Unsupervised Neural Pattern Classifiers for Oral Vowel Pronunciation of Foreign-Accented Speech

Joseph Hecker
Clemson University, heckerj@clemson.edu
UNSUPERVISED NEURAL PATTERN CLASSIFIERS
FOR ORAL VOWEL PRONUNCIATION TRAINING
OF FOREIGN-ACCENTED SPEECH

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Electrical Engineering

by
Joseph C. Hecker
December 2006

Accepted by:
Dr. John N. Gowdy, Committee Chair
Dr. Robert J. Schalkoff
Dr. Constancio Nakuma
ABSTRACT

This thesis describes the development of unsupervised neural-based pattern classifiers for the training of vowel pronunciations for students learning a foreign language. This paper examines American learners of the French language. A corpus of single word utterances is compiled from a group of native French speakers. Cepstral features are used to train two unsupervised neural pattern classifiers: a self-organizing map and a growing neural gas. The development and justification for the use of these classifiers is presented. The output from the classifier is translated to a bar graph for visual assessment. The degree to which the utterance sounds native is determined by comparing target graphs and those produced by the user. It is concluded that unsupervised classification techniques can be used to develop a pronunciation training system that is independent of the language used to train the system. This allows for pronunciation training to be easily achieved for low-density languages.
DEDICATION

This thesis is dedicated to the love of friends and family. Namely, I would like to dedicate this thesis to Laura Oliveria for her continual inspiration, and to my parents for their unyielding encouragement. It is from their support that I have gained the maturity to succeed in my graduate studies.
ACKNOWLEDGEMENTS

I would like to acknowledge the members of my thesis committee for their guidance in my research. I thank Dr. Gowdy for introducing me to speech processing and capturing my interest in pronunciation training. I thank Dr. Schalkoff for sharing his expertise in pattern recognition and directing me toward the resources needed to complete this project. I thank Dr. Nakuma for offering a linguist’s perspective and providing the real world context for the research. It is under their direction that my education successfully culminated in this project.

I would also like to acknowledge the students who generously volunteered their time to aid in the generation of the training corpus and in testing the finished software.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE PAGE</td>
<td>i</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>ii</td>
</tr>
<tr>
<td>DEDICATION</td>
<td>ii</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>iv</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>vii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>viii</td>
</tr>
<tr>
<td>CHAPTER</td>
<td></td>
</tr>
<tr>
<td>1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>Background</td>
<td>1</td>
</tr>
<tr>
<td>Problem Definition</td>
<td>2</td>
</tr>
<tr>
<td>Organization of Thesis</td>
<td>4</td>
</tr>
<tr>
<td>2. SPEECH THEORY AND PRONUNCIATION TRAINING</td>
<td>5</td>
</tr>
<tr>
<td>Speech Processing Theory</td>
<td>5</td>
</tr>
<tr>
<td>Speech Production System</td>
<td>6</td>
</tr>
<tr>
<td>Characteristics of Speech</td>
<td>7</td>
</tr>
<tr>
<td>Pronunciation Training</td>
<td>19</td>
</tr>
<tr>
<td>Need for Pronunciation Training</td>
<td>20</td>
</tr>
<tr>
<td>Overview of current techniques</td>
<td>20</td>
</tr>
<tr>
<td>3. NEURAL PATTERN CLASSIFICATION TECHNIQUES</td>
<td>24</td>
</tr>
<tr>
<td>Feed-Forward Neural Networks</td>
<td>25</td>
</tr>
<tr>
<td>Brief History and Biological Basis</td>
<td>25</td>
</tr>
<tr>
<td>The Single Neuron</td>
<td>26</td>
</tr>
<tr>
<td>The Multi-layered Feed-Forward Neural Network</td>
<td>30</td>
</tr>
<tr>
<td>Properties of Feed-Forward Neural Networks</td>
<td>32</td>
</tr>
<tr>
<td>Relevance to Pronunciation Training</td>
<td>33</td>
</tr>
</tbody>
</table>
Table of Contents (Continued)

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-Organizing Maps</td>
<td>34</td>
</tr>
<tr>
<td>Learning Vector Quantization</td>
<td>34</td>
</tr>
<tr>
<td>Formulation of a Self-Organizing Map</td>
<td>35</td>
</tr>
<tr>
<td>Neural Gas Algorithms</td>
<td>37</td>
</tr>
<tr>
<td>Efficient Neural Gas</td>
<td>40</td>
</tr>
<tr>
<td>Hierarchical Overlapped Neural Gas</td>
<td>41</td>
</tr>
<tr>
<td>Enhanced Neural Gas</td>
<td>42</td>
</tr>
<tr>
<td>Growing Neural Gas</td>
<td>44</td>
</tr>
<tr>
<td>Robust Growing Neural Gas</td>
<td>45</td>
</tr>
<tr>
<td>Radial Basis Function Networks</td>
<td>48</td>
</tr>
<tr>
<td>Hybrid Techniques</td>
<td>49</td>
</tr>
<tr>
<td>Distance Measurements</td>
<td>51</td>
</tr>
<tr>
<td>Visualizations of Unsupervised Clustering</td>
<td>52</td>
</tr>
<tr>
<td>4. EXPERIMENTATION AND RESULTS</td>
<td>56</td>
</tr>
<tr>
<td>Database</td>
<td>56</td>
</tr>
<tr>
<td>Corpus Design</td>
<td>56</td>
</tr>
<tr>
<td>Sample Collection</td>
<td>57</td>
</tr>
<tr>
<td>Feature Extraction</td>
<td>60</td>
</tr>
<tr>
<td>Network Training</td>
<td>62</td>
</tr>
<tr>
<td>Self-Organizing Map</td>
<td>62</td>
</tr>
<tr>
<td>Robust Growing Neural Gas</td>
<td>64</td>
</tr>
<tr>
<td>System Implementation</td>
<td>67</td>
</tr>
<tr>
<td>5. CONCLUSIONS</td>
<td>71</td>
</tr>
<tr>
<td>Results of the Self-Organizing Map</td>
<td>71</td>
</tr>
<tr>
<td>Results of the Robust Growing Neural Gas</td>
<td>72</td>
</tr>
<tr>
<td>Further Discussion and Future Works</td>
<td>72</td>
</tr>
<tr>
<td>BIBLIOGRAPHY</td>
<td>75</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1. First two formants values for English and French vowels</td>
<td>13</td>
</tr>
<tr>
<td>4.1. List of words recorded for use in the training set</td>
<td>57</td>
</tr>
<tr>
<td>4.2. SOM classification</td>
<td>64</td>
</tr>
<tr>
<td>4.3. RGNG network classification</td>
<td>66</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Diagram of the speech production system</td>
<td>6</td>
</tr>
<tr>
<td>2.2</td>
<td>IPA phonetic chart</td>
<td>8</td>
</tr>
<tr>
<td>2.3</td>
<td>Block diagram of the speech production system</td>
<td>11</td>
</tr>
<tr>
<td>2.4</td>
<td>Time-amplitude graph of the French word ‘classe’</td>
<td>12</td>
</tr>
<tr>
<td>2.5</td>
<td>Frequency response of the vowel sound [a]</td>
<td>12</td>
</tr>
<tr>
<td>2.6</td>
<td>Vowel formant structures of English and French</td>
<td>14</td>
</tr>
<tr>
<td>2.7</td>
<td>Cepstral features from one frame of speech</td>
<td>19</td>
</tr>
<tr>
<td>3.1</td>
<td>Biology of the neuron</td>
<td>26</td>
</tr>
<tr>
<td>3.2</td>
<td>Possible activation functions for the artificial neuron</td>
<td>28</td>
</tr>
<tr>
<td>3.3</td>
<td>The artificial neuron</td>
<td>30</td>
</tr>
<tr>
<td>3.4</td>
<td>A generic multi-layered feed-forward neural network</td>
<td>31</td>
</tr>
<tr>
<td>3.5</td>
<td>A neural unit for a radial basis function network</td>
<td>50</td>
</tr>
<tr>
<td>4.1</td>
<td>Experimental setup for speech acquisition</td>
<td>58</td>
</tr>
<tr>
<td>4.2</td>
<td>A screenshot from the speech collection program</td>
<td>59</td>
</tr>
<tr>
<td>4.3</td>
<td>Resulting topology of the SOM after training</td>
<td>63</td>
</tr>
<tr>
<td>4.4</td>
<td>MDL plot for one word set</td>
<td>67</td>
</tr>
<tr>
<td>4.5</td>
<td>MDL plot for two word set</td>
<td>68</td>
</tr>
<tr>
<td>4.6</td>
<td>Visualization of a native sounding utterance [y]</td>
<td>69</td>
</tr>
<tr>
<td>4.7</td>
<td>Visualization of mispronouncing [y] as [u]</td>
<td>69</td>
</tr>
</tbody>
</table>
List of Figures (Continued)

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.8. Visualization of a native sounding utterance [a].</td>
<td>70</td>
</tr>
<tr>
<td>4.9. Visualization of a mispronouncing [a] as [α].</td>
<td>70</td>
</tr>
</tbody>
</table>
CHAPTER 1
INTRODUCTION

Background

Speech is the most natural form of communication for the majority of the world’s people. The ability to speak is one of the most prevalent distinctions between our species and all others. Yet this innate ability has evolved in such a way that thousands of languages are currently in use worldwide. Historically, the inability of people of different languages to communicate orally with one another has acted as a barrier between cultures; whereas common language has acted as a unifier [Lang 1986] [Ben-David 1996]. With the globalization of politics and economies, increasing language education is becoming a necessity.

Traditionally, native speakers of a maternal language (L1) interested in learning a second language (L2) would be required to invest a great deal of classroom time with an instructor formally trained in the L2 vocabulary, grammar, pronunciation, and culture. With the rapid acceleration of education technology it is no longer necessary to learn language in a classroom environment. A growing number of software tools are being developed specifically for the teaching of second languages and are categorized as Computer-Assisted Language Learning (CALL) programs. CALL curricula incorporate many facets of a given language from simple vocabulary to insights on culture through a wide range of text and multimedia interfaces. The benefit of CALL is that the student can define the schedule and pace at which the material is learned. As a result CALL programs require a wide variety of learning materials so that every subject area can be examined.

However, due to both biological and psychological reasons, native sounding pronunciations are often difficult to learn. Traditionally, the most effective way to learn to speak with a native-sounding accent was to immerse one’s self in that culture.
Language immersion programs are effective in teaching native-sounding pronunciations by harnessing the human brain’s innate language learning mechanism. Modern CALL programs are beginning to investigate this problem through the use of speech processing software.

**Problem Definition**

The purpose of this thesis is to develop software able to assist a user in the acquisition of a native sounding accent in a non-maternal language. CALL technology is a broad topic in and of itself, with a large emphasis on teaching culture and grammar. However, the focus of this thesis is to study the aspects of speech processing needed in the learning of pronunciation. Pedagogical issues will be addressed concisely in order to provide context for the application of analytical, mathematical techniques.

Speech processing is the name given to the niche of signal analysis used in modeling human speech. Speech processing can be broken down into three broad categories: speech coding, speaker recognition, and speech recognition. Speech coding is the label for methods that attempt to decompose speech in such a way that it can be represented using the minimum number of bits. Applications for speech coding include digital speech communication and storage. Speaker recognition falls into the category of biometrics; attempting to verify or identify an individual by acoustic properties of the individual’s voice. Speech recognition describes the genre of computational techniques used to determine what was said by a speaker, commonly used for human-computer interaction. It is this category which is essential for CALL.

A speech recognition system can be broken into two subcategories, digital signal processing (DSP) and pattern recognition. Signal processing techniques are used to acquire the speech sample and derive features of the signal that will be useful in determining what the speaker has said. Pattern recognition describes the classification of features into meaningful categories. In terms of pronunciation training, the overall goal is to obtain a speech sample from the student and classify it in such a manner
so that it can be determined whether or not an utterance can be considered a native-sounding pronunciation.

In constructing a speech recognizer, a general methodology is to first obtain a set of data on which to train the system. A database of speech from native L2 speakers is acquired. A set of features is derived from the speech database. Depending on the requirements of the application, a different set of features may be used. Common features in speech recognition applications include cepstral coefficients and linear prediction coefficients. These values are used to train the pattern recognizer.

There are many different types of pattern recognizers: statistical, syntactical, and neural. Statistical pattern recognizers use the probability distributions of the dataset to develop a classification scheme based often on a maximum likelihood requirement. The syntactical pattern recognizers often use grammars to discriminate between patterns. Neural methods are those which use mathematical models derived from biological systems, namely the human brain, to classify data. The techniques explored here within are all neural based methods namely the feed-forward neural network, the self-organizing map, and the neural gas algorithm. These systems must all be trained for a specific pattern recognition application. Issues that arise in each of these scenarios are the ability to discriminate, generalization, training, and computational effort.

Since a great many differences exist between the thousands of living languages, it was decided for this research that a CALL program would be developed for a specific L1 and L2, English and French, respectively. Techniques are developed in a general manner so that they can be ported to other languages. Also, this application focuses only on the pronunciations of oral vowels since the consonant structure of these two languages is reasonably similar. Nasal vowels are not considered in this study. Oral vowels also are the most dominant sounds in a given word. From this point on, the term vowel refers exclusively to oral vowels. Since often a language learner has difficulty hearing the subtleties of a foreign-language pronunciation, the output of each of these networks will provide a visualization of the student-user’s
pronunciation. It has been hypothesized that the ability to produce correct utterances can often precede the ability to consciously discern the distinction [Listerri 1995]. Thus a visual assessment of the pronunciation is justified.

Organization of Thesis

Chapter 2 provides an overview of the speech production system; addressing the linguistic concerns in pronunciation training and providing an overview of current pronunciation training techniques. Chapter 3 discusses the neural-based pattern recognition techniques. Three architectures are investigated: the feed-forward neural network, the self-organizing map, and the neural gas. Chapter 4 describes the experimental setup and results of applying the concepts described in the two preceding chapters. Chapter 5 summarizes the experimental results and indicates the future directions of this research.
CHAPTER 2
SPEECH THEORY
AND PRONUNCIATION TRAINING

The exact number of languages in the world is the topic of debate for linguists, but it is well established that there are several thousand spoken languages worldwide. Some languages have similar traits as others, while some are unique. Languages can be categorized most generally by vocabulary and grammar, and more subtly by accents and tone. The human speech production system is a network of organs that act in unison to convey language conceived in the brain. Since each of these spoken languages use the same articulators to produce speech, it is logical to assume that any person with the right training should be able to produce sounds in any language. However in practice this is often difficult to achieve.

This chapter provides an overview of the speech production system. The current model of the vocal tract is described and signal processing techniques useful in speech recognition are developed. Once an understanding of the mathematics involved in speech recognition is established, current implementations of these techniques are described in their application to the problem of pronunciation training.

Speech Processing Theory

O’Shaughnessy describes the speech chain as the concatenation of events from a speaker’s conception of spoken language to the processing of the language in the listener’s brain [O’Shaughnessy 1987]. It has been the goal of speech processing engineers to mathematically describe these events in order to transform the problem into one solvable by machine. One such application is pronunciation training, which requires an understanding of the mathematics of speech.
Speech processing can be seen as a classic signal processing problem. From an engineer’s perspective, speech is simply a stochastic signal from which a wide range of properties can be derived. These properties stem from models of the biological system.

Speech Production System

The speech production system is formed through the collaboration of a number of articulator organs in the abdomen, throat, and head. Sound is generated when a pulse of air is forced from the lungs through the trachea and out of the mouth and nose. The tongue, teeth, and lips shape the pulse of air in such a way that the pressure-wave resonates in the vocal tract creating a single phoneme.

![Diagram of the speech production system.](image)

Figure 2.1 Diagram of the speech production system.

Phonemes are the most basic part of speech and describe one individual sound. Words are generally made up of many phonemes. The International Phonetic Association (IPA) defines 107 different sounds each represented by a symbol and 55 different modifications these symbols can possibly undergo for all spoken languages in the world [IPA 2005]. Figure 2.2 shows the IPA phoneme chart. To a linguist, each of the phonemes can be described by traits of the articulators: rounding of the lips, position of the occlusion of the tongue, degree to which the mouth is open, path the sound wave travels, and vibration of the glottis [Léon and Bhatt 2005]. It is this set
of features that is used in traditional pronunciation training techniques, a variety of language therapy. As will be shown shortly, these features have distinct mathematical properties as well.

For the pronunciation training task at hand, it is important to note that vowels are all voiced phonemes (those formed with glottal vibrations), have non-transient movement of the articulator organs, and are expressed orally. Vowels are all formed by the position of the tongue and the shape of the lips. These are the speech organs most easily controlled by a language learner.

Characteristics of Speech
Speech characteristics can be categorized into three groups: segmental, sub-segmental, and super-segmental. A segment of speech to the speech processing engineer is a window of speech of finite duration representing a single phoneme. This speech sample can be analyzed in both the time and frequency domains to extract information about the phoneme for the goal of classifying the sound. Sub-segmental speech describes the characteristic of speech that represent only a small portion of a frame of speech. Voice on-set time, the time between stop consonant and vocal chord vibration in a consonant vowel (CV) utterance, is an example of a sub-segmental speech feature. Super-segmental speech characteristics are prosodic features such as intonation and stress. These are often used to add meaning to sentences, such as raising one’s voice at the end of a question. All of these characteristics have features that are dependent on one’s native tongue.

The category of speech features examined in this thesis is the segmental features of foreign-accented speech. However, pronunciation training techniques for super- and sub-segmental features will be described at the end of the chapter with a useful extension to super-segmental features being posed in the concluding chapter.

In order to use digital signal processing techniques, the analog speech waveform, $x_a$, must be digitized. When a speech sample is collected, it is often sampled at a minimum of 8 kHz. The reason for this, as shown below, is because the major features
Figure 2.2 IPA phonetic chart
of the speech signal occur in the frequency band of 0-4000 Hz. In compliance with the Nyquist sampling theorem, 8 kHz is the bare minimum needed to successfully digitize information in this band. In most recognition applications, speech is often sampled at a higher rate (12 kHz for the experiments conducted here). Mathematically, the discrete-time signal, $x(n)$, can be expressed:

$$x(n) = x_a(t)|_{t=nT},$$

(2.1)

for sampling a period $T$ and a sample index $n$.

The majority of classic, long-term signal processing techniques require sequences that exhibit the same behavior for all of time (i.e., auto-correlation). Short-term approximations of signal features have been developed that approximate long-term time functions. Derivation of these techniques can be found in Chapter 4 of [Deller et al. 1993]. In the development of short-term features, speech must be divided into frames for analysis. A frame of speech is generally between 10 and 40 ms in duration and therefore includes several hundred samples. A variety of windowing functions are available for speech processing. In selecting a window type, the trade-offs between width of the main lobe and the attenuation of the side bands must be considered. A narrow main lobe provides less distortion of abrupt changes in the frequency spectrum of the signal, whereas lower side band attenuation reduces the effects of frequency leakage. For the purpose of pronunciation training, the details in the waveform are of the most concern, so a Hamming window is often used because of its reasonably narrow main lobe and its reasonably large attenuation in the sidebands.

Justification of the use of short-term statistics comes from observing the wide-sense stationary behavior of a vowel over time. A signal is wide-sense stationary if:

1. The autocorrelation is a function of time difference only.

2. The expected value of the function is constant.
Therefore, one frame of speech is a valid measure of the segmental feature in speech. On the other hand, super-segmental features take advantage of the changes between frames.

The speech waveform can be modeled as the response of a glottal excitation source oscillating at variable frequency (F0) and the concatenation of tubes that filter the oscillation. The block diagram of this system is shown in 2.3. The sound wave exits the body through both the oral and nasal cavities, with all vowels being exclusively produced orally. Basic filter theory will dictate that the excitation source is convolved with the filter created by the vocal tract to produce the output. This is essential in the development of speech features. The speech production can be expressed in both the time and z domains as:

\[ s(t) = e(t) * g(t) * h(t) * r(t), \]  
\[ S(z) = E(z)G(z)H(z)R(z), \]

where \( e \) represents the time domain excitation function, \( g \) represents the shaping of the glottal pulse, \( h \) represents the vocal tract shaping of the sound wave, and \( r \) represents the radiation of the sound at the lips. The \( * \) is the convolution function. The excitation function is represented as a sequence of ideal discrete-time impulses. The imperfect nature of the pressure wave emitted from the glottis factored in the selection of the glottal shaping filter.

Due to the periodic nature of the excitation of the speech waveform, a vowel can be shown to be a power signal over a finite period of time. This is evident in Fig. 2.4. A discrete-time signal, \( x(n) \) consisting of N points is determined to be a power signal if the power \( P_x \) is finite and nonzero.

\[ P_x = \lim_{N \to \infty} \frac{1}{2N + 1} \sum_{n=-N}^{N} |x(n)|^2. \]
Continuing the development of short-term features, a short-term discrete Fourier transform (DFT) can be defined over one frame of speech. The Fourier transform is a method by which the frequency content of a signal is extracted from a windowed time domain signal $f(n;m)$ through multiplication by an orthogonal basis function and summation over time. Mathematically, it is expressed

$$S(k;m) = \begin{cases} \sum_{n=m-N+1}^{m} f(n;m)e^{-jk(2\pi/N)n}, & k = 0, \ldots, N-1, \\ 0, & \text{elsewhere}, \end{cases}$$

(2.5)

where $j$ is $\sqrt{-1}$, $N$ is the length of the window, and $k$ is related to frequency as $\frac{\omega N}{2\pi}$. Zero-padding the frame of speech is also necessary to suppress frequency leakage caused by generating the frame [Oppenheim and Schafer 1999].

The magnitude of resulting graph shows the resonant frequencies of the vocal tract with the frequency response of the system, shown as the smooth envelope, Fig. 2.5. This envelope clearly shows the resonant frequencies, called formants, that define the sound. Formants are labeled in ascending order according to frequency. Formants F1 and F2 are the dominating sounds in the phoneme with formants F3 and F4 enhancing the quality of the sound. Relating these resonant frequencies to the biologic
system and the linguistic interpretation of the sound shows that F1 is related to the amount of oral and pharyngeal constriction whereas F2 is related to the position of the hump of the tongue. Location and height of the spectral peaks are the primary factors in determining a given sound. However, bandwidth has been shown to aid in the recognition of competing sounds [de Cheveigné 1999].

Each phoneme can be described by a unique set of frequency-domain characteris-
<table>
<thead>
<tr>
<th>Vowel</th>
<th>Language</th>
<th>F1</th>
<th>F2</th>
</tr>
</thead>
<tbody>
<tr>
<td>[i]</td>
<td>English</td>
<td>300</td>
<td>2500</td>
</tr>
<tr>
<td></td>
<td>French</td>
<td>275</td>
<td>2250</td>
</tr>
<tr>
<td>[y]</td>
<td>English</td>
<td>275</td>
<td>n/a</td>
</tr>
<tr>
<td></td>
<td>French</td>
<td>250</td>
<td>1800</td>
</tr>
<tr>
<td>[I]</td>
<td>English</td>
<td>350</td>
<td>19500</td>
</tr>
<tr>
<td></td>
<td>French</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>[e]</td>
<td>English</td>
<td>400</td>
<td>2100</td>
</tr>
<tr>
<td></td>
<td>French</td>
<td>375</td>
<td>2200</td>
</tr>
<tr>
<td>[ɔ]</td>
<td>English</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td></td>
<td>French</td>
<td>375</td>
<td>1600</td>
</tr>
<tr>
<td>[ɛ]</td>
<td>English</td>
<td>550</td>
<td>1800</td>
</tr>
<tr>
<td></td>
<td>French</td>
<td>550</td>
<td>1800</td>
</tr>
<tr>
<td>[œ]</td>
<td>English</td>
<td>600</td>
<td>1200</td>
</tr>
<tr>
<td></td>
<td>French</td>
<td>550</td>
<td>1450</td>
</tr>
<tr>
<td>[a]</td>
<td>English</td>
<td>750</td>
<td>1700</td>
</tr>
<tr>
<td></td>
<td>French</td>
<td>750</td>
<td>1350</td>
</tr>
<tr>
<td>[ɔ]</td>
<td>English</td>
<td>750</td>
<td>1100</td>
</tr>
<tr>
<td></td>
<td>French</td>
<td>750</td>
<td>1200</td>
</tr>
<tr>
<td>[ɛ]</td>
<td>English</td>
<td>550</td>
<td>900</td>
</tr>
<tr>
<td></td>
<td>French</td>
<td>550</td>
<td>950</td>
</tr>
<tr>
<td>[ɔ]</td>
<td>English</td>
<td>400</td>
<td>800</td>
</tr>
<tr>
<td></td>
<td>French</td>
<td>375</td>
<td>750</td>
</tr>
<tr>
<td>[U]</td>
<td>English</td>
<td>370</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>French</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>[u]</td>
<td>English</td>
<td>300</td>
<td>900</td>
</tr>
<tr>
<td></td>
<td>French</td>
<td>250</td>
<td>750</td>
</tr>
</tbody>
</table>

Table 2.1 Values of the first two formants (Hz) for vowels in English and French

Table 2.1 shows the formant locations for the vowels in the English and French language. The problem of pronunciation training becomes clearer when these two independent groups are examined simultaneously. Relating average formant frequencies of these two languages with their IPA symbols clearly demonstrates the similarities and differences in phoneme locations for vowels of different languages, as shown in Fig. 2.6. Examining the frequency domain properties of these vowels yields a distinct resonance structure for each vowel in each language [Delattre 1964].

Figure 2.6 also shows the root of the major issues involved in pronunciation train-
The inability of a person to pronounce foreign phonemes can stem from two different sources. First, language learners often have trouble pronouncing L2 phonemes that do not exist in L1. This is the case with English learners of the French language. The French phoneme [y] does not exist in the English language, so often times English speakers will substitute in a phoneme that is relatively close sounding, such as [u]. It is interesting to note that Portuguese speakers from Brazil will substitute [i] instead of [u] as a result of the psychological partitioning of the phoneme space [Dalby and Kewley-Port 1999].

In most speech recognition applications, formant locations prove to be insufficient for determining with great accuracy what was spoken. Several sets of features have been developed to provide a more complete representation of a frame of speech. Two commonly excepted models of speech features are the linear predictive coefficients and the cepstral coefficients. It has been found that a set of ten linear predictive coefficients can model a speech waveform of 100-200 discrete samples [O’Shaughnessy 1987]. Cepstral coefficients behave similarly. Both the linear predictive coefficients and the cepstral coefficients will be examined for their use in a pronunciation training system.
Linear Prediction

The linear prediction model uses autocorrelation statistics from a frame of speech to estimate the transfer function that models the vocal tract as a minimum phase filter. In understanding the workings of linear prediction, a detailed model of the biologic speech production system is required. The relation to linear prediction then becomes evident.

The physics of a pressure wave of air through a tube is well known. Just as blowing into a soda bottle produces a simple tone, forcing air through the vocal tract produces a more complex tone. The reason is that the contour of the vocal tract forces the wave to resonate at a variety of different frequencies, resulting in the formants observed in the frequency-domain analysis of a speech waveform. Correspondingly, the accepted model of the speech production system is the concatenation of a sequence of tubes.

Termed the Lossless-Tube model, a series of tubes, in which no energy is lost during propagation, can produce frequency domain characteristics similar to that of the vocal tract. In terms of a system function, the lossless tube model can be formalized by a linear difference equation of the all-pole form

\[ H(z) = H_0 \frac{1}{1 - \sum_{i=1}^{R} a(i)Z^{-i}}, \] (2.6)

where \( H_0 \) is a gain term and the coefficients \( a(i) \) represent a minimum phase system. The goal of linear prediction is to determine a set of coefficients \( \alpha(i) \) that equals the \( a(i) \) terms.

While interpreting linear prediction can be explained in several different manners, such as inverse filtering and spectral flattening, only the system identification interpretation will be examined here. The goal of the system identification interpretation is to design a filter that minimizes the mean squared residual error when presented with a sample of speech with a requirement that the 0\(^{th}\) coefficient be equal to 1. Differentiating the expectation function of the squared residual error yields,
where \( r_s(\eta) \) and \( r_{es}(\eta) \) are the autocorrelation functions of the speech sample and the autocorrelation function of the residual error with the speech function, respectively, at time \( \eta \). This set of \( M \) equations is called the LP normal equations when residual error is equal to zero, or the coefficients perfectly model the system.

Solving for the coefficients of the LP-normal equations defines the all-pole model in a mean squared error sense. The Levinson-Durbin algorithm is an efficient method by which the LP-normal equations are solved that makes use of the Toeplitz structure inherent in the matrix form of the equations. The Levinson-Durbin algorithm is summarized in Alg. 2.1.

\begin{align}
\sum_{i=1}^{M} \alpha(i) r_s(\eta - i) - r_{es}(\eta) = 0 & \quad \eta = 1, 2, \ldots, M, \tag{2.7}
\end{align}

Algorithm 2.1 Levinson-Durbin algorithm for efficient computation of the linear predictive coefficients.

\begin{algorithm}
\begin{itemize}
    \item \textbf{Require:} \( f(n;m) = s(n)(w(m - n) \text{ and the initial energy of the error } \xi^0(m) = r_s(o;m) \)
    \item \hspace{1em} \textbf{for} \( l = 1, 2, \ldots, M \) \textbf{do}
    \item \hspace{2em} Calculate the reflection coefficient as
    \[ \kappa(l;m) = \frac{1}{\xi^{l-1}(m)} \left( r_s(l;m) - \sum_{i=1}^{l-1} \alpha^{l-1}(i;m) r_s(l - i;m) \right) \]. \hspace{1em} (2.8)
    \item \hspace{2em} Calculate the LP coefficients for the current order, \( l \),
    \[ \alpha^l(l;m) = \kappa(l;m), \] \hspace{1em} (2.9)
    \[ \alpha^l(i;m) = \alpha^{i;m} - \kappa(l;m) \alpha^{l-1}(l - i;m) \] for \( i = 1, \ldots, l - 1 \). \hspace{1em} (2.10)
    \item \hspace{2em} Calculate the new energy of the error by,
    \[ \xi^l(m) = \xi^l(m)[1 - (\kappa(l;m))^2]. \hspace{1em} (2.11) \]
    \item \hspace{2em} Increment \( l \)
    \item \hspace{1em} \textbf{end for}
\end{itemize}
\end{algorithm}
The LP coefficients can be related to one another through a distance measure known as the Itakura-Saito distance. The Itakura-Saito distance is a maximum likelihood measure of the logarithmic spectral similarity between two frames of speech [Furui 2001]. If one speech frame is defined in terms of the LP coefficients by the denominator of the minimum phase system $\alpha$ and a second frame of speech by $\beta$, then the distance from the first frame to the second frame can be expressed

$$d[\alpha(m), \beta(m')] = \log \left( \frac{\beta^T(m')\tilde{R}(m)\beta(m')}{\alpha^T(m)\tilde{R}(m)\alpha(m)} \right), \quad (2.12)$$

where

$$\tilde{R}(m) = \begin{bmatrix} r(0;m) & r^T(m) \\ r(m) & R(m) \end{bmatrix} \quad (2.13)$$

The matrix $R$ is the autocorrelation matrix of the speech signal comprised of autocorrelation vectors $r$. By nature, the Itakura-Saito distance is not a symmetric measurement. In other words, the distance from $\alpha$ to $\beta$ is not equivalent to the distance from $\beta$ to $\alpha$, which is a result of the autocorrelation matrix being derived for one particular set of LP coefficients. To overcome this, a symmetric version is often employed

$$d[\alpha(m), \beta(m')] = \frac{1}{2} \left[ d[\alpha(m), \beta(m')] + d[\beta(m'), \alpha(m)] \right]. \quad (2.14)$$

Cepstral Analysis

Cepstral analysis is a term given to the features of the speech waveform in a domain, called the quefrency domain, where the roles of time and frequency are interchangeable. Cepstral analysis takes advantage of the mathematical structure of the speech production system. The two major components of a speech waveform are the glottal source and the vocal tract shape. Since the glottis is vibrating at a rate
significantly faster than the rate at which the vocal tract is changing shape, the two components can be separated by taking a logarithm. This operation is performed on the frequency domain representations of the signal since the components of this model are multiplicative in the frequency domain and thus additive in the quefrency domain. Using the short-term discrete-time formulation of features, the cepstrum can be expressed formally as

\[ c(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left\{ \log \left| \sum_{l=m-N+1}^{m} f(l; m)e^{-j\omega l} \right| \right\} e^{j\omega n} d\omega. \] (2.15)

In effect, the high-time glottal component can be removed from the low-time vocal tract component by a filtering in the quefrency domain, a practice known as liftering. A graph of the real cepstrum and a liftering operation is shown in Fig. 2.7. The real cepstrum theoretically decays with increasing quefrency, so the liftering operation is often weighted. Researchers have successfully weighted these cepstral values to increase the recognition rates when using a Euclidean distance metric [Paliwal 1982].

Although a Euclidean distance is an effective measurement of closeness between sets of cepstral-coefficients, it is not ideal since some of the characteristics from the glottal excitation may still be present in the signal after liftering. To counter this problem, the LP-Cepstral coefficients were developed. The LP-Cepstra results from a recursive calculation of the complex cepstrum for linear predictive coefficients of orders 1 to \( M \). The complex cepstral coefficients \( \gamma(n; m) \) are defined as

\[
\gamma_\theta(n; m) = \begin{cases} 
\log \hat{\Theta}_0, & n = 0 \\
-\hat{\alpha}(n; m) + \sum_{k=1}^{n-1} \gamma_\theta(k; m)\hat{\alpha}(n - k; m), & n > 0,
\end{cases}
\] (2.16)

where \( \hat{\Theta}_0 \) is the magnitude of the LP model. These complex cepstral coefficients can be related to their real valued components by,
Figure 2.7 Low-time (left-most box) and High-time (right most box) liftering of a set of cepstral features taken from one frame of speech.

\[
\gamma_{\theta}(n; m) = \begin{cases} 
  c_{\theta}(0; m) = \log \Theta_0, & n = 0 \\
  2c_{\theta}(n; m), & n > 0 \\
  0, & \text{otherwise.}
\end{cases} \tag{2.17}
\]

The LP-Cepstra effectively uses two techniques to remove the excitation signal and therefore represents a smoother envelope curve. The ability to use Euclidean distance in assessing the quality of a set of cepstral coefficients is an appealing trait for pronunciation training. Justification for the use of Euclidean distance stems from the formulation of the Fourier transform. The Fourier transform decomposes a signal through multiplication by an orthogonal basis function and summation over time. The resulting function is comprised of orthogonal components. Taking the log magnitude of the Fourier transform maintains its orthogonality. The Euclidean distance function is a metric. Not only does it adhere to the properties that define standardized measurements, it adheres to an intuitive sense of distance.
Pronunciation Training

The goal of pronunciation training is to improve a given person’s L2 pronunciation. This ability is desirable in both human to human communications as well as human to machine communication. Just as people can have difficulty understanding a speaker with an accent, a speech recognizer has similar problems. Since the goal of pronunciation training is to train the speaker and not the listener, these problems are same with respect to this application.

Need for Pronunciation Training

Accented speech is the result of either a speaker’s inability to produce sounds native to L2 but not found in L1, or the speaker’s inability to perceive the difference between phonemes in L2 and L1. It is the goal of pronunciation training software to eliminate both of these sources of error. Since both of these sources are related, the speech signal can be transformed to a different domain. For this reason, feedback in a CALL system is often done visually, but auditory feedback has shown to be useful as well. The remainder of this chapter provides an overview of the techniques researchers have used to examine pronunciation training.

Overview of current techniques

There is a variety of techniques researchers use in developing pronunciation training software. The approaches can be divided into two categories based on the type of feedback given: auditory and visual. Both methods are equally valid, with the difference being the signal processing and classification techniques used.

Traditional auditory feedback methods focus on replaying a student’s utterance preceded by an utterance from a native speaker. It is the goal of the student to discern the differences between the two speech samples and adjust the articulator organs to compensate. One uncontrollable variable in the traditional method is the inability to eliminate the unique features of the native speaker’s voice from the particular part of speech being conveyed. This is mainly an issue in training the super-segmental
feature of pitch. More modern approaches, like those found in [Martin 2004], are focusing on using DSP techniques to measure the student’s fundamental frequency and modify it to fit the native model. The result is that the student hears his/her own re-synthesized voice with the correct intonation.

Visual feedback methods have been used more extensively than auditory feedback and can be implemented in a variety of manners. Primitive pronunciation training techniques output spectrogram images of a student’s utterance and of a native speaker’s utterance. The goal of this is to emphasize that there is in fact a difference between the two pronunciations [Caillaud and Leriche 2003]. Training of sub-segmental feature of voice on-set time (VOT) can be examined visually as well. Native English speakers can be shown to have a longer VOT than native French speakers for a given CV syllable. A time-amplitude plot of a speech frame containing both a consonant and a vowel conveys this information [Morton 1984].

Extending the basic principle of examining the frequency characteristics of an utterance has led to many attempts at displaying a graph of the first two formant frequencies [Gendrot and Adda-Decker 2004] and [Dowd et al. 1998]. These features were selected because of their ease of calculation and their relation to the position of the articulators. The extraction of these features is achieved by peak picking the first two resonances from a frequency-domain plot of a frame of speech.

Template based methods employing dynamic time warping (DTW) have also been used for speech recognition. In [Dalby and Kewley-Port 1999] researchers applied these techniques to pronunciation training concluding that template-based methods were successful because of their inherent distance measures.

Many researchers have also applied various pattern recognition techniques to pro-
nunciation training. Pattern recognition techniques usually use features such as the LPCs or the cepstrum. Recall that these features model the entire speech waveform, and not just the formant locations. Additional information like the locations of other formants and bandwidths can be included in these features. Pronunciation training has been successfully performed with statistical, syntactic and neural pattern recognition techniques.

Currently the majority of continuous speech recognition technologies use hidden Markov models (HMMs) to perform recognition. A HMM is a probabilistic state machine trained to determine to what extent a given utterance was produced by a sequence of states in the model. In [Precoda et al. 2000] visualization is achieved by quantizing log posteriori probability scores and outputting a bar graph of the result. In [Witt and Young 1997] researchers map the target language to the source language through both linear and Gaussian mixtures to yield a relational $a$ posteriori probability score. Methods of scoring super-segmental utterances of speech with HMMs are explored in [Teixeira et al. 2001]. In all cases, a higher score represents a more native-sounding pronunciation.

A more detailed visualization of classification results from a HMM recognizer for segmental features has been developed in [Yasushi et al. 2000]. Feedback information includes not only a graph of the F1-F2 plane, but diagrams of how the speaker can reposition the articulators to achieve the desired sound. This method has been successful in the training of Japanese learners of the English language.

Structural based pronunciation trainers have been implemented in [Asakawa et al. 2005]. Based on the acoustic universal structure of speech described in [Minematsu 2005], phonemes are connected in a tree structure that is robust at eliminating non-linguistic information. The author calculates the proficiency of pronunciation by determining the Bhattacharyya distance of the utterance to an element in the tree.

Neural-based pattern classifiers have been most successful in isolated word recognition. In [Kohonen 1988], Kohonen pioneered a clustering algorithm capable of
recognizing words in an unlimited vocabulary. These techniques are described in detail later.

More recently, Dr. Stephen Zahorian has led research implementing pronunciation training systems built on binary-pair partitioned feed-forward neural networks [Zahorian et al. 1997], [Zahorian and Venkat 1990], and [Zahorian et al. 1998]. The visualization of the results are made on a graph of the F1-F2 plane which is generated by adding a linear output layer onto the neural network classifier. To enhance the learning experience, this plane can be disguised in the form of a game [Zahorian et al. 2002].

The goal of this thesis is two fold. Firstly, the work of Zahorian and Kohonen is extended by developing pronunciation training software using modern unsupervised neural clustering algorithms (neural gas). Secondly, a novel visualization is developed that allows for a visualization assessment of both segmental and super-segmental accent features.
Neural networks are a biologically inspired computational structure that attempts to model parts of the human brain. This chapter introduces the artificial neural network and details the application of divergent architectures to the task of pronunciation training. In [Hawickhorst et al. 1995], three different types of neural architectures were explored in application of speech recognition: radial basis functions, learning vector quantization, and feed-forward neural networks. It was shown that a supervised training method had the best results. Here the supervised method is briefly explored as it leads to the formulation of the unsupervised training methods of self-organizing maps and neural gas algorithms. Radial basis function networks are then introduced for their relevance in function approximation.

In addition to the demonstrated success of neural network classifiers in previous pronunciation training experiments, such methods were chosen over statistical and structural methods, since the area of neural networks has given rise to robust unsupervised techniques. Unsupervised neural networks are capable of autonomously finding correlations in the data and have the potential to reduce the dimensionality of the problem-space. Since the goal of pronunciation training is to provide visual feedback on characteristics relevant to the degree to which an utterance sounds native, unsupervised approaches are justified. Training on a set of 12 to 15 cepstral coefficients, as developed in the previous chapter, provides the potential for a more precise classifier than traditional pronunciation training techniques have provided.

Understanding the biology of the human brain and applying mathematical models of its components are the inspiration for neural computing. The development of a supervised feed-forward neural network is presented and its relevance to pronunciation
training is explained. Next, unsupervised techniques of Kohonen’s self-organizing map and the neural gas are presented, completing the development of the unsupervised neural clustering algorithms applied in the following chapter toward pronunciation training.

Feed-Forward Neural Networks

The feed-forward neural network is a pattern classifier based on the discriminating capability of the human brain. Humans are able to classify a wide variety of signals in fractions of a second. The ability to recognize a face, understand what someone is saying, or react to a stimulus to the body occur significantly faster in the human body than it would take a computer using traditional computational methods highlighting the massively parallel computational structure of the human brain.

Brief History and Biological Basis

The human brain is an extensively interconnected network of cells, called neurons. Neurons are made up of three primary components: the cell body (soma), the axon, and dendrites. The neuron’s axon acts as an input to the soma. The dendrites act as outputs and connect to the axons of other neurons. Most cells have many dendrites. The connection between the dendrites and axons are called synapses and act as a channel for electro-chemical conductance.

In 1943, McCulloch and Pitts developed the first mathematical model of the synaptic process. Training of the neuron, described below, allowed for a basic planar classifier. It was soon discovered that networking these units together can increase the networks classification capability. The classic example is of a two neuron network being able to act as a binary, two input XOR function. A single neuron is unable to achieve this level of classification. The reason behind the improvement is that the second neuron extends the dimensionality of the network capability to that of a
Figure 3.1 Biology of the neuron

hyperplanar classifier [Schalkoff 1991]. This sparked much research into feed-forward neural networks.

During this boom in research, scientists explored other formulations of the neural network paradigm. Recurrent networks offered a method by which artificial neurons could perform the function of memory storage. Two different formulations of this type were developed, the Hopfield network and the auto-associative feed-forward network. Different types of neural units and their effects of reducing the dimensionality of the input space were originally explored by Kohonen in the 1980’s. This spawned the area of self-organizing structure which is described in detail later on. Time-delay neural networks were developed that allowed neural networks to be trained on the spacio-temporal data.

Despite a large research effort, neural networks have not been as widely implemented in practice due to the inability to guarantee the behavior of the network when stimulated by an input pattern that did not occur during the training procedure. However, neural networks have found a niche of application-specific pattern classifiers.

The Single Neuron

The general description of an artificial neuron derived from the McCulloch-Pitts model consists of an arbitrary number of input synapses, with each synapse being
associated with a weight vector, \( \mathbf{w} \). The inputs themselves are real valued numbers. The total input strength to a neuron is labeled \( \text{net} \) and the output, \( o \), of the unit is some function \( f \) of the net activation. Mathematically, for a neuron \( j \), this is described as

\[
o_j = f_j(\text{net}_j),
\]

(3.1)

where

\[
\text{net}_j = \sum_k w_{kj} i_k.
\]

(3.2)

Therefore, the total strength of the activation is a weighted sum of the inputs, and the ability of the neuron to act as a discriminator comes from the weight vector.

The output is dependent on this as well as the unit’s activation function. Several activation functions have been suggested, including linear, sigmoidal, and hardlimiter functions. Gaussian units are discussed at the end of this chapter as they characterize the radial basis function network. In terms of the net activation, the activation functions mentioned are defined as

\[
f(\text{net}) = \begin{cases} 
\text{net} & \text{linear activation} \\
\frac{1}{1+e^{-\text{net}}} & \text{sigmoidal activation} \\
\begin{cases} 
1 & \text{net} \geq 0 \\
0 & \text{otherwise} 
\end{cases} & \text{hardlimiter activation.}
\end{cases}
\]

(3.3)

Graphs of these functions are shown in Fig. 3.2. Sigmoidal activations are the most commonly used in feed-forward neural networks due to their function as a soft-limiter, as evident in the contour of the graph. Around the origin, the activation function is highly sensitive to changes in the net input, whereas away from the origin the unit
becomes saturated.

Figure 3.2 Possible activation functions for the artificial neuron. Figure (a) is a linear unit, (b) is a sigmoidal unit, and (c) is a hardlimiter.

Using this soft-limiting behavior, a training algorithm has been developed for the weight adjustments of the unit. The goal of training is to randomly initialize the weight vectors in such a manner that when presented with an input vector, the unit will be in the active region around the origin. Gradual changes in the weights of the unit will force the unit to be pulled into one of the saturated regions. A technique called Hebbian learning is applied.

Hebbian learning is a correlation-based learning method by which a characteristic matrix, such as that of the weighted linear synapses between neural units, can be expressed in terms of the input to the units and the unit's output. When presented with an input vector $\mathbf{v}$ that produces an output $\mathbf{o}$, the weight matrix is formed as the outer product of these two vectors:

$$W = \mathbf{vo}^T.$$  \hspace{1cm} (3.4)

The multiplication of values of the same sign indicates that the input stimuli support the result of the output, whereas negative products indicate conflict. A method for
adapting the weights results from differentiating an error function with respect to the weight; a detailed derivation can be found in [Schalkoff 1997]. The error function of the network is the sum of the squared difference between the target output and what was realized, mathematically,

$$E = \frac{1}{2} \sum_j (t_j - o_j)^2.$$  \hspace{1cm} (3.5)

The resulting adjustment for a single supervised artificial neural unit weight along the path from input $i$ to output $j$ can be calculated as

$$\Delta w_{ji} = \epsilon(t) \delta_j v_i,$$  \hspace{1cm} (3.6)

where the vector $t$ is the target output known a priori. The derivative of the activation function for sigmoidal activations can be expressed

$$f'(net_j) = \frac{\delta f(net_j)}{\delta_{net_j}} = \frac{1}{1 + e^{-net_j}}(1 - \frac{1}{1 + e^{-net_j}}) = f(net_j)(1 - f(net_j)).$$  \hspace{1cm} (3.7)

The function $\epsilon(t)$ is the learning rate of the network with respect to time and determines the rate at which the network converges. The term $\delta$ of equation in Eq. 3.6 is defined to be the sensitivity of the network weight due to the output value

$$\delta_j = (t_j - o_j)f'_j(net_j).$$  \hspace{1cm} (3.8)

This training method is known as the Delta Rule.

A single neuron, as shown in Fig. 3.3, acts well as a binary classifier, but the real power of neural networks lies in the interconnectivity of neural units. The classic
example of this has been shown in attempts at having a single neuron learn the XOR function. The XOR function can be expressed as the single bit binary output of a two input binary addition.

Figure 3.3 The artificial neuron

The Multi-layered Feed-Forward Neural Network

The classification ability of a system of individual neurons can be enhanced by interconnecting them with other neurons. The first extension of the single neuron model is to allow the input weight to be analyzed by a layer of neural units. Training of such a network is no different than the method described above.

Further recognition capability can be achieved by forming a cascade of neural unit layers. This multi-layered feed-forward structure can be trained in a Hebb-like style similar to that of the single layer and single neuron. However, the layered topology forces calculations of the weight adjustments to cascade backward from the output through the hidden layers to the input layer. Generally, the input layer is a linear layer that acts as a buffer for the input vector. This training technique is called back-propagation with the Generalized Delta Rule since it defines the Delta Rule when the output of a neural unit is not explicitly known.
Since the outputs of the last layer of the multi-layered, feed-forward network are the only unit outputs visible outside of the network, the training of the internal units’ weights relies on calculating a sensitivity of an internal unit to the network response. The connections between the input vector and the units on an internal layer are a weighted combination of output from units in the preceding layer. Also, the output of an internal unit effects the sensitivity of all units that it is connected to on the layer that follows it. The sensitivity for internal weight adaptation can be expressed in general as

$$\delta_j = f'_j(\text{net}_j) \sum_{\forall n} \delta_n w_{nj},$$

(3.9)

where the index $n$ refers to units on the following layer. Adaptations are usually made to the weights on an epoch-based timeline. An epoch is defined to be one iteration through the entire training set. The advantage of this method over adaptations for every input vector is that the net change to the weights is calculated as opposed to individual changes. Individual changes have the ability to be large in magnitude which can drastically alter the behavior of the network, whereas a net change yields
a smoother change. A summary of the training algorithm for a multi-layered feed-forward neural network is given in Alg. 3.1.

**Algorithm 3.1** Training of a multi-layered feed-forward neural network with backpropagation

1: Randomly initialize the weight vectors so that the output of the unit functions will be in the active region of the function (assuming sigmoidal)
2: **while** Error function in Eq. 3.5 has not converged to low error **do**
3:     **for** $i = 1$ to $N$, the number of input vectors **do**
4:         Present input vector $v$ to the network
5:     **end for**
6:     Compute the weight adjustment to the output layer using Eq. 3.6
7:     **for** $k = 1$ to Number of layers **do**
8:         Use Eq. 3.9 to calculate sensitivities
9:         Compute the weight adjustment to the output layer using Eq. 3.6
10:     **end for**
11:     Accumulate the weight adjustments for each weight
12: **end while**

**Properties of Feed-Forward Neural Networks**

In addition to the learning rate function of Eq. 3.6, there are several other useful adaptations and extensions of the feed-forward neural network that have enhanced the classification ability of the system. In obeying a stochastic gradient decent over an error-surface of unknown contour, it is possible to saturate the network units at a sub-optimal point in the error-space. This occurs when a local minimum is found which is not the desired global minimum. To combat this occurrence, a momentum term can be added to the adaptation rule so that it can be reformulated

$$
\Delta w_{ji}(t + 1) = \epsilon(t) \delta_j v_i + \alpha \Delta w_{ji}(t) \tag{3.10}
$$
where the term added takes into account a brief history of the gradient decent. With adjustment to the gain term $\alpha$, the effect of momentum can be controlled. The basic principle is that local minima on the error-contour will not be ‘deep’ and momentum from the previous weight adjustment can overcome the height of the ‘hill’ on the other side.

A key trait of a successful feed-forward neural network is its ability to generalize rules from the input data. The most relevant application of the need for generalization is in the classification of noisy data. One technique that has been shown to aid the ability of the network to generalize rules from the input patterns is weight decay. Weight decay increases precision in classification by decreasing the weights of units that have little effect on the network output. Here the weight adaptation is added to a decaying version of the original weight using

$$w_{ji}(t + 1) = \beta w_{ji}(t) + \Delta w_{ji}. \quad (3.11)$$

Weight decay occurs on all weights in the network, but is only effective at eliminating weights with little sensitivity since weights that contribute more to the solution are reinforced by the original weight adaptation.

Relevance to Pronunciation Training

Feed-forward networks have been applied extensively to the problem of pronunciation training. In [Zahorian et al. 2002], [Zahorian and Venkat 1990], [Zahorian et al. 1998] and [Zahorian et al. 1997], researchers implemented what was termed a binary-pair partitioned neural network. A small multi-layered feed-forward neural network was trained for every combination of phonemes, yielding a system of 741 neural networks. In pronunciation training, when presented with an input vector, the network that was trained on similar speech features would have the highest output
in the ideal case. The output from the other units permits a linear mapping and a visual assessment of distance based on these outputs. With this base system, research was conducted in network training as well as visualization. The system proved to be effective when applied to pronunciation training for people with hearing impairments.

When applying such a technique to the training of foreign accents, knowledge of the entire phonemic structure of both the target language and the native language is essential and the system of binary-pair partitioned neural networks trained accordingly. This structure is random and dependent on language. Therefore, to generalize the pronunciation training problem, unsupervised techniques are required.

Self-Organizing Maps
In the 1980’s, Teuvo Kohonen proposed a neural network architecture drastically different than that of those based on the feed-forward approach [Kohonen 1990]. As opposed to the neural network being characterized by linear combinations of weighted connections between units, each of Kohonen’s neural units is defined by a weight vector of equal dimensionality as the input vectors. Stemming from his work in learning vector quantization (LVQ), Kohonen proposed a self-organizing map (SOM) of neural units that is capable of mapping high dimensional data to lower dimensions while preserving the structure of the high-dimensional data. The SOM, or Kohonen map, is applicable to the analysis of data when the exact distribution of the data is not known, as in the case of an arbitrary language. The algorithm is therefore capable of autonomously determining relations in the training data. This section describes learning vector quantization, its relevance to speech processing and the relationship between LVQ and the SOM. The development and implementation of the SOM is detailed in relation to pronunciation training.

Learning Vector Quantization
The term learning vector quantization describes a technique that is used in the unsupervised clustering of data. As the name indicates, the aim is to autonomously
determine the set of vectors that will quantize a set of data. As in the C-means algorithm, learning vector quantization assumes a priori knowledge of the number of clusters that actually exists in the data. The appeal here for the speech processing engineer is that for a given language the number of distinct phonemes is known. In the English language there are 39 phonemes, 16 of which are vowel sounds.

The learning strategy of LVQ is different than that of a feed-forward network. Although the principle of Hebbian learning is still applied, a competitive learning strategy is employed. Competitive learning describes the technique that is used when learning is not guaranteed to occur for a given unit as it is in the feed-forward neural network approach. Recall that in the feed-forward neural network approach, all units are adjusted by a factor proportional to their sensitivity. The LVQ system is initialized with a set of \( c \) vectors of random values. As the input data is presented to the system, the distance between this vector and the \( c \) cluster centers is calculated. The closest unit to this vector has its weight vector adjusted by

\[
\mathbf{w}(t + 1) = \mathbf{w}(t) + \alpha(t)[\mathbf{v} - \mathbf{w}]
\]

(3.12)

where \( \alpha(t) \) is a learning rate function of time. The style of competitive learning used in LVQ is a winner-take-all, or hard-max, strategy. Learning only occurs for the winning unit. Convergence of the network to the set of ideal vectors is guaranteed when the learning rate is a strictly decreasing function. When this is the case, the effect of a single input on the network units diminishes. Learning vector quantization is applicable to both speech recognition and speech coding.

Formulation of a Self-Organizing Map

In formulating the self-organizing map, Kohonen extends the ideas of LVQ to include a soft-max learning strategy. This strategy can be described as a winner-take-most method, meaning that units other than the winning unit are adjusted. An additional goal is to achieve a dimensionality reduction so that the distribution of clusters
throughout $d$-dimensional space can be observed. Self-organizing maps are often represented as one, two, or three-dimensional structures. In deciding on a structure, some knowledge of the data may be useful. For example, if a set of three data clusters is distributed in three-dimensional space such that one cluster is a projection on another, in a line mapping of this data, the clusters may not be very distinct since they share common attributes in the higher space. In this case a two or three dimensional map might be more useful. In addition to determining the dimensionality of the mapping, the size of the network must also be taken into consideration. Too few units in a map can yield highly overlapped clusters where as too many clusters can be computationally expensive.

The network layout is a third consideration. The two most frequently used layouts in a two dimensional map are square layouts and hexagonal layouts, where each unit is connected to four or six other units, respectively. The winning unit is determined to be the unit with the smallest distance to the single training vector being presented to the system. This unit and its neighboring units are adjusted in the direction of the input vector. To determine the runner-up units for the soft-max training, a neighborhood function, $N_c(t)$, is to be defined. The neighborhood function defines a maximum Euclidean distance on the lower dimensional map for which all units less than this distance from the winning unit are adjusted. As with the learning rate in LVQ, the neighborhood of a unit decreases with time, a behavior that forces convergence of the network. The updating strategy for the unit weights can be expressed as

$$ w(t + 1) = \begin{cases} w(t) + \alpha(t)[v(t) - w(k)] & w(k) \in N_c(t), \\ w(t) & w(k) \notin N_c. \end{cases} \quad (3.13) $$

As in the LVQ algorithm, this adjustment obeys the principle of Hebbian learning. Presentation of the inputs is performed iteratively over many epochs. A summary of the training procedure is shown in Alg. 3.2.
Algorithm 3.2 Training algorithm for Kohonen’s Self Organizing Map

1: Design a sufficiently large map of predetermined layout of units of equal dimensionality as the system inputs. Randomly initialize unit weights.

2: for $i = 1$ to num\_epochs do
3:   for $j = 1$ to num\_input vectors do
4:     Present the network with input $v_j$
5:     Calculate the distance from $v_j$ to all units $w_k$
6:     Adjust all units in the network according to Eq. 3.13
7:     Decrease the size of the neighborhood function
8:   end for
9: end for

The Neural Phonetic Typewriter [Kohonen 1988] is a successful application of the self-organizing map for the purpose of phone recognition. Kohonen made clear the relevance of this technique to the problems of pronunciation training and speech therapy. The Neural Phonetic Typewriter allowed for a visualization of continuous speech compared to the speech for which the network was trained. A user can therefore visually assess how well an utterance matches the set of ideal phonemes. An important conclusion from this work is that the classification takes into consideration all dimensions of the data and is not a method of extracting principal components (i.e., F1 and F2). Therefore, all of the spectral information contained in the feature aids in the classification. The dimensionality reduction and visualization is only a product of the topological arrangement of units in a visual space.

Neural Gas Algorithms

The neural gas algorithm was first introduced in [Martinetz and Schulten 1991]. The algorithm is an extension of Kohonen’s Self-Organizing Map where the physical topology of the neural units is eliminated. The neurons are free to move around in $d$-dimensional space, thus the term neural gas. As in Kohonen’s SOM, the neural units are defined by their weight vectors. However, in the neural gas network, a unit’s ‘neighborhood’ function is replaced by a receptive field. The receptive field relates
an explicit calculation of distance between the input vector to all unit centers. In effect, this allows for the number of neurons adjacent to a given node to depend on the distribution of data. Martinetz showed how the neural gas algorithm is able to align its units to simultaneously learn the distribution of a data set consisting of a uniform line distribution, a uniform planar distribution, and a uniform distribution in volume.

Martinetz states that the advantages of the neural gas algorithm over K-means clustering and the Kohonen map are threefold. First, the neural gas converges faster to low distortion errors, reducing the computation time to successfully train a given system. Second, when the neural gas converges, it achieves a lower distortion error for data it is clustering, indicating a more precise cluster. Third, the convergence of the network follows a stochastic gradient descent.

The computational structure of the neural gas bears strong resemblance to the SOM. Like the Kohonen map, neural units are described only by their weight vector $\mathbf{w}_i$ and are trained with a Hebb-like learning rule. When the network is presented with an input vector $\mathbf{v}$, the adaptation to the network weights is

$$
\Delta \mathbf{w}_i = \epsilon(t) f_i(D_v)(\mathbf{v} - \mathbf{w}_i),
$$

where $\epsilon(t)$ is the learning rate as a function of iteration $t$ such that

$$
\epsilon(t) = \epsilon_i (\epsilon_f / \epsilon_i)^{t/t_{\text{max}}},
$$

and $f_i(D_v)$ is the activation function of the distortion set $D_v$. The distortion in the network is defined to be the set of distances between input vector and every neural gas unit in the network, as expressed in Eq. 3.16.

$$
D_v = \{|\mathbf{v} - \mathbf{w}_i|, \text{for } i = 1, \ldots, N\}.
$$
After all of the distortions are calculated the distortion set is sorted in ascending order and referenced by a set of indices \( k_i \). The activation function is also a function of the receptive field as a function of time \( \lambda(t) \) and is defined as

\[
f_i(D_v) = \exp(-k_i/\lambda(t)), \quad (3.17)
\]

where,

\[
\lambda(t) = \lambda_i(\lambda_f/\lambda_i)^{t/t_{max}}. \quad (3.18)
\]

In other words, the adjustment to the weight vector is a change to the location of all units in the receptive field of the winning unit. The magnitude of the change is the product of the magnitude of the difference between the input unit and neural unit centers, and the learning rate. The direction of the adjustment is such that the neural unit moves toward the input vector. Algorithm 3.3 summarizes the training procedure for the neural gas algorithm.

Shortly after developing the neural gas algorithm, [Martinetz et al. 1993] detailed how the weight adaptation scheme forces a convergence of the network via stochastic gradient descent. The energy function that categorizes the neural gas network can be expressed as:

\[
E_{ng} = \frac{1}{2C(\lambda)} \sum_{i=1}^{c} \int (x)f(k_i(x, w)) \|x - w_i\|^2 \, dx, \quad (3.19)
\]

where

\[
C(\lambda) = \sum_{i=1}^{c} h_\lambda(k_i). \quad (3.20)
\]
and \( p(x) \) is the probability density function of the underlying dataset.

**Algorithm 3.3** Neural Gas Algorithm of Martinetz

**Require:** Initialization of weight vectors \( w_i, i = 1, \ldots, N \). Setting the initial and final sizes of the receptive field of the units \( \epsilon_i \) and \( \epsilon_f \) respectively. Setting the initial and final values of the learning rate, \( \lambda_i \) and \( \lambda_f \) respectively.

1: while Iteration counter \( t < t_{\text{max}} \) do  
2: Present the network with vector \( v \) from the dataset and compute the distortions \( D_v \) according to Eq. 3.16.  
3: Rank the distortions from least to greatest  
4: Adapt the weight vectors according to Eqs. 3.14-3.18  
5: Increment counter \( t \)  
6: end while

In [Questier et al. 2002], researchers applied the neural gas algorithm to the analysis of pharmaceuticals and compared the performances of the neural gas network to \( k \)-means clustering and the Kohonen map. The neural gas network was able to classify data better than the \( k \)-means algorithm while performing equally well as the SOM. The advantage of the neural gas over the SOM is in the reduced time it takes to converge to a solution. These results verify those of Martinetz. Questier et al. also concluded that the neural gas algorithm can offer just as good of a visualization of the data structure as a SOM. This can be realized by concatenating an additional mapping function to the unit centers.

The neural gas algorithm has been enhanced over the last decade by several researchers. Improvements have been in run-time as well as classification performance. In addition, different neural gas structures have emerged, such as the growing neural gas and the hierarchical overlapped network. Each of these variations on the neural gas algorithm benefit performance in some manner.

**Efficient Neural Gas**

Significant efforts have been made in decreasing the computational effort of convergence in the network [Atukorale and Suganthan 1998]. Several steps of the original
neural gas algorithm are replaced with the steps in Alg. 3.4. Specifically, the explicit calculation of distance for all neurons is replaced with an implicit ranking scheme. First, a metric for ordering data is defined as

$$m_i = \frac{d_i - d_{\text{min}}}{d_{\text{max}} - d_{\text{min}}}.$$  \hspace{1cm} (3.21)

The weight adaptation of Eq. 3.14 then becomes

$$\Delta w_i = \epsilon(t) h_i(m_i(v, W))(v - w_i)$$  \hspace{1cm} (3.22)

where

$$h_i(m_i) = \begin{cases} 
  e^{m_i/\lambda(t)} & \text{if } m_i < r\lambda'(t) \\
  0 & \text{otherwise.} 
\end{cases}$$  \hspace{1cm} (3.23)

The ranking is then performed implicitly through exponentiation of the metric during the weight adjustment step of the training algorithm, as in Eq. 3.23.

**Algorithm 3.4** Modifications to original neural gas algorithm to improve run-time.

Require: Replace steps of 3.3 with the following:
1. Determine the minimum and maximum values in the distortion set
2. Make adjustments to the weight vectors as in Eqs. 3.22 and 3.23

The improvement in run-time allows researchers to examine the effects of building a hierarchy of several neural gas layers. The aim of such a structure is to achieve a higher level of discrimination between classes and to achieve this classification on a reasonably small computational order of magnitude.
Hierarchical Overlapped Neural Gas

Stemming from the conclusions of [Atukorale and Suganthan 1998], a hierarchical-overlapped structured for supervised clustering was explored. As set forth in [Atukorale and Suganthan 2000] and analyzed in [Atukorale et al. 2000], the hierarchical-overlapped neural gas (HONG) algorithm can be applied to labeled pattern recognition problems. The hierarchical-overlapped structure is formed by adding several neural gas classifiers to the network on layers above the original layer. In other words, the network response from the original layer is input into the second level neural gases. The advantage of the hierarchical overlapped topology is that the network can further discriminate between input features that would be represented by similar neural units in the original formulation.

Enhanced Neural Gas

Limitations to the original formulation of the neural gas algorithms motivated research to increase the robustness of the algorithm. Common factors that affect performance of the neural gas network are the initialization of the network weights, ordering of the data sequences, and presence of outlying data points. To remedy these problems, the enhanced neural gas algorithm is presented [Qin and Suganthan 2005].

The enhanced neural gas algorithm adds a heuristic to the clustering algorithm that takes into account the history of cluster movement. The main goal of this heuristic is to reduce the ability of an outlying point to cause significant changes to the cluster center. Any input vector that attempts to shift the cluster center a significant amount has its effect decreased. Mathematically this can be developed as

\[
\Delta w_i = \epsilon(t) h_X(m_i(v, W)) \|v - w_i\| \frac{v - w_i}{\|v - w_i\|}.
\]

(3.24)

This formulation multiplies the Eq. 3.22 by one. This emphasizes the effects that
large magnitude differences between vectors has on the cluster centers. A heuristic weighting can be developed that reduces the sensitivity of the adaptation when the magnitude of one sample is much larger than the average. Formally, this can be expressed

\[ \Delta w_i = \epsilon(\text{iter})h_{\text{lambda}}(m_i(v, W))e^{\exp\left(-\frac{\|v - w_i\|}{\beta d^m(0)}\right)}\sigma_i(\text{iter})\frac{v - w_i}{\|v - w_i\|}, \] (3.25)

where \( \sigma \), the parameter used to limit the amount by which the cluster is updated is defined as

\[ \sigma_i^n(\text{iter}) = \sigma_i^n(t) = \begin{cases} d_i^n(t) & \text{if } \|v^m_t - w^i_{\text{iter}}\| \geq d_i^n(t - 1), \\ \|v^m_t - w^i_{\text{iter}}\| & \text{if } \|v^m_t - w^i_{\text{iter}}\| < d_i^n(t - 1), \end{cases} \] (3.26)

and

\[ d_i^n(t) = \begin{cases} \left(\frac{1}{2}[d_i^n(t - 1) + \|v^m_t - w^i_{\text{iter}}\|]\right)^{-1} & \text{if } \|v^m_t - w^i_{\text{iter}}\| \geq d_i^n(t - 1), \\ \frac{1}{2}[d_i^n(t - 1) + \|v^m_t - w^i_{\text{iter}}\|] & \text{if } \|v^m_t - w^i_{\text{iter}}\| < d_i^n(t - 1), \end{cases} \] (3.27)

where

\[ d_i^n(0) = \left(\frac{1}{N} \sum_{j=1}^{N} \frac{1}{\|v^m_t - w^m_{iN}\|}\right)^{-1}. \] (3.28)

Developing the neural gas in this manner decreases the sensitivity to initialization. During the presentation of the first several input sequences to the network, the probability of an input vector being viewed as an outlier by the network is high, so the cluster centers will not move dramatically. The true underlying distribution of the
data will gradually pull the unit center into its correct location, as the location of the cluster is reinforced. Once the cluster centers have been well established, the effect of outliers will remain low. Applying such a technique is useful when the quality or the order of the input signals is unknown.

Growing Neural Gas
A further extension of the original neural gas algorithm is to allow the network to be able to adapt the number of units so that the final system neither over-fits the data nor remains sparse. Fritzke introduced this concept with respect to growing cell structures in [Fritzke 1992]. A networked pair of cells are permitted to move and spawn new cells while learning the topology of the underlying data structure.

An initial network of two cells is created. Data vectors are iteratively presented to the network. The network error, as a function of the distance between the input vectors and the unit centers, is calculated. After a predefined number of iterations, a new cell is added to the network between the cell with the largest error and its neighbor (as defined by a connection) that has the largest error. The new cell becomes connected to these two preexisting cells. The neighborhood of a winning cell is defined to be any other cell that has a connection to the winner. The age of each cell in terms of the time it was last updated is stored. The age variable allows the removal of cells that are not active in the weight adjustment phase. This technique prevents the network from becoming over-fitted, while the growing mechanism ensures the data does not become under-fitted. The adaptation to the winning cell’s weights is defined to be

$$\Delta w_i = \epsilon (v - w_i),$$

(3.29)

where the value of $\epsilon$ is the learning rate for winning cell. The direct topological neighbors of the winning cell are also adjusted, but with a lower value of $\epsilon$. 
Fritzke later applied the concept of growing cell structures with a hard-max adaptation rule to the neural gas algorithm with a soft-max adjustment [Fritzke 1995]. Similar to the growing cell structures, a growing neural gas initialized with a small number of clusters increases its cardinality while minimizing the error function in Eq. 3.19. The Growing Neural Gas uses a competitive Hebbian learning adaptation, as previously defined, for weight adjustments. A summation of the algorithm used to implement the Growing Neural Gas is shown in Alg. 3.5.

Robust Growing Neural Gas
Successful implementations of networks utilizing these extensions to the neural gas algorithm exhibit good classification performance. Researchers have examined a Robust Growing Neural Gas (RGNG) and a Hierarchical Overlapped Growing Neural Gas (HOGNG) for both labeled and unlabeled data.

In the RGNG algorithm described in [Qin and Suganthan 2004], the authors combine the techniques for enhanced performance of the neural gas algorithm with Fritzke’s growing techniques. The major finding of this work is that the RGNG is able to determine the ideal number of clusters for the data set. There is a major distinction here between the network proposed by Fritzke and that of Qin. The aim of the GNG of Fritzke is to continually add units into the network so that the entire underlying topology is aligned with units in a manner that reflects the probability density of the underlying dataset. In this formulation if the network were allowed to saturate and become overtrained, there will be one neural unit for every point in the training set. The aim of the Qin algorithm is to determine clusters in the data. In other words, each unit in the network represents a grouping of many elements in the training set. The distribution of the underlying data is accounted for by the size of the receptive field of each unit. An additional parameter is added to the weight adaptation rule of the neighboring units that acts as a method to repulse two adjacent clusters. The justification for this is to prevent two units from identifying coincident clusters. The weight adaptation rule becomes
Algorithm 3.5 The Growing Neural Gas Algorithm

Require: Initialization of small population (2) neural units. Initialize the initial and
final learning rate values $\epsilon_i$ and $\epsilon_f$. Initialize the connection matrix $C$ to the null
set. Define the maximum allowable number of clusters as $\text{max_units}$. Define in
terms of number of iterations the length of time between growth stages, $\text{max_iter}$. 
Set the epoch counter $m = 0$ and the iteration counter $t = 0$. The total number
of iterations is therefore expressed as $\text{num_iterations} = mN + t$ where $N$ is the
 cardinality of the input set $\mathbf{V}$.

1: while number of clusters $< \text{max_units} \text{ OR error function } >$ some threshold do
2: Set $t = 1$
3: Assign $\mathbf{X} = \mathbf{V}$
4: for $m = 0$ to $\text{max_iter} - 1$ do
5: for $t = 0$ to $N - 1$ do
6: Select input vector $\mathbf{x}_{m}^{t}$ randomly from $\mathbf{X}$
7: Determine the winning unit and the runner-up
8: if A connection does not already exist as defined by $C$ then
9: Create a connection between the winner and runner-up units.
10: Assign an age of the connection to 0
11: end if
12: Adapt the weight vectors of all the units connected to the winner according
to the Hebb-like rule
13: Increment the age of all connections to the winning unit
14: Remove any connections with $\text{age} > \text{max_age}$
15: if There exists a unit with no connections then
16: Remove the connectionless unit
17: end if
18: Increment t
19: Remove $\mathbf{x}_{m}^{t}$ from $\mathbf{X}$
20: end for
21: end for
22: if $\text{max_units}$ has not been reached then
23: Determine unit with the largest accumulated error $w_{\text{max_error}}^{1}$
24: Determine which neighbor of $w_{\text{max_error}}^{1}$ has the largest error, label this
$w_{\text{max_error}}^{2}$
25: Add a new unit at the mid-point between these two units and create connec-
tions to these units from the new node
26: Remove original edge between $w_{\text{max_error}}^{1}$ and $w_{\text{max_error}}^{2}$
27: Decrease the error of the two original clusters and interpolate an error for
the new cluster
28: Decay the error associated with all clusters by a factor $\beta$
29: end if
30: end while
\[ \Delta w_i = \begin{cases} 
\epsilon_n \sigma_i(\text{iter}) \frac{(v-w_i)}{\|v-w_i\|}, & \text{if } w_i \text{ is the winner,} \\
\epsilon_n \sigma_i(\text{iter}) \frac{(v-w_i)}{\|v-w_i\|} + \exp\left(\frac{-d_{\text{adj}}}{\varsigma}\right) \beta \frac{d_{\text{adj}}}{|N_{\text{adj}}|} \frac{(v-w_i)}{\|v-w_i\|}, & \text{if } w_i \text{ is adjacent to the winner.} 
\end{cases} \]

(3.30)

where \( \varsigma \) determines the width of the exponentially weighted repulsion. The ranking function \( h(\cdot) \) from Eqs. 3.24 and 3.25 is no longer needed as the connection matrix defines the neighborhood of the winning unit.

The ideal cardinality of the set of growing neural gas units is determined by casting the error function in terms of a minimum description length (MDL). MDL is a measure borrowed from information theory that has been successfully applied to neural networks as cited in [Qin and Suganthan 2004]. The technique assesses the cost in terms of bits of encoding the neural model with a fixed precision. It can be expressed as

\[
\text{MDL}(V, W) = cK + N \log_2 c + \kappa \sum_{i=1}^{c} \sum_{x \in S_i} \sum_{k=1}^{d} \max \left( \log_2 \left( \frac{|x_k - w_{ij}|}{\eta} \right), 1 \right) + |O| K. \tag{3.31}
\]

The first term is the cost of using \( K \) bits to represent the \( c \) clusters. The second term is the cost of using \( \log_2 c \) bits as a reference for the \( N \) input vectors to the cluster centers. The third term is the cost of encoding the residual error caused by associating an input vector in the receptive field with a cluster center. The last term is the cost of using \( K \) bits to classify outlying data points as their own cluster. Mathematically, this can be determined using

\[ |O| = |\{ \Delta L < 0 \}| \tag{3.32} \]
where

$$
\Delta L = (N - 1)\log_2(c - \phi) + K
- \left\{N\log_2(c) + \sum_{k=1}^{d} \max \left(\log_2\left(\frac{||x_k - w_{ij}||}{\eta}\right), 1\right)\right\} - \phi K.
$$

\tag{3.33}

The term $\phi$ equals 1 if the unit weight, $w_i$, has only one input vector, $x$, in its receptive field, otherwise it is 0.

This technique for network training inherits the property of stochastic gradient descent from the growing neural gas. With iteration through the algorithm the cost of residual errors will decrease as new units are added to the network, as will the number of outliers. There is a clear trade-off between the gain of reducing residual error and the cost of adding new units. The point at which the minimum network error occurs is determined to have the ideal number of clusters according to the MDL principle. The use of MDL in conjunction with the enhancements of the RGNG have been shown to perform better than implementations of an MDL criterion with the original GNG.

**Radial Basis Function Networks**

The purpose of the radial basis function (RBF) network is to process data based on its spatial location in the network. The biological basis for these structures is derived from the localization of certain types of functions in the brain. For example, visual stimuli are processed in a part of the brain known as the visual cortex whereas auditory information is processed in a separate part of the brain known as the audio cortex. Although it is not understood for certain why this occurs, it is a well observed trait [Hawkins 2004]. Neural network engineers have made use of modeling RBF as a connection of neural units categorized by a weight vector and a Gaussian activation. The net input of unit $i$ can be expressed

$$
net_i = ||v - w_i||^2,
$$

\tag{3.34}
where, just as before, vector $\mathbf{v}$ is the input vector and $\mathbf{w}_i$ is the weight vector that characterizes the unit. The activation function is therefore

$$f(\text{net}_i) = e^{-\text{net}_i/\sigma_i^2},$$

(3.35)

where $\sigma_i^2$ is the variance of the $i^{th}$ Gaussian distribution. The fundamental difference between this type of activation and the activation of the sigmoidal and hard-limiter is that the RBF unit determines the proximity of the input vector to the unit. The linear input units defined in Eqs. 3.3 behave as a binary classifier. The training of a radial basis function network is significantly different from that of the method for training feed-forward networks described in Alg.3.1. It should be evident that the net activation of the RBF unit is the square of the distortions used in the NG and SOM. Therefore, the neural gas algorithm or the self-organizing map are valid methods of training radial basis function networks [Schalkoff 1997].

Often, a layer of weighted linear units is added to a RBF network so that the output of the units can be transformed into a meaningful value. Since the Gaussian activation function is a continuous function over the reals, all RBF units in the network will have some level of output for any input vector, although in an ideal situation one RBF unit will dominate. Figure 3.5 diagrams a sample RBF unit.

In [Leonardis and Bischof 1998], a RBF network of decreasing cardinality is shown to cluster data according to the minimum descriptive length principle. Fritzke also has proposed a structure for growing radial basis function units using the growing neural gas algorithm in [Fritzke 1996]. The size of the receptive field is determined by averaging the distances from one neural unit to all of the others and using this value as the variance for the RBF unit. This technique implies hyper-spherical receptive fields for all units.
Hybrid Techniques

Research has also been performed in combining the hierarchical and growing techniques to achieve better clustering. The techniques of growth and hierarchical overlapping neural gas networks and growing radial basis function networks is implemented in [Cao and Suganthan 2003]. The aim of the research is to categorize the type of motion found in a sequence of video images. The procedure applies an unsupervised GNG on the bottom layer of the hierarchical structure. The units are then transformed into RBF units which act as the input into the layers one level up on the hierarchy. The GNG algorithm is then applied to the units on the second layer. As before, the final classification of the input vectors is achieved by combining the output of the second-layer GNGs.

In [Doherty et al. 2005], researchers reformulated the hierarchical structure to produce a top-down tree of neural units. The TreeNG algorithm bears great similarity to the GNG algorithm. A tree structure is instantiated with a root node representing the entire set of GNG cluster centers. The hierarchy is formed through the evolution of the GNG where edge deletion and insertion are independent functions of time. When edges are deleted in the GNG such that two clusters are no longer connected to one another, a leaf node is added to the tree structure to represent the new cluster.
In the case of the first iteration, the root becomes a leaf and a new root node pointing to the two new leaves is created. The end result is a tree structure that is representative of the underlying distribution of data. Since growth is a function of time instead of error, the validity of the resulting structure is greatly dependent on the growth and decay parameters.

**Distance Measurements**

The clustering capability of any classification technique is greatly influenced by the distance measurement used in network training and assessment. Often when nothing is known about the data set, a Euclidean distance measurement is used. A Euclidean distance assumes that each component has an equal effect on the data and is independent of all other components. Developing a classifier with these assumptions can result in sub-optimal solutions. Exploring other distance measurements is often a recommended technique.

One distance measurement that accounts for statistical relations between components in the data vector is the Mahalanobis distance. The Mahalanobis distance measure is a generalization of the Euclidean distance. Formally, the distance between a vector $\mathbf{a}$ and a vector $\mathbf{b}$ can be defined

$$d(\mathbf{a}, \mathbf{b}) = \sqrt{[\mathbf{b} - \mathbf{a}]\Sigma^{-1}[\mathbf{b} - \mathbf{a}]^T}$$  \hspace{1cm} (3.36)$$

where $\Sigma^{-1}$ is the inverse of the covariance matrix of the data set. In the case where $\Sigma$ is equal to the identity matrix the Mahalanobis distance reduces to a Euclidean distance. When $\Sigma$ is diagonal, it reduces to a weighted Euclidean distance. In the general case, the off-diagonal terms represent correlations between elements in the dataset.

In pattern classification applications, the covariance of the data from the different classes is not known *a priori* and must therefore be learned through network
training. Gustafson and Kessel developed a technique to learn the covariance of the underlying dataset while performing network training with the fuzzy c-means algorithm [Gustafson and Kessel 1979]. The fuzzy c-means algorithm is similar to that of the crisp c-means algorithm where the distinction lies in the fact that datum may be partially contained in more than one class of data. At each phase in the algorithm when the new set of mean vectors are calculated, the covariance matrix of the data vectors that are associated with each mean is calculated as well. This matrix is then used to train the next iteration of the fuzzy c-means algorithm through use in the Mahalanobis distance. Improvements have since been made to the original Gustafson-Kessel clustering algorithm that provide a more robust calculation of covariance matrices [Babuska et al. 2002]. Such a technique would be required if the data set contains linearly distributed data. C-means algorithms are inherently supervised methods, since the number of classes must be known \textit{a priori}. The relation of Fritzke’s growing neural gas to fuzzy systems is described in [Fritzke 1997]. However, the relation between the Gustafson-Kessel fuzzy covariance matrix and a growing neural gas network remains to be explored. An advantage of a variable covariance matrix is that the receptive field of a neural unit is no longer restricted to a hyper-spherical shape, allowing for the possibility of a more precise classifier.

\textbf{Visualizations of Unsupervised Clustering}

Now that the principles of the self-organizing map and the neural gas are understood with respect to the clustering of data, their utility in visualizing the data is explored. The dimensionality of a self-organizing maps inherently provides a simplistic visualization of the data. However, when the actual distance between points in the map is of concern, as in pronunciation training, the map does not provide sufficient results. The reason for this is that in a rectangular lattice map, it is incorrect to assume that the Euclidean distance observed between neural units in the reduced dimensionality translates to a Euclidean distance in the higher dimensional space. Hence, the self-
organizing map only reveals the structure of the data. Other methods, such as vector projections, can be used to derive a more meaningful relationship.

In [Vesanto 1999], the self-organizing map cluster centers are projected onto an output space. Several visualizations of the output space are explored, such as one and two dimensional mappings as well as visualization based on color and on size of the units. The issue in creating the mapping to a space that can be visually assessed is that the mapping of unit centers in the higher dimensional space can be non-linear. To overcome this problem, a technique known as Sammon’s projection is utilized.

An efficient method for computing the Sammon’s projection was formulated in [Maetschke 2004]. Sammon’s projection defines an error function of the mapping between vectors in a high dimensional space $H$ to a low dimensional space $L$ as,

$$E(t) = \frac{1}{\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d_{ij}^h \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (d_{ij}^h - d_{ij}(t))^2}{d_{ij}^h}$$

(3.37)

where the variable $d_{ij}^h$ is the distance between units $i$ and $j$ in the high dimensional space and $d_{ij}$ is the distance between projections of vectors $i$ and $j$ in the lower dimensional space. Functions of the lower dimensional space are a function of time because it is this set of vectors that is to be trained in the mapping. The distance terms can be expressed by Euclidean distance such that

$$d_{ij}^h = \sqrt{\sum_{k=1}^{d} (w_{ik} - w_{jk})^2}$$

(3.38)

and

$$d_{ij}(t) = \sqrt{\sum_{k=1}^{d} (y_{ik}(t) - y_{jk})^2(t)}$$

(3.39)

where vectors $w$ represent the cluster centers determined by clustering and $y$ represent
the vectors in the dimensionally reduced output space. The training algorithm obeys a stochastic gradient descent when the vectors $y$ are adjusted by the following Hebb-like rule:

$$\Delta_i(t) = \frac{1}{n-1} \sum_{j \neq i}^{n-1} v_{ij}(t)$$  \hspace{1cm} (3.40)

where

$$v_{ij}(t) = \left( \frac{d_{ij}^h}{d_{ij}^l(t)} - 1 \right) (y_j(t) - y_i(t)).$$  \hspace{1cm} (3.41)

The point of convergence is simply a ratio of the error functions and a constant dictating the accuracy, $\epsilon$. Thus,

$$\frac{E(t+1)}{E(t)} < 1 + \epsilon.$$  \hspace{1cm} (3.42)

A summary of the algorithm is shown in Alg. 3.6. After training of this output function, each of the cluster centers in the high dimensional space can be represented in the lower dimensional space by simple indexing. The distance between vectors in the high dimensional cluster space is guaranteed to be proportional to the distance in the lower dimensional space.

**Algorithm 3.6** Simplified calculation of Sammon’s projection using gradient descent.

**Require:** Neural clustering algorithm has been formed a set of cluster weights $\mathbf{W}$
1: Initialize a set of vectors $\mathbf{Y}$ of equal cardinality as $\mathbf{W}$ and of lower dimension
2: while Error function does not satisfy Eq. 3.42 do
3: \hspace{1cm} Adjust the weight vectors $\mathbf{Y}$ according to Eq. 3.40 and Eq. 3.41.
4: \hspace{1cm} Calculate the resulting error function
5: end while
This nonlinear method of controlling the mapping from the cluster-space to the visual-space allows for a variety of visualizations to be used. Methods for pronunciation training previously explored have been related to F1-F2 plots. One difficulty of using a projection method for visualization is that the layout of the lower dimensional vectors must be pre-specified in order to imply relationships in the data such as an F1-F2 plot. This mapping is therefore a supervised technique.

A bar-graph interpretation of the output is effective for the truly unsupervised case. For an ideal classifier, only one RBF unit will have significant output. The outputs of the RBF units of a target and a test sample can be plotted against one another and their similarity assessed visually. This technique has been applied in pronunciation through the interpretation of HMM outputs, as previously mentioned.
The techniques covered in the two previous chapters are now combined to create functional pronunciation training software. The experiment is set up as follows. A database of sound samples is acquired and analyzed using the signal processing techniques described in Chap. 2. A set of cepstral features are extracted. The neural techniques developed in Chap. 3 are then trained on the features extracted from the database. The output of the network is converted into a visualization for interfacing the student with the system.\footnote{Software developed for this project can be found at http://people.clemson.edu/~jhecker/thesis.zip}

**Database**

**Corpus Design**

In developing a valid corpus for vowel pronunciation, several linguistic factors must be taken into consideration. Most importantly is the completeness of the list of words to be spoken. The list must include all vowel sounds. For the French language, there are 12 oral distinct vowels which will be the focus of the pronunciation training system. Linguistically, this reduces bias in the sounds the speaker is producing, but it also can affect the performance of the neural training algorithms. To ensure the success of the neural classifier, the number of utterances in each phoneme class should be equal. Even though the enhanced neural gas algorithm is intended to minimize the negative effects of clusters with different numbers of elements, we want to ensure that there is not a cluster that is discarded as noise.

Entire words are used instead of isolated vowels since in-word pronunciations are the most natural. In this corpus consonant-vowel (CV) and consonant-vowel-consonant (CVC) syllables are used. The amount of movement of the articulators...
<table>
<thead>
<tr>
<th>Phoneme</th>
<th>Word to be pronounced</th>
</tr>
</thead>
<tbody>
<tr>
<td>[a]</td>
<td>pattes, classe, gare</td>
</tr>
<tr>
<td>[o]</td>
<td>gâteaux, pâtés, pas</td>
</tr>
<tr>
<td>[ɛ]</td>
<td>cote, port, bonne</td>
</tr>
<tr>
<td>[ɛ]</td>
<td>rez, ces, des</td>
</tr>
<tr>
<td>[a]</td>
<td>rester, c’est, fitait</td>
</tr>
<tr>
<td>[œ]</td>
<td>je, petit, demain</td>
</tr>
<tr>
<td>[i]</td>
<td>pire, qui, parti</td>
</tr>
<tr>
<td>[o]</td>
<td>beau, côte, tôl</td>
</tr>
<tr>
<td>[œ]</td>
<td>neuf, seul, peur</td>
</tr>
<tr>
<td>[ɔ]</td>
<td>peu, pleut, deux</td>
</tr>
<tr>
<td>[u]</td>
<td>coup, tous, bouvez</td>
</tr>
<tr>
<td>[y]</td>
<td>bu, tu, queue</td>
</tr>
</tbody>
</table>

Table 4.1 List of words recorded for use in the training set (before randomization).

from the first consonant to the vowel effects the quality of its production. A diverse set of consonant frames is to be used to add diversity to the training set and capture the vowel sound in a variety of natural utterances.

Another important feature of the word list is ordering. The pronunciation quality of a given word in a list can in fact be dependent on the preceding words. If two consecutive words have the same vowel, the articulators might not retract to their natural position before the pronunciation of the second word. To ensure that no two sequential utterances had the same vowel, the word list is randomized according to phoneme class.

In this experiment, the randomized word list is repeated by the speaker three times. Added redundancy helps eliminate any mispronunciations by the speaker.

Sample Collection

All recordings for the training set are taken inside of a controlled acoustical environment booth, as shown in Fig. 4.1. The recording booth contains a microphone, headphones, computer mouse, and a chair from which the speaker controls the recording software. All cables attach to the computer via connectors in the wall of the booth.
A M-Audio Mobile Pre audio amplifier is used to interface the microphone’s XLR jack to the computer and acts as the primary soundcard for the computer. A computer monitor is placed outside of the window of the booth facing in, giving the speaker view of the recording software which is controlled totally by mouse. The speaker is instructed to sit 4 inches away from the microphone and to have the face of the microphone slightly tilted away from the speaker’s mouth. In pre-recording tests, this set-up proved successful in limiting large variations in the amplitude of the speech waveform and prevented any of the plosive consonants from causing the signal to be clipped. The preamplifier has an LED display illuminating if clipping occurs; any sample that clipped is rerecorded.

![Figure 4.1 Experimental setup for speech acquisition](image)

The recording software, as shown in 4.2, interfaces a randomized list of words with the user and offers the user the ability to record, playback and rerecord utterances registered during the session. The speaker is instructed to ensure clean pronunciation
Figure 4.2 A screenshot from the speech collection program

by playing back the utterance and rerecording it if necessary. When the speaker
presses record, a 1.5 second window begins during which the word displayed in the
center of the screen is to be pronounced. After this period of time has elapsed, a tone
is played over the headset to notify the user that recording has ended.

The entire corpus is comprised of five native French speakers, one French and
four Belgian. All speakers in the recording of the database are male and of regular
build. For the purpose of pronunciation training of American learners of the French
language, the differences between the French of France and that of Belgium are
negligible. In total 360 utterances were recorded and stored for processing.
Feature Extraction

Collection and processing of the training corpus is implemented using Matlab. The speech is sampled at 12 KHz, well above the minimum 8 kHz to avoid aliasing. The first step in converting the audio files into features useful in classification is to pass the speech through a pre-emphasis filter, $H(z)$,

$$H(z) = \frac{1}{1 - 0.97z^{-1}}. \quad (4.1)$$

This filter increases the energy of the high frequency components of the sample so that they are closer in magnitude to the low-frequency spectrum.

Since the goal of the software is to train the pronunciation of vowels, the portion of the waveform containing the vowel must be located and extracted from the 1.5 second utterance. Since vowels are power signals, a straightforward method of locating the vowel portion of the frame of speech is to extract the frame of speech with the highest short-term energy, defined as

$$E_s(m) = \sum_{n=m-N+1}^{m} s^2(n). \quad (4.2)$$

This method is not sufficiently robust in extracting vowels from frames of speech where the vowel is preceded by high energy plosive consonants like [p]. To improve the automated vowel extraction, a zero crossing measure is used in conjunction with the calculation of short-term energy. The zero-crossing measure is defined as

$$Z_s = \frac{1}{N} \sum_{n=m-N-1}^{m} \frac{|sgn[s(n)] - sgn[s(n-1)]|}{2} w(m-n), \quad (4.3)$$

where the function $sgn(.)$ is the algebraic sign of the argument, $s(n)$ is the speech
signal, and \( w(n) \) is the windowing function. The zero-crossing measure will be high when the speech signal contains a noise-like sound and low for segments containing voiced speech. Plosive sounds will have a relatively high zero-crossing measure compared to vowel sounds and can be effectively eliminated by thresholding. The method of combining short-term energy and zero-crossing measure is successful in extracting frames of speech containing only vowels. The validity of the sample was assessed manually.

Algorithm 4.1 Vowel location in CVC syllable

**Require:** A set of speech utterances, \( S \), has been recorded and sampled at 12kHz of duration 1.5 seconds. A \( N=360 \)pt Hamming window, \( w(n) \) is defined for \( n = 1...N \). Define \( \text{max\_index} \) as the index of the frame from which the feature will be extracted. Define \( \text{max\_energy} \) as the largest energy of any frame in the sample and set its initial value to zero.

1. for Index \( m = 1, N/2, N, \ldots, 1/(1.5s \times 12kHz) \). do
2. 
   Create the speech frame from sample \( s(m : m + N) \in S \) by \( f(1 : N) = s(m : m + N) \times w(n) \) for \( m = 1, N/2, N, \ldots, 1/(1.5s \times 12kHz) \).
3. Calculate the short-term energy, \( e \), in the frame as in Eq. 4.2.
4. if Energy \( e \) is greater than \( \text{max\_energy} \) then
5. 
   Set \( \text{max\_energy} = e \)
6. end if
7. end for
8. for Index \( m = 1, N/2, N, \ldots, 1/(1.5s \times 12kHz) \) do
9. 
   Calculate the zero-crossing measure for the frame as in Eq. 4.3.
10. if The energy in the current frame is 90% of the max and the zero-crossing measure is approximately zero then
11. 
   Set \( \text{max\_index} \) to the index of the current frame
12. end if
13. end for

After determining which frames of speech contain the vowel, it is then required to develop a feature representative of the frame. Both the cepstrum and the linear predictive coefficients are widely accepted features for speech recognition. For the experiments performed here, the LP-cepstrum have been selected because the success of a pronunciation training software hinges on the robustness of the distance measure.
Recall that the measure of assessing LP coefficients requires the augmented autocorrelation matrix from which the speech sample was extracted. Therefore the measure is influenced by the word from which the feature was extracted as well as the speaker who spoke the utterance. Also the distance and distortion measurements that define the original neural gas clustering algorithm are implemented with a distance measurement that is independent of sample statistics. A natural method for calculating distance measures for LP-cepstrum is using the Euclidean metric.

As developed in previous chapters, the short-term real LP-cepstra are the desired feature set for the neural pattern classifier. The windowed 360 point frame of speech is then zero padded to 512 points to prevent aliasing in the DFT.

**Network Training**

In determining the best clustering method for the data, a variety of techniques are considered. A simple c-means algorithm is implemented, with $c = 12$, to see if this simple technique is applicable to the problem. The cluster centers that results from an initial test did not obey the prediction that clustering would occur along phonemic boundaries. This behavior can be explained linguistically by the fact that vowel characteristics are often competing in nature. Mathematically, this trait is manifested as a highly overlapped feature space.

Overlapping data imply the need for a fuzzy membership with respect to a given cluster. Such characteristics, as described in the last chapter, are well suited for clustering by self-organizing map and neural gas.

**Self-Organizing Map**

Following the training prescription set forth in Alg. 3.2, the SOM can be trained using the speech features previously developed. A square 15 x 15 unit topology is selected for network training. The network was trained for 10,000 iterations, meaning that each of the features derived from the utterances are repeatedly presented to the
network. The learning rates from the winning unit are linearly decreasing functions of iteration on the range 1 to 0.001, while the learning rate for the neighboring units are linearly decreasing functions of iteration on the range 0.5 to 0.00005. The neighborhood function of the network is originally determined to be half the size of the neural map, as recommended in [Kohonen 1988]. This network function is decreased linearly with iteration until the neighborhood function only includes the winning unit. The samples from the ordered training set are presented to the network in a random manner so that the network is not originally biased toward the lowest indexed data class.

![Figure 4.3 Resulting topology of the SOM after training.](image)

After training, the only units that are consistently updated with the soft-max adaptation rule will be active when presented with a training sample. Table 4.2 indicates which units in the 15 x 15 grid are activated when presented with a sample from the training set. Each column represents a phoneme and each row an utterance of that phoneme. Each vowel sound was spoken in two different consonant frames, indicated by the horizontal line at the mid point in the table. It is easily observed that the SOM classifies the data by phoneme. Figure 4.3 shows a graph of the total activations of the neural units. Of importance here is the structural relationship between peaks in the graph. It is observed that all samples of the phonemes [i], [y], and
[u] fall along the first row of the network’s topological map. The samples for [a] and [o] fall along the bottom row. This trend emphasizes the dominance of the first and second resonant frequencies of the vowel utterances. A similar structure is observed in Fig. 2.6. Relating this structure to the IPA phonetic chart (Fig. 2.2) confirms the relation between the lip-rounding and tongue occlusion rules to these frequencies.

Table 4.2 Cluster numbers associated with known phonemic category in the SOM network. Each column represents a phonemic group and row represents an utterance of that vowel. The horizontal line further divides the utterances by word.

One significant observation from the results of the SOM is that there exists a distinction in the speech feature for vowel phones from utterances with different consonant frames. This result confirms that consonants have a great effect on the quality of adjacent vowels.
Robust Growing Neural Gas

The Robust Growing Neural Gas algorithm, as developed in Chap. 3, is also applied to the set of vowel features extracted from database of native speakers. The implementation presents 300 input vectors per growth stage to a network that is initialized with two random units. The predefined maximum number of clusters is defined to be 30 which is sufficiently large for a vowel pronunciation training system. The maximum age of a connection between two nodes is defined to be 100 iterations.

A constant Euclidean distance metric is used in the calculation of network variables. Attempts at using a Gustafson-Kessel fuzzy covariance matrix for distance calculations proved to be fruitless as the RGNG network was unable to grow. When a new unit is created, a covariance matrix equal to the identity matrix is used for the initial calculation of distance. During the network adaptation phase, new covariance matrices for each unit is recalculated by computing the covariance of all training samples for which that vector is the winner. With well separated artificial data with varying degrees of intracluster covariance, such a technique was observed to follow similar behavior as a constant Euclidean distance. However, initializing the covariance matrix of a new unit the identity matrix prevents grow with highly overlapped clusters. The off-diagonal terms of the other units’ covariance matrices force their distances to be less than that provided by the Euclidean distance. Therefore, a neural unit using a Euclidean metric for the distance calculation will never be the winning unit and will eventually be removed from the network.

A second covariance matrix initialization technique was implemented to combat the problems observed with the identity matrix. The new covariance matrix is initialized to the covariance of the winning unit. This technique did not prove sufficient for artificial data as the new units were restricted in the directions they could move, resulting in the inability to align with the underlying data topology. Therefore fuzzy covariance distance measure are not implemented in the current system.

Table 4.3 provides a similar interpretation of the network results as is given for the
SOM. Here similar trends in classification occur. The RGNG performs unsupervised classification based on a combination of phoneme category and word from which the feature is extracted.

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>$\alpha'$</th>
<th>$\epsilon$</th>
<th>$\theta$</th>
<th>$i$</th>
<th>$o$</th>
<th>$\rho$</th>
<th>$u$</th>
<th>$y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>10</td>
<td>23</td>
<td>6</td>
<td>7</td>
<td>10</td>
<td>26</td>
<td>19</td>
<td>9</td>
</tr>
<tr>
<td>21</td>
<td>10</td>
<td>23</td>
<td>6</td>
<td>7</td>
<td>15</td>
<td>26</td>
<td>19</td>
<td>17</td>
</tr>
<tr>
<td>21</td>
<td>9</td>
<td>23</td>
<td>6</td>
<td>2</td>
<td>10</td>
<td>26</td>
<td>19</td>
<td>9</td>
</tr>
<tr>
<td>21</td>
<td>10</td>
<td>23</td>
<td>6</td>
<td>7</td>
<td>10</td>
<td>26</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>21</td>
<td>10</td>
<td>23</td>
<td>6</td>
<td>2</td>
<td>10</td>
<td>26</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>21</td>
<td>9</td>
<td>23</td>
<td>6</td>
<td>2</td>
<td>10</td>
<td>26</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>18</td>
<td>10</td>
<td>17</td>
<td>6</td>
<td>2</td>
<td>10</td>
<td>26</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>21</td>
<td>10</td>
<td>17</td>
<td>6</td>
<td>2</td>
<td>11</td>
<td>26</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>21</td>
<td>10</td>
<td>17</td>
<td>6</td>
<td>2</td>
<td>11</td>
<td>26</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>22</td>
<td>22</td>
<td>23</td>
<td>10</td>
<td>7</td>
<td>2</td>
<td>24</td>
<td>20</td>
<td>11</td>
</tr>
<tr>
<td>22</td>
<td>22</td>
<td>23</td>
<td>10</td>
<td>7</td>
<td>2</td>
<td>24</td>
<td>20</td>
<td>11</td>
</tr>
<tr>
<td>22</td>
<td>22</td>
<td>23</td>
<td>10</td>
<td>7</td>
<td>2</td>
<td>24</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>17</td>
<td>18</td>
<td>23</td>
<td>7</td>
<td>7</td>
<td>23</td>
<td>24</td>
<td>20</td>
<td>11</td>
</tr>
<tr>
<td>22</td>
<td>18</td>
<td>23</td>
<td>7</td>
<td>7</td>
<td>23</td>
<td>24</td>
<td>20</td>
<td>7</td>
</tr>
<tr>
<td>22</td>
<td>21</td>
<td>23</td>
<td>7</td>
<td>7</td>
<td>23</td>
<td>24</td>
<td>20</td>
<td>9</td>
</tr>
<tr>
<td>17</td>
<td>22</td>
<td>23</td>
<td>10</td>
<td>9</td>
<td>23</td>
<td>24</td>
<td>20</td>
<td>15</td>
</tr>
<tr>
<td>17</td>
<td>18</td>
<td>23</td>
<td>15</td>
<td>9</td>
<td>23</td>
<td>24</td>
<td>20</td>
<td>15</td>
</tr>
<tr>
<td>17</td>
<td>21</td>
<td>23</td>
<td>15</td>
<td>9</td>
<td>23</td>
<td>24</td>
<td>20</td>
<td>15</td>
</tr>
</tbody>
</table>

Table 4.3 Cluster numbers associated with known phonemic category in the RGNG network. Each column represents a phonemic group and row represents an utterance of that vowel. The horizontal line further divides the utterances by word.

Figure 4.4 shows the MDL error as a function of network size. The minimum in the graph occurs at 26 units. However, only 20 of these units actually affect the network. This would indicate that if training were longer, the inactive units would eventually be removed by the network pruning techniques of the RGNG. Twenty units are still larger than the desired 12. However, it is acceptable seeing how the network makes distinctions based on consonant frame. This training set consisted of 24 words. Observing the classification of training samples as the network grows reveals that classification along phonemic categories is crisp when the network has
fewer units and emphasizes the dominance of the F1-F2 traits. As the network grew to have more units, the classification became increasingly dependent on the consonant frame. This result can be interpreted as showing how the network self-adjusts for the highly overlapped feature space.

Since a completely unsupervised pronunciation training system requires the cardinality of the set of network units to equal the number of phoneme classes (12 in French), a RGNG is trained for all phoneme classes but only with no variation of consonant frame within that class. As expected, the MDL error function reaches a well defined minimum at 12 clusters, as illustrated in Fig. 4.5. It is this system that can be used for unsupervised pronunciation training.

System Implementation

With the success of the pattern classifiers, the pronunciation training portion remains to be implemented. The visualization of classification is the bar-graph interpretation of data since this method is truly an unsupervised technique. The program functions as follows. First, the user creates a target utterance by pressing the appropriate button on the screen, as shown in Figs. 4.6-4.9. Upon doing so, a random utterance
is selected from the training set and played back over the computer’s speakers. The
target output is then displayed on the screen. The user than presses the record button
and attempts to pronounce the word that was just played. The output from the radial
basis function layer of the user is now displayed adjacent to that of the target selected
from the database.

If the two utterances are similar in pronunciation, then the output of the units
will produce similar outputs. Figure 4.6 shows a visualization of a native sounding
pronunciation of the phoneme [y]. Figure 4.7 shows a visualization of pronouncing the
phoneme [y] as [u], which is a common error among English learners of the French
language. The unit with the largest output corresponds to the phone’s class. As
previously noted, the sound [y] is well separated in phoneme space, a trait which is
accounted for here by only one large peak appearing on the bar graph. In the case
of vowels that are not well separated from similar sounds, more than one peak will
appear in the graph. When this is the case, examining the similarity of all peaks is
required. Figure 4.8 shows a native sounding pronunciation of the vowel [a], whereas
Fig. 4.9 shows the common mispronunciation of [a] as [α]. The option for the user to
replay both the target utterance and the previously recorded sample allows for both
auditory and visual interpretations of the results.
Figure 4.6 Visualization of a native sounding utterance [y].

Figure 4.7 Visualization of mispronouncing [y] as [u].
Figure 4.8 Visualization of a native sounding utterance [a].

Figure 4.9 Visualization of a mispronouncing [a] as [α].
CHAPTER 5
CONCLUSIONS

The unsupervised neural methods for vowel pronunciation training explored in this research provide interesting insights on the algorithms used as well as on the language for which the methodology was tested. This chapter will provide a discussion of the results stemming from the SOM and RGNG approaches to the problem.

Results of the Self-Organizing Map

The benefits of the SOM approach is that a rough visualization is achieved naturally through the topology of the units. This approach is the method implied by the ‘Neural Phonetic Typewriter.’ For explicit pronunciation training, each unit being represented by a radial basis function allows for a measurement of pronunciation accuracy. Radial basis functions are a natural method for this assessment since the size and shape of the receptive field of the units in the system are uniform throughout the topology. A square topology implies a $d$-dimensional hyperspherical receptive field.

By observing the graphical location of the units on the map, the structure and relation of vowels in the target language are revealed. The dominance of the first and second resonant frequencies in vowel sounds are apparent when the SOM is labeled with the phonemes of the features used in the network training. The location of the vowel sounds on the map is highly correlated to the location of the occlusion of the tongue and the spacing of the lips. This trend is verified by comparing the VOWELS section in Fig. 2.2 and Fig. 2.6 with the cluster locations in Fig. 4.3. The preservation of this structure allows for the lip rounding rules to be used as linguistic modifications to the user’s speech.
An advantage of the SOM approach over a RGNG approach, is that a SOM requires the selection of exemplars for initialization of the RBF output layer. An exemplar would be any unit for which a significant number of training samples are the closest. Therefore, the number of clusters can be constrained after network training. This is not the case with the RGNG network. Setting a high threshold for this value would force in-class differences to have little effect on the pronunciation system.

Results of the Robust Growing Neural Gas
Although the RGNG algorithm does not provide an innate visualization of the data, generating visualizations through bar graph plots of the RBF outputs is simple to achieve. Another effective technique in the assessment of neural gas results is by ensuring that the training samples from the same vowel sounds are all nearest to the same unit. In the original assessment of the results, it was observed that the algorithm had difficulty distinguishing between utterances of the phonemes \([a]\) and \([\alpha]\). From a pattern recognition perspective, this would appear to be an error in the classification. However, it has been shown that modern phonological trends show the disappearance of the phoneme \([\alpha]\) from common usage. Since all of the speakers used to generate the training set are not trained linguists, the results are valid as they show a current trend in modern French linguistics. A highlight of the unsupervised approach is that it can capture the evolution of a language. Although an explicit correction to the user’s speech is not available in the RGNG approach, it does provide an effective method in determining the degree to which the user sounds native.

Further Discussion and Future Works
The main implication of the results of the experiments described is that unsupervised neural clustering can achieve pronunciation training without any language specific information. The importance of the dimensionality reduction of the output visualization is that more information can be displayed at one time. Unlike the cumbersome plots of the formant frequencies, which occupies all of the visual space, using a color
to signify correct pronunciation in phoneme can be coupled with prosodic information for the simultaneous training of sub-segmental and super-segmental accent features. As segmental features are the most important in distinguishing between English and French phonemes, differences in the intonation of the speech can also identify the origin of a speaker even if the segmental features are native-sounding. Intonation is even more important in tonal languages like Chinese and Yoruba.

Continuing research in this area will examine the effects of a continuous speech corpus on the ability of the feature extraction to accurately extract vowels and on the ability of the clustering algorithms to discriminate between phoneme classes. Achieving pronunciation independent of language hinges on the availability of recordings of the target language. Since samples low-density languages are by nature more difficult to ascertain, the application of these techniques must allow for network training to be successful on a corpus that is easily obtained, such as from media broadcasts.

Also, extending these techniques to pronunciation training of consonants is important since many languages exhibit consonants that are not common in other languages. For example, the phoneme \( [r] \) exhibits a wide variety of language specific traits. In English, the \( [r] \) sound can be considered a back-vowel whereas the French phone is a noise-like unvoiced sound and the Spanish phone is front vowel-like sound that exhibits characteristics of unvoiced stops. An effective language independent system requires the consideration of all of these features.

In terms of algorithm development, future research should examine the relation of the fuzzy Mahalanobis distance to the RGNG network. Using an adaptive Mahalanobis distance for a neural unit’s distance measurements, changes the \( d \)-dimensional shape of the receptive field of the neural unit from a \( d \)-dimensional hypersphere provided by the Euclidean metric. This trait is relevant to the weight adaptation calculation in terms of defining which samples are closest to which units and affects the span of the cluster repulsion term. Also, an adaptive covariance matrix should naturally limit the effect of outliers in the data set, reducing the need for a historical
distance measurement. Formalizing these characteristics has the potential to increase the robustness and accuracy of the growing neural gas algorithm.
BIBLIOGRAPHY


