduced at the feature level.

7.4 Internal Representations

Investigations of the internal representation suggest that the network can learn to pay attention to relevant linguistic information in the speech signal and self-organize its knowledge in a way that seems to agree with our knowledge. Furthermore, examination of the connection weights reveals that when the number of hidden units is sufficiently large, the connection vectors between the hidden and output layers are quite orthogonal to each other. Thus they can be used as a set of basis functions and need not be updated during training. The amount of computation required for training can therefore be reduced.

While it is well known that MLP with two hidden layers can approximate any continuous functions arbitrarily well, recent study has shown that with a sufficient number of hidden units, a network with one hidden layer can also approximate any continuous functions [28,29]. This result, combined with our observation that the connection weights between the hidden and output layers need not be trained if there is a sufficient number of hidden units, suggests that it may be possible to approximate any continuous functions by cascading two single-layer perceptrons (SLP's). Since the "upper" SLP needs no training, it may be possible to pre-determine the target signals in the hidden layer. Once these target signals can be specified, the "lower" SLP can be trained without back-propagating errors from the output layer. In other words, the error signals for the hidden units can be generated without feeding information to the output layer. As a result, computational requirements for training the network can be significantly reduced. However, the target signals in the hidden layer often cannot be uniquely determined, since the number of hidden units is often larger than the number of output units. Investigation into procedures for pre-determining the target signals for the hidden layer is an area that deserves further research effort.

7.5 Network Characteristics

The characteristics of the network were examined in several different ways. Our studies showed that the network can approach an asymptotic performance in a few iterations through the training set, suggesting that most of the learning occurs in the first few iterations. If the number of training tokens is fixed, significant improvement in performance cannot be expected by simply iterating through the training set repeatedly. However, this asymptotic performance depends on the number of training tokens, and it improves quite linearly with the logarithm of the number of training tokens. As the number of training tokens increases, the performance results on the training and test data eventually converge. If the number of hidden units is fixed, significant improvement in performance cannot be expected by simply having more training tokens. However, this performance depends on the number of hidden units. If there are sufficient training data, increasing the number of hidden units can improve the performance of the network.

The performance difference between the training and test data provides some indications for improving the performance of the network. When the difference is small, the network can generalize well to the test data. However, it may also indicate that the capability of the network is limited by the number of hidden units. Therefore, the performance can potentially be improved by increasing the size of the network. When the difference is large, the network can pay attention to detailed but irrelevant information in the training data. There is a great deal of flexibility in the network and its performance can be improved by using more training data.

While the most appropriate number of hidden units may very well depend on the specific task or other conditions, our experimental evidence suggests that the number of units in the hidden layer immediately before the output layer should, in general, be chosen to be at least $log_2 N_O$, where N_O is the number of output units. This seems to agree with our intuition, since at least $log_2 N_O$ bits are needed to encode the output classes. Investigation into random connections suggests that the number of units in

the hidden layer immediately after the input layer should be chosen to be at least N_O, resulting, on the average, in one hyperplane for each class. However, the number of hidden units should not be too large, since the performance may decrease if there are too many connection weights to estimate, using the limited amount of training data.

The use of the nonlinear sigmoid function in the hidden and output layers can improve the discriminating capability of the network. If the nonlinear function were not used in the *hidden* layers, MLP would become SLP. If the nonlinear function were not used in the *output* layer, more hidden units might be needed.

Recent results have suggested that no more than one hidden layer is needed for classification in MLP [28,29,62]. Our experimental result shows that the performance of a network with two hidden layers is indeed quite comparable with that of a network with one hidden layer. This result also seems to agree with the previous result that problems that are difficult for a network with one hidden layer are also difficult for a network with two hidden layers [62].

These experiments collectively suggest that the performance may depend on a number of variables such as the number of training iterations, amount of training data, number of hidden units, amount of input information, and the use of the nonlinear sigmoid function. Although the results provide some insights for choosing these different variables, much further work needs to be pursued to quantify their relationships. Some of the issues that deserve quantification include the number of connection weights that can be estimated reliably for a fixed number of training tokens, the best possible performance for a fixed number of training tokens and hidden units, the amount of training data needed before the performance on the training and test data would converge, and the forms of nonlinear functions that are appropriate for phonetic recognition.

7.6 Comparisons with Traditional Techniques

In Chapter 5, the performance of the network was compared with those of traditional techniques based on Bayes decision theory [35]. Although traditional techniques have a well-defined mathematical formalism, their performance can depend on a number of factors when the amount of training data is limited. For example, the performance of parametric techniques depends on whether the form of the parametric model matches the underlying probability distributions. The reliability of estimating the underlying distributions using nonparametric techniques can depend on factors such as the distance metric and the local geometry in the feature space. Experimental results suggest that when the amount of training data is limited, it is possible for MLP to yield higher performance than the k-nearest neighbor and Gaussian classifiers. Analysis of the complexity shows that although MLP requires the greatest amount of computation for training, its requirements during actual classification are quite low.

While our study suggests that MLP can yield higher performance than the two traditional techniques for our task of vowel recognition, we have no proof that the same result applies to any other tasks. By gaining a better understanding of the problem, performance of traditional techniques can potentially be improved. A topic of future research would be to discover and investigate the types of classification problems for which MLP can yield higher performance than traditional techniques. In addition, it would also be worthwhile to develop more efficient algorithms for training the network [4,19,105].

7.7 Error Metrics and Initializations

Although the MLP does not need to make specific assumptions about the form of the underlying probability functions or distance metrics, its performance can be affected by the choice of the error criterion and the initial state of the network. Our experiments indicate that the use of a weighted mean squared error and/or center

initialization can improve the performance of the network. Furthermore, transfer initialization was studied in the context of rapid speaker adaptation. By incorporating some *a priori* knowledge into the network, the performance can be improved, especially when the amount of training data is very limited. However, these are only simple examples of some alternatives. It would be of interest to develop error measures that directly reflect the performance of the network or initialization procedures that capture specific characteristics of the data.

There are many more dimensions in which the characteristics of the network can be improved. For example, the momentum and gain terms in Equation 1.15 can be made adaptable during training [19,116]. The second order derivative of the high-dimensional error surface can be used to improve the learning speed [105]. The temperature in Equation 1.1 can be trained. The connection weights can be modulated by activations of the basic units [57]. These are only some specific examples illustrating some of the directions that need to be pursued before we can fully exploit the framework of MLP. The lack of thorough coverage can only indicate the diversity and complexity of this rapidly growing area.

7.8 Applications to Acoustic-Phonetic Labeling of Continuous Speech

In order to make our study more manageable, our task was constrained to classification of the vowels in American English. Given a time region obtained from the time-aligned transcription, the network determines which one of the 16 vowels was spoken. Although the performance compares favorably to that of human listeners, the network has only been used as a discriminator and is not as yet readily applicable to the task of recognizing continuous speech. When applied to continuous speech recognition, there are at least two fundamental issues that must be explored. First, the network must not rely on the time regions provided by the transcription, since the transcription is unknown during actual speech recognition. Second, the network must

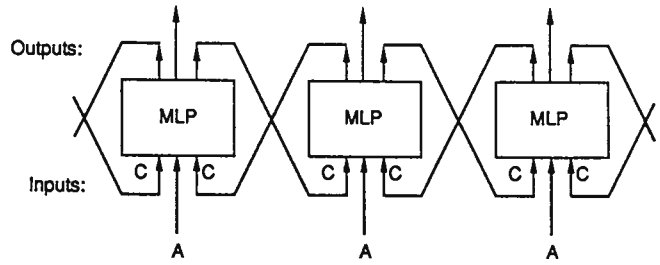


Figure 7.1: Possible basic structure for applying MLP to continuous speech recognition. A stands for acoustic inputs, whereas C stands for contextual inputs.

need to deal with other classes of sounds. Thus other important issues such as proper input representations for different sounds must be addressed. This section discusses how MLP can be exploited for acoustic-phonetic labeling of continuous speech.

Figure 7.1 shows a possible basic structure for applying MLP to continuous speech recognition. The inputs to each network module correspond to measurements including acoustic attributes and contextual information at different times in the speech signal. Although perfect contextual information would not be available during the recognition process, hypotheses about the phonetic contexts can potentially be obtained from processing of the adjacent modules. By allowing adjacent modules to communicate and exchange information, the output of one module can affect the decision of its adjacent module, which in turn, can refine the decision of the original module. In other words, a relaxation process [65,111] may be adopted to account for contextual variations. Alternatively, the contextual input of a module can be as simple as the acoustic input of its adjacent module.

This basic structure can be exploited in a variable-rate segment-based approach or a fixed-rate frame-based approach. In a segment-based approach, acoustic landmarks

or time regions in the speech signal can first be determined or hypothesized. Since acoustic cues for phonetic contrasts are often encoded at specific times in the speech signal, establishing the landmarks can potentially facilitate subsequent extraction or integration of relevant acoustic cues. For example, while it is reasonable to perform classification of vowels based on measurements made at the left, middle, and right regions of a vowel token, classification of the place of articulation for a stop can be more reliable by making measurements near the beginning of the stop release. Therefore description of the speech signal in a segmental level may provide a flexible framework for applying our acoustic-phonetic knowledge [50].

Once the acoustic landmarks are established, relevant acoustic attributes at different times in the speech signal can be measured. Although these time regions are only hypotheses and are often more error-prone than the time-aligned transcription, the resulting measurements can still be used as inputs to each module to perform classification. Of course, the network must be able to invalidate some of the hypothesized time regions.

In a fixed-rate frame-based approach, the control strategy becomes simpler since the process of establishing acoustic landmarks can be bypassed. Acoustic input to each module can correspond to measurements at each frame of speech. However, incorporating acoustic-phonetic information can also be more difficult [50].

7.9 Concluding Remarks

The research reported in this thesis is concerned with the use of ANN's for phonetic recognition. To limit the scope of our investigation, we have chosen to focus on vowel classification using MLP's. Within this context, we have found that ANN's can indeed provide an effective and exciting alternative for phonetic classification. However, further studies of MLP and other types of ANN's are clearly necessary in order for us to gain a more global understanding of the effectiveness of various ANN's as pattern classifiers. Future research should also include theoretical and

experimental explorations into the statistical and limiting behavior of ANN's, as well as the conditions for their optimal use.

While the input representation that we employ is well motivated, it is possible that this somewhat impoverished signal representation may account for the limiting performance of the network in our experiments. Since we are primarily interested in *relative* classification performance and the basic characteristics of the network, we have not been greatly concerned with the adequacy of the input. In the future, we must begin to explore alternative representations so that we can achieve higher classification performance. This issue will be particularly important as we expand from vowels to include other speech sounds, since synchrony spectra may not be entirely appropriate for these sounds. In addition, by classifying different speech sounds, we will truly explore the ability of ANN's to allow radically different sources of information to interact, cooperate and compete.

Our encouraging results on the use of MLP's for phonetic classification lead us to speculate that the network may be applicable to the problem of recognition, i.e. detection as well as classification. An interesting topic for future research would be to investigate how MLP's or other ANN's can be extended to the recognition of continuous speech, or how ANN's can be integrated with other techniques for continuous speech recognition.

We approached this investigation with the belief that ANN's might offer a flexible framework for us to make use of our improved, albeit incomplete, speech knowledge. This belief appears to be substantiated by our experimental results. However, one must not place undue emphasis on the use of self-organizing techniques as a substitute for knowledge. We must continue to strive for a better understanding of the human speech communication process, so that such knowledge will one day enable us to build speech recognition systems with performance approaching that of humans.

Appendix A

Cross-talk in the Hopfield Network

As we have pointed out in Chapter 1, the energy minimization procedure of the Hopfield network may get stuck at an undesirable local minimum. Although the precise relationship between the local minima and the energy function is still not fully understood, the following sections discuss some of the potential problems and present a *supervised* procedure to eliminate some of the spurious local minima from the energy landscape of the network [86].

A.1 Cross-talk between Stored Patterns

When the Hopfield network is used to store mN-dimensional patterns, the connection weights, w_{ij}, can be determined according to Equation 1.8. In vector notation, the connection weight matrix

$$W = \sum_{k=1}^{m} P^k (P^k)^t \tag{A.1}$$

where P^k is a column vector and stands for the k^{th} pattern to be stored. From Equations 1.1 and A.1, the output vector of the network in the l^{th} iteration,

$$\begin{aligned} Y_l &= S(Z_l) \\ &= S(WY_{l-1}), \end{aligned} \tag{A.2}$$

where Z_l stands for the input vector to the sigmoids in the l^{th} iteration, with z_i being its i^{th} element. If the initial output vector is set to the q^{th} stored pattern, i.e. $Y_0 = P^q$,

$$\begin{aligned} Y_1 &= S(Z_1) \\ &= S(WP^q) \\ &= S(\sum_{k=1}^{m} P^k (P^k)^t P^q) \\ &= S(\sum_{k=1}^{m} P^k R_{kq}), \end{aligned} \tag{A.3}$$

where R_{kq} stands for the correlation between P^k and P^q. If P^q is uncorrelated with $P^k \; \forall k \neq q$, then $R_{kq} = 0 \; \forall k \neq q$. From Equation 1.6, Equation A.3 becomes

$$Y_1 = P^q. \tag{A.4}$$

Thus if each of the stored patterns is orthogonal to all other stored patterns, and if the output vector of the network, Y, is initially the same as one of the stored patterns, then the output vector will not change. In other words, the energy landscape has a local minimum at each of the m orthogonal stored patterns.

However, if the stored patterns are not orthogonal, $R_{kq} \neq 0 \; \forall k$. Thus Z_1 in Equation A.3 is a weighted sum of all the stored patterns. In other words, due to the cross-talk between the stored patterns, there is no guarantee that a local minimum is located at each of the stored patterns.

In order to gain a better understanding, a study similar to Gold's was performed [53]. A binary stored pattern was arbitrarily chosen from the steady state

region of each of the following 7 vowels in the /b/-vowel-/t/ environment: /i, ɪ, æ, ɑ, o, ɔ, u/. After computing the weight matrix according to Equation A.1, the network was tested using the same stored patterns. Only three of the seven stored patterns remained unchanged. Each one of the other four stored patterns changed and converged to a nearby local minimum where the energy was lower than that of its original intended location. The fact that a stored pattern can migrate away from its intended location suggests that the weight matrix is unable to create local minima at the stored patterns.

To further test the auto-associative memory, 50 test patterns were generated from each of the stored patterns, making a total of 350 test patterns. Each generated test pattern was obtained by randomizing 50% of the bits of the corresponding stored pattern. It was found that only 22% of the test patterns were able to retrieve their corresponding stored patterns. Furthermore, 43% out of the 350 test patterns converged to the same locations their corresponding stored patterns converged to. In other words, 57% of the test patterns converged to some spurious local minima. Thus the cross-talk between the stored patterns can shift the local minima from their intended locations as well as create spurious local minima that may have no physical meaning. Figure A.1 illustrates a possible energy landscape in one dimension. The circles represent the stored patterns. We can see that pattern A is located above a local minimum of the energy landscape while pattern B is located at a local minimum. There are also spurious local minima at different locations.

A.2 Suppression of Cross-talk

The creation of local minima by using Equation 1.8 can be interpreted as applying a "downward" force to the energy landscape at each of the stored patterns, P^k. Thus a local maximum could similarly be created by applying an "upward" force to the energy landscape. Furthermore, if an upward force of appropriate magnitude is applied at an undesirable local minimum, the two forces in opposite directions can potentially

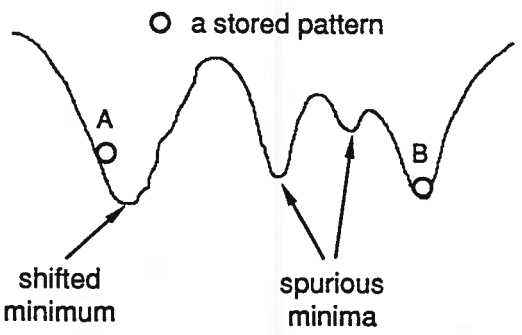


Figure A.1: Possible energy landscape for the Hopfield network in one dimension.

cancel each other, thus removing the undesirable local minimum.

When used as an auto-associative memory, the desirable local minima are known *a priori* and are located at the stored pattern vectors. As a result, the undesirable local minima can potentially be found through use of training data. No upward force needs to be applied if a training token converges to the desired local minimum. However, an upward force can be applied if a training token converges to an undesirable local minimum as shown in Figure A.2. On the one hand, the upward force must not be so large that the entire landscape is disturbed. On the other hand, the force must be large enough to suppress the local minimum. This suggests that a slight force can be applied repeatedly to an undesirable local minimum until it disappears:

$$\Delta W = -\delta Y^u (Y^u)^t \tag{A.5}$$

where δ is a small positive constant and Y^u is an undesirable local minimum.

An experiment was performed to study the effectiveness of such a supervised train-

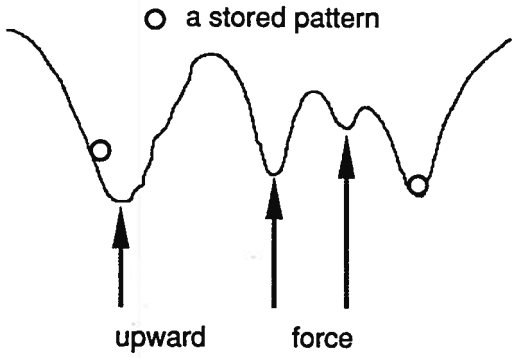


Figure A.2: Undesirable local minima can be suppressed by applying appropriate upward force.

ing technique. Fifty training tokens were obtained by randomizing 50% of the bits of each stored pattern, representing a total of 350 training tokens. The training set was repeatedly presented to the network. The slight upward force specified in Equation A.5 was applied once each time an undesirable local minimum was encountered. The small constant, δ, was chosen to be 0.01. Figure A.3 shows the performance on the training data as a function of the number of iterations through the training set. We can see that before any upward force is applied, only 22% of the training tokens converge to the appropriate local minima. However, over 99% converge to the appropriate local minima after 7 iterations of training.

After applying the above suppression procedure, the network was tested in two ways. First, the stored patterns were used as test vectors. It was found that all these patterns stayed unchanged. In other words, when the test patterns are error-free, the stored patterns can be retrieved. Second, a new set of 350 test tokens were generated the same way the training tokens were obtained. It was found that 97% of the test

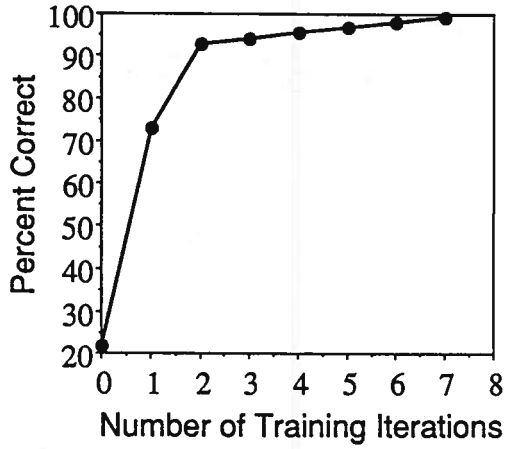


Figure A.3: Performance of the Hopfield network on training data.

patterns converged to the desired local minima. In other words, the suppression of the cross-talk improves the retrieval accuracy of the network from 22% to 97%.

Appendix B

Connection Weight Patterns

B.1 Examples for Vowels

As discussed in Section 4.2.1, the network can learn to pay attention to spectral information near the formant frequencies. This section shows some more examples of the connection weight patterns to different output units of a SLP, which was trained using the vowel tokens in Database I.

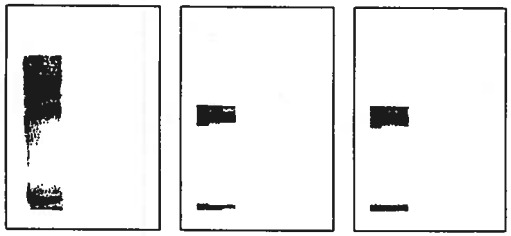


Figure B.1: Internal representation with no hidden layers: spectrographic displays for the original signal, input signal, and the connection weights to an output unit that corresponds to the vowel /i/.

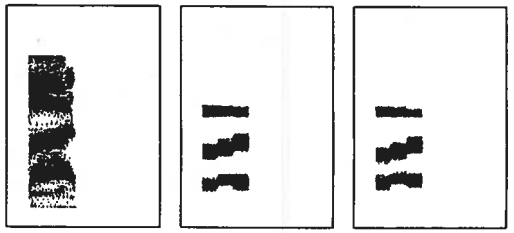


Figure B.2: Internal representation with no hidden layers: spectrographic displays for the original signal, input signal, and the connection weights to an output unit that corresponds to the vowel /ɑ/.

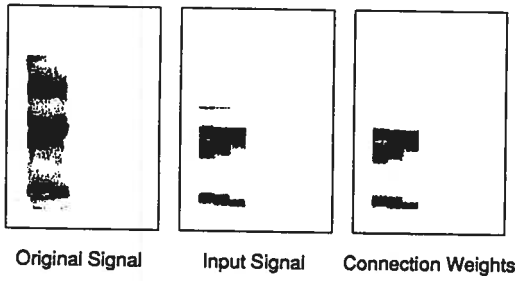


Figure B.3: Internal representation with no hidden layers: spectrographic displays for the original signal, input signal, and the connection weights to an output unit that corresponds to the vowel /ɜ˞/.

B.2 Examples for Distinctive Features

As discussed in Section 4.2.1, the network learns to extract relevant acoustic properties from the speech signal that correspond to the distinctive features. This section shows more examples of the connection weight patterns for the distinctive features. The SLP was trained using the vowel tokens in Database II. Each figure shows the connection weight patterns to the units of presence (+) and absence (-) of the feature. For comparison, the input signals of two examples are also shown.

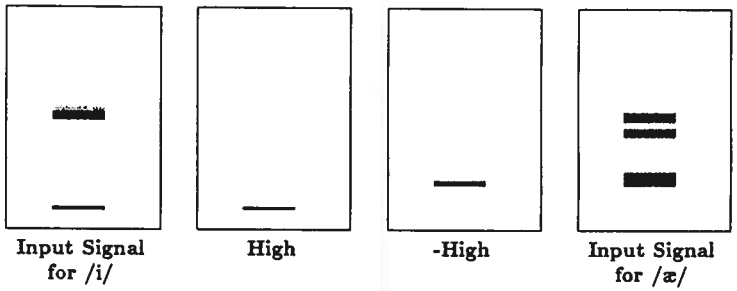


Figure B.4: Internal representation with no hidden layers: spectrographic displays for the connection weights to output units that correspond to the high and -high features, and the input signals of two examples, /i/, a high vowel, and /æ/, a -high vowel. The connection weights for the high and -high features are the greatest at the frequencies where the high and -high vowels typically have their first formants.

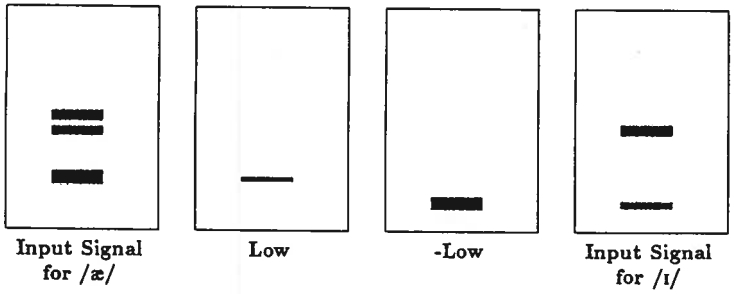


Figure B.5: Internal representation with no hidden layers: spectrographic displays for the connection weights to output units that correspond to the low and -low features, and the input signals of two examples, /æ/, a low vowel, and /ɪ/, a -low vowel. The connection weights for the low and -low features are the greatest at the frequencies where the low and -low vowels typically have their first formants.

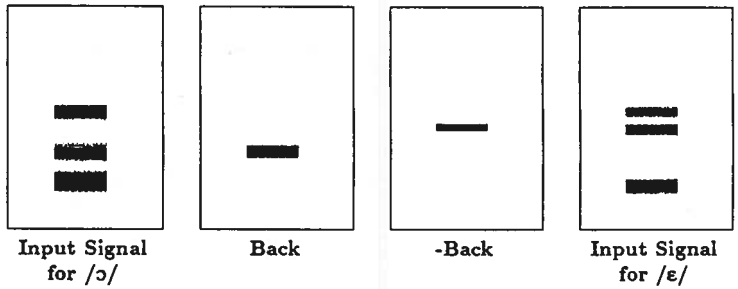


Figure B.6: Internal representation with no hidden layers: spectrographic displays for the connection weights to output units that correspond to the back and -back features, and the input signals of two examples, /ɔ/, a back vowel, and /ɛ/, a -back vowel. The connection weights for the back and -back features are the greatest at the frequencies where the back and -back vowels typically have their second formants.

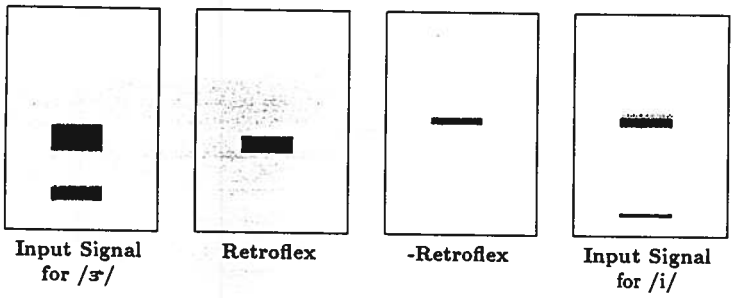


Figure B.7: Internal representation with no hidden layers: spectrographic displays for the connection weights to output units that correspond to the retroflex and -retroflex features, and the input signals of two examples, /ɝ/, a retroflex vowel, and /i/, a -retroflex vowel. The connection weights for the retroflex feature are the greatest at the frequencies where the retroflex vowel, /ɝ/, typically has its second and third formants. The connection weights for the -retroflex feature are the greatest at the frequencies typically above the third formant frequency of a retroflex vowel, where not much energy can be found.

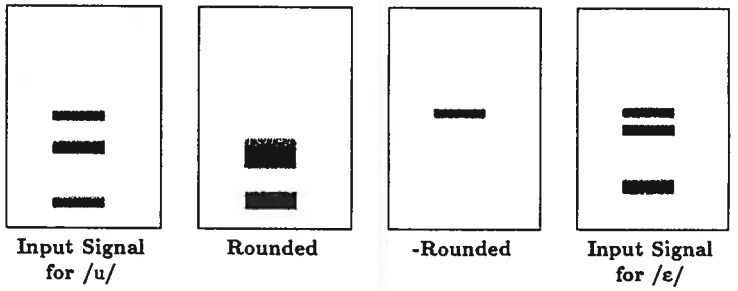


Figure B.8: Internal representation with no hidden layers: spectrographic displays for the connection weights to output units that correspond to the rounded and -rounded features, and the input signals of two examples, /u/, a rounded vowel, and /ɛ/, a -rounded vowel. While the connection weights for the rounded feature are more difficult to interpret from visual inspection, those for the -rounded feature are the greatest at the frequencies typically above the third formant frequency of a rounded vowel, where not much energy can be found.

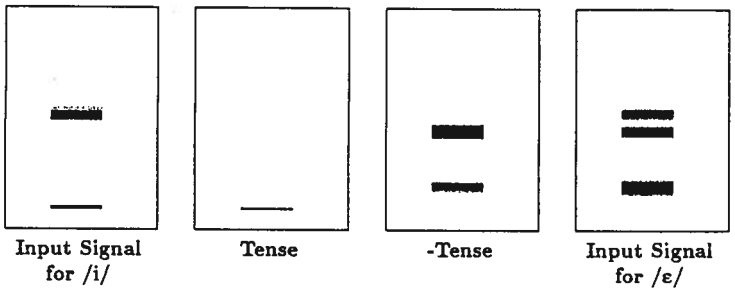


Figure B.9: Internal representation with no hidden layers: spectrographic displays for the connection weights to output units that correspond to the tense and -tense features, and the input signals of two examples, /i/, a tense vowel, and /ɛ/, a -tense vowel. While the connection weights for the -tense feature are more difficult to interpret from visual inspection, those for the tense feature are the greatest at the frequencies where the -low vowels typically have their first formants, presumably because all the tense vowels are -low vowels according to Table 3.4.

Appendix C

Training and Test Speakers

This appendix lists the 500 training and 50 test speakers in Database IV, which was constructed from the TIMIT database. The first letter of each item indicates whether the speaker is male or female, whereas the next 4 characters identify the speaker, and the last digit encodes the dialect of the speaker.

C.1 Training Speakers

FAEM0-2	FAKS0-1	FALK0-3	FALR0-4	FASW0-5
FAWF0-5	FBAS0-4	FBCG1-8	FBCH0-6	FBJL0-5
FBMH0-5	FBMJ0-4	FCAG0-4	FCAJ0-2	FCAL1-5
FCDR1-5	FCEG0-8	FCFT0-4	FCJS0-7	FCLT0-8
FCMG0-3	FCMH0-3	FCMH1-8	FCMM0-2	FCRH0-4
FCRZ0-7	FCYL0-2	FDAC1-1	FDAS1-2	FDAW0-1
FDHC0-7	FDJH0-3	FDML0-1	FDMY0-5	FDRD1-2
FDRW0-6	FDTD0-5	FDXW0-2	FEAC0-2	FEAR0-5
FECD0-1	FEDW0-4	FELC0-1	FEME0-3	FETB0-1
FEXM0-5	FGCS0-3	FGDP0-5	FGJD0-4	FGMB0-5
FGMD0-5	FGRW0-3	FGWR0-7	FHEW0-5	FHLM0-2
FISB0-7	FJAS0-2	FJCS0-5	FJDM2-6	FJKL0-2
FJLG0-3	FJLR0-3	FJMG0-4	FJRE0-2	FJRP1-7
FJSA0-5	FJSJ0-8	FJSP0-1	FJWB1-4	FJXM0-5

FJXP0-4 FKAA0-2 FKDE0-7 FKDW0-4 FKFB0-1
FKLC0-4 FKLH0-8 FKMS0-3 FLAC0-3 FLAG0-6
FLAS0-7 FLET0-7 FLHD0-4 FLJD0-3 FLJG0-5
FLKD0-4 FLKM0-4 FLMA0-2 FLMC0-2 FMAF0-4
FMAH1-7 FMBG0-8 FMCM0-4 FMEM0-1 FMGD0-6
FMJB0-2 FMJF0-3 FMJU0-6 FMKF0-2 FMLD0-8
FMMH0-2 FMPG0-5 FNKL0-8 FNLP0-5 FNMR0-4
FNTB0-3 FPAB1-7 FPAD0-6 FPAS0-2 FPAZ0-3
FPKT0-3 FPLS0-8 FPMY0-5 FRAM1-2 FRJB0-6
FRLL0-2 FRNG0-4 FSAG0-5 FSAH0-1 FSCN0-2
FSDC0-5 FSEM0-4 FSGF0-6 FSJG0-5 FSJS0-3
FSKC0-3 FSKL0-2 FSKP0-5 FSLB1-2 FSMA0-1
FSMM0-5 FSMS1-5 FSPM0-7 FSRH0-2 FTAJ0-6
FTBR0-1 FTBW0-5 FTLH0-7 FTMG0-2 FUTB0-5
FVKB0-7 FVMH0-1 MABC0-6 MADC0-3 MADD0-7
MAEB0-4 MAFM0-7 MAHH0-5 MAJC0-8 MAKB0-3
MAKR0-3 MARC0-2 MBCG0-8 MBDG0-3 MBGT0-5
MBJK0-2 MBJV0-2 MBMA0-4 MBMA1-6 MBML0-7
MBOM0-7 MBPM0-5 MBSB0-8 MBTH0-7 MBWM0-3
MBWP0-4 MCAE0-6 MCAL0-3 MCCS0-2 MCDC0-3
MCDD0-3 MCDR0-4 MCEM0-2 MCEW0-2 MCHH0-7
MCHL0-5 MCLM0-5 MCMJ0-6 MCPM0-1 MCRC0-5
MCRE0-7 MCSH0-3 MCTH0-7 MCTM0-2 MCTT0-5
MCTW0-3 MCXM0-8 MDAC0-1 MDAC2-5 MDAW1-8
MDBB0-2 MDBB1-3 MDBP0-2 MDCD0-4 MDCM0-7
MDDC0-3 MDED0-7 MDEF0-3 MDEM0-2 MDHL0-5
MDHS0-3 MDJM0-3 MDLB0-2 MDLC0-3 MDLC2-2
MDLD0-2 MDLF0-7 MDLH0-3 MDLM0-7 MDLR0-7
MDLS0-4 MDMA0-4 MDMT0-2 MDNS0-3 MDPB0-7
MDPS0-2 MDRB0-5 MDRD0-6 MDRM0-4 MDSJ0-5
MDSS0-2 MDSS1-3 MDTB0-3 MDVC0-7 MDWA0-5
MDWH0-5 MDWK0-5 MDWM0-3 MEAL0-6 MEDR0-1
MEFG0-2 MEGJ0-5 MEJL0-6 MEJS0-8 MESD0-6
MESG0-4 MESJ0-6 MEWM0-5 MFER0-5 MFGK0-5
MFRM0-4 MFWK0-4 MFXV0-7 MGAG0-4 MGAR0-7
MGAW0-7 MGES0-5 MGJC0-4 MGLB0-3 MGMM0-4
MGRL0-1 MGRP0-4 MGRT0-7 MGSH0-5 MGSL0-7
MGWT0-2 MHBS0-7 MHIT0-5 MHMG0-5 MHMR0-3
MHPG0-3 MHRM0-2 MHXL0-7 MILB0-3 MJAC0-4

MJAE0-2 MJAI0-7 MJAR0-2 MJBG0-2 MJBR0-3
MJDC0-4 MJDE0-2 MJDM0-5 MJDM1-4 MJEB1-1
MJEE0-4 MJES0-3 MJFH0-5 MJFR0-7 MJHI0-2
MJJB0-3 MJJG0-3 MJJJ0-4 MJJM0-7 MJKR0-3
MJLB0-4 MJLG1-3 MJLN0-8 MJLS0-4 MJMA0-2
MJMM0-4 MJMP0-3 MJPG0-5 MJPM0-2 MJPM1-4
MJRA0-7 MJRF0-4 MJRG0-5 MJRH0-4 MJRH1-3
MJRK0-6 MJSR0-4 MJSW0-1 MJTC0-8 MJTH0-8
MJVW0-3 MJWG0-5 MJWS0-4 MJWT0-1 MJXA0-5
MJXL0-4 MKAG0-7 MKAH0-2 MKAJ0-2 MKAM0-4
MKCH0-3 MKDB0-7 MKDD0-8 MKDT0-2 MKES0-6
MKJL0-7 MKJO0-2 MKLN0-6 MKLS0-1 MKLS1-3
MKLW0-1 MKRG0-8 MKXL0-3 MLBC0-4 MLEL0-4
MLIH0-5 MLJB0-4 MLJC0-4 MLJH0-4 MLLL0-4
MLNS0-3 MLSH0-4 MMAA0-2 MMAB0-3 MMAB1-5
MMAG0-2 MMAM0-3 MMAR0-3 MMBS0-4 MMCC0-5
MMDB0-6 MMDB1-2 MMDG0-7 MMDM0-4 MMDM1-5
MMDM2-2 MMDS0-2 MMEA0-8 MMEB0-3 MMGC0-4
MMGK0-2 MMJB1-3 MMLM0-8 MMRP0-1 MMSM0-3
MMVP0-5 MMWB0-5 MMWH0-3 MMWS0-8 MMXS0-2
MNET0-4 MNJM0-7 MNLS0-7 MPAB0-7 MPAM0-8
MPAM1-6 MPAR0-7 MPCS0-4 MPDF0-2 MPEB0-4
MPFU0-7 MPGH0-1 MPGL0-2 MPGR0-1 MPGR1-6
MPLB0-4 MPMB0-5 MPPC0-2 MPRB0-2 MPRD0-3
MPRK0-4 MPRT0-4 MPWM0-4 MRAB0-2 MRAB1-4
MRAI0-1 MRAV0-5 MRBC0-3 MRCG0-1 MRCS0-7
MRCW0-2 MRCZ0-2 MRDM0-8 MRDS0-3 MREB0-1
MREE0-3 MREH1-3 MREM0-7 MREW1-5 MRFL0-4
MRGG0-2 MRGM0-4 MRGS0-2 MRHL0-2 MRJH0-2
MRJM0-2 MRJM3-5 MRJM4-7 MRJO0-1 MRJR0-6
MRJS0-6 MRJT0-2 MRKO0-4 MRLD0-5 MRLJ0-2
MRLK0-8 MRLR0-2 MRMB0-6 MRML0-5 MRMS0-2
MRMS1-7 MRPC0-7 MRPC1-7 MRPP0-5 MRRE0-8
MRRK0-5 MRSP0-4 MRTC0-3 MRTJ0-3 MRTK0-3
MRVG0-5 MRWA0-3 MRWS0-1 MRWS1-5 MRXB0-6
MSAS0-5 MSAT0-2 MSDB0-7 MSDH0-5 MSEM1-5
MSES0-7 MSJK0-6 MSLB0-8 MSMR0-6 MSMS0-4
MSRR0-5 MSTF0-4 MSVS0-6 MTAA0-3 MTAB0-7
MTAS0-4 MTAS1-2 MTAT0-5 MTAT1-2 MTBC0-2
MTCS0-8 MTDB0-2 MTDP0-5 MTDT0-3 MTER0-7

MTHC0-3	MTJG0-2	MTJM0-3	MTJS0-1	MTJU0-6
MTKD0-7	MTLC0-7	MTLS0-4	MTMT0-5	MTPF0-1
MTPG0-3	MTPP0-3	MTPR0-7	MTQC0-4	MTRC0-4
MTRR0-1	MTRT0-4	MTWH0-7	MTWH1-7	MVJH0-3
MVLO0-5	MVRW0-7	MWAC0-5	MWBT0-1	MWCH0-5
MWDK0-3	MWEM0-5	MWEW0-2	MWGR0-3	MWJG0-3
MWRP0-7	MWSB0-2	MWSH0-5	MWVW0-2	MZMB0-2

C.2 Test Speakers

FAJW0-2	FDFB0-3	FDMS0-4	FDNC0-2	FJEM0-1
FJRB0-8	FJWB0-2	FKKH0-5	FLBW0-4	FLNH0-6
FLOD0-5	FMAH0-5	FPAC0-7	FREH0-7	FREW0-4
FSAK0-4	FSJW0-3	MAEO0-7	MAPV0-3	MARW0-4
MBAR0-7	MBEF0-3	MBNS0-4	MCMB0-5	MDAS0-5
MDLC1-7	MDLR1-7	MDWD0-2	MGAF0-3	MGJF0-3
MGXP0-4	MJDH0-6	MJEB0-2	MJRP0-2	MKCL0-4
MKDR0-7	MKLT0-5	MLNT0-3	MRAM0-5	MRES0-8
MRJM1-2	MRKM0-5	MROA0-4	MSFH0-4	MSFH1-5
MSFV0-3	MSJS1-1	MSMC0-4	MSRG0-4	MTMR0-2

Bibliography

[1] Bahl, L.R., Cole, A.G., Jelinek, F., Mercer, R.L., Nadas, A., Nahamoo, D., and Picheny, M.A., "Recognition of isolated word sentences from a 5000-word vocabulary office correspondence task," in *Proc. ICASSP-83*, 1983.

[2] Bahl, L.R., Jelinek, F., and Mercer, R., "A maximum likelihood approach to continuous speech recognition," *IEEE Trans. PAMI*, Vol. PAMI-5, No. 2, March, 1983.

[3] Barto, A.G., and Anandan, P., "Pattern-recognizing stochastic learning automata," *IEEE Trans. Systems, Man, and Cybernetics*, Vol. SMC-15, No.3, May/June 1985.

[4] Becker, S. and Cun, Y.L., "Improving the convergence of back-propagation learning with second order methods," *Proceedings of the 1988 Connectionist Models*, Summer School, Carnegie Mellon Universtiy, June 17-26,1988.

[5] Bengio, Y., and De Mori, R., "Use of neural networks for the recognition of place of articulation," *Proc. ICASSP*, New York, 1988.

[6] Bentley, J.L., "Multidimensional divide-and-conquer," *Communications of the ACM*, Vol. 23, No.4, April, 1980.

[7] Bentley, J.L., Weide, B.W., and Yao, A.C., "Optimal expected-time algorithms for closest point problems," *ACM Trans. on Mathematical Software*, Vol. 6, No. 4, December, 1980.

[8] Bladon, R.A.W., "Speaker normalization by linear shifts along a bark scale," *Proc. 10th International Congress of Phonetic Sciences*, Netherlands, 1983.

[9] Blumstein, S.E. and Stevens, K.N., "Acoustic invariance in speech production: evidence from measurements of the spectral characteristics of stop consonants," *Journal of the Acoustical Society of America*, Vol. 66, 1979.

[10] Blumstein, S.E. and Stevens, K.N., "Perceptual invariance and onset spectra for stop consonants in different vowel environments," *Journal of the Acoustical Society of America*, 1980.

[11] Blumstein, S.E. and Stevens, K.N., "Phonetic features and acoustic invariance in speech," *Cognition* 10, 1981.

[12] Bourlard, H. and Wellekens, C.J., "Speech pattern discrimination and multilayer perceptrons," Manuscript M.211, Phillips Research Laboratory, Brussels, Belgium.

[13] Burr, D.J., "Experiments on neural net recognition of spoken and written text," *IEEE Trans. ASSP*, Vol.36, July 1988.

[14] Bush, M.A., Kopec, G.E., and Zue, V.W., "Selecting acoustic features for stop consonant identification," *Proc. ICASSP-83*, 1983.

[15] Bush, M.A., Kopec, G.E., and Lauritzen, N., "Segmentation in isolated word recognition using vector quantization," *Proc. ICASSP-84*, San Diego, 1984.

[16] Buzo, A., Gray, A.H., Gray, R.M., and Markel, J.D., "Speech coding based upon vector quantization," *IEEE Trans. ASSP* Vol. ASSP-28, No. 5, October, 1980.

[17] Carbonell, N., Damestoy, J., Fohr, D., Haton, J., and Lonchamp, F., "Aphodex, design and implementation of an acoustic-phonetic decoding expert system," *Proc. ICASSP-86*, Tokyo, Japan.

[18] Carpenter, G.A., and Grossberg, S., "Neural dynamics of category learning and recognition: attention, memory consolidation, and amnesia," in J. Davis, R. Newburgh, and E. Wegman (eds.) *Brain Structure, Learning, and Memory*, AAAS Symposium Series, 1986.

[19] Chan, L.W. and Fallside, F., "An adaptive training algorithm for back propagation networks," *Computer Speech and Language 2*, 1987.

[20] Chomsky, N. and Halle, M. *The Sound Pattern of English*, Harper & Row, 1968.

[21] Cohen, R., Baldwin, G., Berstein, J., Murveit, H., and Weintraub, M., "Studies for an adaptive recognition lexicon," *Proc. DARPA Speech Recognition Workshop*, Report no. SAIC-86/1644, 1987.

[22] Cole, R. & Scott, B., "Towards a theory of speech perception," *Psychological Review*, 81, 1974.

[23] Cole, R.A., Stern, R.M., and Lasry, M.J., "Performing fine phonetic distinctions: templates versus features," in *Invariance and Variability in Speech Processes*, J.S. Perkell and D.H. Klatt, Eds. Hillsdale, NJ: Lawrence Erlbaum Assoc., 1985.

[24] Cole, R.A., Rudnicky, A.I., Zue, V.W., and Reddy, D.R., "Speech as patterns on paper," in *Perception and Production of Fluent Speech*, R.A. Cole (ed.), Hillsdale, NJ: Lawrence Erlbaum Assoc., 1980.

[25] Cole, R.A. and Zue, V.W., "Speech as eyes see it," in *Attention and Performance VIII*, R.S. Nickerson, (ed.), Hillsdale, NJ: Lawrence Erlbaum Assoc., 1980.

[26] Cover, T.M., "Learning in pattern recognition," in *Methodologies of Pattern Recognition*, S. Watanabe, ed., Academic Press, New York, 1969.

[27] Cover, T.M. and Hart, P.E., "Nearest neighbor pattern classification," *IEEE Trans. Info. Theory*, IT-13, January, 1967.

[28] Cybenko, G., "Continuous valued neural networks with two hidden layers are sufficient," *Tufts University*, 1988.

[29] Cybenko, G., "Approximation by superpositions of a sigmoidal function," *Tufts University*, 1988.

[30] Davis, S.B., Mermelstein, P., "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. Acoust, Speech, and Signal Processing*, Vol. ASSP-28, No.4, August 1980.

[31] De Mori, R., Laface, P., and Piccolo, E., "Automatic detection and description of syllabic features in continuous speech," *IEEE Trans. Acoustics, Speech, and Signal Processing*, Vol. ASSP-24, No.5., October, 1976.

[32] Denker, J.S., "AIP conference proceedings 151, neural networks for computing," *American Institute of Physics*, New York, 1986.

[33] Devijver, P.A. and Kittler, J., "Pattern recognition: a statistical approach," *Prentice Hall*, 1982.

[34] Dixon, N.R. and Silverman, H.F., "The 1976 modular acoustic processor (MAP)," *IEEE Trans. Acoust, Speech, and Signal Processing*, Oct. 1977.

[35] Duda, R.O. and Hart, P., *Pattern classification and scene analysis*, John Wiley & Sons, 1973.

[36] Egan, J., "Articulation testing methods II," OSRD Report No. 3802, *U.S. Dept. of Commerce Report PB 22848*, 1944.

[37] Elman, J.L, and Zipser, D., "Learning the hidden structure of speech," ICS Report 8701, U. of California, San Diego, 1987.

[38] Elman, J.L., and McClelland, J., "Exploiting lawful variability in the speech wave," in *Invariance and variability in speech processes*, edited by Perkell, J., and Klatt, D., Lawrence Erlbaum Associates, Publishers, 1986.

[39] Espy-Wilson, C.Y., "A phonetically based semivowel recognition system," in *Proc. ICASSP-86*, 1986.

[40] Fallside, F., Chan, L.W., "Connectionist models and geometric reasoning," *CUED/F-CAMS/TR.266, Cambridge University*, 1986.

[41] Feldman, J., Fanty, M., and Goddard, N., "Computing with structured neural networks," *IEEE Computer*, March, 1988.

[42] Feng, M.W., Kubala, F., Schwartz, R., and Makhoul, J., "Improved speaker adaptation using text dependent spectral mappings," *Proc. ICASSP-88*, New York, 1988.

[43] Fisher, W.E., Doddington, G.R., and Goudie-Marshall, K.M., "The DARPA Speech Recognition Research Database: Specifications and Status," *Proceedings of the DARPA Speech Recognition Workshop* Report No. SAIC-86/1546, February, 1986.

[44] Flanagan, J.L., "Speech analysis synthesis and perception," *Springer-Verlag,* New York, 1965.

[45] Fralick, S.C. and Scott, R.W., "Nonparametric Bayes risk estimation," *IEEE Trans. Info. Theory,* IT-17, July, 1971.

[46] Friedman, J.H., Bentley, J.L., and Finkel, R.A., "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. on Mathematical Software,* Vol. 3, No. 3, September, 1977.

[47] Gallant, S. and Smith, D., "Random cells: an idea whose time has come and gone ... and come again," *First International Conference Neural Network,* IEEE, June 1987.

[48] Gallager, R.G., *Information theory and reliable communication,* John Wiley and Sons, 1968.

[49] Ghitza, O., "Robustness against noise: the role of timing-synchrony measurement," *Proc. ICASSP 87,* 1987.

[50] Glass, J.R. and Zue, V.W., "Detection and recognition of nasal consonants in American English," in *Proc. ICASSP-86,* Tokyo, 1986.

[51] Glass, J.R. and Zue, V.W., "Signal representation for acoustic segmentation," *Proc. First Conf. on Speech Science and Tech.,* 1986.

[52] Glass, J.R., *Finding acoustic regularities in speech: applications to phonetic recognition,* Ph.D. Thesis, Massachusetts Institute of Technology, 1988.

[53] Gold B., "Hopfield model applied to vowel and consonant discrimination," *Lincoln Laboratory Technical Report 747,* June 1986.

[54] Gold, B., and Lippmann, R.P., "A neural network for isolated word recognition," *Proc. ICASSP-88,* New York, 1988.

[55] Gold, B. and Rabiner, L.R., "Parallel processing techniques for estimating pitch periods of speech in the time domain," *Journal of the Acoustical Society of America 46,* 1969.

[56] Grossberg, S., "Neural networks and natural intelligence," MIT Press, 1988.

[57] Hinton, G.E., "A parallel computation that assigns canonical object-based frames of reference," *Proc. IJCAI 7,* Vancouver, Canada.

[58] Hinton, G.E., Sejnowsky, T.J., and Ackley, D.H., "Boltzmann machines: constraint satisfaction networks that learn," *Technical Report CMU-CS-84-119,* Carnegie-Mellon University, 1984.

[59] Hoel, Port, and Stone, "Introduction to statistical theory," Houghton Mifflin, 1971.

[60] Hopfield J., "Neural networks and physical systems with emergent collective computational abilities," *Proc. National Acad. Sci.,* Vol 79, April 1982.

[61] Hopfield J., "Neurons with Graded Response have Collective Computational Properties like those of two-state Neurons," *Proc. Natl. Acad. Sci.* USA. May 1984.

[62] Huang, W.Y., and Lippmann, R.P., "Neural net and traditional classifiers," *IEEE Conference on Neural Information Processing Systems,* Colarado, 1987.

[63] Huang, W.M., Lippmann, R.P., and Gold, B., "A neural net approach to speech recognition," *Proc. ICASSP-88,* New York, 1988

[64] Huang, W.Y. and Lippmann, R.P., "Comparisons between conventional and neural net classifiers," *First International Conference on Neural Network,* IEEE, June 1987.

[65] Hummel, R.A. and Zucker, S.W., "On the foundations of relaxation labeling processes," *IEEE Trans. on PAMI,* Vol. PAMI-5, No.3, May 1983.

[66] Hunt, M.J., and Lefebvre, C., "Speaker dependent and independent speech recognition experiments with an auditory model," *Proc. ICASSP-88,* New York, 1988.

[67] Jakobson, R., Fant, G., and Halle, M., *Preliminaries to speech analysis,* MIT Press, Cambridge, MA., 1963.

[68] Jordan, M.I., "Serial order: a parallel distributed processing approach," *ICS Report 8604,* University of California, San Diego, May 1986.

[69] Kamn, C.A., Landauer, T.K., and Singhal, S., "Training an adaptive network to recognize demisyllables in continuous speech," 1988 IEEE Workshop on Speech Recognition.

[70] Kirkpatrick S., Gelatt C., and Vecchi, M, "Optimization by simulated annealing," *Science* Vol. 220, Number 4598, May 1983.

[71] Klatt, D.H., "Review of the ARPA speech understanding project," *J. Acoust. Soc. Amer.,* Vol. 62, No.6, Dec. 1977.

[72] Klatt, D.H., "The problem of variability in speech recognition and in models of speech perception," in *Invariance and Variability in Speech Processes,* J.S. Perkell and D.H. Klatt, Eds. Hillsdale, NJ: Lawrence Erlbaum Assoc., 1985.

[73] Klatt, D.H., "Models of phonetic recognition I: Issues that arise in attempting to specify a feature-based strategy for speech recognition," in *Proceedings of Montreal Symposium on Speech Recognition,* July, 1986.

[74] Klatt, D.H., "Linguistic uses of segmental duration in English: acoustic and perceptual evidence," *Journal of Acoustical Society of America,* Vol. 59, No. 5.

[75] Klatt, D.H. and Stevens, K.N., "On the automatic recognition of continuous speech: implications of a spectrogram-reading experiment," *IEEE Trans. on Audio and Electroacoustics,* Vol. AU-21, No.3, 1973.

[76] Kohonen, T., *Self-Organization and Associative Memory,* Springer-Verlag, Berline, 1984.

[77] Kohonen, T., Torkkola, K., Shozakai, M., Kangas J., and Venta, O., "Phonetic typewriter for Finnish and Japanese," *Proc. ICASSP-88,* New York, 1988.

[78] Kohonen, T., Masisara, K., and Saramaki, T., "Phonotopic maps - insightful representation of phonological features for speech representation," *Proceedings IEEE 7th Inter. Conf. on Pattern Recognition,* Montreal, Canada, 1984.

[79] Kosko, Bart, "Adaptive inference in fuzzy knowledge networks," *Proc. IEEE International Conf. on Neural Networks,* San Diego, June 1987.

[80] Kuchera, H., and Francis, W.N., *Computational analysis of present-day American English,* Brown University Press, Providence, R.I., 1967.

[81] Lamel, L.F., Kassel, R., and Seneff, S., "Speech database development: design and analysis of the acoustic-phonetic corpus," *Proc. DARPA Speech Recognition Workshop,* Report no. SAIC-86/1546, 1986.

[82] Lea, W.A., *Trends in Speech Recognition,* Englewood Cliffs, NJ: Prentice-Hall, 1980.

[83] Lee, K.F., and Hon, H.W., "Large-vocabulary speaker-independent continuous speech recognition using HMM," *Proc. ICASSP-88,* New York, 1988.

[84] Lesser, V.R., Fennell, R.D., Erman, L.D., and Reddy, D.R., "Organization of Hearsay II speech understanding system," *Proc. ICASSP-75,* 1975.

[85] Leung, H.C., "Some phonetic recognition experiments using artificial neural nets," *Proc. ICASSP-88,* New York, 1988.

[86] Leung, H.C., Area Exam Paper. MIT, 1987.

[87] Leung, H.C., "A procedure for automatic alignment of phonetic transcriptions with continuous speech," *Proc. ICASSP-84,* San Diego, 1984.

[88] Leung, H.C., *A procedure for automatic alignment of phonetic transcriptions with continuous speech,* S.M. thesis, MIT, Cambridge, MA, 1985.

[89] Lewis, P.M. and Coates, C.L., *Threshold Logic,* John Wiley & Sons, 1967.

[90] Lippmann, R.P., "An introduction to computing with neural nets," *IEEE ASSP Magazine,* April 1987.

[91] Lippmann, R.P., "Review of Neural Networks for Speech Recognition," *Neural Computation* 1, 1989.

[92] Lippmann, R.P., Gold, B., and Malpass, M.L., "A comparison of Hamming and Hopfield neural nets for pattern classification," *MIT Lincoln Laboratory Technical Report,* TR-769, 1987.

[93] Lisker, L, "Rapid vs. rabid: A catalogue of acoustic features that may cue the distinction," *Haskins Laboratories, Status Report on Speech Research,* SR-54, 1978.

[94] Lubensky, D., "Learning spectral-temporal dependencies using connectionist networks," *Proc. ICASSP-88,* New York, 1988.

[95] Makhoul, J. and Schwartz, R., "Ignorance modeling," in *Invariance and Variability in Speech Processes,* J.S. Perkell and D.H. Klatt, Eds. Hillsdale, NJ: Lawrence Erlbaum Assoc., 1985.

[96] Makhoul, J., Schwartz, R., and El-Jaroudi, A., "Classification capabilities of two-layer neural nets," *Proc. ICASSP-89,* Glasgow, Scotland, May 1989.

[97] Massaro, D.W. and Oden, G.C., "Evaluation and integration of acoustic features in speech perception," *J. Acoust. Soc. Amer.,* 67(3), March 1980.

[98] McClelland, J.L. and Rumelhard, D.E., "An interactive activation model of context effects in letter perception. Part I: An account of basic findings." *Psychological Review,* 88, 1981.

[99] Mercier, G., Callec, A., Monne, J., Querre, M., Trevarain, O., "Automatic segmentation, recognition of phonetic units and training in the keal speech recognition system," *Proc. ICASSP-82,* Paris, France.

[100] Minsky, M., and Papert, S., *Perceptrons.* MIT Press, Cambridge, MA, 1969.

[101] Minsky, M.L. and Selfridge, O.G., "Learning in random nets," *Information Theory, Fourth London Symposium,* C. Cherry, ed., Butterworth, London, 1961.

[102] Nearey, T., *Phonetic feature systems for vowels,* Ph.D. dissertation, University of Connecticut, 1977.

[103] Newell, A., "Intellectual issues in the history of artificial intelligence," in *The Study of Information: Interdisciplinary Messages,* F. Machlup and U. Mansfield, eds., John Wiley and Sons, New York, 1983.

[104] Newell, A., Barnett, J., Forgie, J.W., Green, C.C., Klatt, D.H., Licklider, J.C.R., Munson, J., Reddy, D.R. and Woods, W.A., "Speech understanding systems: final report of a study group," *Amsterdam, The Netherlands: North-Holland/American Elsevier,* 1973.

[105] Parker, D.B., "Optimal algorithms for adaptive networks: second order back propagation, second order direct propagation, and second order Hebbian learning," *Proc. IEEE First International Conference on Neural Networks,* San Diego, 1987.

[106] Peterson, G.E. and Barney, H.L., "Control methods used in a study of the vowels," *Journal Acoust. Soc. Amer.*, Vol. 24, 1952.

[107] Phillips, M.S., "Speaker independent classification of vowels and diphthongs in continuous speech," *Proc. of the 11th International Congress of Phonetic Sciences*, Estonia, USSR, 1987.

[108] Prager, R.W., Harrison, T.D., and Fallside, F., "Boltzmann machines for speech recognition," *Computer Speech and Language*, 1986.

[109] Rabiner, L.R., Levinson, S.E., and Sondhi, M.M., "On the application of vector quantization and hidden Markov models to speaker-independent, isolated word recognition," *The Bell System Technical Journal*, April 1983.

[110] Rabiner, L.R. and Myers, C.S., "Connected digit recognition using a level-building DTW algorithm," *IEEE Trans. Acoust., Speech, Signal Process.*, Vol. ASSP-29, No.3, June 1981.

[111] Rumelhart *et al.*, *Parallel Distributed Processing*, MIT Press, 1986.

[112] Rohwer, R., Renals, S., and Terry, M., "Unstable connectionist networks in speech recognition," *Proc. ICASSP-88*, New York, 1988.

[113] Rosenblatt, F., *Perceptrons and the theory of brain mechanisms*, Spartan Books, 1962.

[114] Rossen, M.L., Niles, L.T., Tajchman, G.N., Bush, M.A., Anderson, J.A., Blumstein, S.E., "A connectionist model for constant-vowel syllable recognition," *Proc. ICASSP-88*, New York, 1988.

[115] Rtischev, Dimitry, "Speaker adaptation in a large-vocabulary speech recognition system," S.M. thesis, January 1989.

[116] Rumelhart, D.E., Hinton, G.E., and Williams, R.J., "Learning representations by back-propagating errors," *Nature*, Vol. 323, October 1986.

[117] Schwartz, R., personal communication, June 1988.

[118] Schwartz, R., Chow, Y., Roucos, S., Krasner, M., and Makhoul, J., "Improved hidden Markov modeling of phonemes for continuous speech recognition," *Proc. ICASSP-84*, San Diego, 1984.

[119] Schwartz, R.M., Chow, Y.L., and Kubala, F., "Rapid speaker adaptation using a probabilistic spectral mapping," *Proc. ICASSP-87*, Dallas, TX, 1987.

[120] Sejnoha, V., "Speaker normalization transformations for automatic recognition," *J. Acoust. Soc. Amer.*, 74, S17, 1983.

[121] Sejnowsky, T., personal communication.

[122] Seneff S., "A computational model for the peripheral auditory system: application to speech recognition research," *Proc. ICASSP-86*, Tokyo, 1986.

[123] Seneff S., "Vowel recognition based on 'line-formants' derived from an auditory-based spectral representation," *Proc. of the 11th International Congress of Phonetic Sciences*, Estonia, USSR, 1987.

[124] Shikano, K., Lee, K.F., and Reddy, R., "Speaker adaptation through vector quantization," *Proc. ICASSP-86* Tokyo, Japan, 1986.

[125] Stevens, K.N., "Acoustic correlates of some phonetic categories," *J. Acoust. Soc. Amer.*, Vol.63, No.3 1980.

[126] Stevens, K.N., *Course notes for Speech Communication*, MIT, 1984.

[127] Stevens, K.N. and Blumstein, S.E., "Invariant cues for place of articulation in stop consonants," *Journal of the Acoustical Society of America*, 1978, 64.

[128] Stevens, K.N. and Blumstein, S.E., "The search for invariant acoustic correlates of phonetic features," in *Perspectives on the study of speech*, P.D. Eimas and J.L. Miller Ed. Lawrence Erlbaum Assoc., 1981.

[129] Stevens, K.N., "Models of phonetic recognition II: An approach to feature-based recognition," *Proceedings of Montreal Symposium on Speech Recognition*, July, 1986.

[130] Syrdal, A.K., "Aspects of a model of the auditory representation of American English vowels," *Speech Communication* 4, 1985.

[131] Tou, J.T. and Gonzalez, R.C., *Pattern Recognition Principles*, Addison-Wesley, 1974.

[132] Touretzky, D., Hinton, G., and Sejnowsky, T., editors, *Proceedings of the 1988 Connectionist Models*, Carnegie Mellon University, 1988.

[133] Waibel, A., Hanazawa, T., and Shikano, K., "Phoneme recognition: neural networks vs. hidden Markov models," *Proc. ICASSP-88*, New York, 1988.

[134] Wakita, H., "Normalization of vowels by vocal tract length and its application to vowel recognition," *IEEE Trans. ASSP-25*, 1977.

[135] Watrous, R.L., "Connectionist speech recognition using the temporal flow model," 1988 IEEE Workshop on Speech Recognition.

[136] Weinstein, C.J., McCandless, S.S., Mondshein, L.F., and Zue V.W., "A system for acoustic-phonetic analysis of continuous speech," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, February 1975.

[137] Wolf, J.J. and Woods, W.A., "The HWIM speech understanding system," *Proc. ICASSP-77*, 1977.

[138] Woods, W.A., "Motivation and overview of SPEECHLIS: an experimental prototype for speech understanding research," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, February 1975.

[139] Zue, V.W., "The use of speech knowledge in automatic speech recognition," *Proceedings of the IEEE*, November 1985.

[140] Zue, V.W. and Cole, R.A., "Experiments on spectrogram reading," *Proc. ICASSP-79*, 1979.

[141] Zue, V.W., "Models of phonetic recognition III: The role of analysis by synthesis in phonetic recognition," in *Proceedings of Montreal Symposium on Speech Recognition*, July, 1986.

[142] Zue, V.W., *Acoustic characteristics of stop consonants: a controlled study*, Ph.D. Thesis, Department of Electrical and Computer Science, MIT., 1976.

[143] Zue, V.W. and Laferriere, M., "Acoustic study of medial /t,d/ in American English," *Journal of Acoustical Society of America*, Vol. 66. No.4, Oct. 1979.

[144] Zue, V.W. and Lamel, L.F., "An expert spectrogram reader: a knowledge-based approach to speech recognition," *Proc. ICASSP-86*, Tokyo, 1986.

[145] Zue, V.W. and Seneff, S., "Transcription and alignment of the TIMIT database," *Second Symposium on Advanced Man-Machine Interface through Spoken Language*, Hawaii, 1988.

[146] Zue, V.W., Glass, J., Phillips, M., and Seneff S., "Acoustic segmentation and phonetic classification in the summit system," *Proc. ICASSP-89*, Scotland, 1989.