The Recognition of New Zealand English Closing Diphthongs Using Time-Delay Neural Networks

JOHN ROBERT KIRKLAND

A thesis presented for the degree of Doctor of Philosophy in Electrical and Electronic Engineering at the University of Canterbury, Christchurch, New Zealand.

January 1995

Abstract

As a step towards the development of a modular *time-delay neural network* (TDNN) for recognizing phonemes realized with a New Zealand English accent, this thesis focuses on the development of an *expert module* for closing diphthong recognition. The performances of *traditional* and *squad-based* expert modules are compared speaker-dependently for two New Zealand English speakers (one male and one female). Examples of each kind of expert module are formed from one of three types of TDNN, referred to as *basic-token TDNN*, *extended-token TDNN* and *sequence-token TDNN*.

Of the traditional expert modules tested, those comprising extended-token TDNNs are found to afford the best performance compromises and are, therefore, preferable if a traditional expert module is to be used. Comparing the traditional and squad-based expert modules tested, the latter afford significantly better recognition and/or false-positive error performances than the former, irrespective of the type of TDNN used. Consequently, it is concluded that squad-based expert modules are preferable to their traditional counterparts for closing diphthong recognition. Of the squad-based expert modules tested, those comprising sequence-token TDNNs are found to afford *consistently* better false-positive error performances than those comprising basic- or extended-token TDNNs, while similar recognition performances are afforded by all. Consequently, squad-based expert modules comprising sequence-token TDNNs are recommended as the preferred method of recognizing closing diphthongs realized with a New Zealand accent.

This thesis also presents results demonstrating that squad-based expert modules comprising sequence-token TDNNs may be trained to accommodate multiple speakers and in a manner capable of handling both uncorrupted and highly corrupted speech utterances.

Acknowledgements

This thesis, and the research that preceded it, would not have been possible without the advice, support and encouragement of many of friends, colleagues and organizations. I would like to thank my supervisor, Dr Kathy Garden, for her guidance during my research into speech recognition and assistance in making fruitful contacts with staff of the Linguistics and Psychology Departments. I am also grateful for the assistance afforded me by the other staff members of the Electrical and Electronic Engineering Department, particularly Helen Devereux (*speaker HD*) for her speech samples, Professor Des Taylor for reading much of this thesis and Dermit Sallis for his advice and assistance with hardware matters. My thanks also to Mike Surety and Dave van Leeuwen for their assistance with my computer related problems.

I am indebted to Catherine Watson for her unwavering support and encouragement during the preparation of this thesis. I would also like to thank the other members of the *speech group* - Tracy Clark, Andrew Elder, William Thorpe and Bill Kenedy - who, like Catherine, provided considerable insights into speech and its processing.

I am also indebted to Dr Nicola Woods (Linguistics) for the many fruitful discussions we had concerning speech from a Linguist's perspective and for reading and commenting on chapter 2. My thanks also to Dr Paul Russell and Dr Gillian Rhodes (Psychology) for introducing me to artificial neural networks.

I would like to acknowledge the financial assistance of Telecom New Zealand Limited and the University Grants Committee. I am also indebted to Rochester and Rutherford hall for their financial assistance and accommodation for the period I served as House Tutor. My thanks to the board, the staff (particularly Frank, Mea and Ray) and all the student residents during 1990-1992, for making life outside research worthwhile and interesting.

I am extremely grateful to my friends and fellow band members Chris Pemberton, Steve Money and John Crequer for keeping me sane and "in time", while conducting my research.

Finally, I would like to thank my family, Mum, Dad and Andrew for their continuing support and encouragement to finish this thesis!

Contents

List of Abbreviations

§	Section.
ANN	Artificial neural network.
BT_N	An expert module comprising a squad of *N* basic-token TDNNs.
CL	Clean.
dB	Decibel.
DFT	Discrete Fourier Transform.
ET_N	An expert module comprising a squad of *N* extended-token TDNNs.
Hz	Hertz.
kHz	Kilo Hertz.
log	Logarithm (base 10).
LP-TDNN	Large phonemic time-delay neural network.
LVQ	Learning Vector Quantizer.
max.	Maximum.
min.	Minimum.
MSD	Minimum significant difference.
msec.	Milli-second.
secs.	Seconds.
SNR	Signal-to-noise ratio.
SPL	Sound pressure level.
ST_N	An expert module comprising a squad of *N* sequence-token TDNNs.
TDNN	Time-delay neural network.
V	Voltage.

List of Symbols

/ /	Contains a phoneme symbol or phonemic transcription.
∞	Infinity.
α	The level of significance for a statistical test.
ε	A variable computed during Cochran's generalized Q-test.
η	An ANN's learning rate (it may use more than one).
A	The Agreement threshold.
$\boldsymbol{d}$	A vector containing the desired activations of an ANN's output nodes
$\mathscr{E}(p)$	The McClelland error of an ANN in response to $\boldsymbol{x}(p)$.
$\mathscr{E}_{av}$	The average error between an ANN's responses and those desired during training.
F_0	The fundamental frequency of vocal cord vibration.
F_i	The i^{th} formant.
F_{-3dB}	The filter cut-off frequency.
$F_{Nyquist}$	The Nyquist frequency (half F_{samp}).
F_{samp}	The frequency of digital speech signal sampling.
$\boldsymbol{o}$	A vector containing the activations of an ANN's output nodes.
o_j	The output of the j^{th} node within an ANN.
o_b	The output of the bias node within an ANN.
Q	The statistic produced by Cochran's generalized Q-test.
$Q_\alpha(\varepsilon,\nu)$	The critical value for Cochran's generalized Q-test.
T_{samp}	The time interval between digital speech signal samples.
v_j	The input to the j^{th} node within an ANN.
$h(n)$	The n^{th} sample of a Hamming window function.
w_{ij}	The weight value associated with the weighted connection joining the i^{th} and j^{th} nodes within an ANN.
$\boldsymbol{w}(t)$	A vector containing an ANN's current weights.
$\boldsymbol{w}*$	The weights solution obtained during ANN training.
$\boldsymbol{w}(0)$	The initial weights used at the commencement of ANN training.
$\Delta\boldsymbol{w}(t)$	A vector containing the current weight changes for an ANN's weights
$W(i)$	The i^{th} coefficient produced by the Waibel transform.
$\boldsymbol{x}$	A token suitable for an ANN.

Chapter 1

Introduction

Speech is perhaps the most frequently used and fastest means of communication between two or more human beings (Miller 1981). It is, therefore, not surprising that humans might desire to interact with other entities in their environments using speech. For example, conventional computer software, such as the word processor used to generate this document, might be enhanced both by being able to transcribe a human speaker's utterances and by being able to react to his or her spoken commands. Similarly, other machines with which humans interact may well benefit from a simpler interface afforded by *automated speech recognition*.

The central problem of concern in this thesis is the automated recognition of English utterances produced with a New Zealand accent. This problem is of interest for the following reasons. First, it is evident that recognition systems designed to suit one accent of a language such as English, may not be well suited to other accents of this language. Clarke (1993) found that systems using speech features best suited to the recognition of American English words, are less suitable for recognizing the same words spoken with a New Zealand accent. It is, therefore, desirable to develop recognition systems for each accent of a given language, rather than just for the popular accents for which speech databases exist.[1] The knowledge gained from this initial exercise may then be used to develop a general approach to the recognition of such a language and possibly to develop a system capable of recognizing all of its accents simultaneously.

Second, the automated recognition of New Zealand English speech is of interest because some linguists, like Holmes (1994), believe that changes occurring within the pronunciation of this accent, particularly in the vowels, are likely to occur in other accents of English in the future. Consequently, problems associated with the recognition of New Zealand English speech currently, may well emerge when attempting to recognize other accents of English in the future.

Third, the New Zealand accent, like other accents of English, contains diphthong phonemes whose realizations are difficult to recognize using traditional approaches based on

[1]The major accents of American English are popular when developing speech recognition systems as a consequence of the TIMIT (Zue and Seneff 1988) and other speech databases.

time-delay neural networks (TDNNs, see §4.2), due to their extended durations. Recognition of *closing diphthong* realizations produced with a New Zealand accent is further complicated by the significant overlap of spectral features exhibited by realizations of /ai/ and /ei/. As discussed in §5.1.4.1, these realizations may cause TDNNs like those proposed in Waibel *et al* (1989a) and Hataoka and Waibel (1990) to produce ambiguous responses.

To progress towards the recognition of New Zealand English using a system based on TDNNs, this work focuses on recognizing closing diphthongs realized with a New Zealand accent. Approaches based on three TDNN architectures are reported and compared, these being referred to as *basic-token TDNN*, *extended-token TDNN* and *sequence-token TDNN* (see §4.2.3). Basic-token TDNN closely resembles the network proposed by Waibel *et al* (1989a) and has been applied successfully in experiments attempting Japanese phoneme recognition (Waibel *et al* 1989a; Waibel *et al* 1989b; Miyatake *et al* 1990; Sawai 1991a; Minami *et al* 1991). Extended-token TDNN resembles the most successful TDNN proposed by Hataoka and Waibel (1990) for the recognition of American English diphthong realizations. In contrast to basic-token TDNN, extended-token TDNN uses longer duration tokens to better represent closing diphthong realizations. Sequence-token TDNN is a novel TDNN approach developed in this work and is intended specifically for the recognition of closing diphthongs realized with a New Zealand accent.

The next section presents a brief overview of the field of automated speech recognition. More extensive reviews may be found in Clark (1993), Roe and Wilpon (1993), Owens (1993), Picone (1993), Ainsworth (1988) and Dixon and Martin (1979). This is followed in §1.2 by a summary of the contributions made to the field of automated phoneme recognition by the work presented in this thesis. Finally, this chapter concludes with an overview of the content to be found in the remaining chapters of this thesis.

1.1 Automated Speech Recognition: An Overview

Automated speech recognition is described variously as a *pattern recognition/classification* problem, or as a speech-to-text conversion task (Roe and Wilpon 1993; Morgan and Scofield 1991; Church 1987). It is sometimes distinguished from the problem of *speech understanding* which requires the use of linguistic and other forms of knowledge to interpret the meaning of an utterance, in addition to identifying its form (Church 1987). Increasingly, however, this distinction is being eroded by the realization that accurate speech recognition may also require the use of such knowledge to supplement the information obtained from speech utterances (Church 1987).

In this thesis, *automated speech recognition* is defined generally as the process whereby tangible speech utterances are transformed into discrete abstract units through the

actions of a machine. It is assumed that a speaker's utterances partially represent the *abstract messages* he or she intends to communicate to one or more listeners. It is further assumed that such messages may also be represented by *models* utilizing *discrete abstract units* (see §2.2) and that these units may be represented within a machine. Ideally, the transformations applied during speech recognition convert tangible speech utterances into representations that are more concise, comprises fewer unique entities and are more suitable for machine manipulation. However, they also yield two fundamental problems; what abstract units should be used and how do these units relate to speech utterances and to one another?

In this thesis, a hierarchy of abstract units, representing progressively higher levels of abstraction between tangible speech utterances and the abstract messages they represent, is assumed. This hierarchy is borrowed from *segmental theories* of language and speech proposed by linguists and contains the fundamental abstract units referred to as the *phoneme*, the *morpheme* and the *sememe* (these are discussed in §2.2.1). It is assumed that speech utterances may be transformed into sequences of phonemes (the least abstract of the fundamental units) using pattern recognition techniques like those discussed in this thesis. It is envisaged that such sequences might then be used in conjunction with language and task-specific knowledge to derive more abstract representations incorporating morphemes and sememes. Ideally, transformation of speech utterances into sememes (the most abstract of the fundamental units) may permit their *meaning* to be represented within a machine, thereby approaching speech understanding. It is anticipated that such transformation will ultimately be necessary to achieve transcriptions of speech utterances comparable with those of a human listener, such as a court stenographer (Roe and Wilpon 1993).

Within the last twenty five years, approaches to automated speech recognition have included *isolated-word*, *connected-word* and *continuous-speech* recognition systems. In isolated-word recognition, recognition is achieved by matching utterances isolated by intervals of "silence" with pre-stored templates (these utterances may be words or short phrases, see Fu 1980). Connected-word recognition relaxes the need for utterance isolation, but limits utterances to a pre-specified vocabulary (Rabiner and Levinson 1981). Like isolated-word recognition, connected-word recognition is also achieved using pre-stored templates formed from isolated utterances. Continuous-speech recognition generally involves the recognition of *sub-word* units, such as phonemes, syllables, demisyllables or diphones, to permit large vocabulary recognition (Rabiner and Levinson 1981; Lee and Alleva 1992). This approach is considerably more difficult than isolated- or connected-word recognition, since the units to be recognized are less well defined acoustically (Rabiner and Levinson 1981). Systems based on all three approaches have been created for individual speakers (*speaker-dependent systems*) and for groups of speakers (*multi-speaker* and *speaker-independent systems*).

The recognition systems discussed in this thesis attempt phoneme recognition as a step towards continuous-speech recognition. Traditionally, it was anticipated that systems

attempting *automated phoneme recognition* would have to segment speech utterances prior to recognizing the phoneme realizations contained (see Rabiner and Levinson 1981). Fortunately, however, current phoneme recognition systems, like those discussed in this work based on TDNNs, do not require speech signal segmentation, except to select speech portions for system training (see §3.3). This is advantageous since the segmentation of continuous speech into sub-word units is an extremely difficult problem.

Figure 1.1-1 depicts the principal components of current automated speech recognition systems that utilize automated phoneme recognition. Following the acquisition of a speaker's utterance, feature analysis is undertaken to provide the information necessary for phoneme recognition, while eliminating irrelevant information like that arising from a speaker's environment (Roe and Wilpon 1993). Feature analysis is commonly used to produce sequences of spectral features representing the changes in a speech signal's spectral characteristics with time. From these sequences, sequences of phonemes are then derived using pattern recognition techniques. Finally, the phoneme sequences representing an utterance are transformed into traditional text or some other form of symbolic representation using language processing.

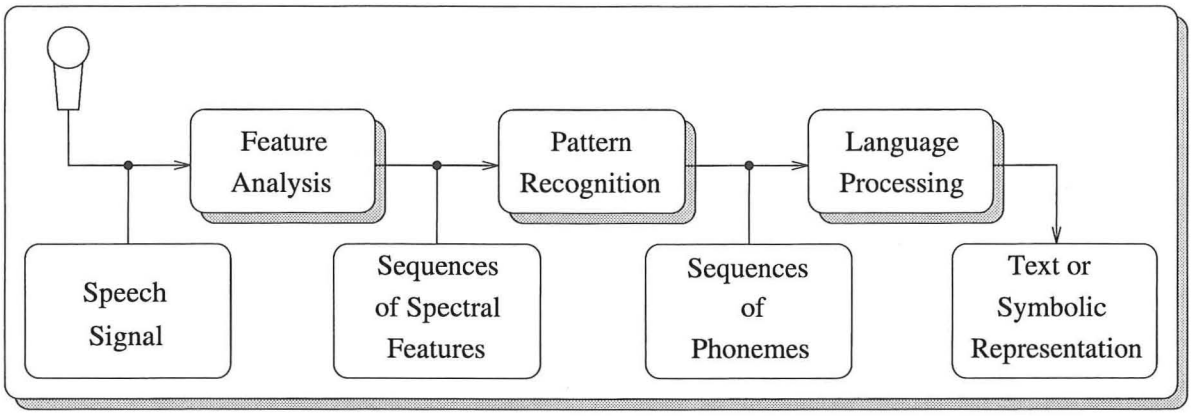

Figure 1.1-1 The principal components of current automated speech recognition system utilizing automated phoneme recognition. Based on Figure 1 in Roe and Wilpon (1993).

As indicated in Figure 1.1-1, pattern recognition plays a central role in automated phoneme recognition. The process of human, or natural, pattern recognition may be viewed abstractly as shown in Figure 1.1-2 (a). *Realizations* of an *object X* are given an appropriate *object index*, I_X, by a human who is modelled by an *opaque mapping*, $R_{op}(\boldsymbol{x})$. To emulate human pattern recognition by machine, $R_{op}(\boldsymbol{x})$ must be replaced by a transparent mapping, $R_{tr}(f(\boldsymbol{x}))$, that may be described precisely to a machine (Pao 1989). Within this expression, $f(\boldsymbol{x})$ transforms a realization, $\boldsymbol{x}$, into a pattern containing a set of *features*, referred to as a *token*. Ideally, each token is transformed by $R_{tr}(.)$ to produce the same object index as a Human would processing $\boldsymbol{x}$.

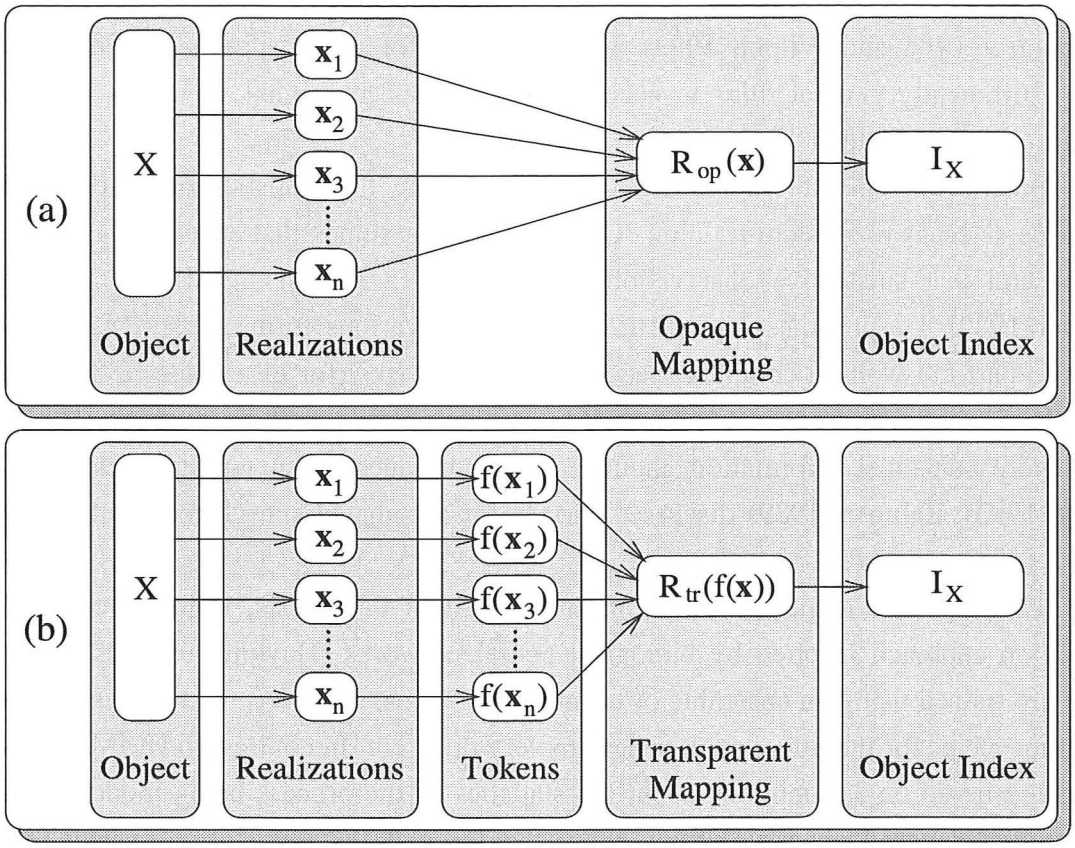

Figure 1.1-2. An abstract view of the process of pattern recognition when (a) conducted by a human and (b) conducted by a machine. Based on Figure 1.1 in Pao (1989).

Within the last 25 years, there have been four basic approaches to the recognition of patterns representing speech utterances (Roe and Wilpon 1993). Examples of these four approaches are referred to as *template matchers*, *rule-based systems*, *hidden Markov models* (HMMs) and *artificial neural networks* (ANNs). All four approaches have been applied to the problem of automated phoneme recognition, as discussed by Clark (1993), Ederveen and Boves (1991), Lee (1990) and Waibel *et al* (1989a), respectively. Usually, the transparent mappings "applied" by systems using these pattern recognition approaches are difficult to express mathematically. However, for each approach, the algorithm for "learning" or "implementing" a transparent mapping may usually be expressed mathematically and in a manner suitable for use by computer.

Template matchers attempt to find the closest match between input feature sets and feature sets "stored" as predefined templates. Unfortunately, such systems are sensitive to temporal distortions caused by variable speaking rate, unless this distortion is minimized using a technique like *dynamic time warping* (Roe and Wilpon 1993). Dynamic time warping minimizes the timing differences between two feature sets by warping the time-axis for one set so that the maximum coincidence is obtained with the other (Clark 1993).

Rule-based systems attempt pattern recognition using a series of rules forming a *decision tree* (Roe and Wilpon 1993). Unfortunately, for many speech recognition tasks, finding sufficiently general rules to accommodate all speech signal variations has proven difficult (Roe and Wilpon 1993).

HMMs attempt pattern recognition by modelling the statistics of an ensemble of utterances selected for system training. This modelling assumes that speech is a Markovian process and that successive observations (portions of a speech signal) are independent (Rabiner 1989). Despite these simplistic assumptions, various forms of HMMs have been shown to perform well on complex speech recognition tasks (for example, see Sagayama *et al* 1992). The principal advantage of HMMs over template matching systems is that they retain more statistical information about training utterances than templates do (Roe and Wilpon 1993). Rabiner (1989) discusses HMMs for automated speech recognition in more detail.

The approach to pattern recognition reported in this thesis, is based on ANNs, a recognition approach inspired by biological neural networks (Haykin 1994). Like HMMs, ANNs are trained using an ensemble of training utterances, however, the training algorithms and philosophies for these two approaches are very different. In contrast to HMMs which are formulated based on assumptions about the statistics of the process being modelled, ANNs are formulated based on assumptions about the *form* of the recognition system required (Bridle 1992). During training, an ANN's free parameters are altered to minimize its *output error*, rather than to *explicitly* model the statistics of its training data, as is the case in HMM training. ANN training is also competitive, forcing an ANN to form powerful *discriminate* functions for object recognition, rather than merely forming independent models of objects, as in HMM training (Song 1992).

Of the four pattern recognition approaches for speech recognition discussed, the approach using HMMs is currently the most successful and widely used (Roe and Wilpon 1993). Despite this, new ANN based approaches, such as TDNNs, are continually evolving and being shown to perform as well as, or better than, HMMs (see Waibel *et al* 1989a). Further, the properties of HMMs and ANNs have lead many researchers to combine both approaches in order to integrate their complementary advantages in one system (for example, see Robinson *et al* 1994; Devillers and Dugast 1994; Lippmann and Singer 1993; Le Cerf and van Compernolla 1993; Gao *et al* 1990). In particular, such combinations link the powerful discriminative ability of ANNs with the ability of HMMs to handle temporal variation.

Despite more than forty years of research in the field of automated speech recognition (Clark 1993; Ainsworth 1988), which has yielded several successful commercial systems in the United States, Canada and Japan, their still exists no system which can perform as well as a human in tasks such as court stenography (Roe and Wilpon 1993; Owens 1993). The work presented in this thesis continues that of other researchers using TDNNs for automated

phoneme recognition (see §4.2). A more detailed discussion of automated speech recognition using sub-word units is presented in §2.5, following on from the discussion of the phoneme in §2.2.2.

1.2 Contributions Made by this Thesis

Following Waibel *et al* (1989b) and Miyatake *et al* (1990), automated phoneme recognition may be achieved using a *modular TDNN*. As discussed in §4.2.2, such networks consist of several *experts modules* that are each trained to recognize a particular class of phonemes and an *arbitration module* that combines the responses of these expert modules. All these modules are formed from TDNNs, though the specific architecture used for each may vary. As a step towards the development of a modular TDNN for recognizing New Zealand English phonemes, this thesis focuses on the development of an expert module for closing diphthong recognition. In so doing, two main contributions are made to the field of automated phoneme recognition.

The first contribution is the development of a new method of using traditional TDNNs wherein *sequences* of TDNN responses are used to signify phonemes, rather than individual responses. This method is applied in the *sequence-token TDNNs* discussed in this thesis and contributes to the superior *false-positive error performances* of expert modules comprising these networks (see §5.1).

The second contribution is the development of a mechanism for *selective attention* for use with TDNN based systems attempting phoneme recognition. Instead of using individual TDNNs to form expert modules as is traditional, this thesis proposes using ensembles of identical and similarly trained TDNNs, referred to as *squads*, to form such modules. In contrast to their traditional counterparts, squad-based expert modules for closing diphthong recognition are able to correctly classify tokens representing these phonemes, while "ignoring" tokens representing phonemes from other classes (see §5.1.4 and §5.2).

1.3 The Remaining Chapters

The topics covered by the remaining chapters of this thesis are as follows.

Chapter 2 presents an overview of speech and automated speech recognition using sub-word units. Tangible and abstract aspects of speech are discussed and related with particular emphasis on the phoneme.

Chapter 3 discusses the speech utterances acquired for the experiments described in chapter 5. This chapter also discusses the speech preprocessing (feature analysis) used in this

work to prepare tokens for expert modules. Finally, chapter 3 discusses selecting speech portions from closing diphthong realizations to create training tokens for TDNNs.

Chapter 4 presents a brief overview of selected topics concerning multi-layer feed-forward artificial neural networks, before presenting a detailed description of TDNNs. In addition, a review of important experimental results concerning TDNNs for automated phoneme recognition is presented, along with a discussion of pattern recognition using *squads*.

Chapter 5 describes several experiments with various types of expert module for closing diphthong recognition. Traditional expert modules comprising individual basic-, extended- and sequence-token TDNNs are compared, *speaker-dependently,* with squad-based expert modules comprising the same types of TDNNs. To further test squad-based expert modules comprising sequence-token TDNNs, several *multi-speaker* experiments are also conducted.

Chapter 6 presents some conclusions and suggestions for further work.

Chapter 2

Speech and Automated Speech Recognition

Speech may be analyzed in many different ways. For instance, its production and perception by humans, or its physical properties as an acoustic waveform, may be analyzed. Alternatively, one may analyze the information it bears concerning the abstract messages it is intended to convey. The development of an automated speech recognition system requires an understanding of all these facets of speech and the relationships between them.

This chapter discusses speech, with emphasis on topics relating to automated phoneme recognition. The next section discusses some physical properties of speech. This is followed in §2.2 by a discussion of segmental linguistic theories concerning speech communication, including those concerning the phoneme. §2.3 and §2.4 discuss the phonology of New Zealand English and diphthongs, respectively, both of which are topics of central interest in this thesis. Finally, §2.5 discusses automated speech recognition in conjunction with sub-word units, with emphasis on automated phoneme recognition.

2.1 Physical Properties of Speech

The following sections present an overview of human speech production (§2.1.1), the properties of speech signals (§2.1.2) and human speech perception (§2.1.3). Frequently, as in this work, speech signals provide the raw materials for automated phoneme and speech recognition. How such signals are generated and perceived by humans provides insights into which of their meany features are of importance to their recognition.

2.1.1 Aspects of Human Speech Production

Speech is produced through the controlled, time ordered, gestures of a speaker's vocal organs. These gestures result in audible and visual cues which may be used by a hearer to discover a speaker's abstract message. Figure 2.1.1-1 depicts examples of both these types of cues. Humans are usually more adept at recovering a speaker's message using audible cues in isolation, than they are using visual cues in isolation. Consequently, it is commonly

assumed by engineers and linguists alike that a speaker's gestures may be represented adequately by the resulting audible cues alone. This assumption is convenient since speech signals, like that shown in Figure 2.1.1-1 (a), may be readily stored and manipulated by machine.[1]

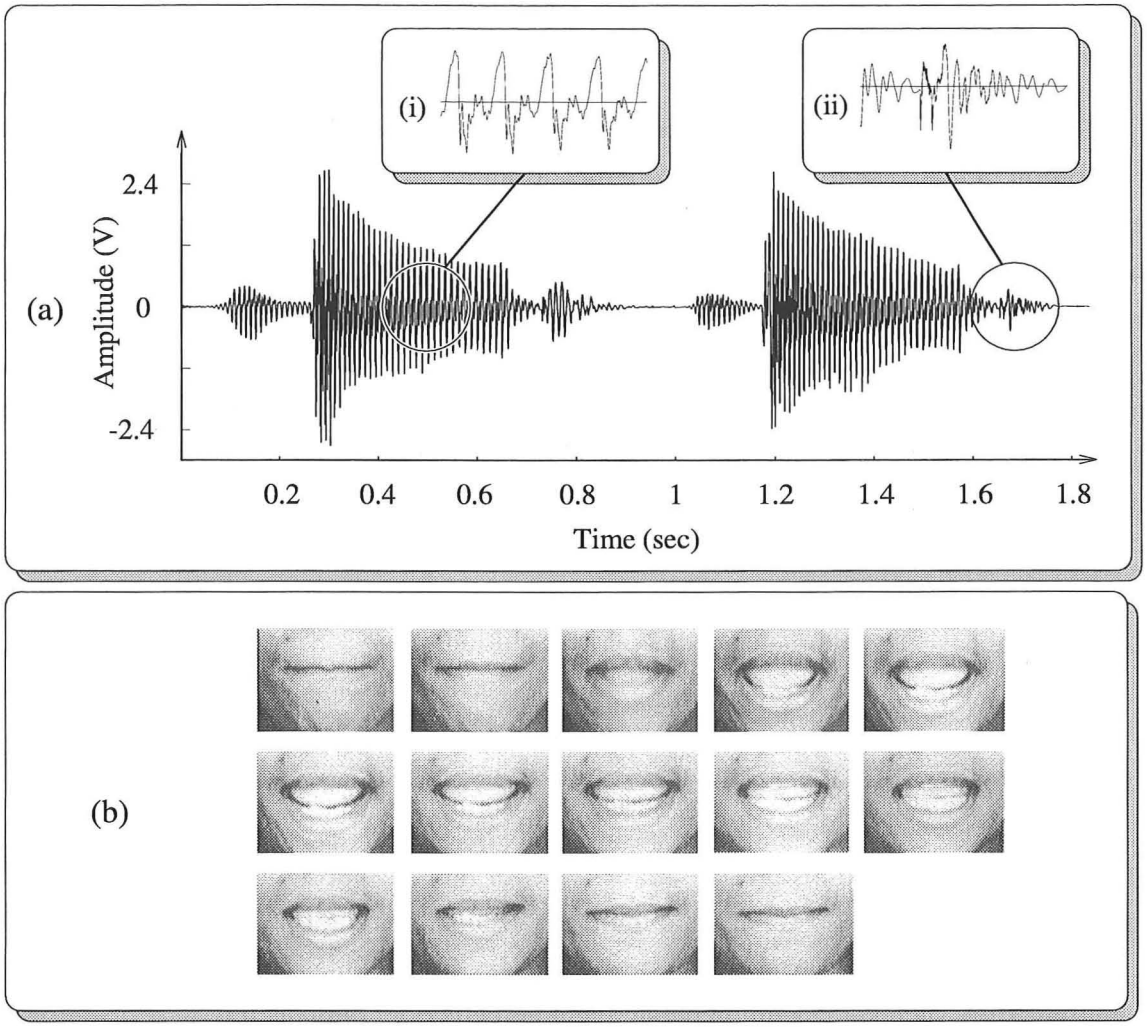

Figure 2.1.1-1. Examples of the (a) Audible and (b) visible cues associated with speech utterances. Part (a) shows a transverse wave representation of an utterance containing the words *bide* (/baid/) and *bade* (/beid/) in succession. Inserts (i) and (ii) show speech portions exemplifying a vocoid and a contoid, respectively. Part (b) shows snapshots of a speaker's lip movements whilst uttering the word *babe* (/beib/).

[1]Not all the articulatory gestures executed during the course of speaking result in audible sound (Singh and Singh 1976). *Prephonatory* articulatory gestures are required to position articulators at the commencement of an utterance, while *postphonatory* gestures return these articulators to their resting positions. Cues associated with such gestures are only available visually to a hearer and are, therefore, not available to automatic speech recognition systems utilizing speech signals alone.

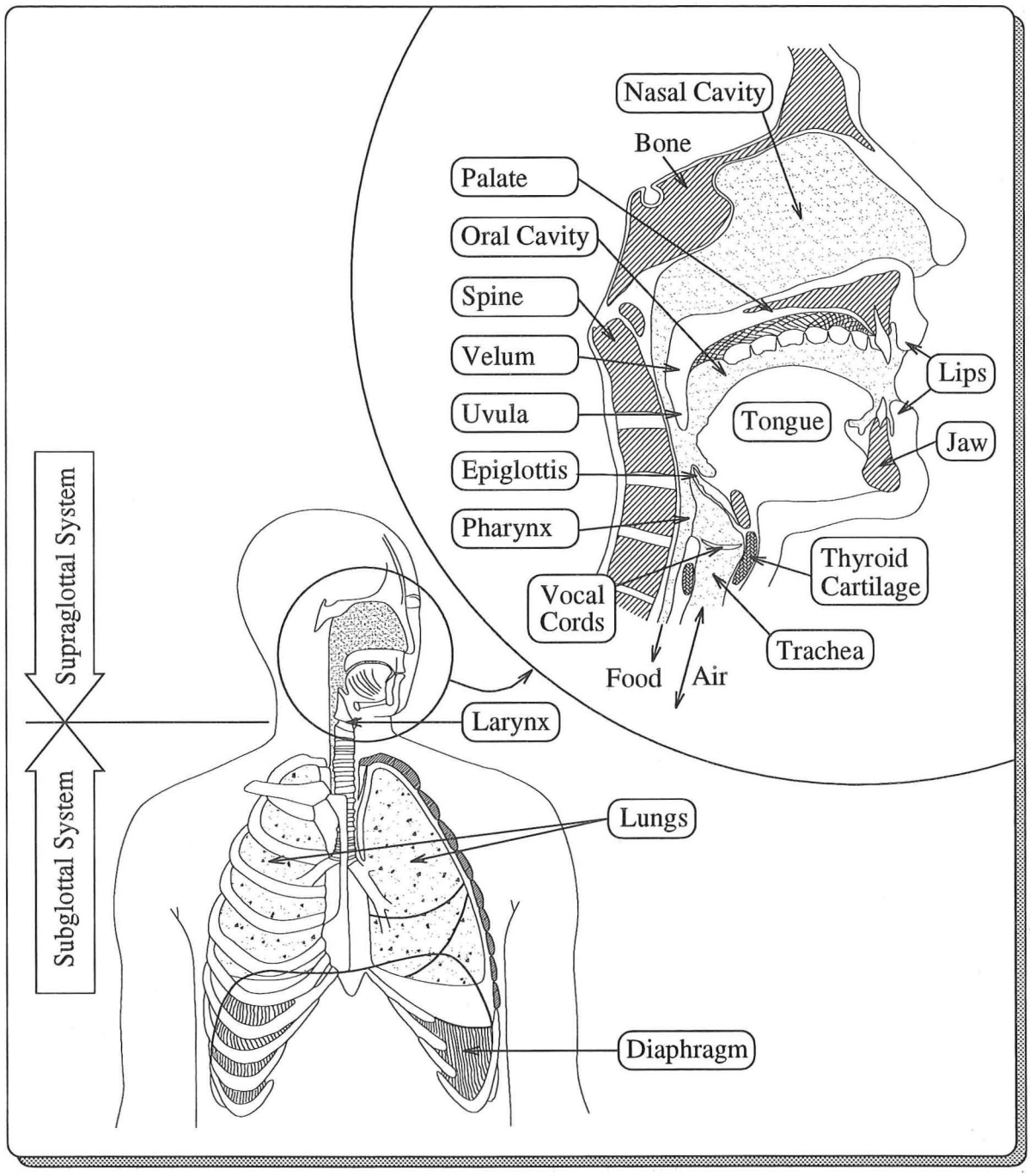

Figure 2.1.1-2. The primary components of a speaker's vocal organs. Air-flow may be initiated by the subglottal system including the lungs and the diaphragm. Speech sounds are caused by constricting or interrupting this air stream using the *articulators* of the *vocal tract* (the air passages above and including the larynx, Crystal 1980). Sound may be emitted from both the nasal and oral cavities (depending on the positioning of the velum) and may be initiated at more than one point in the vocal tract. Adapted from Miller (1981), Crystal (1980b), Clark and Yallop (1990) and Lieberman and Blumstein (1988).

Figure 2.1.1-2 shows the primary components of a speakers vocal organs. The movement of air required for speech is typically initiated by the *subglottal system*, including the lungs and muscles of the chest and abdomen (Fry 1979). Sound is generated by

interrupting or constricting this movement of air through the *supraglottal* system, usually while exhaling.[2] Such limitation of air movement may occur at numerous points within the *vocal tract* (the air passages above and including the larynx (Crystal 1980)) leading to sounds of differing qualities.[3]

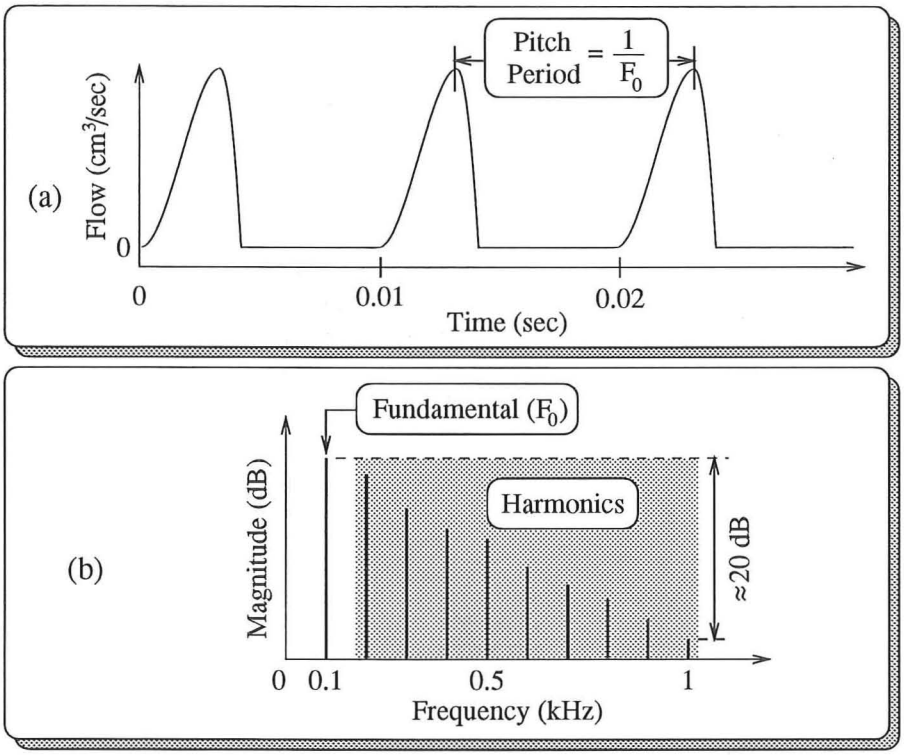

Figure 2.1.1-3. A simplified graph of air flow rate (*volume-velocity*) through the glottis (part (a)) and its spectrum (part(b)). Adapted from Clarke and Yallop (1990) and Lieberman and Blumstein (1988).

A significant number of speech sounds, including the sounds associated with the closing diphthongs studied in this work, are produced in conjunction with *vocal cord* vibration and are referred to as *voiced speech sounds* (Crystal 1980). The fundamental frequency (F_0) of vocal cord vibration may be controlled by a speaker and its perceptual correlate is *pitch* (Clark and Yallop 1990). Figure 2.1.1-3 (a) shows a waveform representing the air-flow through the *glottis* (the opening between the vocal cords, Crystal 1980) during normal voiced speech. Its spectrum (part (b)) is characterized by a series of harmonically related impulses commencing at F_0 (Hz) and spaced F_0 (Hz) apart. As shown in Figure 2.1.1-3 (b), the

[2]Sounds generated while exhaling air are referred to as *egressive* (Crystal 1980). Speech may also comprise sounds made while inhaling which are referred to as *ingressive*.

[3]*Quality*, in the sense used throughout this thesis, refers to the *timbre* of a sound resulting from the range of frequencies that constitute its identity (Crystal 1980).

magnitudes of these impulses are inversely related to their frequencies.

Speech sounds excited *only* by the vibration of the vocal cords are referred to as *vocoids* (Gimson 1989, Crystal 1980). Such sounds are often exemplified by those associated with the vowels of a language. All other speech sounds, such as those excited by constrictions or interruptions elsewhere in the vocal tract, are referred to as *contoids* (Gimson 1989, Crystal 1980). Contoids may be *voiced* or *unvoiced*, the latter being produced without vocal cord vibration (such contoids are also referred to as *unvoiced speech sounds*). The relationships between vowels, consonants, vocoids and contoids are discussed further in §2.3.

A speaker may shift his or her *active articulators*, such as the lips, tongue and lower jaw, relative to *passive articulators*, such as the palate and the upper teeth, to control the quality of sounds produced (Crystal 1980). Consequently, a speaker's vocal organs are capable of producing a continuum of sound qualities implying an infinite number of sound variants. A central issue for speech research is understanding how the sounds used by a speaker for speech are organized to assist the communication of abstract messages. A model of this organization is discussed in §2.2.

2.1.2 Some Properties of Speech Signals

Speech signals represented by transverse waveforms (see Figure 2.1.1-1 (a), for example), are derived from measurements of pressure variations caused by the longitudinal propagation of speech waves through air (zero in Figure 2.1.1-1 signifies atmospheric pressure). Often, such signals exhibit continually changing properties in response the continual fluid motion of a speaker's active articulators. This variation may be observed using "short-time" measurements of the sort discussed in §3.2.

For voiced sounds, in particular vocoids, the envelope of the short-time spectrum is characterized by peaks which are related to the resonant frequencies of a speaker's vocal tract (Lieberman and Blumstein 1988). Figure 2.1.2-1 depicts such a spectrum. The resonant frequencies of a speaker's vocal tract are referred to as the *formants* and are the frequencies at which his or her vocal tract permits the greatest transmission of excitation source energy. Notably, energy is not present at formant frequencies unless these are harmonically related to the fundamental frequency (Lieberman and Blumstein 1988). Despite this, such frequencies may be *estimated* using techniques such as *linear prediction* (Makhoul 1975).

Typically, frequencies spanning a range containing the first two or three formants are required for correct speech perception by humans (Lieberman and Blumstein 1988; Miller 1981). These formants are normally labelled F_1, F_2 and F_3, as in Figure 2.1.2-1, and may exhibit the frequency and bandwidth ranges listed in Table 2.1.2–1 (Witten 1982).

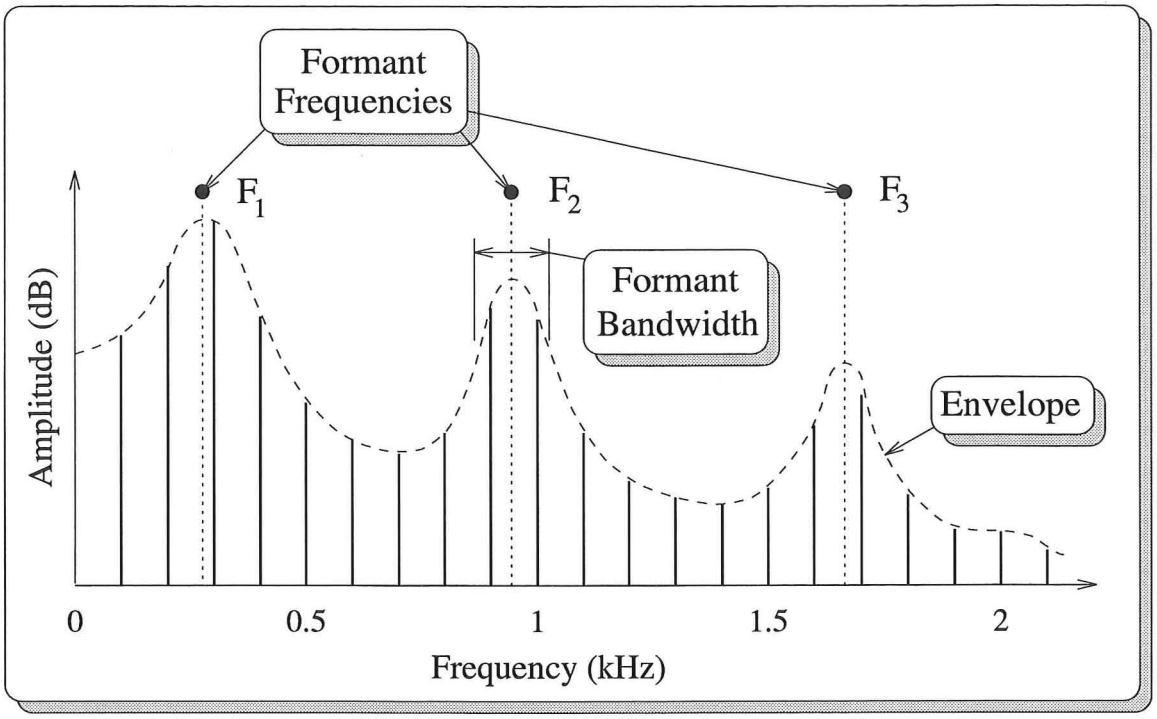

Figure 2.1.2-1. A typical spectrum of a vocoid. Energy is only present at frequencies harmonically related to the fundamental frequency which may not coincide with any of the *formant frequencies* (F_1, F_2, F_3,...) of a speaker's vocal tract.

	Formant Frequency Range (Hz)	Bandwidth Range (Hz)
F_1	100 - 1100	45 - 130
F_2	500 - 2500	50 - 190
F_3	1500 - 3500	70 - 260

Table 2.1.2-1. Approximate frequency and bandwidth ranges of the first three formants associated with human vocal organs. Based on Table 5.1 in Witten (1982).

During voiced speech, the fluid motion of a speaker's active articulators may produce smoothly varying formant frequencies. Observations of such variations, like those depicted in Figure 2.1.2-2, are referred to as *raw formant tracks* (Owens 1993). By "connecting" related formant estimates in these tracks, *smoothed formant tracks* may be obtained using a process known as *formant tracking* (see McCandless 1974; Seneff 1976). Vocoids, such as those associated with the vowels, may be characterized by the positions and movements of their formant frequencies. For example, the final sounds in the words *key*, *she* and *fee* (realizations of the phoneme /i/) are characterized by low first and high second formant frequencies as shown in Figure 2.1.2-3. Some authors refer to these characteristic frequencies as *targets* (Broad and Cleremont 1987; Crystal 1980). During normal continuous speech,

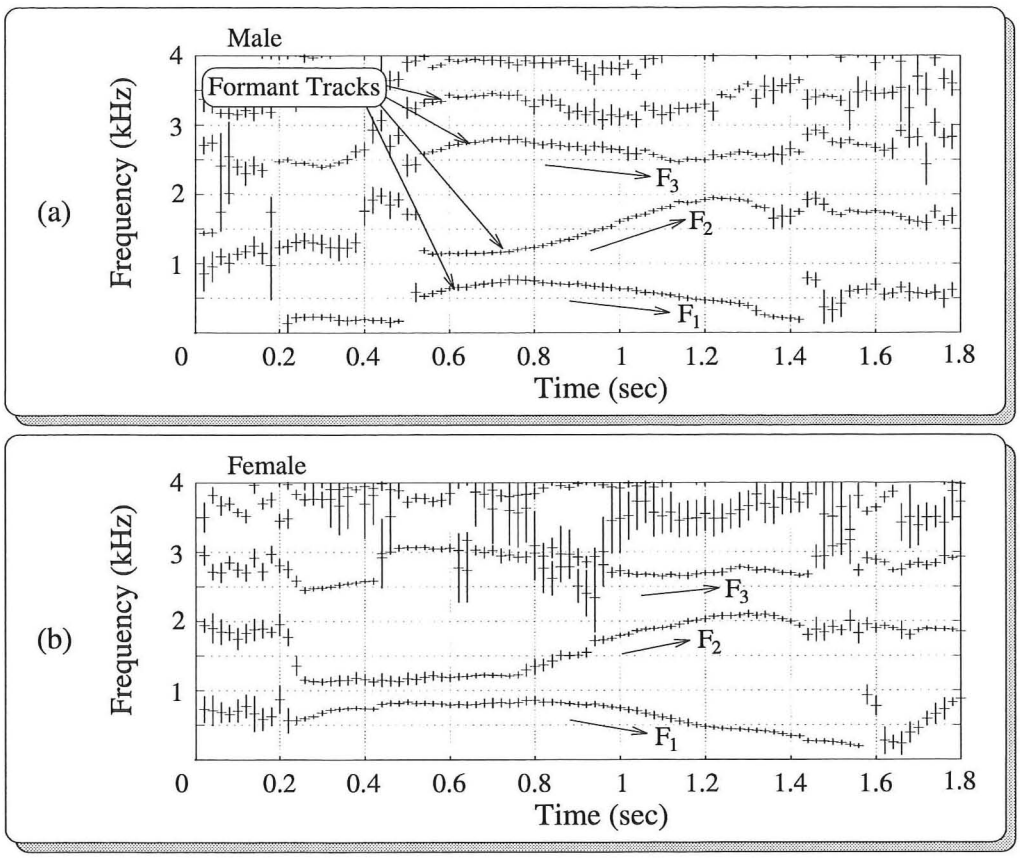

Figure 2.1.2-2. Raw formant tracks for the word *bide* spoken by (a) a male and (b) a female. The elements of each track are computed at discrete time intervals (corresponding to a time-shifted frame, see §3.2) and are represented by '+'s. The vertical members of each '+' indicates the bandwidth of their associated formant estimate.

formant transitions may fail to reach these target frequencies due to the constraints imposed on articulation by the production of adjoining sounds (Broad and Cleremont 1987).

Despite the information inherent in the transitions of the first three formant frequencies, it is uncommon for current phoneme recognition systems to rely solely on this information for phoneme identification (Hanes *et al* (1994) is one recent exception). However, formant tracks do provide one means of identifying speech portions for training token generation (see §3.3) and help to identify sounds which may prove difficult to distinguish (for example, the realizations of /ai/ and /ei/ studied in this work, see §4.2.3.1).

Figure 2.1.2-3 also shows how *speech segments* are defined in this thesis following Fant (1973). Sections of a speech signal bounded by "distinct changes in speech wave structure" are referred to as speech segments. These changes may be observed in several domains, the time and time-frequency domains being common. The relationship between speech segments, features and phonemes is discussed further in §2.2.3. To avoid confusion with speech segments, the phrase *speech portion* is used in this thesis to refer to an arbitrary section of speech signal.

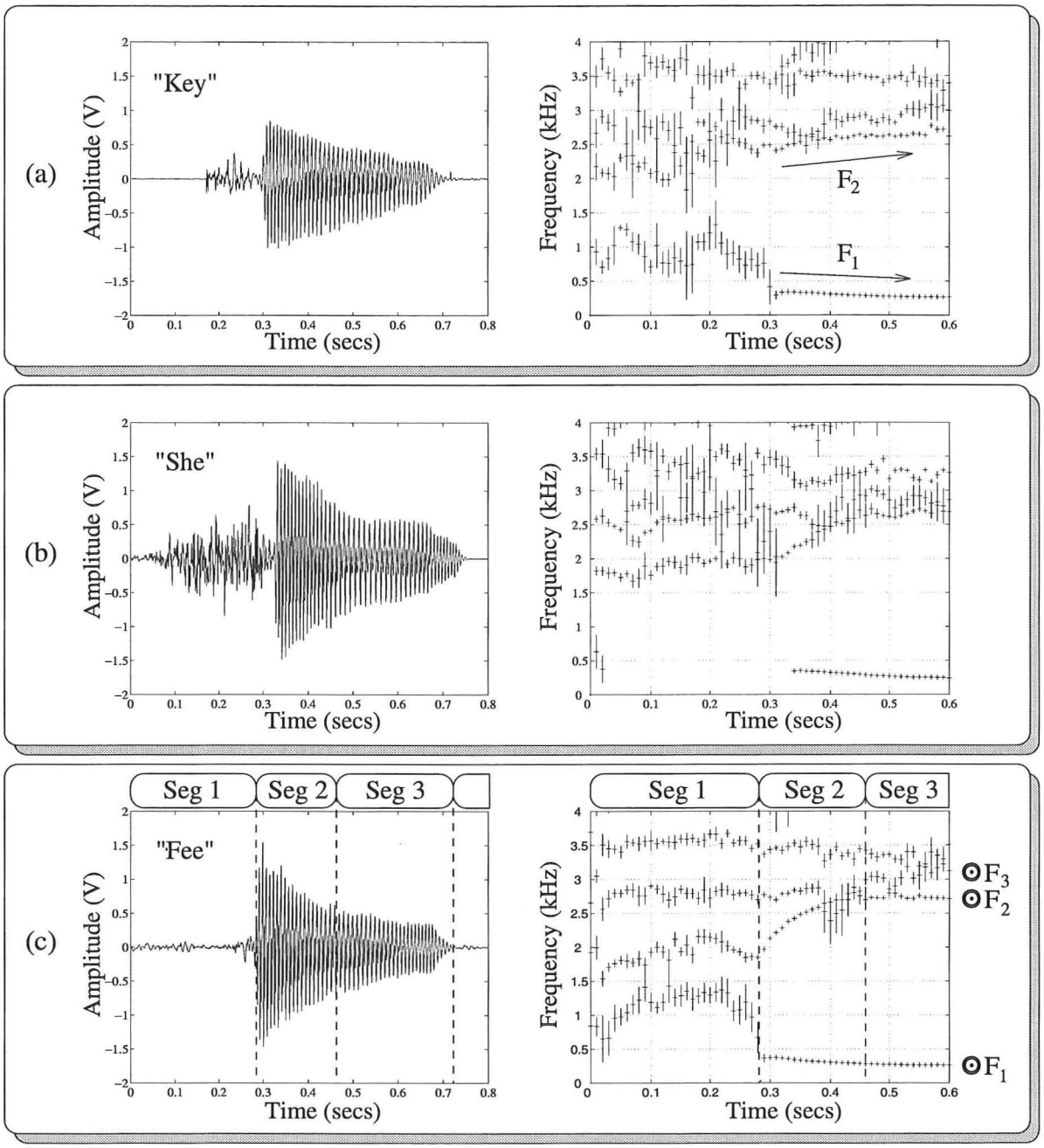

Figure 2.1.2-3. Speech signals (time-domain) and formant tracks (time-frequency domain) for the words (a) *key*, (b) *she* and (c) *fee*, showing how variation in the initial unvoiced sound (realizations of the phonemes /k/, /ʃ/ and /f/, respectively) produces varying cues in both domains. Each unvoiced sound (first 30 msec approximately) exhibits a characteristic "formant pattern" and affects the transition of formants leading into the neighbouring vowel sound (a realization of the phoneme /i/). Part (c) indicates the *target* frequencies for the first three formants which are ideally realized during production of this vowel sound (frequencies indicated by "bull's-eyes"). It also shows the partitioning of the signal associated with *fee* into *segments* corresponding to major "signal changes", such as the onset of voicing at the end of segment one (denoted *Seg 1*), or the completion of formant transitions at the end of segment two (denoted *Seg 2*).

2.1.3 Aspects of Human Speech Perception

During the preparation of speech signals for presentation to a phoneme recognition system, several transformations are typically performed in order to focus attention on features useful for phoneme identification (such transformations were denoted, generally, by *f(.)* in Figure 1.1–2). Some of these transformations are non-linear and are inspired by observations of the physiology and behaviour of the human auditory system. In particular, transformations which approximate the human *auditory spectrum* (Hermanski 1990) are desirable. This spectrum contains relevant information concerning speech sounds between approximately 100 Hz and 7 kHz (Lieberman and Blumstein 1988). Typically, only the magnitude of energy at these frequencies is considered, since the human auditory system is insensitive to phase differences between frequencies (Lieberman and Blumstein 1988).

Human perception experiments show that the frequencies forming a complex sound cannot be individually identified within a certain *critical bandwidth* (Picone 1993). This bandwidth is typically 10% to 20% of a complex sound's centre frequency. It is also evident that humans do not perceive frequency on a linear scale. Consequently, an approximate mapping between usual linear frequency, f (Hz), and a *mel scale*, like

$$\text{mel} = 2595 \log_{10}\left(1 + \frac{\text{f}}{700}\right) \qquad (2.1.2\text{-}1)$$

(Picone 1993), is often employed. This transform attempts to map f onto a scale which is linear *perceptually*. The critical bandwidth concept may be combined with a mel scale transform to create *critical band filterbanks*. Such a filterbank has bandpass filters linearly spaced along a *mel* scale with each filter having a bandwidth equivalent to the critical bandwidth associated with its centre frequency (Picone 1993). An approximate critical band filterbank comprising 16 filters is used for speech signal preparation in this work, following Waibel *et al* (1989a) (see §3.2). Notably, the frequency resolution of such a filterbank decreases with increasing frequency, f, due to the increasing critical bandwidths of its filters (Waibel 1989a).

In contrast to its decreasing frequency resolution, the human auditory system exhibits increasing sensitivity with increasing frequency and is most sensitive at approximately 3.3 kHz. This is particularly noticeable for quiet sound levels, as depicted in Figure 2.1.3-1 which plots perceived equi-loudness contours (in phons) with respect to sound pressure level (SPL, in dB). The variation in human auditory sensitivity may be crudely emulated by taking the logarithm (base 10) of magnitude spectrums prior to their use (*log compression*), or by more elaborate transformations such as those proposed by Hermansky (1990) (*perceptual compression*). Both approaches yield similar results for vocoids (of interest in this work), as demonstrated by the example depicted in Figure 2.1.3-2. In this example, the most significant

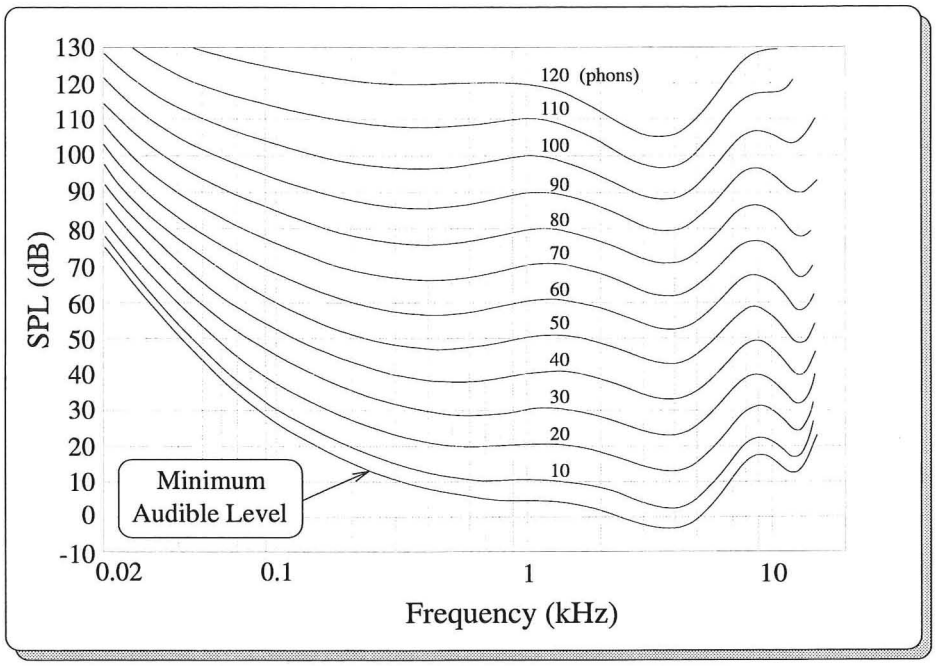

Figure 2.1.3-1. Contours of perceived equi-loudness (in phons) versus sound pressure level (SPL, in dB). The lowest contour corresponds to the minimum audible sound level and the remaining contours are traced such that phons=SPL at 1 kHz. From these contours, the ear appears to be most sensitive at approximately 3.3 kHz for all sound pressure levels. Based on figure in Davis and Davis (1987).

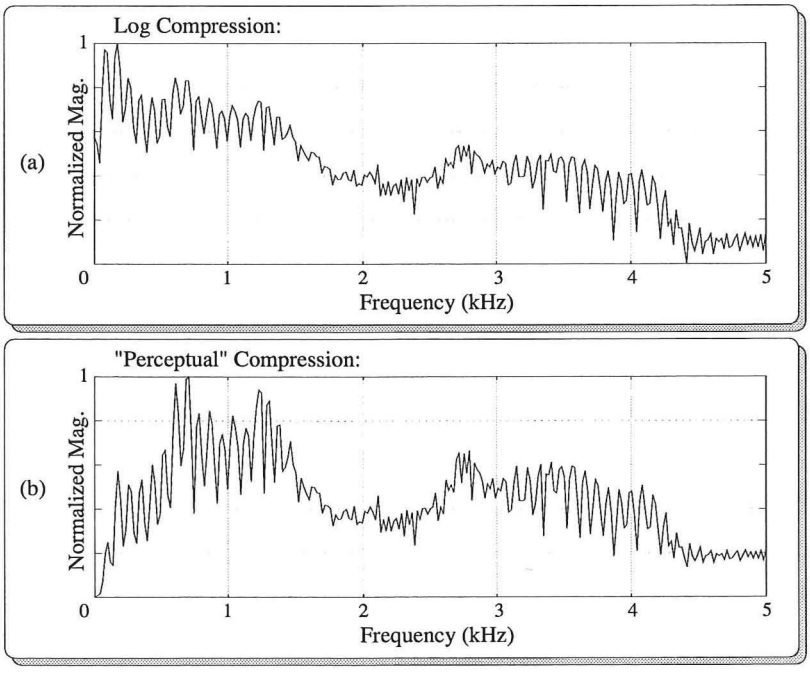

Figure 2.1.3-2. Examples of (a) *log compression* and (b) *perceptual compression* emulating varying human auditory sensitivity (see Figure 2.1.3-1). For vocoids (of primary interest during the experiments discussed in chapter 5), the two approaches yield similar results, due to the tendency of such spectrums to decay at higher frequencies prior to compression.

differences between log and perceptual compression occur at low frequencies approaching $f=0$ (Hz). Following the preprocessing approach proposed by Waibel *et al* (1989a), *log compression* is used in this work to emulate varying human auditory sensitivity (see §3.2).

Experiments with the perception of synthetic speech have revealed many properties of the human auditory system and its processing of speech. For example, such experiments have demonstrated the importance of spectral shape to sound identification, particularly for vocoids (Miller 1981), and to the robustness of speech communication in general. It is estimated that conversation may take place successfully in environments with signal-to-noise ratios (see equation (5.2.1-1)) approaching -6 dB (Miller 1981). In noisy environments, words are easier to perceive correctly in the context of sentences since a listener has a better idea of what to expect (Miller 1981). These expectations exist in the listener's mind rather than in the acoustic signal and are used to compensate for inevitable lapses in speech production and perception. Emulating these expectations by machine requires models of language and speech, such as those discussed next.

2.2 Segmental Linguistic Theories of Language and Speech

This section discusses segmental linguistic theories concerning the form of the abstract messages conveyed by speech, with particular emphasis on theories concerning the phoneme. The next section discusses a simplified model of speech communication arising from these theories. Within this model, phonological processing constitutes one step in the conveyance of abstract messages between a speaker and a hearer. §2.2.2 discusses the phoneme of central importance to automated phoneme recognition. Finally, §2.2.3 briefly discusses the problems associated with modelling speech utterances using phonemes.

2.2.1 The Speech Chain

Figure 2.2.1-1 depicts a simplified *model* of human speech communication, referred to as the *speech chain* (Denes and Pinson 1973). In this model, an abstract message to be communicated undergoes several levels of "encoding" before being realized as speech through the gestures of a speaker's vocal organs. After acoustic transmission, this speech is perceived by a hearer and undergoes several levels of "decoding" in order to *discover* the speaker's message.[4] Though interaction between levels is likely during both the "encoding" and

[4]The term *discover* is used in preference to *recover*, in this context, to emphasize the fact that a speech signal is not infinitely informative about a speaker's intended message. A hearer

"decoding" of an abstract message, particular attention is paid to this phenomenon during "decoding", since it is evident that some messages must be "decoded" at multiple levels simultaneously (Moulton 1969). For automated speech recognition, the processes "conducted" by the *hearer* are of primary interest.

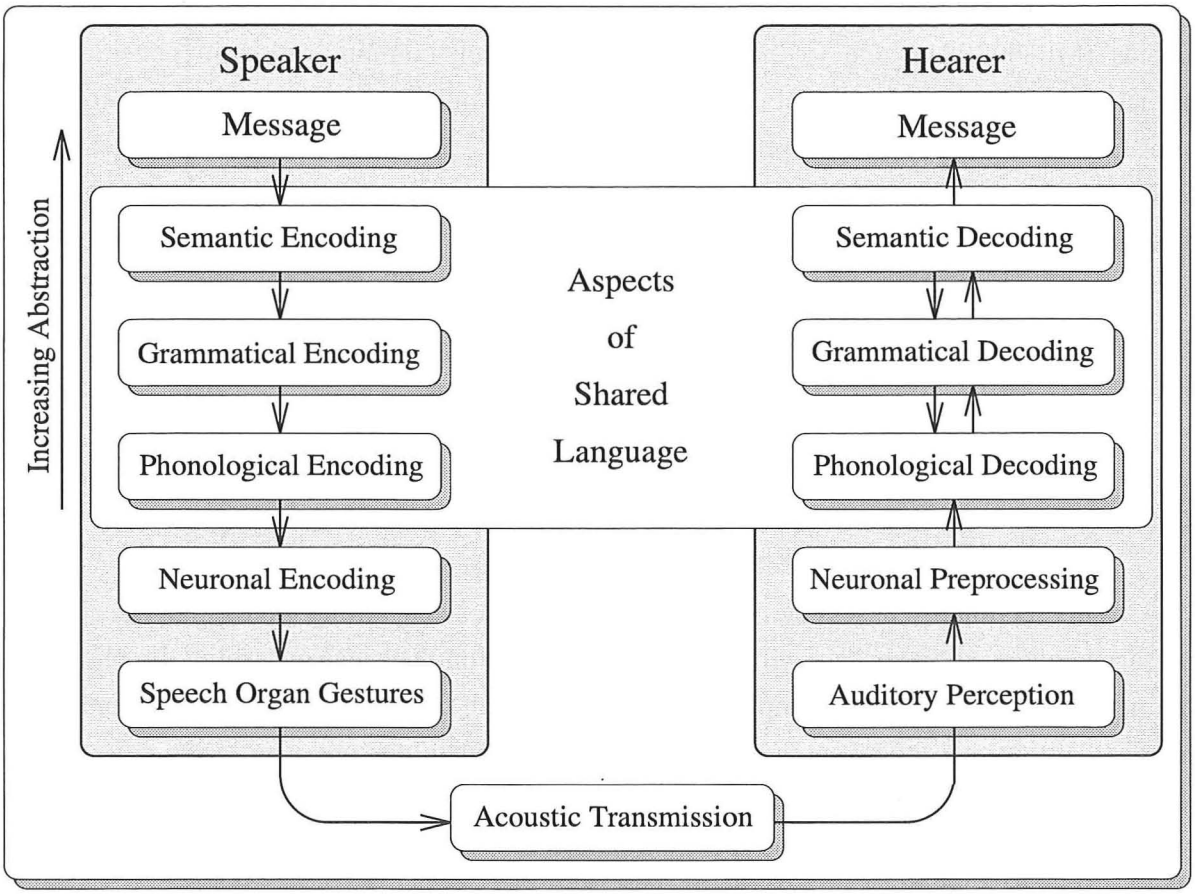

Figure 2.2.1-1. A simplified model of speech communication between humans indicating broadly the steps involved. The *aspects of shared language* are required by both a speaker and a hearer for this process to be successful and are the principal concern of linguistic theories of speech and language. Adapted from Matthei and Roeper (1983), Denes and Pinson (1973), with detail from Moulton (1969).

As indicated by Figure 2.2.1-1, language and speech may be analyzed at various levels. Several of the levels depicted are language specific and are grouped together by the box marked *aspects of shared language* (for convenience, these levels are referred to as the *linguistic levels*, henceforth). The transformations envisaged within these levels are described by segmental linguistic theories, making extensive use of discrete abstract units. Clark and Yallop (1990, page 93) discuss the rational for this approach;

---

must often rely on other sources of information, such as the constraints of the language being spoken and the context of an utterance, to "decode" and understand a message.

> In describing language we need to refer to the units of language. The fundamental reason for this in not just that it is traditional and convenient to refer to sounds and words and syllables and other such elements; it is that language itself depends on discrete and finite options.

Abstract units permit the options available to a speaker creating an utterance, or a hearer interpreting an utterance to be modelled. Within Figure 2.2.1–1, such options exist within the three language specific levels denoted *semantic*, *grammatical* and *phonological* encoding/decoding. The abstract units traditionally associated with these levels are the *sememe*, the *morpheme* and the *phoneme*, respectively (Crystal 1980).

Sememes are used in some semantic theories as minimal units of *meaning* (Crystal 1980). Not all languages share a common set of sememes. For example, German uses the sememe *Uhr* when describing any instrument for time-keeping, whereas English differentiates between portable and non-portable instruments using the sememes *watch* and *clock*, respectively (Moulton 1969). Ultimately, the automated transformation of speech utterances into sememe units may permit their meanings to be represented within a machine, thereby facilitating automated speech understanding, or enhanced automated speech recognition.

Morphemes are the minimal distinctive units of grammar (Crystal 1980) and may correspond to words such as *self*, or parts of words such as *un*. For example, the English word *unselfish* contains three morphemes, *un*, *self* and *ish*. Morphemes are the principal subjects of morphology which is concerned with word structure (Crystal 1980). As well as morphology, the grammatical level of the model in Figure 2.2.1-1 is concerned with the combination of words to form larger entities, such as sentences. Rules for word combination form the basis of the *syntax* of a language and are the principal concern of syntactics (Crystal 1980).

Phonemes are the minimal units of the sound structure of a language (Crystal 1980). These and other related units are described in detail in the next section.

A fundamental theorem of segmental linguistics is that at each *linguistic level* within the model depicted in Figure 2.2.1-1, utterances may be represented by combinations of the level's abstract units. For example, at the phonological level, an utterance of the word *bide* might be represented by a combination of the phonemes /b/, /d/ and /ai/. For convenience, such combinations are often sequentially ordered following western orthographic practices. An utterance of the word *bide*, for example, may be transcribed *phonemically* as /baid/. This ordering may resemble (approximately) the temporal ordering of speech features associated with the realizations of /b/, /d/ and /ai/, however, it *does not* necessarily indicate the order in which these abstract units are *presumed* to be *discovered* by a hearer (see the example concerning the phrase *I was riding/writing...* in the next section).

As the model depicted in Figure 2.2.1-1 suggests, the linguistic levels may be arranged

hierarchically in order of increasing abstraction (the phonological level being the "closest" to tangible speech utterances). Ideally, representations at the higher levels (the grammatical and semantic levels) may be obtained by transforming representations formed at the lower linguistic levels. In this way the higher level representations desired during automated speech recognition might be derived from phonological representations produced by an automated phoneme recognition system.

Currently, the relationships between morphological and phonological representations of speech utterances are being actively researched (Church 1987). Transforming phoneme sequences into morpheme sequences, rather than into word sequences as at present (see Waibel 1992b; Lee 1990), may permit more efficient storage of the *lexicon* required for automated speech recognition (Church 1987). Such transformation may also permit the tight coupling that exists between phonological and morphological processes to be utilized (Church 1987). Research into the relationships between semantic and grammatical representations of speech utterances is also ongoing (for example, see Hinton *et al* 1986a), however, such relationships are still particularly contentious.

From the model of human speech communication depicted in Figure 2.2.1-1, it is assumed in this thesis that a hierarchy of abstract units incorporating phonemes, morphemes and sememes, is necessary to permit automated speech recognition. The work presented in this thesis concentrates on the transformation of speech utterances into phoneme sequences, which are assumed to be the raw materials necessary to form more abstract representations.

2.2.2 The Phoneme and Related Units

Naturally a fundamental concept underlying any approach to automated phoneme recognition is the concept of the phoneme. It is, therefore, regrettable that the phoneme and related concepts are poorly described by some authors when discussing speech processing. For instance, some authors make no distinction between *phonetic* and *phonemic* analyses of speech utterances. Owens (1993, page 85) states that "The aim in phonetic analysis is to derive the phonemic structure of an utterance directly from the speech signal". Consequently, it is common for speech researchers to describe the relationships between acoustic events and phonemes as *acoustic-phonetic mappings* (Owens 1993; Waibel *et al* 1989a; Waibel and Hampshire 1989), rather than *acoustic-phonemic mappings*. Some authors regard phonemes as the "building blocks" of speech utterances, rather than abstract units for describing or modelling them. For example, Morgan and Scofield (1991, page 97) suggest that phonemes are "used to pronounce a word" and that "Phonemes are constructed from permutations of voicing, tongue, mouth, jaw and lip positions". Other authors do not distinguish between *phones* and *phonemes*, despite the former being tangible speech sounds and the latter being

abstract units (as discussed in this section). For example, Lee (1990) refers to the "50 phones in English" (rather than the 50 phonemes in English) and the "phone /t/" (rather than the phoneme /t/). As a consequence of poor descriptions like those above, this section presents a concise review of the phoneme and related concepts of interest to researchers in automated phoneme recognition.

The phoneme is for linguists the most basic unit beyond which it is not worth proceeding when describing the *meaning* of larger units (Gimson 1989). This abstract unit is the principal concern of *phonology* and is the fundamental unit used when describing the sound structure of a language (Crystal 1980). The concept of the phoneme is not a new one. As Arbercrombie (1991) suggests;

> The phoneme idea is found as an explicit concept about 1880, but for a considerable time before we can find it implicit in a number of early writers on language. They may point out, for example, that some differences of sound in a language do not affect meanings; or do not have to be shown in writing; or have no reality for either speaker or hearer, by whom they are 'not felt' or 'not heard'.

The term *phoneme* was coined by the French phonetician A. Dufriche-Desgenettes and first used in a paper he presented to the Société de Linguistique in 1873 (Abercrombie 1991). In 1879, the phoneme was given its current technical sense, as something to be contrasted with, and distinguished from, a speech sound, by the Polish linguist M. Kruszewski (Abercrombie 1991).

A *phoneme* is simply defined as "the smallest unit of sound in a language which can distinguish two words" (Richards *et al* 1985). For example, utterances of the words *pin* and *bin* may differ only in their initial sounds which can be treated as *realizations* of the phonemes /p/ and /b/. This simple (linguistic) definition, though useful, is perhaps misleading since one might conclude that phonemes are *tangible units* of sound. In reality, "phonemes are the abstract units that form the basis for writing down a language systematically and unambiguously" (Ladefoged 1982). To clarify this view, Figure 2.2.2-1 presents an idealized pictorial representation showing *some* of the conceptual relationships between phonemes and speech sounds. This figure, like the notion of the phoneme itself (Clark and Yallop 1990), is extremely controversial. It is unlikely that any linguist would condone such a simple representation, however, it is necessary for engineers to formulate a "model" of the relationships between phonemes and speech sounds in order to progress with the problem of automated phoneme recognition.

The outermost region depicted in Figure 2.2.2-1 represents all the sounds which may be produced by human vocal organs and is referred to as the *sound-space*. The study of these

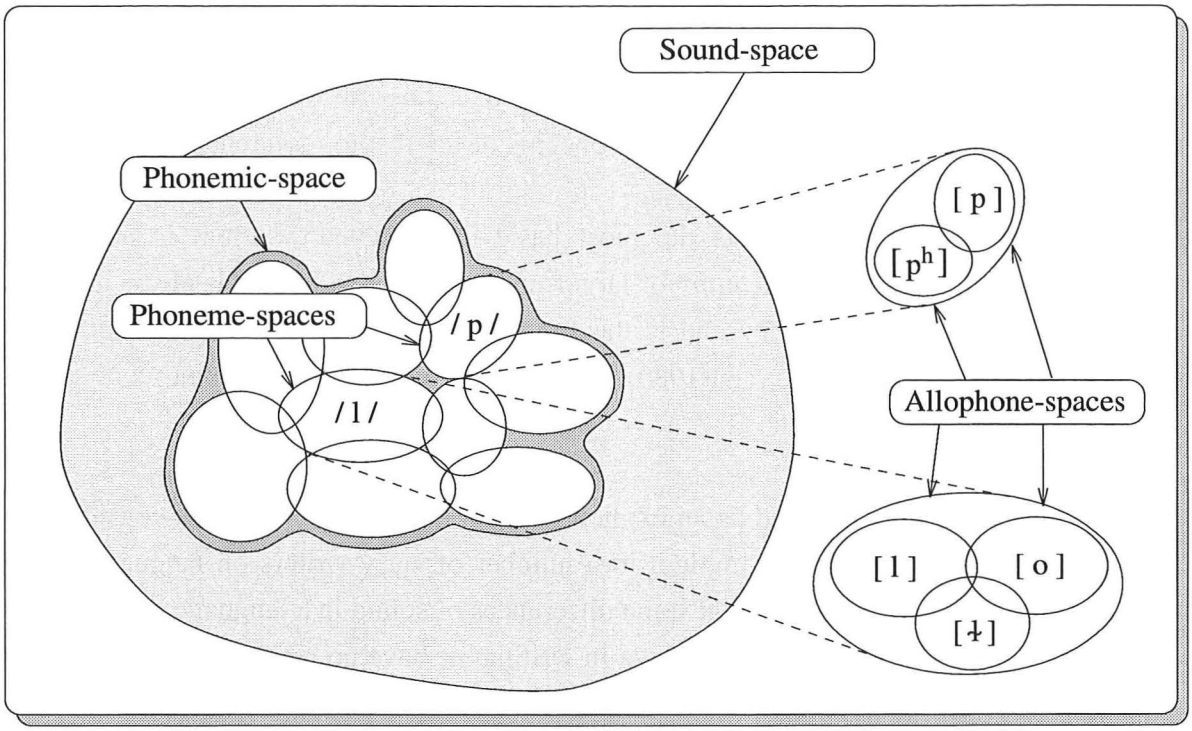

Figure 2.2.2-1. An idealized representation of the conceptual relationships between phonemes, allophones and phones. Sound-space contains all the sounds, or phones, that may be produced by human vocal organs. This space is partially spanned by phonemic-space which contains all the phones used by a particular language or accent. Phonemic-space is sub-divided into phoneme-spaces, representing phonemes, which group phones that similarly affect the meanings of utterances in which they occur. Phoneme-spaces are further sub-divided into allophone-spaces which represent a phoneme's allophones, or its significant phonetic variants. Developed in collaboration with Catherine Watson.

sounds, or *phones*[5], is the principal concern of *phonetics* (Catford 1988) and is language independent.[6] Within this space, a fuzzy framework modelling the sound structure, or *phonology*, of a language may be superimposed. This framework encloses a region referred to as the *phonemic-space* and contains all the phones used by a language or accent. In general, phonemic-spaces are smaller than the sound-space, since languages seldom use all possible phones. In English, for example, phones such as "clicks" are not used during the course of normal speech.

The organization of phonemic-space is the central concern of *phonology*. The phonemic-space may be subdivided into *overlapping* regions representing the phonemes of

[5]A *phone* is "the smallest perceptible discrete segment of sound in a stream of speech" and, within segmental models of speech, is the physical realization of a phoneme (Crystal 1980).

[6]Speech sounds analyzed phonetically are normally represented by symbols enclosed between [] to avoid confusion with phonemes. The later are represented by symbols placed between //. Sequences of symbols placed between [] and // are referred to as *phonetic* and *phonemic* transcriptions, respectively.

a language (these are referred to as *phoneme-spaces*). This division, like the extent and location of phonemic-space, is highly language and accent dependent. A phoneme-space, being a sub-space within sound-space, contains an semi-infinite number of phones. These phones are *realizations of a phoneme* and are grouped together because of their common effect on the meaning of utterances in which they are used. For example, the realizations of the phoneme /p/ in utterances of the words *ape* (/eip/) and *pie* (/pai/) are likely to be *different phones*, yet from the perspective of *meaning*, they serve equally to distinguish *ape* and *pie* from utterances of the words *aid* (/eid/) and *dye* (/dai/), respectively.

The phoneme-spaces depicted in Figure 2.2.2-1 overlap to reflect the fact that certain phones may be "interpreted" as one of two or more phonemes, depending on the context in which they are used. For example, a phone lying in the overlapping region of the phoneme-spaces for /d/ and /t/ may be used when uttering a phrase commencing *I was riding/writing...* (Moulton 1969). This phone is interpreted as a realization of /t/ or /d/ depending on whether *riding* or *writing* is "selected" by a hearer using the context provided by the rest of the phrase. For example, if the entire phase uttered were *I was riding/writing a horse*, this phone would usually be interpreted as a /d/ giving the phrase *I was riding a horse.*[7]

As Figure 2.2.2-1 shows, phoneme-spaces may be further subdivided into overlapping regions whose elements *systematically* vary from those of another phoneme-space sub-region according to the contexts in which they are used (linguistic or social contexts, Crystal 1980). These regions are referred to as *allophone-spaces* in this thesis and delineate the major variants, or *allophones*, of a phoneme.[8] For example, the phoneme /l/ *may* contain allophonic-spaces such as those denoted by [l], [ɫ] and [o] ([o] appears as an allophone of /l/ in models of the Cockney accent of English, Gimson 1989). Allophonic observations are not required in order to represent the meaning of word length utterances. However, they are sometimes necessary when describing continuous speech to remove ambiguities arising from the economy of pure phonemic transcriptions (see §2.2.3).

The division of phonemic-space into regions of phones related by *meaning*, is an abstraction similar to calling a band of the visible light spectrum *orange*. As Catford (1988, page 203) explains,

---

[7]This example also demonstrates that ambiguity at the phonemic level may require semantic knowledge to correct. Both *riding* and *writing* are verbs, therefore, the ambiguity arising in the example cannot be overcome at the grammatical level. Semantic knowledge that *I was riding a letter* and *I was writing a horse* are unlikely phrases, is required.

[8]Following Catford (1988), *allophones* are assumed in this thesis to be abstract units like phonemes, with the exception that their realizations are more uniform. However, it must be noted that some linguists treat allophones as tangible phoneme realizations (Crystal 1980; Bolinger 1975) and transcribe them using symbols within [], as is the normal practice when transcribing phones.

It is important to be aware that phonemes are abstractions or generalizations: they are, that is to say, abstract phonological units, each of which is manifested, or realized, in speech in a number of different ways. You cannot *pronounce* a phoneme. You can only pronounce a specific sound which may be the realization of a phoneme. If you say, for instance, the English word *cat* you are producing a quite specific sequence of sounds. That sequence of sounds is not itself a sequence of phonemes: it is the outward or concrete manifestation, of the sequence of phonemes that we represent in the transcription as /kæt/.

The phonemes of a language are typically identified through the process of *commutation*, the discovery of *minimal pairs* through sound substitution (Gimson 1989, Crystal 1980). The words *pin* and *bin* are an example of a minimal pair which reveal the phonemes /p/ and /b/. Phonemic representations of *pin* and *bin*, /pIn/ and /bIn/, differ only in their initial phonemes which represent, *abstractly and economically*, the differences in sound quality that distinguish these words for speakers of English.

Ideally, observations from numerous minimal pairs, including those where contrasts occur in word initial, medial and final positions, reveal all the phonemes of a language, a set described as a *phonemic solution*. However, it is often necessary to consider the wider patterning of sounds within a language when developing such a solution (Clark and Yallop 1990). For example, in phonemic solutions for English, it is common for the sequence of sounds [tʃ] (as found at the start of the word *chop*) to be treated as one phoneme, the *affricate* /tʃ/, whereas the sequence of sounds [ts] (as found at the end of a word like *spots*) to be treated as two phonemes, the *voiceless plosive* /t/ and the *fricative* /s/. A phonemic solution may be likened to one subdivision of a phonemic-space into phoneme-spaces and is not necessarily the only subdivision possible. Several phonemic solutions for English are currently in use, due mainly to the varying significance attached to vowel quality (Gimson 1989).

The use of minimal pairs to develop a phonemic solution results in an inventory of phonemes given by negative, rather than positive, definitions (Gimson 1989). For example, the essence of /p/ is that it is not /t/ or /k/. To provide positive phoneme definitions, linguists describe the significant (qualitative) features concerning the *voice*, *place* and *manner* by which their realizations are articulated (see Crystal 1980 for a discussion of voice, place and manner). For example, the English phoneme /p/ is described positively by the features *voiceless*, *bilabial*, *oral*, *stop* (see Crystal 1980 for definitions of these terms). Such a feature description may cover many, but not all, realizations of a given phoneme. For example, /p/ may also be realized as a *labio-dental stop* instead of a *bilabial stop* in words like *cup-full*. Features ascribed to the articulation of phoneme realizations are intended to highlight the differences, or *oppositions* (Crystal 1980), between phonemes, rather than to provide a full description.

As discussed above, the realization of a phoneme may vary depending on the phonetic context in which it is used, the major variants being grouped as allophones. This variation is referred to generally as *context-sensitivity* or *co-articulation* (though the latter term is not used consistently by all linguists) (Clark and Yallop 1990). Co-articulation results because, as Clark and Yallop (1990, page 118) state;

> Speech does not consist simply of a string of target articulations linked by simple movement between them. Instead, the articulation of individual segments is almost always influenced by the articulation of neighbouring segments, often to the point of considerable overlapping of articulatory activities.

Co-articulation is necessary to cope with the inherent delay between neuromuscular commands and their associated speech gestures that results from the inertia of a speaker's active articulators (Clark and Yallop 1990). If co-articulation was avoided, normal speaking rates of 150 words a minute, or 3 to 5 phonemes per second (Miller 1981), would be difficult to attain. Consequently, co-articulation is not a needless complication interfering with the ideal properties of speech, but rather is an efficient encoding scheme that ensures good performance despite the constraints imposed by the human vocal organs (Clark and Yallop 1990).

Within linguistic research, phonemes are often used to transcribe speech utterances. *Phonemic transcription* (or *broad transcription*) is the most common and least cumbersome method of indicating the spoken realization of language (Gimson 1989). For example, a utterance of the word *titles* may be transcribed phonemically as /taitlz/. Within this highly sophisticated representation are a number of implicit assumptions (Gimson 1989). First, it is assumed that the phonemes transcribed have predictable realizations as phones based on the phonological rules of their associated language. For example, the *phonetic transcription* (or *narrow transcription*) [tʰäˬëtl̥ɬz̥] representing the phones used when uttering the word *titles*, should be predictable from /taitlz/. Secondly, within a phonemic transcription, the number and nature of the phonemes comprising the phonemic solution assumed, is also implicit. Traditionally, automating phonemic transcription has been the aim of researchers attempting automated phoneme recognition. However, these transcriptions are not without problems as discussed in §2.2.3.

A suitable summary for many of the concepts presented in this section is provided by Gimson (1989, page 221):

> Speech must, therefore, be considered from a phonetic point of view as an ever-changing continuum of qualities, quantities, pitches and intensities. If, for practical purposes, e.g. in a phonetic/phonemic transcription of the spoken language, we treat

> speech as a succession of articulatory or auditory separable units, it is largely because we impose, upon the gross material of speech, entities which we have derived (consciously or unconsciously) from a knowledge of the linguistically significant oppositions operating for any particular language system, i.e. the phonemic categories....
>
> ...It should not, however, be forgotten that such a linear presentation is an abstraction from the concrete material of speech rather than a statement of the gross phonetic material composing the continuum.

2.2.3 Problems With Phonemic Representations

In developing models incorporating the phoneme to describe speech utterances, linguists have identified a number of problems with them. These problems may influence the ability of an automated *speech* recognition system to form a transcription of a speaker's message, or understand its meaning. This section discusses some of the major problems with phonemic models of speech utterances. All but the first of these problems are left for future work.

The first and most obvious problem with phoneme based models is identifying the speech portions which correspond to phoneme realizations. In this thesis, Fant's model relating phonemes and speech signals, depicted in Figure 2.2.3-1, is adopted (Fant 1973, page 22). A speech signal may be partitioned into segments, as discussed in §2.1.2, and shown in parts (a) and (b) of Figure 2.2.3-1. Neighbouring segments may share common features (see part (c)) that may span all, or part, of their length depending of which "changes in waveform structure" are used to establish segment boundaries. Each segment may contain information corresponding to one or more phonemes, as indicated by the phoneme-sound correlation plot in part (d), and each phoneme may span one or more segments, as shown in part (e). Ideally, the speech portion assigned to each phoneme has the highest correlation with that phoneme, giving the ideal segmentation shown in part (f) of Figure 2.2.3-1. In practice, however, phoneme-sound correlation, as depicted in part (d) of Figure 2.2.3-1, is seldom known[9], implying the speech portions associated with the phonemes depicted in part (e) must be established using educated "guesswork". The method by which such speech portions associated with phonemes are identified in this work is discussed in §3.3.

Another significant problem with phonemic representations is caused by the high

---

[9]A system capable of deducing phoneme-sound correlations could also serve as a phoneme recognition system.

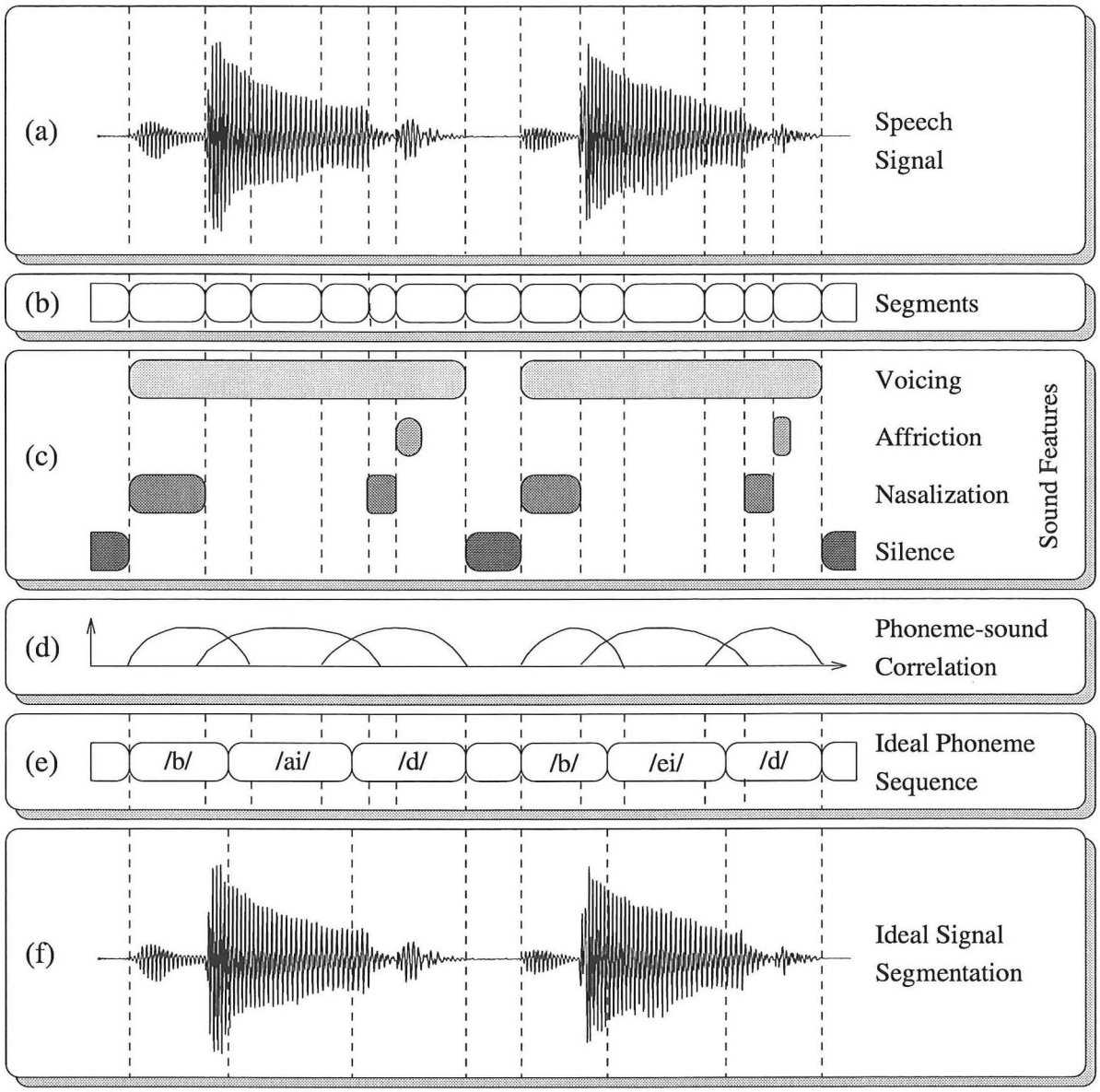

Figure 2.2.3-1. The conceptual relationship between speech segments, features and phonemes after Fant (1973). The speech signal in part (a) (utterances of *bide* and *bade*) may be partitioned into the segments shown in part (b), which may share similar features with one another as indicated by part (c). Depending on the phoneme-sound correlation (part (d)) within its extent, a segment may contain information concerning one or more phonemes (part (e)). Similarly, the ideal extent of a phoneme (part (e)) may include one or more segments. Part (f) shows the ideal segmentation of the speech signal in part (a) into "phoneme realizations" which is only possible when the ideal phoneme sequence (part (e)) is known.

degree of redundancy present in languages such as English (Miller 1981; Gimson 1989). As a consequence of this redundancy, a high level of confusion is tolerable at the phonemic level (Gimson 1989). This fact is utilized by speakers when speaking in a rapid or casual manner. Phonemic transcriptions derived from utterances spoken rapidly or casually may diverge from "ideal" (or *desired*) transcriptions in two ways, both of which preclude simple transformations to determine more abstract linguistic units, such as morphemes and sememes.

The first divergence, *assimilation*, results when the realization of one phoneme approaches that of another due, to the influences of phonetic context (the realizations of neighbouring phonemes, Crystal 1980, Clark and Yallop 1990). For example, the realization of /n/ in *ten bikes* may be more like a realization of /m/ when this phrase is uttered informally. In this case, the phonemic transcription produced by an automatic phoneme recognition system might be /tembaiks/, rather than /tenbaiks/, thereby complicating the deduction of the desired morpheme *ten*.

The second divergence, *elision*, results from the omission of sounds, or groups of sounds, from connected speech (Crystal 1989; Clark and Yallop 1990). For example, realizations of words such as *camera* and *february* may omit entire syllables, causing them to be transcribed phonemically as /kamra/ and /febri/, respectively (Crystal 1980). Once again such sequences complicate morpheme deduction.

Another significant problem with phoneme based models of speech arises from the desire of engineers to use automated phoneme recognition to process utterances containing multiple words. Such usage is not entirely shared by linguists when describing speech *for their purposes*. As Gimson (1989 page 52) points out

> It frequently happens that a phonemic analysis is based on a unit not larger than the word. If any larger section of the utterance is used, the analysis becomes a great deal more complicated.

This point is demonstrated by the two phrases *plum pie* and *plump eye* which may share an identical phonemic transcription /plampaI/ (Gimson 1989). Utterances of these phrases are distinguished principally by the aspiration that accompanies the realization of /p/ in *pie* ([p^h]), which is absent from the realization of the final /p/ in *plump* ([p]). Aspiration, and other aspects of pronunciation, are not commonly presented in traditional phonemic transcriptions (particularly those produced automatically), since they are typically not essential when ascribing meaning to sounds. However, as in the case shown, the recognition of such features may be required *in parallel* with phoneme recognition to avoid ambiguity when attempting to recognize utterances containing multiple words.

The final problem with phoneme based models is that of *phoneme neutralization* (Gimson 1989). When describing the phonology of a language, it happens that some phones may be assigned to one of several phonemes with equal validity. For example, in English the phones directly following the realization of /s/ in *spin*, *steam* and *scum* could be treated as realizations of /p/, /t/ and /k/, respectively, or /b/, /d/ and /g/, respectively. For orthographic reasons, /p/, /t/ and /k/ are commonly used when transcribing these phones, even though they are reputed to have more in common with realizations of /b/, /d/ and /g/ perceptually (Gimson 1989). This approach causes no ambiguity within linguistic models of English, since /ptk/ are

never opposed to /b/, /d/ and /g/ after /s/, however, it is unclear whether this arbitrary decision is equally suited to automated phoneme recognition, which attempts to emulate human perception. This example highlights the need for engineers to seek alternative "phonemic solutions" suited to automated speech recognition, as was done when transcribing the TIMIT speech database (see Zue and Seneff 1988).

2.3 The Phonology of New Zealand English

This section introduces the phonology of New Zealand English and the phonemic solution for this accent adopted in this work. New Zealand English has gradually developed as a distinct variety of English during the course of the last one hundred years (Holmes and Bell 1992). Phonologically, New Zealand English is a variety of English similar to that found in the southeast of England, however, phonetically its is quite distinct (Holmes and Bell 1992; Bauer in print). Differences between British RP[10], American and New Zealand English phoneme realizations provide one motivation for attempting the automated recognition of New Zealand English phonemes, as discussed in chapter 1. Like Australian English, the pronunciation of New Zealand English varies with "social class", causing linguists to define two types of New Zealand English accent, referred to as *general* and *broad* (Bauer 1986). The former is more "cultivated" than the latter and is the accent spoken by both New Zealand English speakers studied in this work (see §3.1)

New Zealand English, according to one phonemic solution, may be described using the 43 phonemes listed in Table 2.3-1. This solution is based upon that proposed by Hawkins (1973) and differs only in the omission of /ə/, which by Hawkins' own admission "cannot be established as a separate phoneme for New Zealand English". The phoneme notation adopted in this thesis is identical to that proposed by Carstairs-McCarthy (1989) (based upon Hawkins 1973), with the exception of the labels associated with the closing diphthongs, which follow the more familiar notation used by Jones (1967).

The phonemes listed in Table 2.3-1 are divided into the broad classes labelled *vowel* and *consonant.*[11] The consonants, of which there are twenty four, are subdivided into groups

---

[10]RP is short for *received pronunciation* and refers to a regionally neutral accent of British English (Crystal 1980).

[11] The convention of using the terms *consonant* and *vowel* to refer to *phonological function* rather than *phonetic form*, proposed by Pike (Clark and Yallop 1990, Gimson 1989, Crystal 1980), is adopted in this thesis. The term vowel is applied to phonemes which typically form the nucleus of syllables rather than to the entire group of phonemes whose realizations are examples of vocoids (Crystal 1980). Consequently, the phonemes /r/, /l/, /j/ and /w/, though typically realized as vocoids, are treated as consonants in English due to their

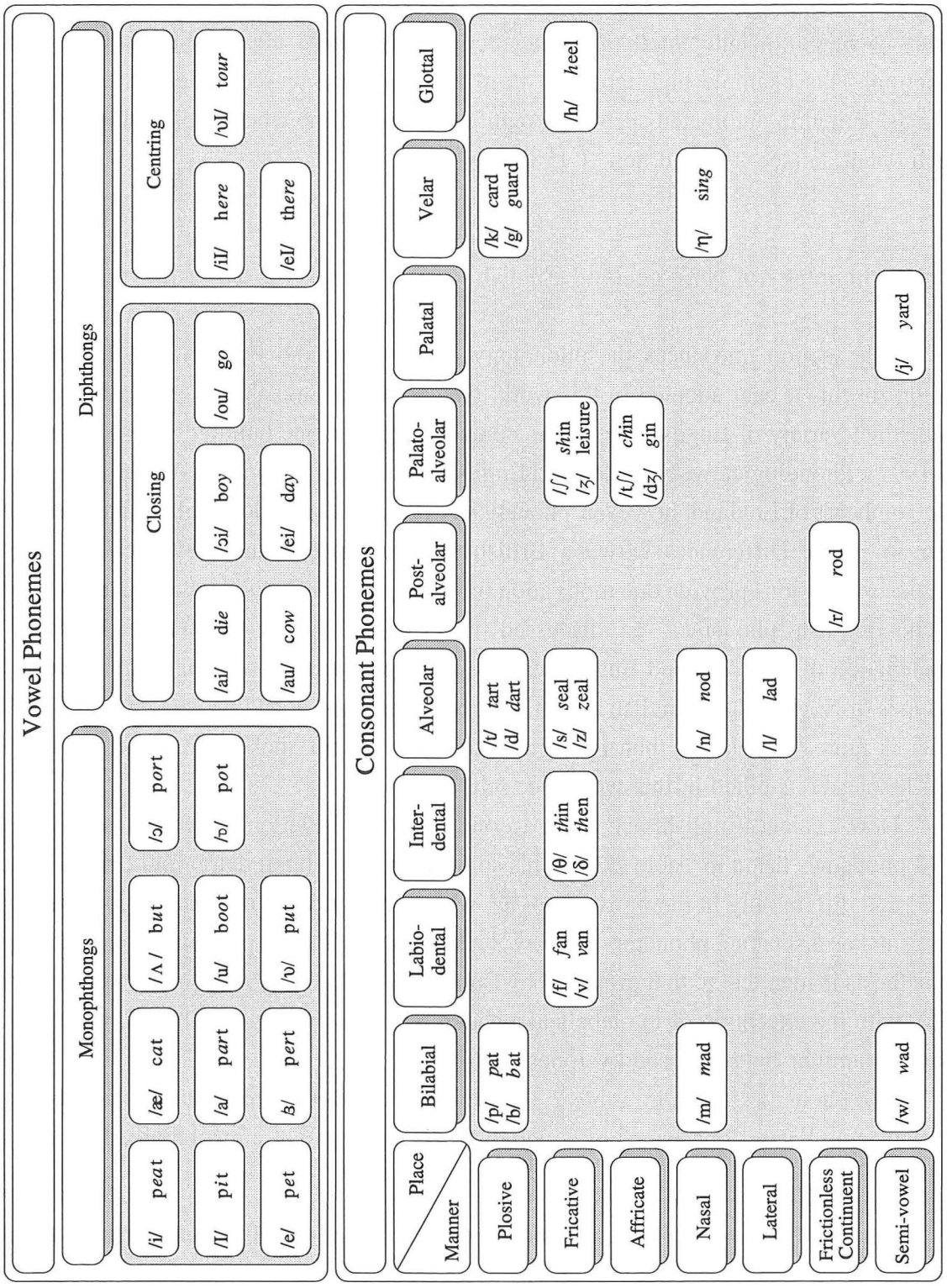

Table 2.3-1. The 43 phonemes of the phonemic solution for New Zealand English adopted in this work, including example words in which they are realized.

---

marginal positioning within syllables.

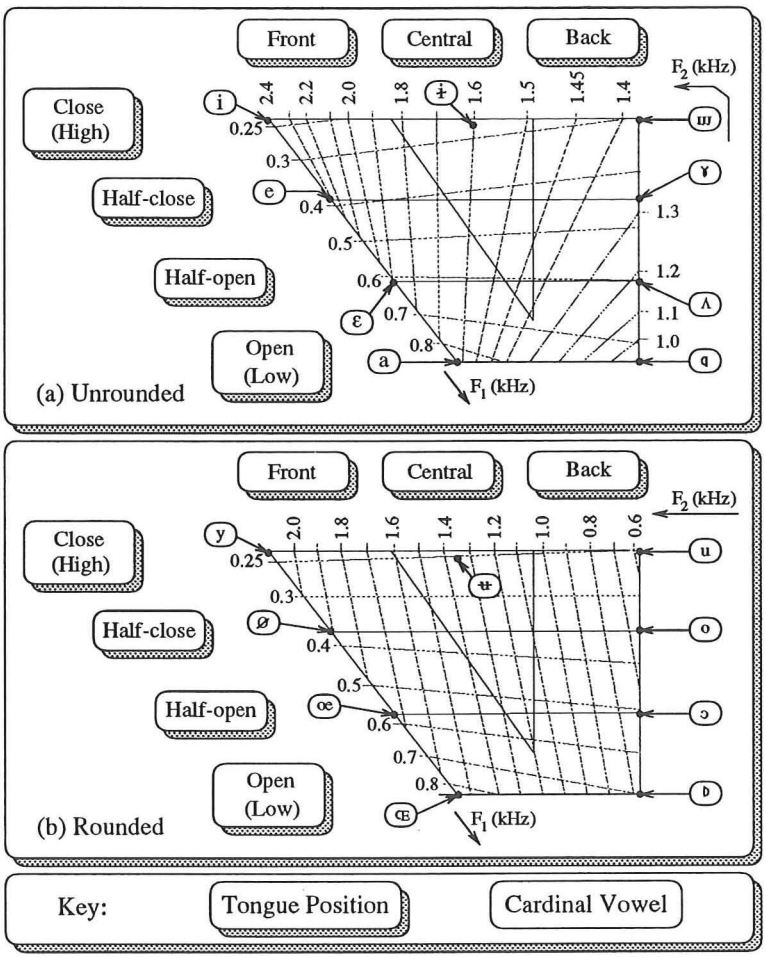

Figure 2.3-2 Cardinal vowel diagrams for (a) vowels realized with unrounded lips (see Figure 2.3-3 (a) and (c)) and (b) vowels realized with rounded lips (see Figure 2.3-3 (b)). The eighteen cardinal vowels (male average) are displayed on two separate diagrams, in this figure, to allow the frequencies of the first two formants, F_1 and F_2 to be indicated (normally these diagrams are superimposed, as in Clarke and Yallop (1990)). The tongue position labels indicate the location of the tongue when producing each cardinal vowel. For example, *i* is produced with the tip of the tongue ("Front") approaching near to the palate ("Close (High)"), placing the vocal tract in a closed formation. Adapted from Catford (1988) and Clarke and Yallop (1990).

based on the primary features of their articulation. The vowels, of which there are nineteen, are subdivided into two separate categories labelled *monophthong* and *diphthong*. The former category contains phonemes whose realizations exhibit little perceptible change in quality with time (sometimes called *pure vowels*), whereas the phonemes contained in the latter category are realized with considerable changes in quality, or *glides* (Crystal 1980).

Unfortunately, the description of vowel realizations, particularly in terms of place, is more complicated than it is for the consonants, since vowels are not produced with such obvious articulator contact or proximity. Consequently, it is common practice to describe vowel quality in terms of a set of language independent reference realizations known as the *cardinal vowels* (Catford 1988). Figure 2.3-2 shows *cardinal vowel diagrams* which represent the "space" occupied by the *rounded* and *unrounded* cardinal vowels for an average adult

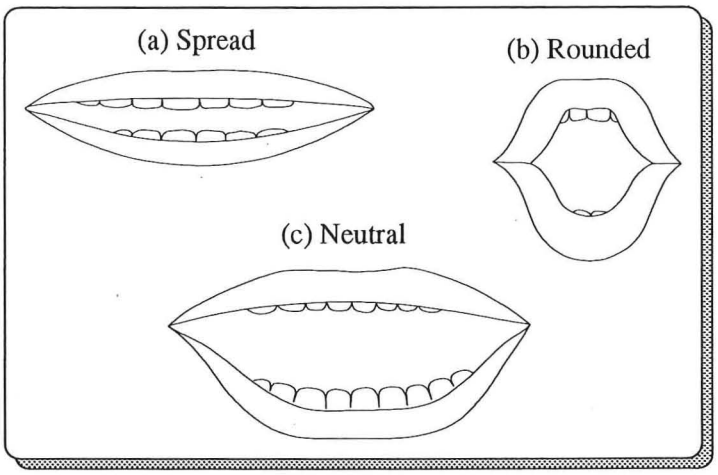

Figure 2.3-3. Various lip configurations that may be observed during normal speech. Vowel realizations are often described as being rounded (b) or unrounded (a or c) depending on the lip configurations used to realize them.

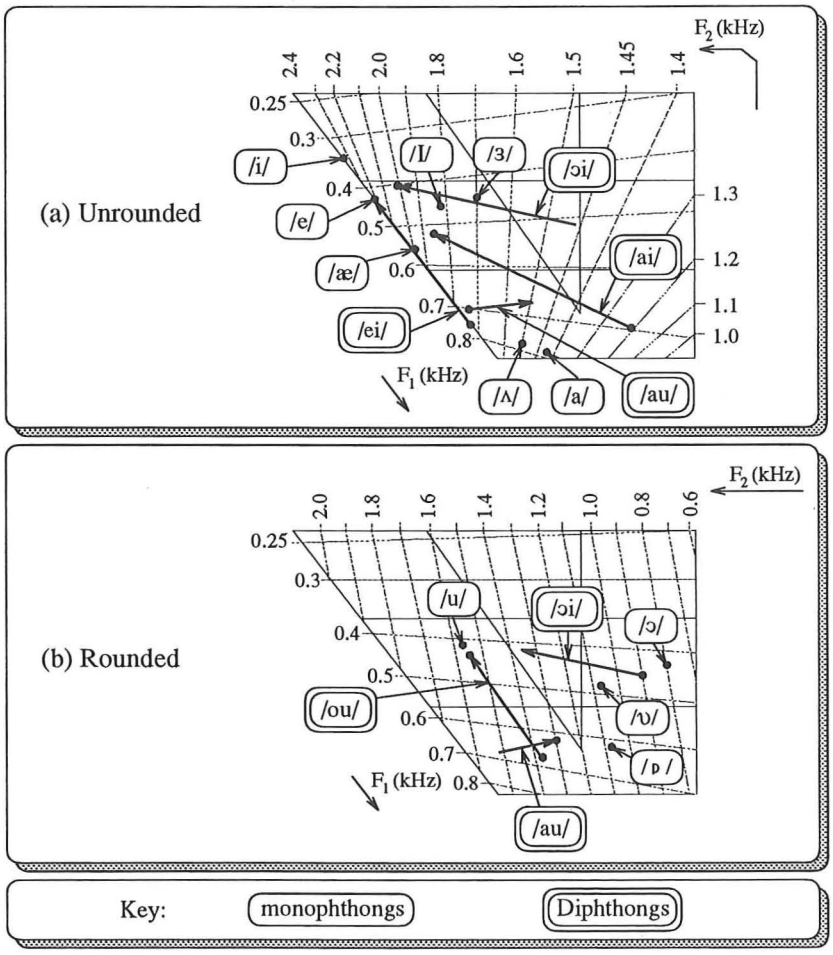

Figure 2.3-4. A comparison between the cardinal vowels (Figure 2.3-2) and the monophthong and closing diphthongs of New Zealand English (average of 25 male speakers). Unfortunately, diphthong representation is complicated in this figure by /ɔi/ and /au/ which transit between rounded and unrounded vowel qualities. Adapted from Maclagan (1982).

male (unfortunately, no similar female diagrams were presented in Catford 1988). The terms rounded and unrounded refer to lip configurations like those depicted in Figure 2.3-3.

Figure 2.3-4 shows the monophthongs and closing diphthongs of New Zealand English (an average of 25 male speakers, Maclagan 1982) superimposed on the cardinal vowel diagrams given in Figure 2.3-2 (the centring diphthongs are ignored for reasons discussed in §2.4). This figure shows that the glides associated with realizations of certain closing diphthongs may contain qualities (approximately) characteristic of the realizations of one or more monophthongs. As a consequence of this proximity, the behaviours of the expert modules for closing diphthong recognition discussed in this thesis have been examined in response to utterances containing monophthong realizations. Interestingly, the average glides associated with /ai/ and /ei/ in Figure 2.3-4 do not exhibit the overlap observed in this work (see Figures 4.2.3.1-3 and 4.2.3.1-4). This fact is most likely due to the averaging of glides produced by speakers with general and broad New Zealand English accents (see Bauer 1986). For the latter, the glides associated with /ai/ and /ei/ are typically more distinctive, as depicted in Figure 2.3-4.

2.4 Diphthongs

This section discusses one particular class of phonemes known as the diphthongs. Table 2.3-1 lists the *closing* and *centring diphthongs* of New Zealand English which constitute just under one half of this accent's vowel phonemes.[12] Closing diphthongs are characterized by glides that tend towards the top of the cardinal vowel diagram shown in Figure 2.3-2 (see Figure 2.3-4), implying their final qualities are produced by more *closed* vocal tract configurations. Centring diphthongs, by contrast, tend to the centre of the cardinal vowel diagram.

Traditionally, diphthongs are denoted using a sequence of two symbols, such as /ai/ or /ei/, to highlight their changing qualities which result from changing vocal tract configurations (Cleremont 1991). Such sequences do not cause ambiguity in phonemic transcriptions of English utterances, since monophthongal sequences, such as /a/ followed by /i/, do not occur (Hawkins 1973). Despite this, the *biphonemic* notation can give a misleading impression of the qualities of a diphthong. /ai/, for example, is *not* realized by concatenating

[12]In recent texts on automated speech recognition (see for example Morgan and Scofield 1991; Owens 1993), diphthongs are often distinguished from vowels, despite the many linguistic texts concerning English which regard *diphthongs as one class of vowels* (see for example, Clark and Yallop 1990; Gimson 1989; Ladefoged 1982). Given the nuclear distributions of diphthongs within English syllables they, like the monophthongs, are regarded as vowels in this thesis.

realizations of the monophthongs /a/ and /i/ (Hawkins 1973). Indeed the qualities associated with each end of a diphthong's glide (marked *diphthong-glide* in Figure 2.4-1) may not resemble any monophthong vowel quality at all (Ladefoged 1982; Lehiste and Peterson 1961).

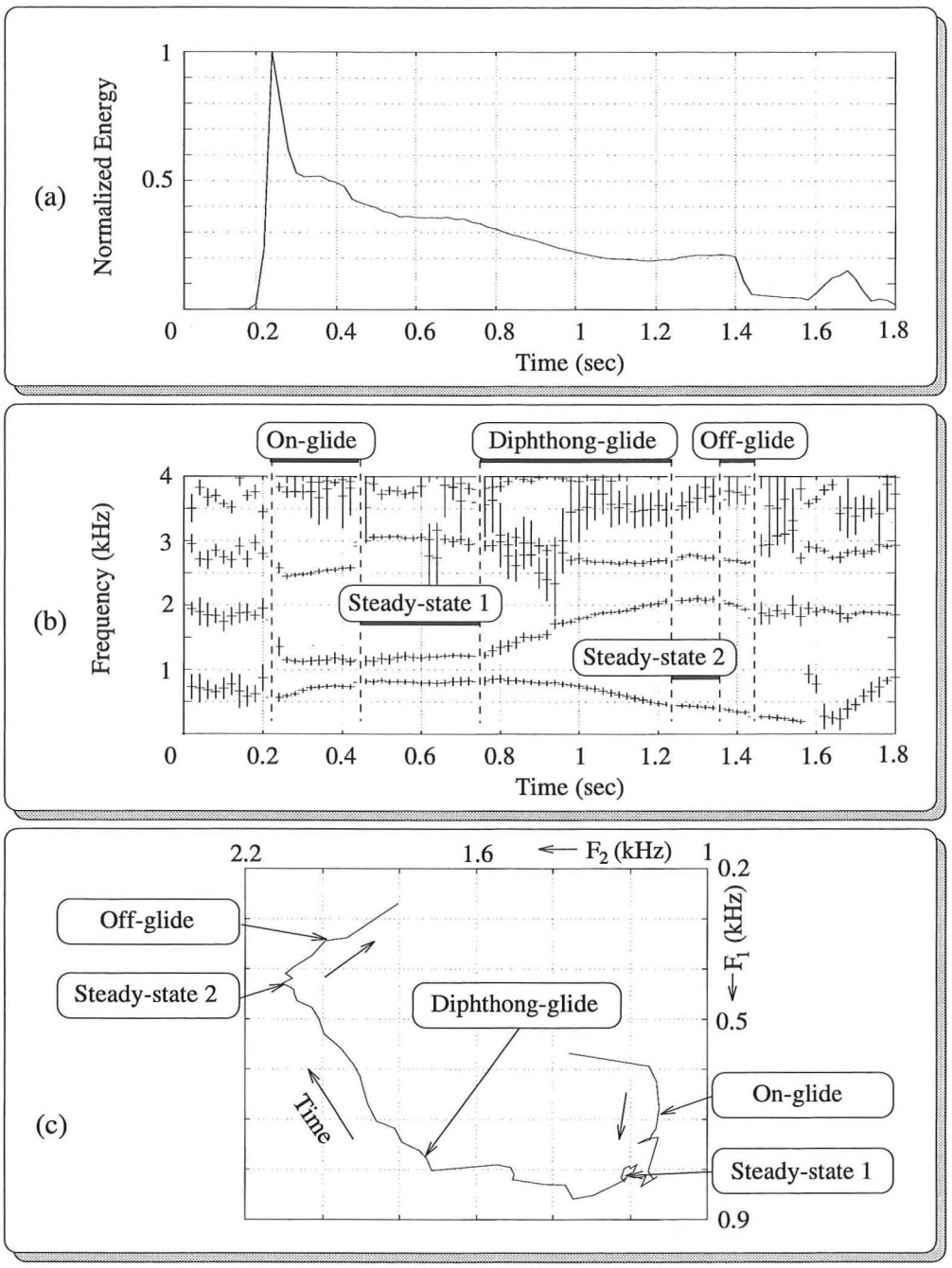

Figure 2.4-1. Instrumental analysis of the word *bide* containing a realization of the closing diphthong /ai/ (female speaker). Part (a) shows a plot of normalized signal energy which decays during /ai/'s realization (the interval from 0.25 to 1.4 seconds approximately) as is typical. Part (b) shows the formant tracks corresponding to the entire utterance which have been subdivided into sections relevant to the realization of /ai/. Part (c) shows the trace of the first two formants in part (b) plotted against one another. This plot omits the time axes so that steady states become single points, thereby emphasizing the *glides*.

Some authors, such as Hawkins (1973), choose to notate diphthongs using a biphonemic sequence with the second element superscripted, for example /a^{i}/ and /e^{i}/. This is intended to reflect the typical prominence of the first quality within diphthong realizations, which is usually longer in duration and larger in amplitude (Ladefoged (1982), Hawkins (1973)). Figure 2.4-1 illustrates this point. The first "steady state" portion (*steady-state 1*) is longer in duration and more intense than the second (*steady-state 2*), as shown by parts (b) and (a) of Figure 2.4-1, respectively. On occasions, this pattern of prominence within diphthong realizations may be reversed with the final quality being emphasized. Consequently, increasing signal energy may not be used as a robust indicator for the absence of a diphthong.

Part (c) of Figure 2.4-1 shows the trace which results when the first two formants in part (b) are plotted against one another. For convenience, this trace is referred to as an F_1-F_2 *trajectory* in this thesis. The unusual axes directions within this plot ensure that the F_1-F_2 trajectory shown resembles the orientation of its associated glide in a traditional cardinal vowel diagram (in this case, the glide for /ai/ in Figure 2.3-4). F_1-F_2 plots of this kind are commonly used during instrumental studies of speech sounds (see Maclagan 1982, for example) and are utilized in this work to select speech portions for training phoneme recognition systems (see §3.3). Such plots reduce "steady-state" speech portions to single points, emphasizing the glides within phoneme realizations.

As well as classifying diphthongs by their glide directions (centring or closing), some authors, following Lehiste and Peterson (1961), divide the diphthongs of a language into *genuine* (or *phonemic*) and *pseudo* (or *non-phonemic*) types (see Cleremont 1991; Bond 1978; Bond 1982). Genuine diphthongs consist of two steady-state portions separated by a glide and are, therefore, considered to contain *two* targets (Bond 1978). The diphthong depicted in Figure 2.4-1 is an example of such a diphthong. By contrast, pseudo diphthongs consist of only one target and a glide within which this target is embedded (Lehiste and Peterson 1961).

The particular diphthongs classified as genuine and pseudo English diphthongs is accent dependent. For example, Bond (1982) classifies /ai/, /au/ and /ɔi/ realized by speakers of American English (from Dayton, Ohio) as genuine diphthongs, whereas Bernard (1970) classifies /ai/, /au/, /ɔi/, /ei/ and /ou/ realized by speakers of "general" Australian English as genuine diphthongs (the last two diphthongs in this case being pseudo diphthongs in American English, Lehiste and Peterson 1961). For Australian accented English, the set of genuine diphthongs corresponds exactly to the set of closing diphthongs. Given the similarities between Australian and New Zealand diphthongs (Maclagan 1982), it is assumed in this thesis that the closing diphthongs of New Zealand English are also its genuine diphthongs.[13]

---

[13]In the experiments described in chapter 5, only the closing diphthongs of New Zealand English were considered. This simplification was made for two reasons. First, it is assumed that only the closing diphthongs of New Zealand English exhibit the true characteristics of a diphthong under the definition given by Lehiste and Peterson (1961). Second, and more

On encountering genuine diphthongs for the first time, it may be wondered why the realizations of such phonemes - the apparent concatenation of two vowel like qualities - should be treated as single phoneme at all? Within the phonology of a language, the classification of complex vowel-like sound sequences depends upon their function. In English certain vocoid sequences are classified as diphthongs because they act as a single vowel forming the nucleus of a syllable (Clark and Yallop 1990). For example, the word *bide* (/baid/) which contains the diphthong /ai/, is considered to be monosyllabic, even by phonologically naive speakers. Other complex sequences, such as those occurring across word boundaries in continuous speech, are regarded as concatenated monophthongs, since their elements belong to different syllables. The complex sequence of /i/ and /a/ resulting from the combination of the words *key* (/ki/) and *arch* (/atʃ/) in the phrase, "The key arch of the building...", is such an example.

Interestingly, phonemic solutions for some languages, such as Japanese (Okada 1991), have no diphthongs at all, whereas others, such as Hong Kong Cantonese (Zee 1991), have more than English. In the case of Japanese, all complex sequences of two vowel-like sounds have "two distinct elements which constitute two different 'morea' linked by a glide" (Dolan and Mimori 1986).[14] Consequently, experiments using TDNNs to recognize Japanese phonemes (Waibel *et al* 1989b) have avoided the problem of diphthong recognition altogether.

Diphthong	Usage (%)
/ai/	1.83
/ei/	1.71
/ou/	1.51
/au/	0.61
/ɔi/	0.14

Table 2.4-1 The average usage of each closing diphthong in colloquial RP English (Gimson 1989). For this accent, the closing diphthongs occur 4.8% of the time and the vowels (all types) 39.2%.

Table 2.4-1 lists the frequency of occurrence of the closing diphthongs in colloquial RP English (Gimson 1989). Assuming these levels are applicable to New Zealand English also, since this accent is morphologically and phonologically similar to RP (Maclagan 1975),

---

importantly, there is currently considerable uncertainty about the number and nature of centring diphthongs within New Zealand English, due to the apparent merger and monophthongalization of /iI/ and /eI/ and the infrequent usage of /uI/ (Maclagan 1982).

[14]A *mora* is a unit of timing; each mora takes the same amount of time to say (Ladefoged 1982).

the diphthongs /ai/ and /ei/ are the most frequently used, closely followed by /ou/. These usage levels are significant, since the experimental results discussed in chapter 5 demonstrate that /ai/ and /ei/ are the most readily confused of the five closing diphthongs of New Zealand English, at least when realized within phonemic contexts including /b/, /d/ and /g/.

The experiments involving closing diphthongs discussed in chapter 5 are motivated by the small number of results reported concerning closing diphthong recognition by TDNNs, particularly for non-American accents. Realizations of the closing diphthongs produced by American and New Zealand English speakers differ significantly, posing different problems for recognition. In particular, American realizations of /ai/ and /ei/ are distinctive since only the latter is a genuine diphthong (see Bond 1982), whereas in New Zealand English, both phonemes are genuine diphthongs whose realizations may share similar spectral features (see Figure 4.2.3.1-4).

2.5 Automated Speech Recognition Using Sub-word Units

This section presents an overview of automated speech recognition in conjunction with sub-word units[15], including the phoneme. These units, and the recognition approaches with which they are commonly associated, are discussed in the next section. This is followed in §2.5.2 by an overview of automated phoneme recognition, the variety of sub-word recognition attempted in this work.

2.5.1 Some Sub-word Units

Attempts at automated speech recognition through *sub-word* unit recognition are motivated strongly by the desire to achieve *large vocabulary* speech recognition (Holmes and Pearce 1993; Lee 1990). Recognition systems based on word recognition have been shown to work successfully for small vocabularies (Lippmann *et al* 1987; Rabiner *et al* 1988), however, increasing the scale of these systems poses several serious problems. For example, word recognition systems must be trained for each word individually (training data may not be shared between words, Lee 1990), implying a large quantity of training data is required. This inconvenience extends to user-added words for which several repartitions are also required to enable training.

In contrast to words, sub-word units permit training data for more than one unit to be

[15]The term *sub-word unit* is one used commonly by engineers, whereas linguists prefer the term *segmental unit.*

derived from a common utterance, reducing the number of utterances required for training. For example, an utterance of the word *bide* (/baid/) may be used to create training data for *three* phonemes (/b/, /ai/ and /d/), rather than just *one* word. This approach becomes desirable when the size of the vocabulary exceeds the number of sub-word units to be recognized.

Lee (1990) suggests good sub-word units should be *consistent* and *trainable* (sentiments echoed by Holmes and Pearce 1993). Consistency implies that different instances of the same sub-word unit should have similar characteristics, while trainability refers to the amount of training data required for successful system operation. Importantly, the degree to which a sub-word unit must be consistent is dependent upon the approach used to recognize it. For example, Lee and Alleva (1992) regard the phoneme as an *inconsistent* unit, when used in conjunction with hidden Markov models (HMMs), since phonemic HMMs perform poorly due to their "broad distributions". Despite this, other authors, such as Waibel (1992), find the phoneme sufficiently consistent to allow good recognition performances in conjunction with artificial neural networks (ANNs). Both authors agree that the limited number of phoneme units (43 for New Zealand English, see Table 2.3-1) makes such units very trainable.

To improve the performance of HMMs for sub-word unit recognition, a number of new sub-word units have been proposed. For example, some authors use *diphone* and *triphone* units to overcome the variation inherent in phoneme realizations. These units are used to better model phoneme realizations and the transitions between them (Lee 1990). Though diphone and triphone HMMs prove more consistent than phoneme HMMs, they are less trainable since there are significantly more of them. Lee (1990) overcomes this problem by using *generalized triphones*, which are clusters of traditional triphones having similar hidden Markov models. This approach significantly reduces the number of sub-word units required making them more trainable.

Despite these and other sub-word units proposed, the phoneme is selected for automated recognition in this thesis since many authors, such as Kasabov (1993), Fallside (1992), Watrous (1990) and Waibel *et al* (1989a, 1989b), have shown that phonemes may be successfully recognized using ANNs.

2.5.2 Automated Phoneme Recognition: An Overview

Automated phoneme recognition is defined in this thesis as the process of assigning abstract phonemic symbols to speech portions in order to produce *phoneme sequences* resembling traditional phonemic transcriptions in symbol ordering. Due to co-articulation, it is likely that such portions may contain features associated with *more than one phoneme.* Consequently, it is assumed that the label assigned to a speech portion during phoneme recognition corresponds to the phoneme most strongly indicated by the features it contains

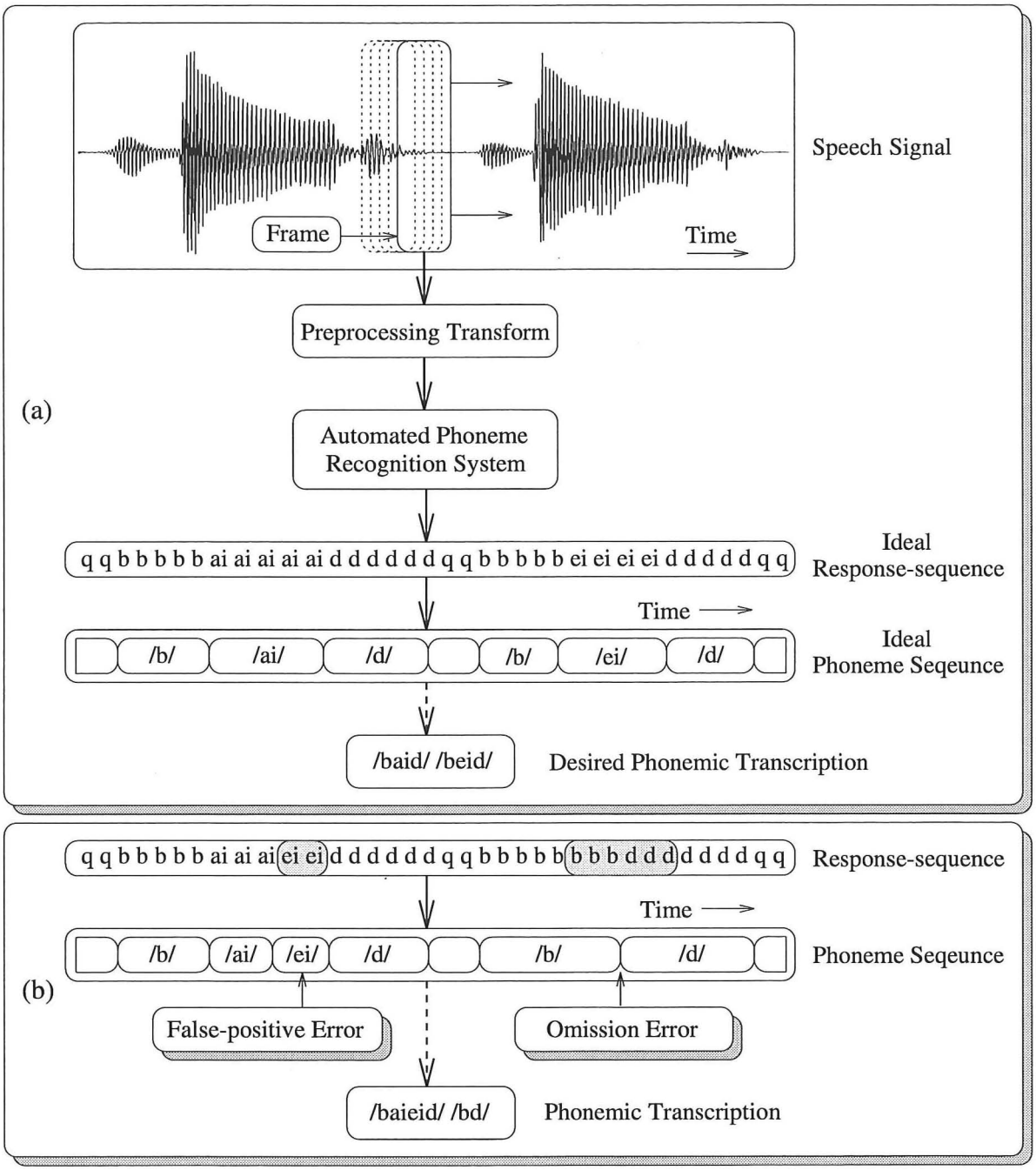

Figure 2.5.2-1. A simplified view of the ideal operation of an automated phoneme recognition system (part (a)) and examples of the types of errors which may occur during non-ideal operation (part (b)). During operation, speech signal portions are selected by a frame sliding along the time axis and transformed into input suitable for a recognition system. This system responds to its input (one response for each frame position) by producing a sequence of phoneme labels (or *q* indicating "silence"), referred to as a *response-sequence.* Ideally (part (a)), this sequence contains contiguous groupings of identical phoneme labels which are readily collapsed to form a phoneme sequence. In practice (part (b)), the response-sequence may contain unwanted responses, leading to false-positive errors, or leave out desired responses, leading to omission errors.

(the phoneme with the highest correlation in terms of Fant's model, see Figure 2.2.3-1).

Figure 2.5.2–1 (a) shows a simplified view of the ideal operation of a phoneme

recognition system. Speech portions are selected by a *frame* which is stepped progressively across a speech signal and transformed to provide input for a recognition system. A *preprocessing transform* is necessary to highlight those features likely to assist phoneme recognition, while eliminating irrelevant features like those arising from a speaker's environment (this transformation is equivalent to the *feature analysis* depicted in Figure 1.1-1).

The recognition system responds to its input by producing a sequence of phonemic symbols, referred to as a *response-sequence* in this thesis. For simplicity, this response-sequence is depicted containing a single phonemic symbol for each frame position processed, though in practice it is likely to contain several alternatives, ranked in order of likelihood (this arrangement may be necessary to handle ambiguous phoneme realizations, such as that arising in the *I was riding/writing* example discussed in §2.2.2). Ideally, a response-sequence contains contiguous groups of replicated symbols corresponding to individual phoneme realizations and may be readily collapsed to form an ideal *phoneme sequence*, like that depicted in Figure 2.5.2-1 (a). Phoneme sequences contain information concerning the identity, extent and relative temporal positions of phoneme realizations observed. Such sequences may be further simplified to produce representations approximating traditional phonemic transcriptions. However, for the purposes of automated speech recognition, it is desirable to retain as much information as possible concerning observed phoneme realizations, to aid subsequent processing.

When training an automated phoneme recognition system, the aim is to make the recovery of an ideal phoneme sequence as easy and reliable as possible. Ideally, the examples used to train a system should constrain its behaviour such that only desired responses are produced. In practice, however, this is not always possible leading to two types of errors, referred to as *omission* and *false-positive errors* (Miyatake *et al* 1990 refers to these as deletion and insertion errors, respectively). An omission error occurs when the realization of a given phoneme present in an utterance does not elicit this phoneme's symbol from a recognition system. By contrast, a false-positive error occurs when a phonemic symbol is produced that does not correspond to the phoneme realization actually being processed. Examples of these errors are depicted in Figure 2.5.2-1 (b).

Chapter 3

Speech Acquisition and Preparation

This chapter discusses the acquisition and preparation of the speech utterances used in the experiments described in chapter 5. §3.1 discusses the vocabulary of the utterances acquired as well as the method by which these utterances were digitally sampled. This is followed in §3.2 by a discussion of the preprocessing used in this work to prepare speech signals for phoneme recognition. Finally, §3.3 discusses the method used to select speech portions from closing diphthong realizations to generate training tokens for TDNNs.

3.1 Speech Acquisition and Vocabulary

For the experiments discussed in chapter 5, two adult speakers of *general New Zealand English* (see §2.3), one female (*speaker HD*) and one male (*speaker JK*), were recorded. Both speaker's utterances were recorded digitally in an anechoic chamber using an IBM personal computer equipped with a 16 bit analogue-to-digital converter board (an *SX-10* by Antex Electronics). A *sampling frequency* of F_{samp}=10 kHz was used to ensure the transitions of each speaker's first three formants were captured (for realizations of the closing diphthongs, these lie below 3.2 kHz on average, as shown in Figure 4.2.3.1-4). An anti-aliasing filter was provided automatically by the *SX-10* board with its *cutoff frequency* set to F_{-3dB}=4.4 kHz. Microphone amplification was achieved using a custom built amplifier with a maximum gain of 60 dB.[1]

Figure 3.1-1 shows the frequency-domain characteristics of the analogue channel used when recording speech in this work (the channel from the amplifier's input to the input of the *SX-10* board's analogue-to-digital converter). Within the frequency range of the first three formants observed for speakers JK and HD (see the lighter shaded region in Figure 3.1-1 (a)), the magnitudes of aliased frequencies (frequencies exceeding the Nyquist frequency $F_{Nyquist}$=5 kHz, Owens 1993; see the darker shaded regions in Figure 3.1-1 (a)) are attenuated by at least 70 dB, as suggested necessary by Owens (1993). Within this same frequency range, the phase

[1]Such a high gain is necessary to permit *natural* speech utterances from quieter speakers to be recorded without undue contamination as a consequence of *quantization noise* (see Stremler 1982 for a discussion of quantization noise).

response of the analogue channel is approximately linear as shown in Figure 3.1-1 (b). No attempt has been made to correct for the phase lags at higher frequencies in this response, since phase information is ignored during the preparation of speech signals for automated phoneme recognition is this work (see §3.2).[2]

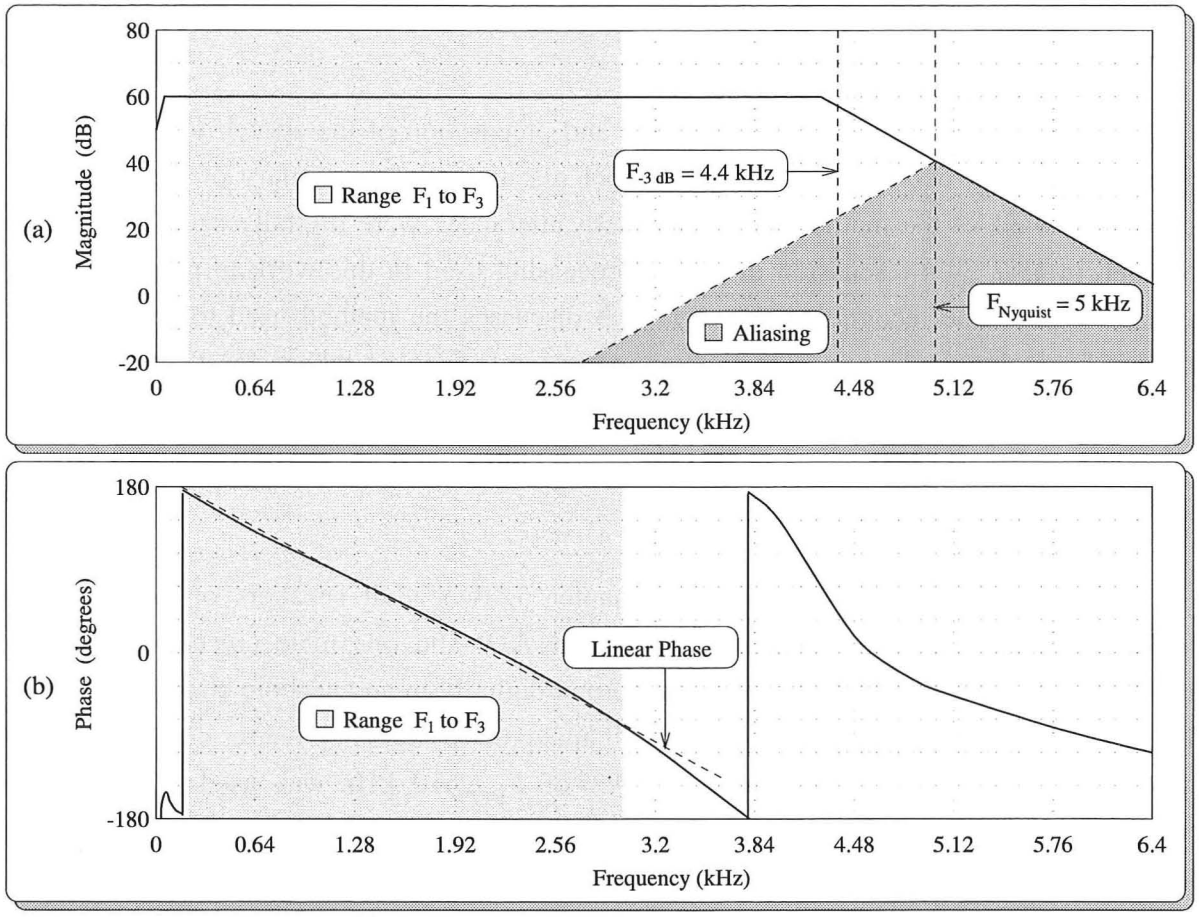

Figure 3.1-1. The (a) magnitude response and (b) phase response of the analogue channel (less the microphone) used for recording speech in this work. The magnitude response decays rapidly after the *cut-off* frequency (F_{-3dB}) so that aliased frequencies (frequencies exceeding $F_{Nyquist}$=5 kHz) are attenuated heavily, as indicated by the darker shaded region marked *Aliasing*. Within the range of the first three formants exhibited by speakers JK and HD (the lighter shaded region marked *Range F_1 to F_3*), aliased frequencies are attenuated by at least 70 dB (part (a)) and the phase response is approximately linear (part (b)).

The next section discusses the utterances recorded to train and test examples of the expert modules for closing diphthong recognition discussed in §4.2.3. This is followed in §3.1.2 by a discussion the utterances recorded to test the abilities of the squad-based expert modules discussed in §4.2.3 to "ignore" input corresponding to monophthong realizations.

---

[2]This approach is common and is based on the observation that the Human auditory system is insensitive to phase differences *between frequencies* (see §2.1.3).

3.1.1 Closing Diphthong Syllables

Following the initial experiments involving the recognition of Japanese phoneme realizations in conjunction with TDNNs (see §4.2.1 and Waibel *et al* 1989a; Waibel *et al* 1989b; Haffner *et al* 1989; Sawai *et al* 1989; Miyatake *et al* 1990; Minami *et al* 1990; Sawai 1991a), realizations of the five closing diphthongs of New Zealand English were recorded in the context of short *isolated* utterances. In particular, each of the five closing diphthongs was recorded in the phonemic contexts listed in Table 3.1.1–1 to permit observations of isolated and syllable initial, medial and final realizations.

Final	Initial			
	/b/	/d/	/g/	/_/
/b/	/b_b/	/d_b/	/g_b/	/_b/
/d/	/b_d/	/d_d/	/g_d/	/_d/
/g/	/b_g/	/d_g/	/g_g/	/_g/
/_/	/b_/	/d_/	/g_/	/_/

Table 3.1.1-1. The sixteen phonemic contexts in which each of the five closing diphthongs of New Zealand English was recorded in this work. In each context, '_' represents one of the five closing diphthongs.

In New Zealand English, all five closing diphthongs (particularly /ei/) may be realized in isolation (represented by /_/ in Table 3.1.1-1) during informal speech, though only realizations of /ai/ constitute formal English words (the words *eye* and *I*). Some of the phoneme combinations listed in Table 3.1.1-1, like /beib/ (*babe*) and /baid/ (*bide*), represent common words in New Zealand English, while others, like /doud/ (in *endowed*) and /gaid/ (in *brigade*), constitute syllables appearing in common words. Some combinations, like /gɔig/ and /gɔib/, constitute nonsense syllables which appear in no common English words at all. Such utterances are necessary for completeness when training a system for closing diphthong recognition, since words comprising these unusual phoneme combinations may become common at any time. For example, /geig/ only appears in the name *Geiger* associated with the well known term *Geiger counter* (entering English circa 1928, Korff 1955). Dolan and Mimori (1986) also use isolated nonsense syllables during their experiments with English diphthongs.

For both speakers JK and HD, four realizations of each closing diphthong in each phonemic context listed in Table 3.1.1-1 were sampled, giving a total of 320 isolated utterances per speaker. Within each speaker's utterances, a total of 800 phoneme realizations were recorded, including 320 closing diphthong realizations (64 realizations of each) and 480 voiced plosive realizations (160 realizations of each). Each utterance was stored in an individual file surrounded by short intervals of "silence".

Though the syllable contexts listed in Table 3.1.1-1 are only a small fraction of those possible in English, the realizations of each closing diphthong recorded in these contexts still exhibit significant variation, particularly in the on- and off-glides. For example, Figure 3.1.1-1 shows F_1-F_2 trajectories estimated from the *medial* realizations of /ai/ produced by speaker HD (36 realizations in total; note that Figure 2.4-1 (c) shows the trajectory associated with one of these realizations). Though all aspects of these realizations exhibit variation, the off-glides in particular vary significantly depending of the identities of the final voiced plosive realizations.

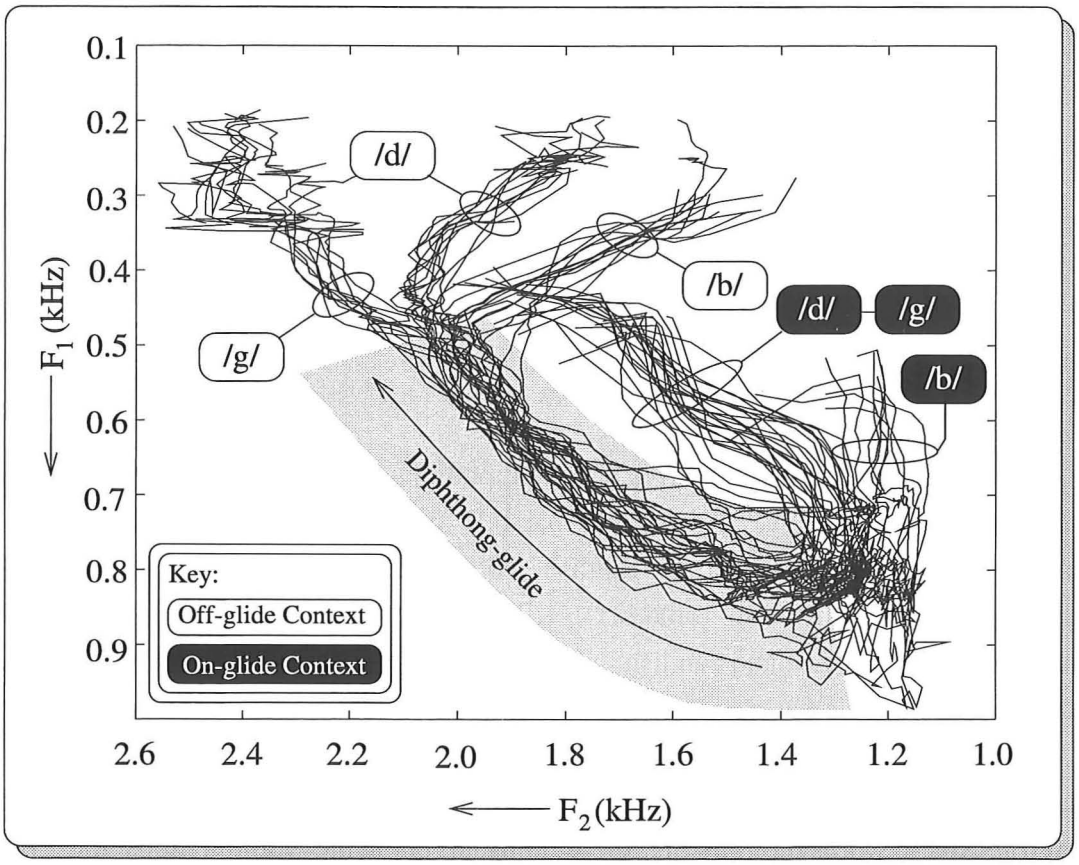

Figure 3.1.1-1. F_1-F_2 trajectories for syllable medial realizations of /ai/ produced by speaker HD. The shaded region highlights the diphthong-glides associated with all 36 realizations shown, which all transit in the direction indicated by the arrow. The on- and off-glide contexts marked indicate the identities of the voiced and unvoiced plosives that influence the trajectories with which they are associated.

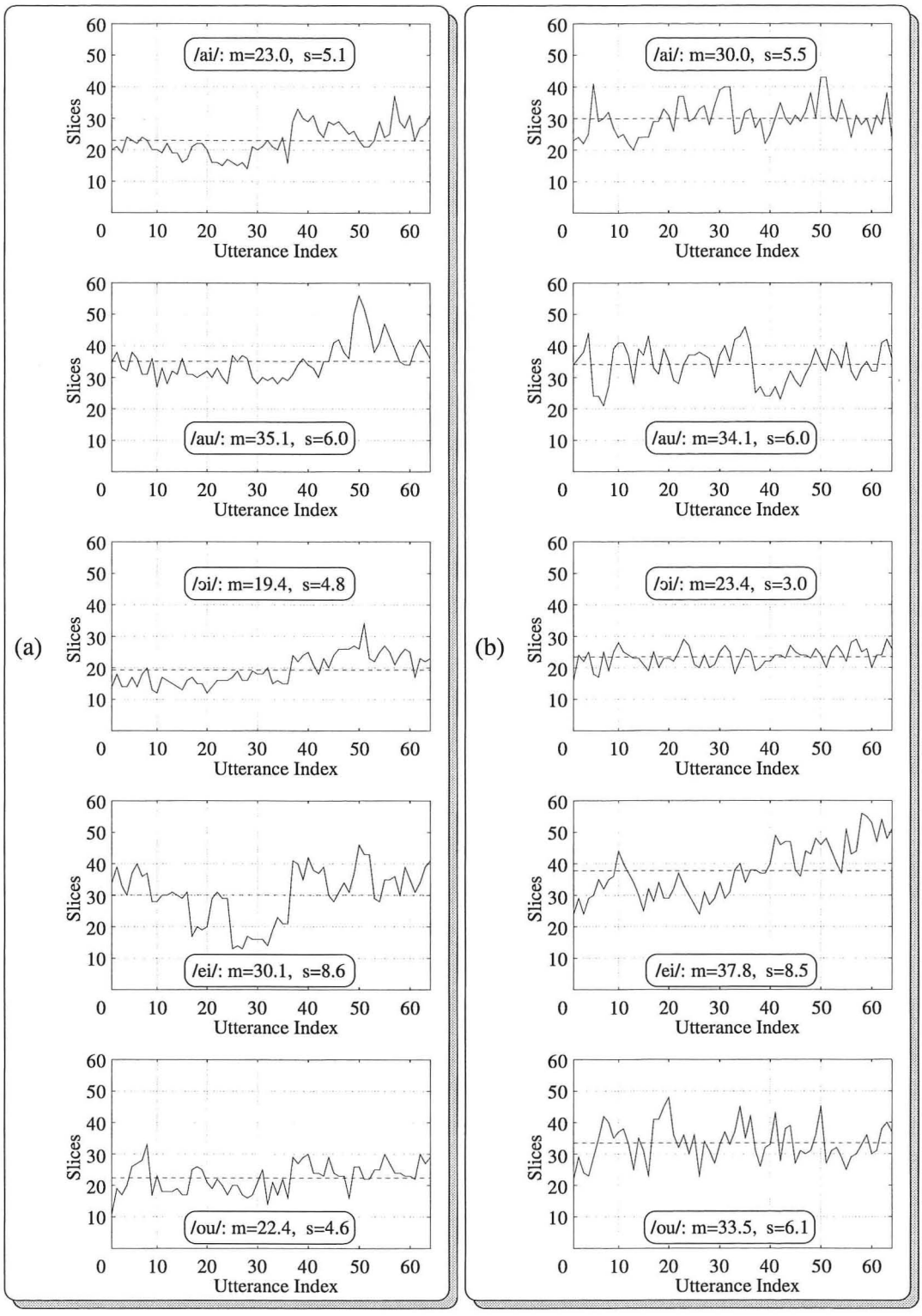

Figure 3.1.1-2. Plots of diphthong-glide lengths (measured in token slices, see text) for (a) speaker JK and (b) speaker HD. Within each plot, values for the mean (m) and standard deviation (s) of the 64 diphthong-glide lengths displayed are given. The mean glide length is also indicated in each plot by the horizontal dashed line.

As the shaded region in Figure 3.1.1-1 indicates, the *diphthong-glides* associated with a closing diphthong's realizations may be similar, irrespective of the context in which it is realized. It is assumed in this thesis that such glides are characteristic of their associated diphthong's realizations and efforts are made to include them in tokens representing these phonemes (see §3.3). Figure 3.1.1-2 shows the diphthong-glide *lengths* measured from the closing diphthong realizations recorded for speakers JK and HD. These lengths were estimated manually using the software package *STEP* discussed in §3.3 and are expressed in *slices* indicating the approximate sizes of the tokens required to *fully* represent them (see §3.2 for a discussion token slices). The average slice length of 28.9 slices for all 640 closing diphthong realizations considered (an average length of 26 and 31.7 slices for speakers JK and HD, respectively) is nearly twice the number of slices used by Waibel *el al* (1989a) to form tokens representing Japanese phoneme realizations (15 slices). To more fully represent (on average) the diphthong-glides of speaker JK's and speaker HD's closing diphthong realizations, *extended-token TDNN* (see §4.2.3.1) uses tokens containing 30 slices. The benefits of this approach are evaluated by comparing the performances of expert modules comprising these TDNNs with those comprising basic-token TDNNs which use tokens containing 15 slices (see §5.1).

3.1.2 Monophthong Syllables

To permit testing of the abilities of squad-based expert modules for closing diphthong recognition to "ignore" monophthong realizations (see §5.1.4.2 and §5.2.2), utterances containing such realizations were sampled for speakers JK and HD. One realization of each monophthong in Table 2.3-1, excluding /a/, was recorded in each of the sixteen contexts listed in Table 3.1.1-1, giving 160 isolated utterances per speaker (realizations of /a/ were ignored due to their similarity to realizations of /ʌ/ in New Zealand English; see Figure 2.3-4). Within each speaker's utterances, a total of 400 phoneme realizations were recorded, including 160 monophthong realizations (16 realizations of each) and 240 voiced plosive realizations (80 realizations of each).

3.2 Speech Preprocessing

As discussed in §2.5.2, speech signals to be processed by an automated phoneme recognition system must first be converted into a suitable form by a *preprocessing transform.* This section discusses the preprocessing transform used in this work, which was first proposed by Waibel *et al* (1989a). This transform converts a speech signal into a *spectrogram*

comprising a series of mel-scaled spectrums from which *tokens* suitable for a TDNN may be derived. It is now explained in conjunction with Figures 3.2-1 and 3.2-2.

Preprocessing commences by computing *mel-scale* magnitude spectrums of the speech samples contained within a sliding frame (see Figure 3.2-1). This frame contains N=256 speech samples (25.6 msec of speech when F_{samp}=10 kHz) and is shifted by 50 samples (5 msec) between successive spectrum estimations to permit short-time spectral characteristics to be tracked (Owens 1993; Morgan and Scofield 1991). For each frame position, a mel-scale magnitude spectrum is estimated as follows. First the frame's samples, $s(nT_{samp})$ $n=0,1,...,N-1$ (see Figure 3.2-1 (a)) are scaled by the samples of a *DFT-even*[3] *Hamming window function*, $h(nT_{samp})$, given by

$$h(nT_{samp}) = \begin{cases} 0.54 - 0.46\cos\left(\dfrac{2\pi n}{N}\right), & n = 0,1,...,N-1 \\ 0, & \text{otherwise} \end{cases} \tag{3.2-1}$$

(Harris 1978). The scaled samples, $s(nT_{samp})h(nT_{samp})$ $n=0,1,...,N-1$ (see Figure 3.2-1 (b)), are then transformed into the frequency-domain by evaluating the *discrete Fourier transform* (DFT) given by

$$S(k) = \sum_{n=0}^{N-1} s(nT_{samp})\,h(nT_{samp})\,e^{-j\frac{2\pi nk}{N}}, \quad k = 0,1,...,N-1 \tag{3.2-2}$$

where $S(k)$ is the complex-valued spectral estimate at the frequency $f=kF_{samp}/N$ and $T_{samp}=1/F_{samp}$ is the sampling interval (Owens 1993; Morgan and Scofield 1991). The *linear-scale* magnitude spectrum given by $|S(k)|$, $k=0,1,2...N-1$ (Figure 3.2-1 (c) shows the logarithm of this spectrum), is then compressed into a *mel-scale* log-magnitude spectrum (Figure 3.2-1 (d)) using the transform proposed by Waibel and Yegnanarayana (1981) (henceforth referred to as a *Waibel transform* for convenience). The i^{th} coefficient produced by the Waibel transform, $W(i)$ (assuming F_{samp}=10 kHz as in Waibel and Yegnanarayana 1981), is given by

[3]A sampled window function is DFT-even when $w(n)=w(N-n)$ for n=1,2,...,N-1 and $w(n=0)$ is unmatched (where N is the number of samples forming the window) (see Harris 1978).

$$W(i) = \log_{10}\left(\frac{\left|S\left(k_s(i)\right)\right|}{2} + \sum_{k=k_s(i)+1}^{k_e(i)-1} \left|S(k)\right| + \frac{\left|S\left(k_e(i)\right)\right|}{2} \right) \tag{3.2-3}$$

where i=1,2,...,16 and the indices k_s and k_e are listed in Table 3.2–1 for each value of i.[4]

As shown in Figure 3.2-1 (d), the frequency bands associated with the sixteen coefficients produced by the Waibel transform vary in width depending on frequency. This variation is intended to simulate the varying frequency resolution of human auditory perception, as discussed in §2.1.3. *Log compression* is also used within the Waibel transform to simulate varying human auditory sensitivity (see §2.1.3 also). Apart from simulating human perception, the Waibel transform also reduces the amount of data required to represent spectrum estimates (from 129 coefficients representing the positive frequencies of the linear-scale magnitude spectrum, $|S(k)|$, down to 16). This was done to reduce the number of free parameters in a TDNN, thereby simplifying its training (see §4.2).

i	k_s	k_e	f_s (Hz)	f_e (Hz)
1	0	2	0	78
2	2	6	78	243
3	6	10	243	391
4	10	14	391	547
5	14	18	547	703
6	18	22	703	859
7	22	26	859	1015
8	26	30	1015	1172
9	30	35	1172	1367
10	35	41	1367	1602
11	41	48	1602	1875
12	48	57	1875	2227
13	57	68	2227	2656
14	68	81	2656	3164
15	81	97	3164	3789
16	97	116	3789	4531

Table 3.2-1. Indices and frequencies associated with the sixteen mel-scale coefficients produced by the Waibel transform. The i^{th} mel-scale band commences at $k=k_s(i)$ and ends at $k=k_e(i)$, which corresponds (approximately) to the start and end frequencies f_s and f_e, respectively.

[4]The frequencies associated with the indexes listed in table 3.2-1 differ from those listed by Waibel and Yegnanarayana (1981) who incorrectly associate $S(k=2)$ with (approximately) 117 Hz instead of (approximately) 78 Hz. When F_{samp}=10 kHz, a 256 point DFT has a frequency resolution of (approximately) 39.1 Hz. Consequently, all the frequencies listed in Waibel and Yegnanarayana (1981), expect $S(k=0)$=0 Hz, are in error by (approximately) 39.1 Hz.

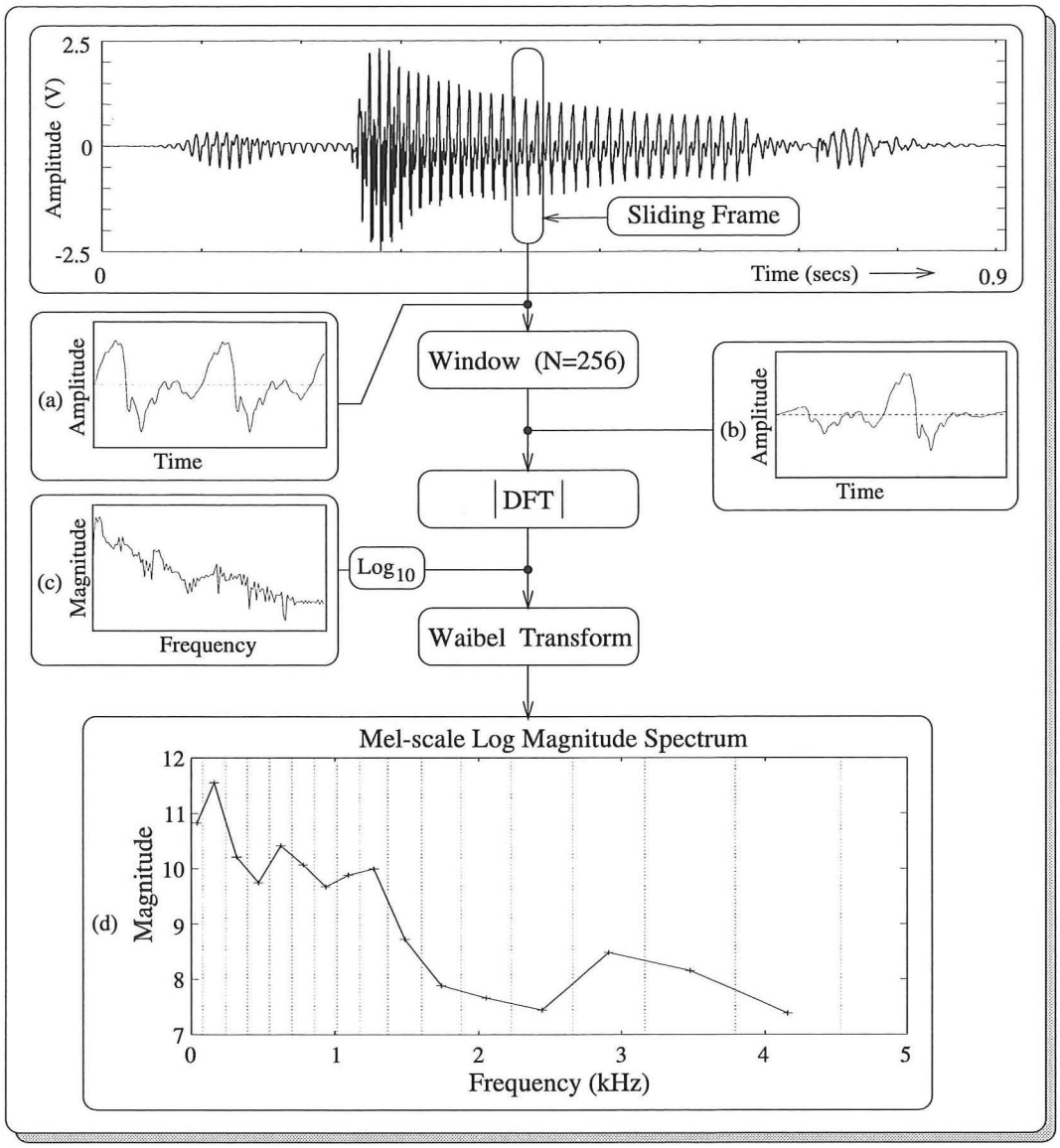

Figure 3.2-1. The method by which speech signals are processed in this work to produce mel-scale log magnitude spectrums. A group of N=256 speech samples (part (a)) is selected by a sliding frame and scaled by a (Hamming) window (the results of this scaling are shown in part (b)). The scaled speech samples are then transformed using a discrete Fourier transform (DFT) to give a magnitude spectrum (ultimately; see part (c)) whose coefficients are further transformed into a mel-scaled log magnitude spectrum (part (d)) using the Waibel transform.

Having evaluated a sequence of mel-scaled log-magnitude spectrums for a speech signal (or portion thereof), preprocessing is completed by averaging pairs of these spectrums (coefficient wise) to form the columns, or *slices*, of a spectrogram like that depicted in Figure 3.2-2 (b). In this representation of a speech signal, the vertical axis constitutes *mel-scaled* frequency and the horizontal axis constitutes time. Note that each slice of a spectrogram is created from a speech portion that is offset by 10 msec from that used to create its neighbouring slices, implying a 10 msec slice rate (Waibel *et al* 1989a). Though low in

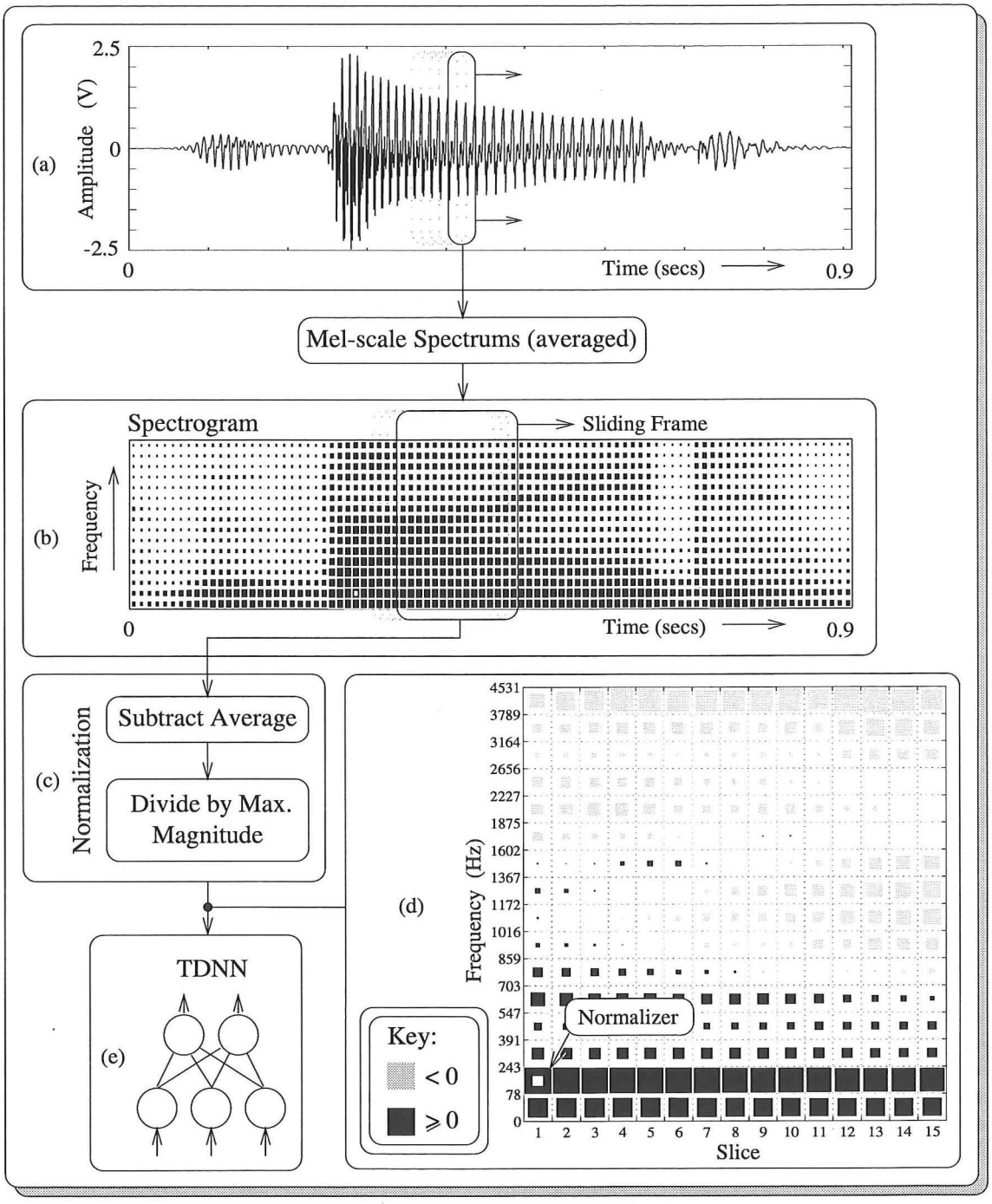

Figure 3.2-2. The method by which mel-scale log magnitude spectrums are combined to form a *spectrogram* from which tokens for a TDNN may be derived. A speech signal (part (a)) is first processed as in Figure 3.2-1 to give a sequence of mel-scaled log magnitude spectrums which are then averaged in pairs to produce a spectrogram (part (b)). Groups of slices from this spectrogram are then selected (again by a sliding frames) and normalized (part (c)) to produce a token like that in part (d). This token is depicted using a Hinton diagram in which the black squares represent the magnitudes of *positive* coefficients and the grey squares represent the magnitudes of *negative* coefficients.

frequency resolution, particularly at higher frequencies, this spectrogram contains sufficient information to permit phoneme recognition, as evidenced by the excellent expert module

recognition performances presented in chapter 5.

From spectrograms like that depicted in Figure 3.2-2 (b), tokens for a TDNN may be derived by selecting several slices (like the group enclosed within the frame shown in Figure 3.2-2 (b)) and normalizing these so that their coefficients all lie between ±1 (Waibel *et al* 1989a). Normalization is accomplished by first subtracting the average coefficient magnitude from each coefficient in a group of slices selected and then dividing by the largest resulting coefficient *magnitude* (Waibel *et al* 1989a; see Figure 3.2-2 (c)). This approach is referred to as *token-wide normalization* and results in tokens like that depicted in Figure 3.2-2 (d) (this token is depicted using a Hinton diagram in which the black squares represent the magnitudes of *positive* token coefficients and grey squares represent the magnitudes of *negative* coefficients; see Hinton and Sejnowski 1986). In this thesis, the coefficient with the largest magnitude after the average coefficient magnitude is subtracted, is referred to as the *normalizer* and is represented in a token by a unit sized square (the maximum size) with a hollow centre. The frequencies listed in Figure 3.2-2 (d) correspond to the limits of the bands associated with the critical band filters used by the Waibel transform (see Table 3.2-1).

During the normal operation of a TDNN based system for phoneme recognition, the frame used to select spectrogram slices for token generation is advanced *slice-by-slice* (Minami *et al* 1991), as depicted in Figure 3.2-2 (b). This approach enables the recognition of phoneme realizations *without* careful pre-segmentation of speech signals and is the reason why TDNN based systems produce repeated phonemic symbols in response to each phoneme realization processed (see Figure 4.2.2-1).

3.3 Speech Portion Selection For Training

When training a TDNN based system for phoneme recognition, a set of tokens with *known* phonemic identities is required. To generate *training tokens* representing a given phoneme, appropriate speech portions must first be selected from its realizations. Traditionally, such speech portions have been selected about the "centres" of a phoneme's realizations (see Waibel *et al* 1989a; Waibel *et al* 1989b; Miyatake *et al* 1990). For certain phonemes, such "centres" are assumed to correspond, conveniently, to "obvious" *time-domain* features within their realizations. For instance, Waibel *et al* (1989a) assumes the vowel onsets within voiced plosive realizations correspond to their "centres". This assumption permits simple speech portion selection based on time-domain observations. Unfortunately, however, the realizations of many phonemes, such as the closing diphthongs, may exhibit no obvious time-domain features, since they are vocoids.

One might argue that the "centre" of a vocoid could be deduced by interpolating between obvious time-domain features corresponding to neighbouring phoneme realizations.

However, this approach assumes that neighbouring realizations exhibit such features and that the "centre" being sort lies medially between them. Clearly, in utterances of words like *waylay* (/weilei/), the first assumption is unlikely to be valid when attempting to locate the "centres" of /ei/'s realizations. For the first realization in particular, /ei/'s neighbouring phonemes /w/ and /l/ are, themselves, realized as vocoids and may not provide the required time-domain features to permit the "centre" of /ei/'s realization to be located. Consequently, a more sophisticated method of determining the "centres" of phonemes realized as vocoids is required to permit speech portions representing them to be selected.

In this work, speech portions representing phonemes realized as vocoids are selected using *time-domain* "centres" determined from *frequency-domain* observations. In the frequency-domain, vocoids are characterized by formant tracks (see §2.1.2) which may be readily estimated and displayed. For a given phoneme (whose realizations are vocoids), the characteristic features of such tracks may be assessed simply by viewing several of its realizations *simultaneously* in the frequency-domain (see Figure 3.1.1-1 for example). These features may then be used to locate *frequency-domain* "centres" that in turn may be used to compute *time-domain* "centres" for speech portion selection. This approach is used in this work to generate training tokens to suit the different TDNNs discussed in §4.2.3.[5]

The remainder of this section discusses a purpose written software package for finding the time-domain "centres" of vocoids and selecting speech portions using them. For convenience, this package is referred to as *STEP*, short for *Speech Training Example Preparation*. STEP is intended to reduce the human effort required to select speech portions from vocoids, while still allowing the validity of the portions selected to be easily checked. Figure 3.3-1 shows a simplified flow diagram of STEP's operation with graphical inserts resembling those it produces. Its operation is now explained in the context of processing closing diphthong realizations.

Using STEP, the realizations of a given closing diphthong are first processed to produce sets of smoothed formant tracks. These tracks are determined by estimating the formant frequencies and bandwidths associated with the speech samples selected by a *sliding frame*. For convenience, this frame is identical to that used to create the mel-scaled spectrograms discussed in the previous section. For each frame position, estimates of the formant frequencies and bandwidths are determined from a *linear predictive model* of the speech samples contained, using the method given in Chandra and Lin (1974). These estimates are then used to form smoothed formant tracks using a *tracking algorithm* based on

[5]Note that for sequence-token TDNNs, the "centres" selected are within the realizations of *sub-phoneme objects*, rather than within the realizations of *phoneme objects*, as for basic- or extended-token TDNNs.

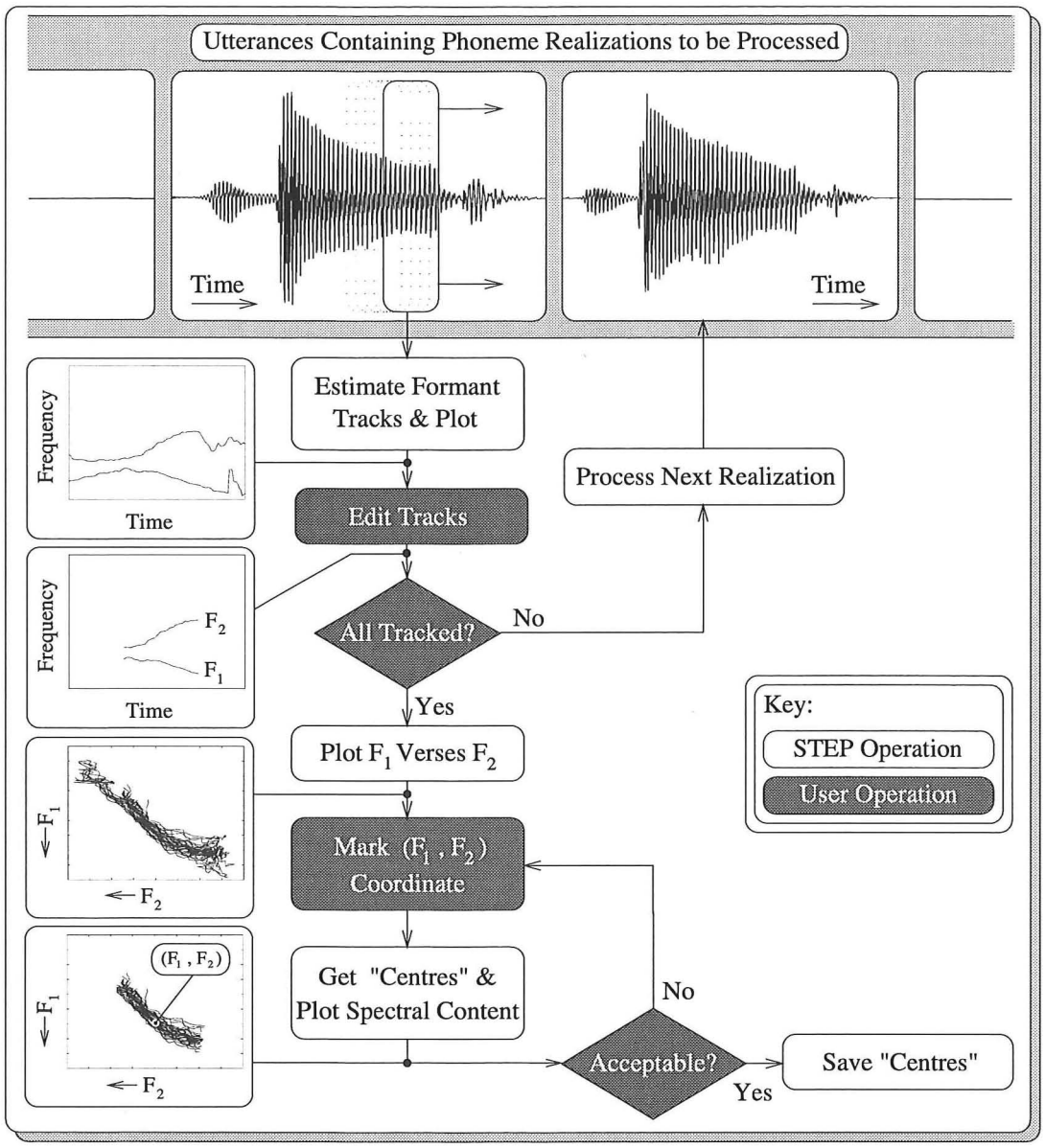

Figure 3.3-1. A flow diagram of the operation of *STEP* when processing closing diphthong realizations.

that proposed by McCandles (1974) (see also Seneff 1976; Owens 1993).

Having determined the sets of smoothed formant tracks for a given closing diphthong, these are then manually cropped to prevent STEP from assigning time-domain "centres" to incorrect speech samples. For example, the formant tracks leading to the F_1-F_2 trajectories depicted in Figure 3.1.1-1 are cropped to give the trajectories shown in Figure 3.3-2, to prevent time-domain "centres" from being incorrectly assigned to speech samples associated with the on- or off-

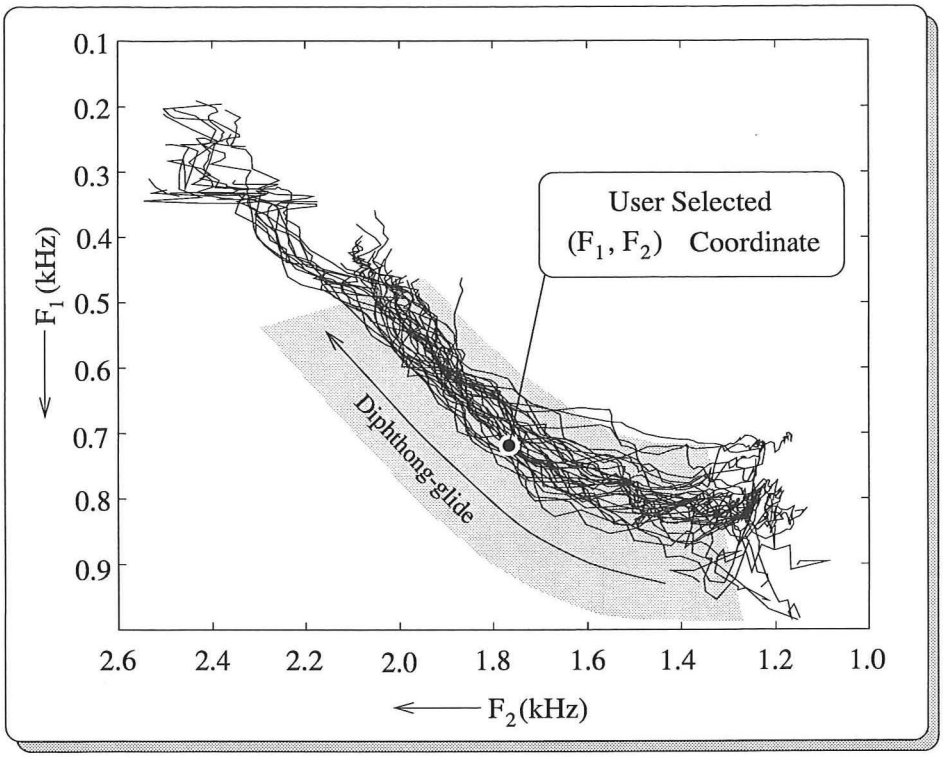

Figure 3.3-2. Examples of cropped F_1-F_2 trajectories corresponding to realizations of /ai/ by speaker HD.

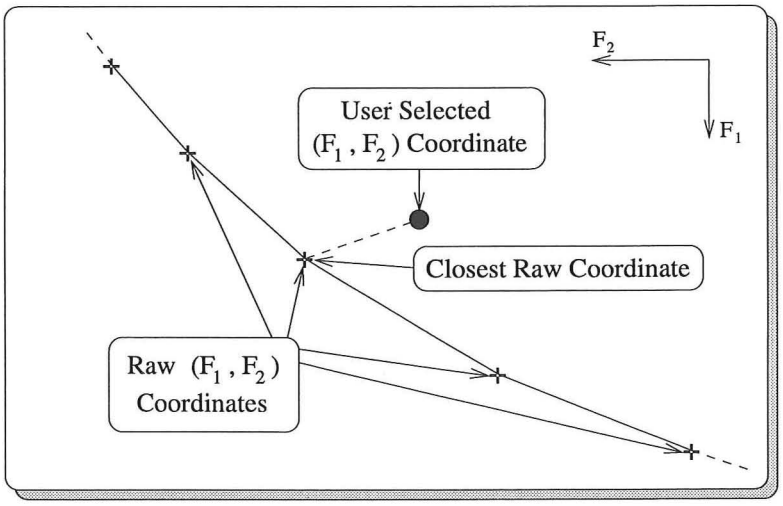

Figure 3.3-3. Finding the closest raw (F_1,F_2) coordinate on an F_1-F_2 trajectory from a user selected coordinate.

glides.[6] Having cropped the formant tracks, F_1-F_2 trajectories are then plotted to permit a user

[6]On- and off-glides are peripheral in closing diphthongs realizations and vary with context as shown in figure 3.1.1-1. In some contexts these glides may lie close to the diphthong-glides of such realizations, causing STEP to mistake them for diphthong-glides unless they are cropped.

to select an (F_1,F_2) coordinate (a frequency-domain "centre") based on the frequency-domain features observed. Such a coordinate in shown in Figure 3.3–2.

From the user selected (F_1,F_2) coordinate, STEP then determines the nearest *raw* (F_1,F_2) coordinate on each of the F_1-F_2 trajectories displayed, as depicted for one trajectory in Figure 3.3-3.[1] Knowing the speech samples used to compute these raw (F_1,F_2) coordinates, STEP then determines the time-domain "centre" of each phoneme realization being processed. In addition, STEP also displays the F_1-F_2 trajectories of the speech portions that would be selected using these "centres" (these portions are of a known *fixed length* suitable for token generation). By observing these trajectories, a user may locate their (F_1,F_2) coordinate to best capture the appropriate frequency-domain features of the phoneme realizations being processed. This coordinate may be relocated repeatedly until a suitable one is found. Once a satisfactory (F_1,F_2) coordinate is selected, details of the time-domain "centres" and speech portions selected are stored for each phoneme realization processed. Using this information, tokens like those discussed in §5.1.1 may be generated.

[1]These coordinates correspond to the raw F_1 and F_2 formant frequencies found during formant estimation in conjunction with the sliding frame. For convenience, the closest raw coordinate is found using a Euclidean distance measure.

Chapter 4

ANNs and Automated Phoneme Recognition

Within recent years a number of artificial neural network (ANN) approaches to the problem of phoneme recognition have been proposed including traditional multi-layer perceptrons (or multi-layer feed-forward networks), Kohonen's learning vector quantizer (LVQ), time-delay neural networks (TDNNs), time-delay LVQ and recurrent networks (see Lippmann 1989; Morgan and Scofield 1991). Of these, the "most promising" results have been reported in conjunction with various forms of TDNNs (Lippmann 1989). In this work, TDNNs of the form proposed by Waibel *et al* (1989a) are used to form expert modules to recognize closing diphthongs realized with a New Zealand accent. The next section presents a brief overview of multi-layer feed-forward networks, including details of their training. This is followed in §4.2 by a discussion of TDNNs for phoneme recognition, including a discussion of those used in this work. Finally, §4.3 discusses the use of network ensembles, or *squads*, to improve the performance of expert modules for phoneme class recognition.

4.1 Multi-Layer Feed-Forward ANNs

As depicted in Figure 4.1-1, a multi-layer feed-forward ANN consists of an input layer, one or more hidden layers and an output layer of simple processing units referred to as *nodes* (Haykin 1994; note this author uses the term *neuron* instead of *node*). These nodes are intended to simulate (crudely) the processing action of biological neurons and are interconnected by a series of weighted connections intended to simulate (again crudely) the axons, dendrites and synapses that link biological neurons. In feed-forward networks, all weighted connections are directed "forwards" from the input layer towards the output layer, implying nodes nearer the output may not influence those nearer the input. Traditionally, as with TDNNs, these connections only join pairs of nodes in adjacent layers and may or may not link all possible pairings, implying two layers may be *partially* or *fully connected* (see Figure 4.1-1).

For all the ANNs discussed in this thesis, the *output* (or *activation*), o_j, of an arbitrary node, N_j, *in the hidden or output layers* is related sigmoidally to its input, v_j, by

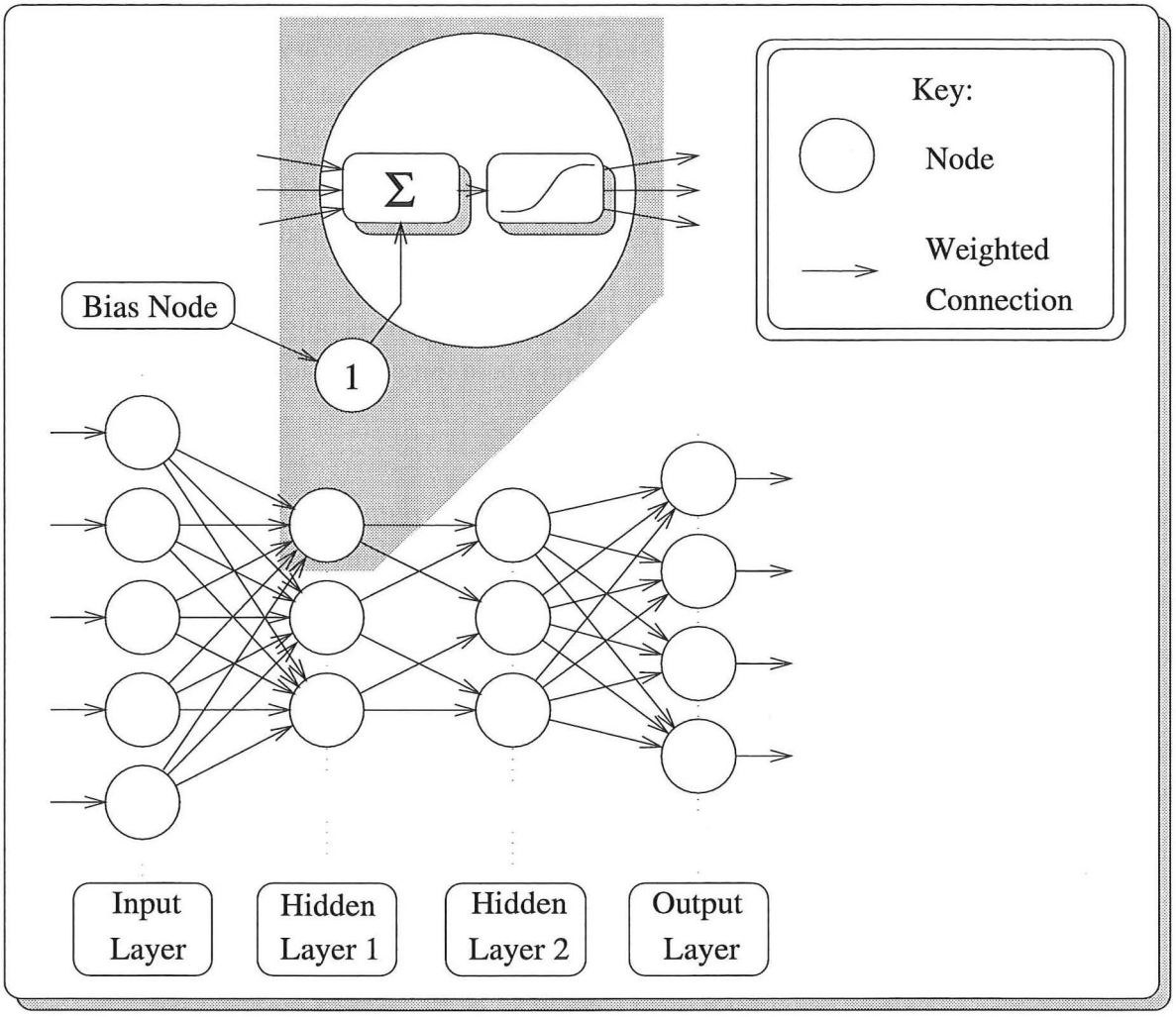

Figure 4.1-1. A stylized multi-layer feed-forward artificial neural network. This example has an input layer, two hidden layers and an output layer of *nodes* joined by *weighted connections*. Each node produces an output by first summing its inputs, including an input from a fully active *bias node*, and then performing a transformation on this sum (a sigmoidal transformation in this figure). In an ANN like that shown, adjacent layers may be *fully connected* (like the input and first hidden layers), implying all possible node pairs are joined by weighted connections, or *partially connected* (like the first and second hidden layers), implying some node pairs are not connected.

$$o_j = \frac{1}{1+e^{-v_j}} \tag{4.1-1}$$

where v_j is given by

$$v_j = \sum_{m=0}^{M-1} w_{ij} o_i(m) + w_b o_b \,, \tag{4.1-2}$$

In this expression, $o_i(m)$ is the output of one of the M previous layer nodes connected to node N_j and w_{ij} is the weight associated with this connection. Similarly, w_b is the weight associated

with the weighted connection joining node N_j to its network's bias node (note o_b=1 permanently). From equation (4.2-1), the output of node N_j may vary between 0 (*inactive*, implying v_j=-∞) and 1 (*fully active*, implying v_j=+∞). In contrast to the other layers, the output of an arbitrary node, N_k, in the input layer is given by the k^{th} element of a token. Consequently, when tokens like those proposed by Waibel *et al* (1989a) are used (see Figure 3.2-2 (d) for an example of such a token), the activation of each input layer node may vary between ±1 .

For an ANN like that in Figure 4.1-1, the activations of its output nodes are functions of its weights and the input applied to it. This may be expressed mathematically using

$$\boldsymbol{o}=\boldsymbol{F}(\boldsymbol{x},\boldsymbol{w}), \tag{4.1-3}$$

where $\boldsymbol{o}$ is a vector containing the activations of an ANN's *output layer nodes*, $\boldsymbol{w}$ is a vector containing all the weights associated with its weighted connections and $\boldsymbol{x}$ is a token (a matrix in this work, see Figure 3.2-2 (d)) (Haykin 1994). Such an ANN may be *trained* to respond usefully to tokens by altering the values of its weights. Following Waibel *et al* (1989a), the TDNNs used in this work (see §4.2) are training using the *back-propagation algorithm*. This *supervised learning algorithm* incrementally alters an ANN's weights with the intention of reducing the differences between its responses and those desired for a set of *training tokens*.

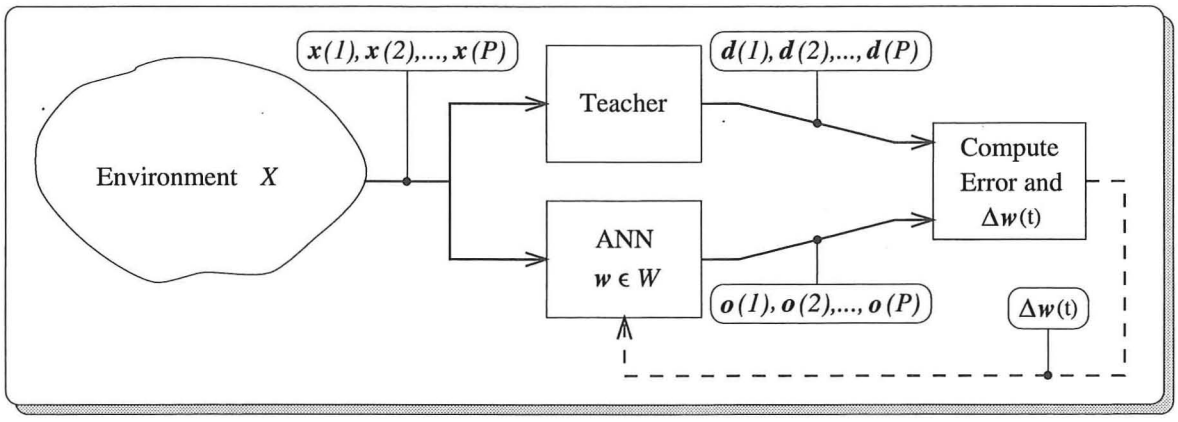

Figure 4.1-2. A model of supervised ANN training. A sample of tokens, $\boldsymbol{x}(p)$ p=1,2,...,P, is derived from an environment of interest (environment X) and processed by both a *teacher* and an ANN. From the responses of these two entities, an error representing the differences between them is evaluated and used as a bases for adapting the free parameters of the ANN (in this figure signified by the set of weights $\boldsymbol{w}$, representing one point in a multi-dimensional weight space W) so that its responses more closely resemble those of the teacher. Based on Figure 2.13 in Haykin (1994).

Figure 4.1-2 depicts a model of supervised learning. A set of training tokens $\boldsymbol{x}(p)$, p=1,2,...,P, is obtained from an environment of interest (environment X) and processed by a teacher who produces a set of desired responses, $\boldsymbol{d}(p)$ p=1,2,...,P. The same tokens are also processed by a trainee ANN which produces a set of responses $\boldsymbol{o}(p)$, p=1,2,...,P. These

responses are compared with those produced by the teacher and if not "sufficiently similar" are used to determine a vector of *weight changes* $\Delta w(t)$. A trainee ANN's weights are then updated using

$$w(t+1) = w(t) + \Delta w(t) \tag{4.1-4}$$

where $w(t+1)$ and $w(t)$ are its new and existing weights, respectively (Haykin 1994).

Initially, a trainee ANN's weights are set to small random values. These initial weights, denoted $w(0)$, are then altered using small weight changes in conjunction with equation (4.1-4) until training is terminated. Ideally, at the completion of training an ANN's responses closely resemble those of its teacher *for all tokens that may be obtained from environment X*. A network that approaches this ideal is said to have *generalized* well from the information present in its training tokens. For convenience, the final weights obtained during ANN training are denoted w^* in this thesis.

When using the back-propagation algorithm for ANN training, the i^{th} element of $\Delta w(t)$ may be evaluated using

$$\Delta w_i(t) = -\eta \frac{\partial \mathscr{E}_{av}}{\partial w_i(t)}, \tag{4.1-5}$$

where η is the *learning rate* used to control the magnitude of weight changes, $w_i(t)$ is the i^{th} element of w and $\mathscr{E}_{av}$ is a measure of the *average* difference, or *error*, between a trainee ANN's responses and those desired when processing $x(p)$, $p=1,2,...,P$. This method of evaluating $\Delta w(t)$ corresponds to the *batch mode* of back-propagation learning and is used exclusively in this work (alternative modes of back-propagation learning are discussed by Haykin 1994; Haffner 1989). In batch mode, a trainee ANN's weights are only updated using equation (4.1-4) once *all* the training tokens (one *epoch* of tokens, Haykin 1994) have been processed in conjunction with the current weights $w(t)$.

The average error, $\mathscr{E}_{av}$, may be evaluated using

$$\mathscr{E}_{av} = \frac{1}{P}\sum_{p=1}^{P} \mathscr{E}(p) \tag{4.1-6}$$

where $\mathscr{E}(p)$ is the error between a trainee ANN's response to the token $x(p)$ and that desired. In this work, $\mathscr{E}(p)$ is evaluated using McClelland's error measure to accelerate training (see Haffner *et al* 1989). Assuming a trainee ANN has M output nodes, $\mathscr{E}(p)$ is given by

$$\mathscr{E}(p) = -\sum_{m=0}^{M-1} \ln\left[1 - \left(e_m(p)\right)^2\right] \tag{4.1-7}$$

where

$$e_m(p) = d_m(p) - o_m(p) \tag{4.1-8}$$

and $d_m(p)$ and $o_m(p)$ are the m^{th} elements of $\boldsymbol{d}(p)$ and $\boldsymbol{o}(p)$, respectively. As well as being used in the evaluation of weight changes, $\mathscr{E}_{av}$ is also used in this work to terminate back-propagation learning. In particular, ANN training is terminated when $\mathscr{E}_{av}$ falls below a "sufficiently small threshold" (examples of these thresholds are given in chapter 5).

Back-propagation learning is complicated mainly by the need to evaluate equation (4.1-5) for each weight in an ANN. Haykin (1994) discusses the evaluation of equation (4.1-5) for the weights of a *fully connected* multi-layer feed-forward ANN. Appendix 2 discusses the more complicated problem of evaluating this equation for each *unique weight* in a TDNN. As discussed in §4.2, TDNNs are only *partially connected* and share common weights between several weighted connection *replicas*.

Back-propagation learning may be used to solve a variety of training problems including function approximation, prediction and pattern recognition (Haykin 1994). In this work, it is used to train TDNNs for pattern recognition. An ANN may be used to recognize tokens (patterns) representing the realizations of several objects by assigning one *object index* (see Figure 1.1-2) to each of its output nodes. During training, such an ANN is taught (ideally) to respond to each training token with *one* output node highly active ($o \approx 0.9$) and the remaining output nodes nearly inactive ($o \approx 0.1$). For each training token, the output node made active is that associated with the object whose realization it *predominantly* represents. During the operation of an ANN trained in this manner, the *most-active rule* may be used to determine the object index it selects when processing each token (pattern) presented. This rule may be stated formally as

> The object index selected by an ANN as best signifying the identity of a token it is processing is that associated with its most active output node.

In this thesis, one object index is assigned to each output node of all the TDNNs discussed (see §4.2). In addition, all the performance results presented in chapter 5 were obtained in conjunction with the most-active rule.

Unfortunately, ANN training using the back-propagation algorithm can be slow since the *learning rate*, η (see equation (4.1-5)), is identical for all weights and remains constant throughout training (Haykin 1994). The next section discusses a variant of the back-propagation algorithm in which these conditions are relaxed. This variant was used to train all the TDNNs discussed in this thesis.

4.1.1 The Delta-Bar-Delta Learning Rule

The delta-bar-delta learning rule is based upon several heuristics intended to accelerate the convergence of the traditional back-propagation algorithm (Jacobs 1988; Haykin 1994). These heuristics suggest that each *free parameter* of an ANN (its weights in this thesis) should have its own learning rate and that these rates should be allowed to vary during training. Further, they suggest that when $\Delta w_i(t)$ continually exhibits the same sign, its associated learning rate should be increased and when $\Delta w_i(t)$ alternates in sign, its learning rate should be decreased.

The delta-bar-delta learning rule may be expressed mathematically as follows. In place of equation (4.1-5), the weight change for each weight, w_i, is given by

$$\Delta w_i(t) = -\eta_i \frac{\partial \mathscr{E}_{av}(t)}{\partial w_i(t)} \tag{4.1.1-1}$$

where η_i is now *unique*. When the weights are updated at the end of each epoch (batch mode weight update), the learning rates are also modified using

$$\eta_i(t+1) = \eta_i(t) + \Delta\eta_i(t) \tag{4.1.1-2}$$

where $\eta_i(t+1)$ and $\eta_i(t)$ are the new and current learning rates, respectively, and the rate change $\Delta\eta_i(t)$ is given by

$$\Delta\eta_i(t) = \begin{cases} +\kappa & \text{if } \overline{\rho}_i(t-1)\rho_i(t) > 0 \\ -\phi\,\eta_i(t) & \text{if } \overline{\rho}_i(t-1)\rho_i(t) < 0 \\ 0 & \textit{otherwise} \end{cases} \tag{4.1.1-3}$$

In this expression $\rho_i(t)$ is given by

$$\rho_i(t) = \frac{\partial \mathscr{E}_{av}(t)}{\partial w_i(t)}, \tag{4.1.1-4}$$

and contains the current *sign* of the weight change $\Delta w_i(t)$, $\overline{\rho}_i(t)$ is given by

$$\overline{\rho}_i(t) = (1-\theta)\rho_i(t) + \theta\overline{\rho}_i(t-1) \tag{4.1.1-5}$$

and contains an exponential average of the current and previous weight changes (and their signs) and κ, ϕ and θ are training constants.

In this work, the training constants κ, ϕ and θ were set to 1.5, 0.1 and 0.5, respectively, to train all the TDNNs created. In addition, the individual learning rates were

not permitted to fall below η_{min}=0.001, the value to which each was initialized at the commencement of training. These values were selected during cursory training trials with *basic-token TDNNs* (see §4.2.3.1) and found to be satisfactory for all the TDNNs created in this work.

Haykin (1994) compares the effectiveness of the traditional back-propagation algorithm for ANN training with the delta-bar-delta learning rule. His example demonstrates that the latter may be significantly faster and more successful at reducing average ANN error ($\mathscr{E}_{av}$) than the former, provided it is used in conjunction with *batch mode* weight updating.

4.2 Time-Delay Neural Networks

TDNNs of the form used in this thesis were first proposed in Waibel *et al* (1989a), though their basis lies in the ANNs reported by Lang and Hinton (1988). The TDNN architecture was originally conceived for phoneme recognition to overcome certain problems associated with *fully connected* multi-layer ANNs. Such networks usually require accurate pre-segmentation of speech signals (accurate selection of speech portions for token generation) *during training and normal operation* (Lippmann 1989), and have many free parameters implying large quantities of training data may be required (Lang and Hinton 1988).

Simple multi-layer feed-forward ANNs, like that in Figure 4.1-1, process their inputs in parallel to produce an output. Such networks have no "memory" of earlier inputs and are, therefore, incapable of processing object realizations whose characteristic features are displaced in time. A simple method of overcoming this problem is to store information over time for simultaneous presentation to such networks (Lippmann 1989). Unfortunately, however, this approach requires accurate pre-segmentation of temporal events (Lippmann 1989), or training with sufficient data to accommodate the variation caused by poor pre-segmentation (Lang and Hinton 1988). Regrettably, both these approaches entail an undesirable increase in computational effort.

The TDNN architecture described in this section avoids the need for accurate pre-segmentation of speech signals for phoneme recognition by incorporating *shift invariant feature detectors* in its structure. Such detectors are a consequence of two structural modifications to the architecture of a traditional fully-connected multi-layer ANN. First, the number of weighted connections (degree of *connectivity*) is dramatically reduced so that the nodes in each hidden layer of a TDNN only have a "localized view" of the nodes in their preceding layer. Second, these nodes (and the weights associated with the connections feeding them) are replicated to reduce a TDNN's dependence on the positioning of features within the tokens it processes (this approach to *shift-invariance* is also used by Fukushima 1980). Both modifications have the effect of reducing the number of free network parameters (*unique*

weights), permitting network training in conjunction with smaller data sets (Lang and Hinton 1988). The second modification also improves the tolerance of TDNNs to inaccurate pre-segmentation of speech signals during training (Waibel 1992a) and during normal operation (Waibel 1989a), compared to similar fully connected ANNs (Grayden and Scordilis 1992).

As proposed by Waibel *et al* (1989a), a TDNN consists of an output layer, two hidden layers and an input layer. The output and second hidden layers both contain one node corresponding to each object to be recognized. The number of nodes in the input layer is equivalent to the number of coefficients produced by the Waibel transform (16 in this work, see §3.2). Figure 4.2-1 shows the two ways in which TDNNs are traditionally depicted (see Waibel *et al* 1989a). In the first depiction (Figure 4.2-1 (a)), each node in the top three layers is connected to one or more nodes in a preceding layer via *multiple* weighted connections incorporating different amounts of delay (an example of this connectivity for one node in each of these layers is shown). These nodes are also connected to a bias node via individual weighted connections incorporating no delay. For example, each output node is connected to *one* node in the second hidden layer via *nine* weighted connections, each having a different delay (an integer multiple of τ). By contrast, each node the second hidden layer connects to each of the eight nodes in the first hidden layer via *five* weighted connections, each having a different delay (again, an integer multiple of τ). Connections between the input and first hidden layers are similar to those between the first and second hidden layers.

For convenience in Figure 4.2-1 (a), the weighted connections feeding higher level nodes are depicted originating from the outputs of *tapped delay-lines*, where the output of each tap is delayed by an integer multiple of a fixed delay τ (τ is equivalent to the mel-scale spectrogram slice rate of 10 msec; see 3.2). Thus, for example, each node in the second hidden layer is fed via a *unique* set of 40 weighted connections (with unique weight values) connected to the *same* tapped delay-line outputs.

Figure 4.2-1 (b) shows the second way in which TDNNs are traditionally depicted (Waibel *et al* 1989a). In this depiction, the network is *unfolded* in time and represented by a *partially connected* multi-layer feed-forward ANN, with the input and hidden layers represented by matrices of nodes (depicted using Hinton diagrams; see Hinton and Sejnowski 1986). Within these matrices, the elements of a column correspond to the nodes shown in part (a) and the rows contain *time replicas* of these nodes representing their activations at different time instances (equivalent to the taps of the delay lines shown in part (a)). Consequently, for the TDNN depicted, the layers from the input layer to the output layer contain 16, 8, 5 and 5 *unique nodes* and 15, 13, 9 and 1 time replicas, respectively.

Naturally the weighted connections in Figure 4.2-1 (a) and (b) are the same, though they are depicted somewhat differently. In contrast to the former, the latter shows the weighted connections with *unique weights* at *one* time instant. For comparison, Figure 4.2-2 shows the *same* weighted connections at two other time instances. The connections in Figure

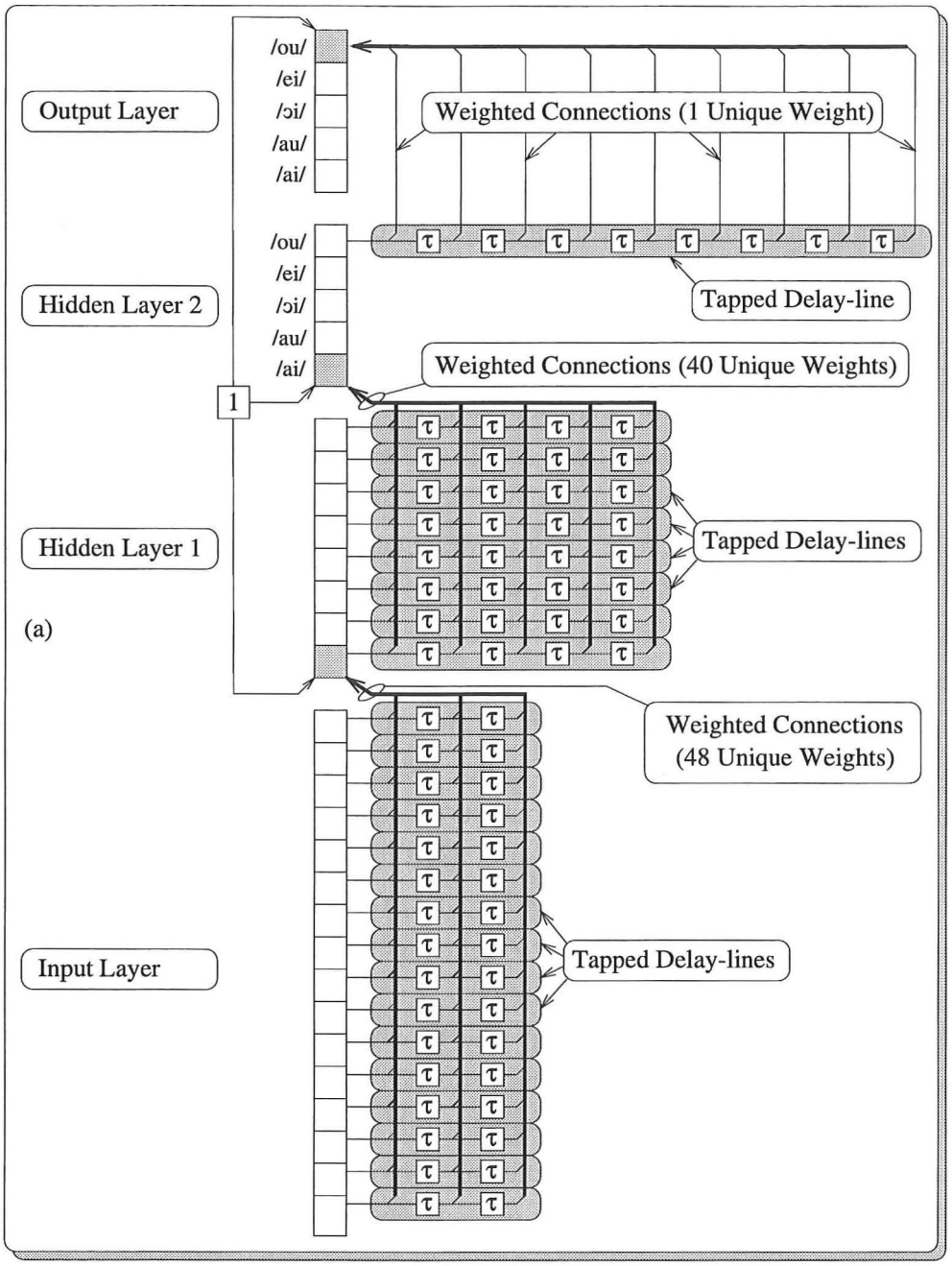

Figure 4.2-1. The two traditional methods for depicting a TDNN (in this case, a *basic-token TDNN* for processing closing diphthong realizations, see §4.2.3.1). Part (a) shows the nodes present in each layer of a TDNN and its bias node (these nodes are represented by squares for convenience). The weighted connections associated with the shaded nodes in the top three layers demonstrate how the nodes in these layers are connected to those in their preceding layers. As well as having access to the instantaneous outputs of nodes in their preceding layers, the nodes in the top three layers also derive input from earlier outputs, delayed by a multiple of the interval τ. For convenience, these delayed outputs are viewed as originating from *tapped delay lines*, as shown.

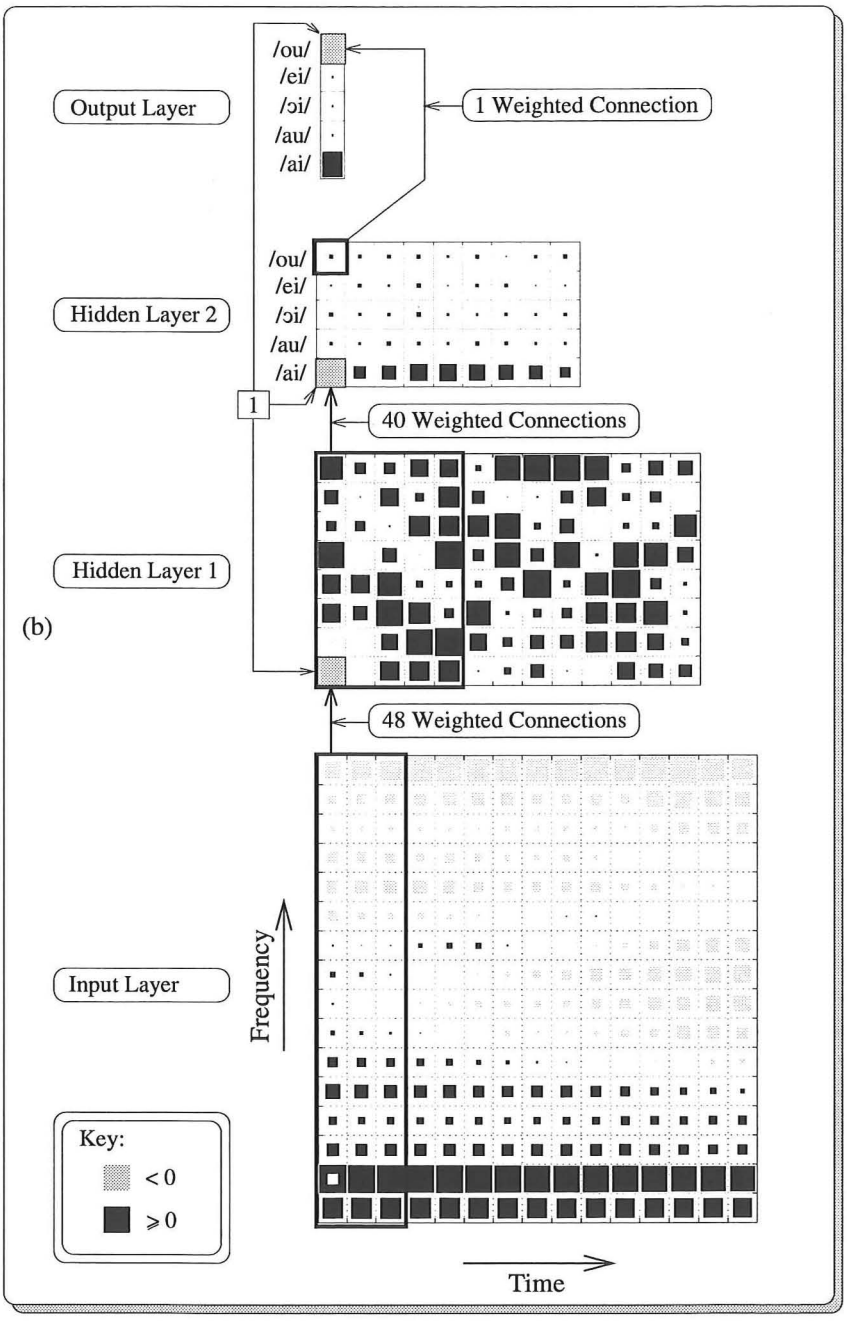

Figure 4.2-1*(cont)*. Part (b) shows the TDNN depicted in part (a) *unfolded in time* to give a partially connected multi-layer feed-forward network. The nodes in each *row* of the lower three layers are *time replicas* of one another and show the activations of the nodes in part (a) *over time*. The nodes in each *column* of the top three layers are joined to a "window" of nodes in their preceding layers by weighted connections with *unique weights* (examples of these connections are indicated for the same shaded nodes in the top three layers as in part (a)).

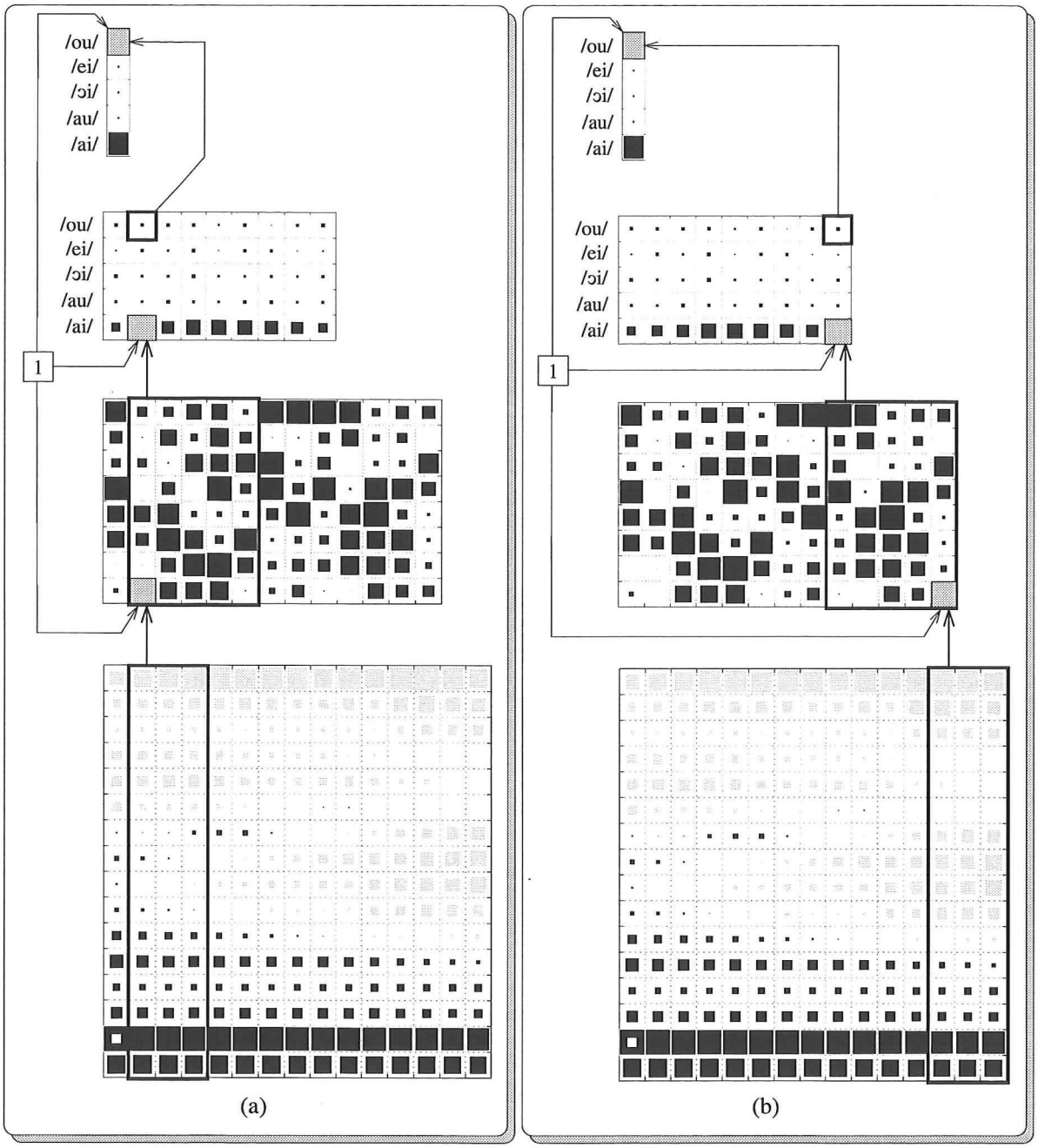

Figure 4.2-2. The same weighted connections as shown in Figure 4.2-1 (b) except offset by (a) *one* time-step (τ) and (b) by the maximum time-step possible while still influencing the same TDNN response.

4.2-2 (a) are offset by *one* time-step (τ) from those shown in Figure 4.2-1 (b), while those in Figure 4.2-2 (b) are offset by the largest time-step possible while still influencing the *same* TDNN response.

From the unfolded depiction of a TDNN in Figure 4.2-1(b), the structural modifications to a traditional fully-connected ANN discussed earlier become clear. In Figures 4.2-1 (b) and 4.2-2, each node in the first hidden layer receives input from connections to a

"window" of 48 nodes in the input layer (16 unique nodes by 3 time replicas) and a bias node. For each *unique* node in the first hidden layer (each node in a column), these connections have unique weights which are *shared* by the replicated connections feeding all its time replicas (see Figure 4.2-2).

The second hidden layer is similarly connected to the first with the exception that each *unique* node in this layer (each node in a column) is joined to a window of 40 nodes (8 unique nodes by 5 time replicas) in the first hidden layer by 40 weighted connections with unique weights. Once again, replicas of each unique node are fed by connection replicas sharing the same set of weights. As pointed out in Waibel *et al* (1989a);

> The choice of a larger 5 frame window in this layer was motivated by the intuition that higher level units should learn to make decisions over a wider range in time based on more local abstractions at lower levels.

This intuition is extended further in the output layer. Each output layer node is joined to a *single row* of replicas in the second hidden layer by weighted connections all sharing the same weight. Each output layer node is intended to integrate the activations of its row of counterparts in the second hidden layer for a fixed time interval.

Like other multi-layer feed-forward ANNs (see §4.1), TDNNs may be trained using the back-propagation algorithm (Waibel *et al* 1989a). In this work, McClelland's error measure (see equation (4.1-7)) and the delta-bar-delta learning rule (see §4.1.1) were used to accelerate TDNN training, as discussed in §4.1. Appendix 2 presents a derivation of the expressions for the weight changes required to update the *unique weights* of a TDNN using equation (4.1-4). This derivation assumes batch mode weight update and is complicated primarily by the partial and non-uniform pattern of connections between a TDNN's layers. Expressions for the weight changes, $\Delta w(t)$, required during TDNN training are given by equations (A2.1-12), (A2.1-13), (A2.2-10), (A2.2-11), (A2.3-12) and (A2.3-13) (note that (t) is dropped from the notation used to denote these changes in appendix 2 for simplicity).

The TDNNs used in the experiments discussed in chapter 5 (see §4.2.3), maintain the fundamental TDNN architecture proposed by Waibel *et al* (1989a). This architecture has been applied successfully to Japanese phoneme recognition (see §4.2.1) and it is, therefore, interesting to discover how it performs on New Zealand English before engaging in significant architectural modifications.

The next section discusses the major experimental results concerning phoneme recognition using TDNNs reported by other authors. This is followed in §4.2.2 by a discussion of *modular TDNNs*, the state-of-the-art phoneme recognition systems for which the expert modules discussed in this work are intended. Finally, the architectures of the three TDNNs used to form these expert modules are discussed in §4.2.3.

4.2.1 Major Experimental Results with TDNNs

The first experiments with TDNNs of the form used in this thesis were reported by Waibel *et al* (1987) (subsequently published as Waibel *et al* 1989a) and concerned the recognition of the Japanese voiced plosives /b/, /d/, and /g/, using a TDNN resembling that in Figure 4.2-1. Realizations of these phonemes were obtained from common Japanese words uttered in *isolation* by three Japanese male speakers (all professional announcers). For these three speaker's, TDNNs were trained *speaker-dependently* and gave "recognition rates" of 98.8%, 99.1% and 97.5% for *aligned test tokens.*[2] Waibel *et al* (1989a) found these rates were "considerably higher" than those for similarly trained HMMs, which achieved only 92.9%, 97.2% and 90.9% for the same three speakers and test conditions.

Having shown TDNNs could perform well for Japanese voiced plosives, Waibel and his colleagues then demonstrated the same approach could be applied successfully to six other classes of Japanese phonemes (Waibel *et al* 1989b; these classes appear in Figure 4.2.1-1). Following this success, they then turned to the wider problem of devising a TDNN architecture to recognize *all* the phonemes of Japanese *simultaneously*. As a step towards finding such an architecture, Waibel and his colleagues first attempted to enlarge their TDNN for voiced plosive recognition to recognize both the voiced and unvoiced plosives. Unfortunately this network (a TDNN resembling that in Figure 4.2-1 (b) with six output nodes) proved extremely difficult to train, leading Waibel and his colleagues to conclude that a single *monolithic TDNN* was impractical for recognizing all Japanese phonemes (Waibel *et al* 1989b). Consequently, they next sort a method of combining the individual TDNNs they had successfully trained for each phoneme class.

An initial attempt to combine the individual TDNNs for voiced and unvoiced plosive recognition using the most-active rule (see §4.1), succeeded only in reducing their individual classification performances from 98.3% and 98.7%, respectively, to 60.5% when combined. It was found, as in this thesis, that *individual TDNNs* trained for one phoneme class may respond highly actively to realizations of phonemes from other classes. Consequently, it was concluded that to combine the voiced plosive TDNN with the unvoiced plosive TDNN, the two networks would need to be fused together as one network and *some* portions retrained. Several different schemes for combining and partially retraining these networks were devised. The most successful of these schemes achieved a (speaker-dependent) classification performance of 98.6% for test tokens representing both voiced and unvoiced plosive realizations (Waibel *et al* 1989b).

---

[2]Aligned test tokens are generated from carefully selected speech portions of known identity. They differ from training tokens (see §3.3) only in that they are not experienced by an ANN during training. Recognition rates obtained using aligned test tokens are referred to as *classification performances* in this thesis (see §5.1.1).

Using the principles learned from combining their individual TDNNs for voiced and unvoiced plosive recognition, Waibel and his colleagues continued to combine individual TDNNs and eventually produced a *modular network* capable of recognizing realizations of all 23 Japanese phonemes and *q* for "silence" (this network was first discussed briefly by Waibel *et al* 1989c and then in detail by Miyatake *et al* 1990; Minami *et al* 1991; Sawai 1991a). Figure 4.2.1-1 depicts their *modular TDNN* which is entitled *LP-TDNN*, short for *large phonemic time-delay neural network*. Within this network, seven *expert modules* provide *intra-class* discrimination of phoneme realizations corresponding to seven phoneme classes (six consonant classes and one vowel class), one module (*c-class*) provides *inter-class* discrimination for the consonant classes and one module (*speech-silence*) detects the presence or absence of speech. All of these expert modules share the same input layer (and tokens) and consist of two further *independent* hidden layers. For each expert module, these layers resemble the first three layers of the TDNN depicted in Figure 4.2-1 (b) and are connected in a similar fashion. None of the expert modules in Figure 4.2.1-1 contain the integrating output layer of the TDNN depicted in Figure 4.2-1 (b), since this function is transferred to the *arbitration module* in LP-TDNN.

The arbitration module depicted in Figure 4.2.1-1 is tasked with combining the activations of the various expert modules as is discussed in the next section. This module consists of two layers of nodes (and the weights feeding them) resembling the top two layers of the TDNN depicted in Figure 4.2-1 (each output node integrates the activations of a row of nodes in the final hidden layer). The arbitration module is apparently densely connected to the nine expert modules, as depicted in Figure 4.2.1-1, though the specific details of this connectivity have *not* been made public. The next section discusses some general concepts concerning the form and function of modular TDNNs for phoneme recognition.

Miyatake *et al* (1990) observed that the practice of training expert modules for each phoneme class first and then combining and retraining these to form an LP-TDNN, can cause the combined network to produce large numbers of false-positive errors (such errors are defined in §2.5.2). Consequently, they trained their LP-TDNNs as single integrated networks, with the hope that false-positive errors would be reduced by "lateral inhibition" (presumably acting between the expert modules). The first LP-TDNN created by Miyatake *et al* (1990) was trained using tokens generated from speech portions "centred" on the phoneme realizations produced by one Japanese male speaker. On isolated-word test utterances containing 13 974 phonemes realizations, this network correctly recognized 95.8% of these realizations, but produced 8 698 false-positive errors.

In an attempt to improve upon the performance of their first LP-TDNN, Miyatake *et al* (1990) trained a second LP-TDNN with tokens aligned and *misaligned* with the centres of phoneme realizations. This was done to better constrain the acoustic-phonemic relationships learned by the second LP-TDNN by labelling and presenting more of each phoneme's

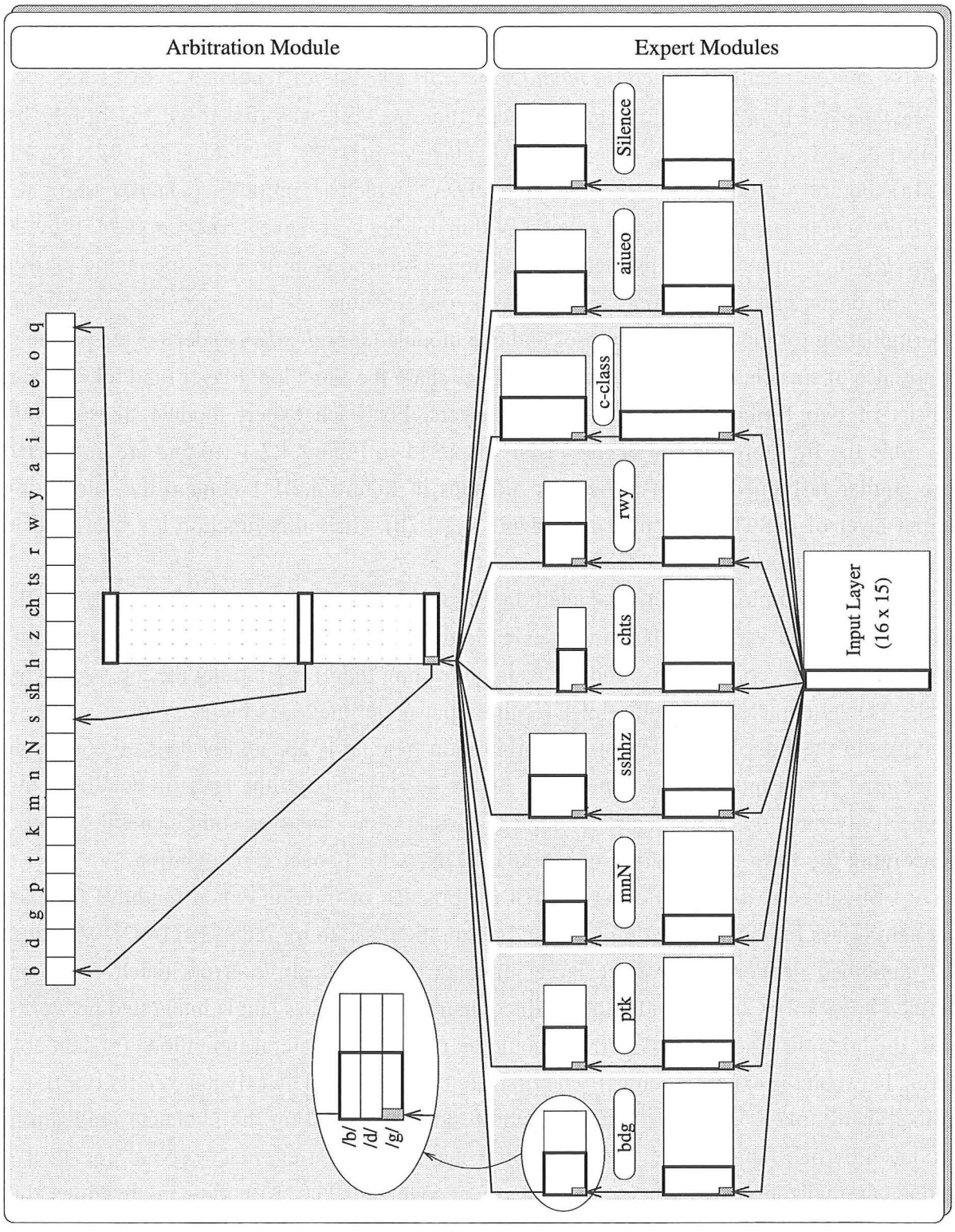

Figure 4.2.1-1. The architecture of LP-TDNN used for Japanese phoneme recognition. This network comprises seven *expert modules* providing *intra-class* discrimination of phoneme realizations corresponding to seven phoneme classes, one expert module (*c-class*) providing *inter-class* discrimination for the consonant classes and one module (*speech-silence*) to detect the presence or absence of speech. The outputs of the expert modules are combined by an arbitration module to produce responses signifying one of the 23 Japanese phonemes or "silence" (q). Based on Figure 1 in Miyatake *et al* 1990.

realizations. On the test utterances used previously, the second LP-TDNN correctly recognized 98% of the phoneme realizations, while producing 3 236 false-positive errors. Despite producing far fewer false-positive errors than their first LP-TDNN, Miyatake *et al* (1990)'s second LP-TDNN still made a large number of these errors, leading them to express the need for "new training methods" to improve upon this situation.

Sawai (1991a) further tested the second LP-TDNN reported by Miyatake *et al* (1990) by processing 1 940 phoneme realizations extracted from 278 *continuous-speech* Japanese phrases (all previous tests used realizations from isolated words). Under test, the recognition performance of this network fell to 81.2%, while producing 926 false-positive errors. This decrease in recognition performance was attributed to "the different co-articulatory effects between word speech and continuous speech" (Sawai 1991a). To rectify this problem, the second LP-TDNN was retrained using tokens representing phoneme realizations observed in continuous-speech phrases as well as those used previously. This led to an improved recognition performance of 89.1% in conjunction with the test set of 1 940 phoneme realizations, while only 500 false-positive errors were made.

The experimental results discussed so far have all concerned *speaker-dependent* TDNNs or LP-TDNNs trained for one of three Japanese male speakers. Sagayama *et al* (1992), discussing ATR's *ATREUS* project, compares the performances of full speaker-dependent and speaker-independent speech recognition systems incorporating LP-TDNNs for phoneme recognition.[3] These systems are found to afford recognition rates of 65% and 68%, respectively, on a Japanese *phrase recognition* task.[4] Unfortunately, however, these results are found to be worse than those for similar systems incorporating HMMs, the best of which achieve 94% and 83% for speaker-dependent and speaker-independent phrase recognition, respectively. Notably, however, the performances of systems incorporating LP-TDNNs reported by Sagayama *et al* (1992), are for networks trained using phoneme realizations from isolated words only. By contrast the best performing HMMs reported were trained using phoneme realizations from isolated-word and continuous-speech phrases. Consequently, in light of the results reported by Sawai (1991a), the comparisons reported by Sagayama *et al* (1992) are somewhat biased in favour of the systems incorporating HMMs.

Aside from the incomplete training afforded the systems incorporating LP-TDNNs compared by Sagayama *et al* (1992), their relatively poor performances compared to systems

---

[3]It is assumed that the results presented by Sagayama *et al* (1992) refer to LP-TDNNs, though this is not clearly stated in their report. The other experiments with LP-TDNNs discussed in this section were also conducted at ATR Interpreting Telephony Research Laboratories, Japan.

[4]The performance figures quoted in this section are only approximate since they are presented graphically by Sagayama *et al* (1992).

incorporating HMMs might also be attributed to the large numbers of false-positive errors made by their component LP-TDNNs. Rather than attempting to find a new training method to overcome these errors, as suggested by Miyatake *et al* (1990), this thesis proposes a new method of forming expert modules from TDNNs. Instead of using a single TDNN (or part thereof) as previous researchers have, it is proposed that expert modules be formed from ensembles of TDNNs, referred to as *squads*. This concept is discussed further in §4.3 and the performances of *traditional* and *squad-based* expert modules for closing diphthong recognition are compared in §5.1.

Apart from the experimental work already discussed in this section, that by Hataoka and Waibel (1990) is also relevant to this thesis. These authors discuss the recognition of American English vowel realizations using various *speaker-independent* TDNNs. As a consequence of preliminary experiments with vowel realizations from the TIMIT database, Hataoka and Waibel (1990) suggest a TDNN with a larger input layer (more node replicas) is necessary when attempting to recognize diphthong realizations. The best ("large sample") classification performance reported by these authors is 82.38% for a TDNN comprising an input layer with 20 node replicas, which they trained to recognize American English diphthongs. Consequently, the benefits of using a TDNN with an extended input layer to recognize closing diphthongs realized with a New Zealand accent is examined in this work (see the discussion of *extended-token TDNN* in §4.2.3.1).

A significant feature of the results discussed in this section, particularly those used to compare various recognition systems, is that none were reported in conjunction with any form of statistical significance testing. In fairness, the lack of such statistical evidence is not confined to reports concerning phoneme recognition by TDNNs. As Gillick and Cox (1989) state;

> It is common practice for researchers to test two or more [speech or phoneme recognition] algorithms together and then to make claims about their relative efficacy on the basis of the test results. However, these claims are seldom backed by any evidence that any difference in performance is statistically significant; indeed, most papers show an almost complete lack of awareness of the importance of comparing the results of experiments in a way that takes account of variability and uncertainty in a principled manner.

Consequently, in this thesis an attempt is made to analyze the significance of performance differences where possible.

4.2.2 Modular TDNNs for Automated Phoneme Recognition

As discussed in the previous section, the problem of recognizing all the phonemes of Japanese eventually led to the development of *modular TDNNs*, like LP-TDNN, since *monolithic TDNNs* proved too difficult to train. Figure 4.2.2-1 shows the basic structure and operation of a modular TDNN for phoneme recognition. A common token is fed to a set of *expert modules* which are either trained to recognize phoneme realizations corresponding to one phoneme class (*intra-class recognition*), or to recognize the classes to which phoneme

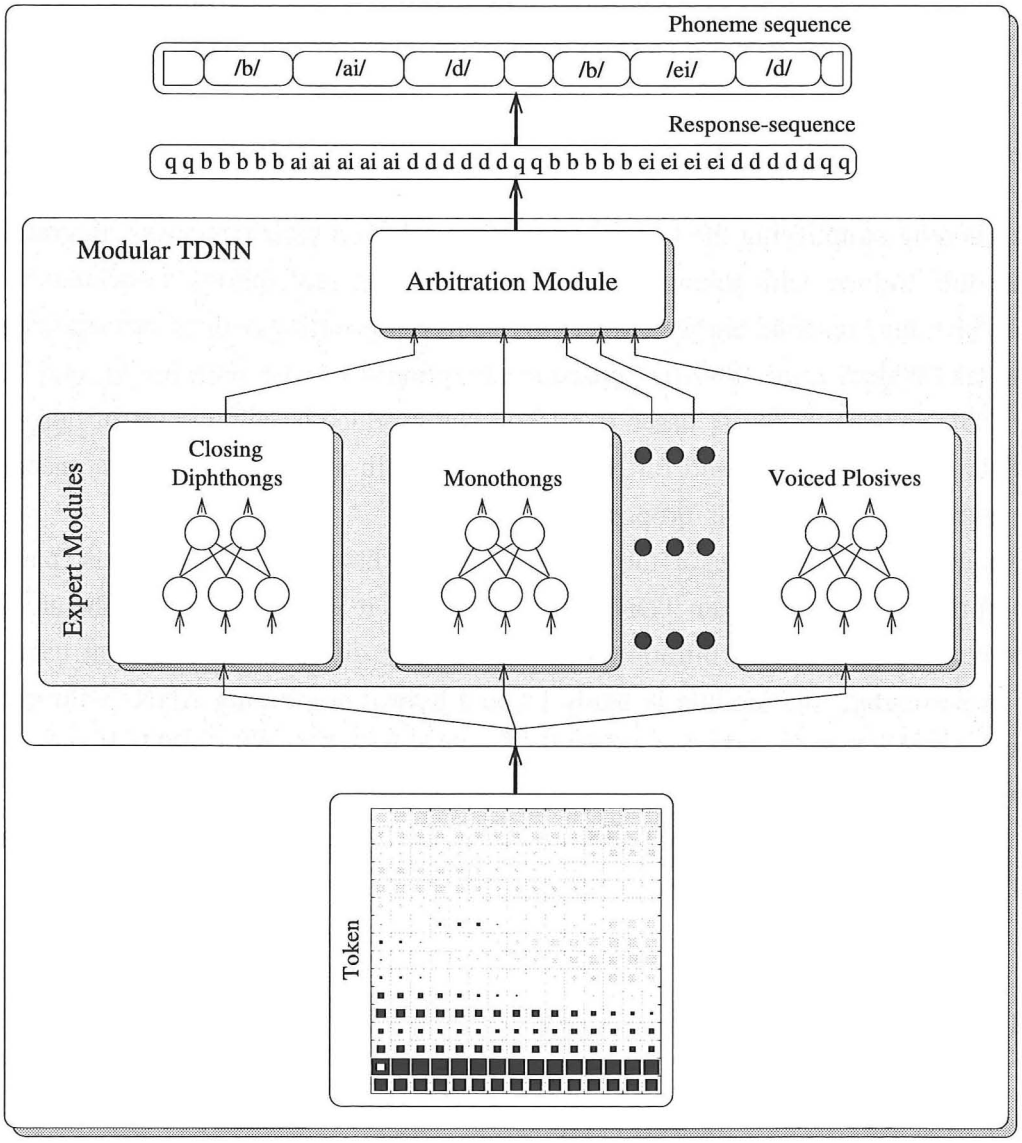

Figure 4.2.2-1. The basic structure and operation of a modular TDNN for phoneme recognition. Tokens obtained from a spectrogram like that in Figure 3.2-2 are fed to a set of expert modules whose responses are combined by an arbitration module to form a response-sequence.

realizations belong (*inter-class recognition*). The outputs of these expert modules are combined by an *arbitration module*, whose task is to decide which classification(s) to enter in a modular TDNN's response-sequence. For simplicity this sequence is depicted containing a single phonemic symbol for each token processed, though in practice it is likely to contain ranked alternatives as discussed in §2.5.2.

Within a modular TDNN, the task of recognizing all the phonemes of a language is partitioned into simpler sub-tasks associated with a set of phoneme classes. *Phoneme class partitioning* is enforced by limiting the number of output nodes associated with each expert module. It enables such modules to be trained "easily", which in turn enables a modular TDNN to be formed more readily than a monolithic TDNN. Unfortunately, however, simplifying an expert module's classification task reduces the constraints on it to respond desirably to *all possible input*. Ideally, an expert module should only respond actively to tokens representing phoneme realizations associated with its phoneme class and remain inactive for all other tokens. This behaviour is desired to reduce conflicts between expert modules, thereby simplifying the task of arbitrating between their responses. Regrettably, an expert module trained with tokens representing phoneme realizations associated with one phoneme class, may respond highly actively to tokens representing realizations associated with other classes (Waibel *et al* 1989b). Consequently, phoneme class partitioning may result in modular TDNNs (like those discussed in §4.2.1) that produce large numbers of false-positive errors. This problem is discussed further in §4.3, wherein a solution based on ensembles of ANNs, referred to as squads, is proposed.

To accommodate some of the expert modules discussed in this thesis, particularly those comprising sequence-token TDNNs (see §4.2.3.2), it is anticipated that the architecture of the module required for arbitration will take a very different form to that used in LP-TDNN. In particular, this module is likely to be a hybrid combining ANNs with traditional serial or fuzzy algorithms (see for example Kasabov and Shishkov 1993; Kasabov 1993). Following the approach used by Waibel and his colleagues (see §4.2.1), the exact form of this module is left until the properties of appropriate expert modules for recognizing all the New Zealand English phonemes have been determined.

4.2.3 Three TDNN Architectures for Closing Diphthong Recognition

As discussed in chapter 1, the experiments described in chapter 5 concern closing diphthong recognition using expert modules comprising one of three types of TDNN, referred to as *basic-*, *extended-* and *sequence-token TDNNs*. Following the success of Waibel *et al* (1989a), the *initial aim* of these experiments was to create and test expert modules comprising basic-token TDNNs (these resemble closely the TDNN proposed by Waibel *et al* 1989a; see

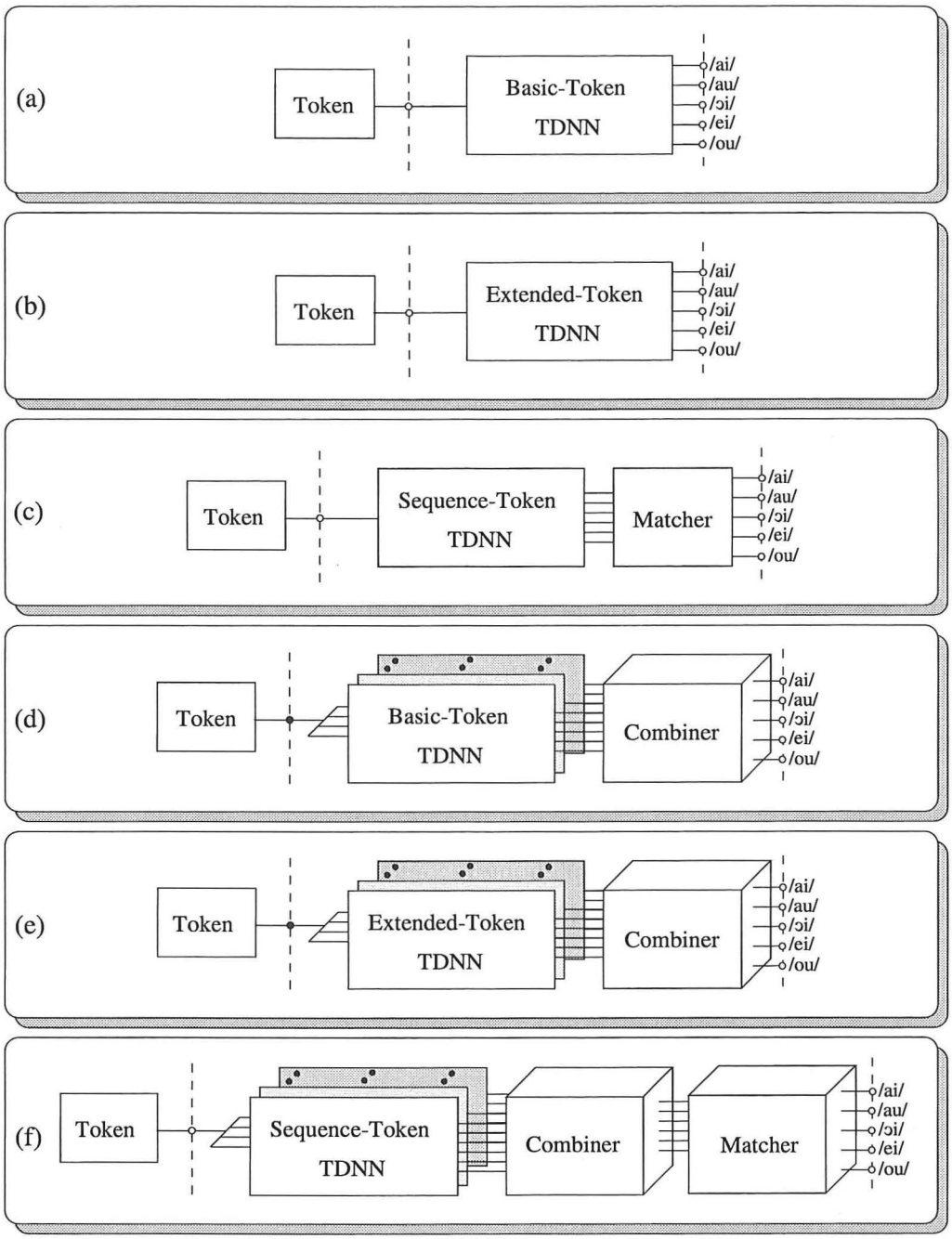

Figure 4.2.3-1. Outlines of the traditional ((a), (b) and (c)) and squad-based ((d), (e) and (f)) expert modules for closing diphthong recognition compared in this work. Traditional expert modules comprise individual TDNNs, whereas squad-based expert modules comprise ensembles of TDNNs whose responses are combined by a *combiner*. The expert modules comprising sequence-token TDNNs ((c) and (f)) also contain *matchers* to convert the output sequences produced by their component TDNNs into phonemic symbols.

Figure 4.2-1). This aim was altered, however, following Hataoka and Waibel's suggestion that TDNNs for diphthong processing should have an extended input layer (see §4.2.1). Consequently, the aim become to create and compare expert modules comprising basic-token TDNNs with similar modules comprising *extended-token TDNNs*.

During the initial experiments with expert modules comprising basic- and extended-token TDNNs, observations of the former suggested another method of using traditional TDNNs to form expert modules for closing diphthong recognition. In particular, it was observed that expert modules comprising basic-token TDNNs would (undesirably) respond with consistent *sequences* of different most active output nodes, when processing certain closing diphthong realizations (most notably those of /ai/ and /ei/; see Figure 5.1.4-1). Consequently, expert modules comprising TDNNs trained to respond *intentionally* with such sequences, referred to as *sequence-token TDNNs*, were conceived and the experiments discussed in this thesis extended to include them.

Figure 4.2.3-1 depicts the broad outlines of the expert modules for closing diphthong recognition tested in this work. These modules are of two kinds, referred to as *traditional* ((a) through (c)) and *squad-based expert modules* ((d) through (f)). The former comprise individual basic-, extended-, or sequence-token TDNNs, as is traditional (see Waibel *et al* 1989b; Miyatake *et al* 1990), whereas the latter comprise ensembles of such networks. Comparing the traditional and squad-based expert modules depicted, the latter require *combiners* to form *collective responses* from the responses of their individual component TDNNs. Compared to the modules (of both kinds) comprising basic- and extended-token TDNNs, those comprising sequence-token TDNNs require *matchers* to convert the sequences produced by their component TDNNs into phonemic symbols.

The next section discusses the basic- and extended-token TDNNs used in this work to form expert modules for closing diphthong recognition. This is followed in §4.2.3.2 by a discussion of the sequence-token TDNN used.

4.2.3.1 Basic- and Extended-token TDNNs

Figure 4.2-1 depicts the architecture of the basic-token TDNNs used to process closing diphthong realizations in this work. This architecture is identical to that proposed by Waibel *et al* (1989a) for voiced plosive recognition, with the exception that the top two layers have five rather than three (unique) nodes to accommodate the five closing diphthongs. Similarly, Figure 4.2.3.1-1 depicts the architecture of the extended-token TDNNs used to process closing diphthong realizations. This architecture differs from that in Figure 4.2-1 only in that there are more node and weight replicas in the first three layers. In particular, the input layer contains 30 node replicas corresponding to an input token of 30 slices. As discussed in §3.1.1, tokens of this length were chosen to capture (on average) the full extents of the diphthong-glides produced by the two New Zealand English speakers sampled.

For both basic- and extended-token TDNN, each output layer node (and associated second hidden layer node) is arbitrarily assigned a phonemic symbol corresponding to one of

the closing diphthongs. During training, one of these nodes is "forced" to be highly active (o=0.9 ideally) in response to each training token, while the remainder of the output nodes are "forced" to be nearly inactive (o=0.1 ideally). This process is depicted in Figure 4.2.3.1-2 for both types of TDNN in conjunction with training tokens representing a realization of /ai/.

Following the approach used by Waibel *et al* (1989a) and Hataoka and Waibel (1990), the basic- and extended-token TDNNs created in this work were trained using tokens

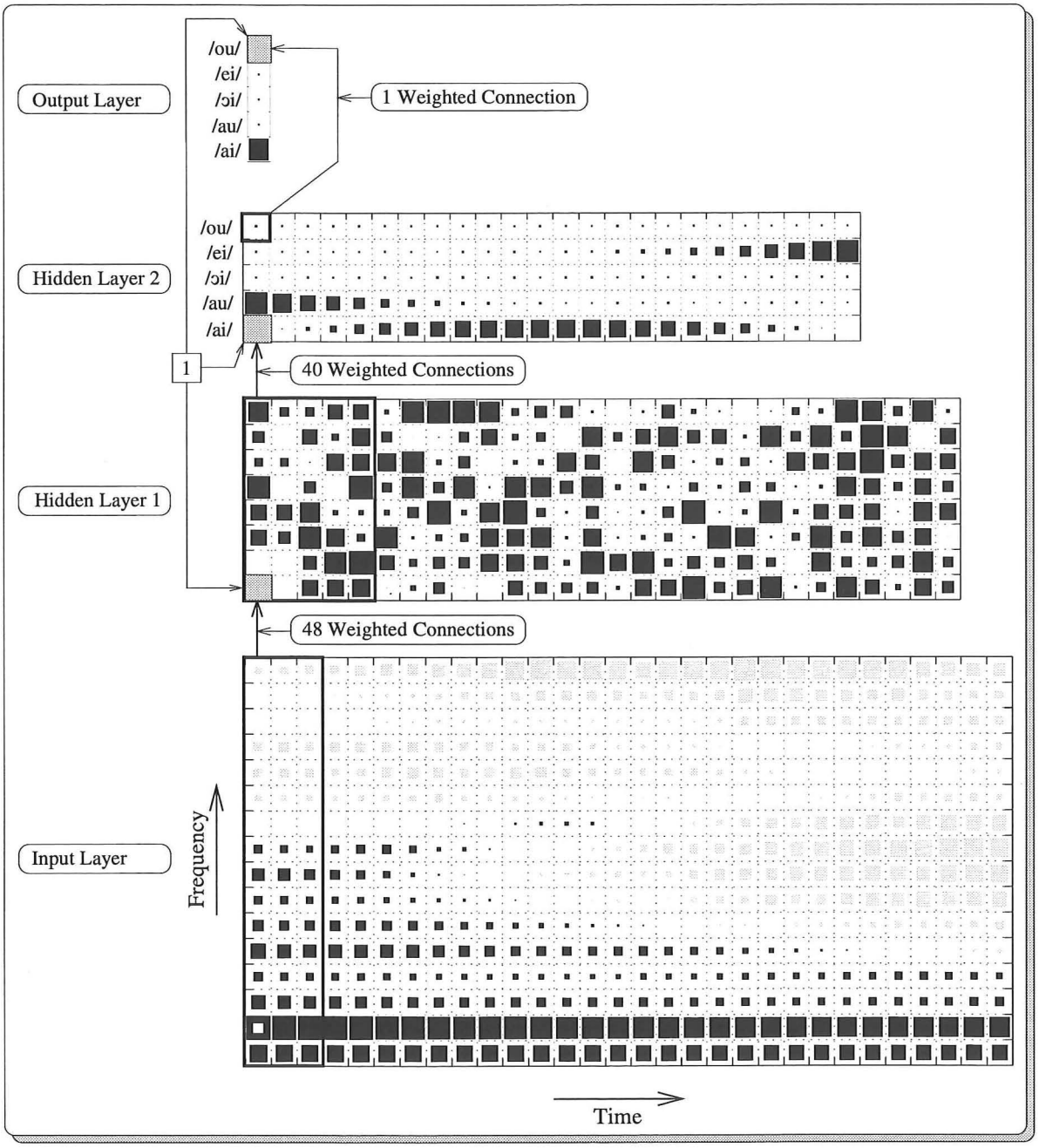

Figure 4.2.3.1-1. The architecture of the *extended-token TDNNs* used to process closing diphthong realizations in this work.

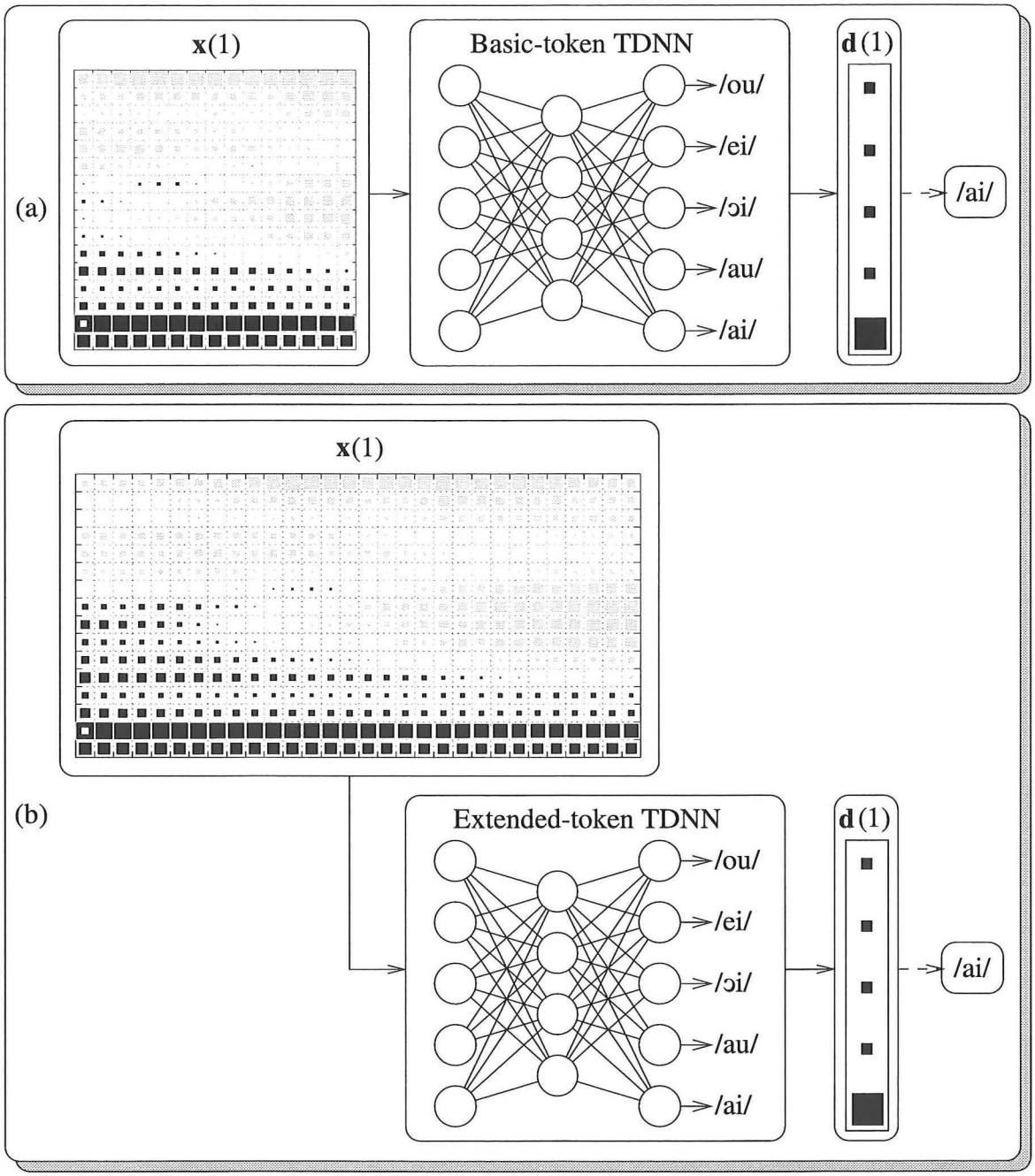

Figure 4.2.3.1-2. A training token, ***x(1)***, and the desired response, ***d(1)***, for (a) basic-token TDNN and (b) extended-token TDNN. The tokens shown both represent realizations of /ai/, hence it is desired that the output node associated with /ai/ for each TDNN depicted be highly active ($o \approx 0.9$) and the remaining output nodes nearly inactive ($o \approx 0.1$). When the output node associated with /ai/ for each TDNN is most active, this is assumed to signify that the current token being processed represents a realization of this closing diphthong.

generated from speech portions selected about the "centres" of closing diphthong realizations. For each speaker, the (F_1,F_2) coordinates depicted in Figure 4.2.3.1-3 were used in conjunction with *STEP* to locate the *frequency-domain* "centres" of the realizations corresponding to each closing diphthong (see §3.3). In this work, these "centres" are assumed to lie medially within

the common diphthong-glide region shared by a given closing diphthong's realizations (see Figure 3.1.1-1 for example). Given the frequency-domain "centres", *STEP* was then used to determine the *time-domain* "centres" of each speaker's closing diphthong realizations and to select appropriate length speech portions about these "centres". These speech portions were

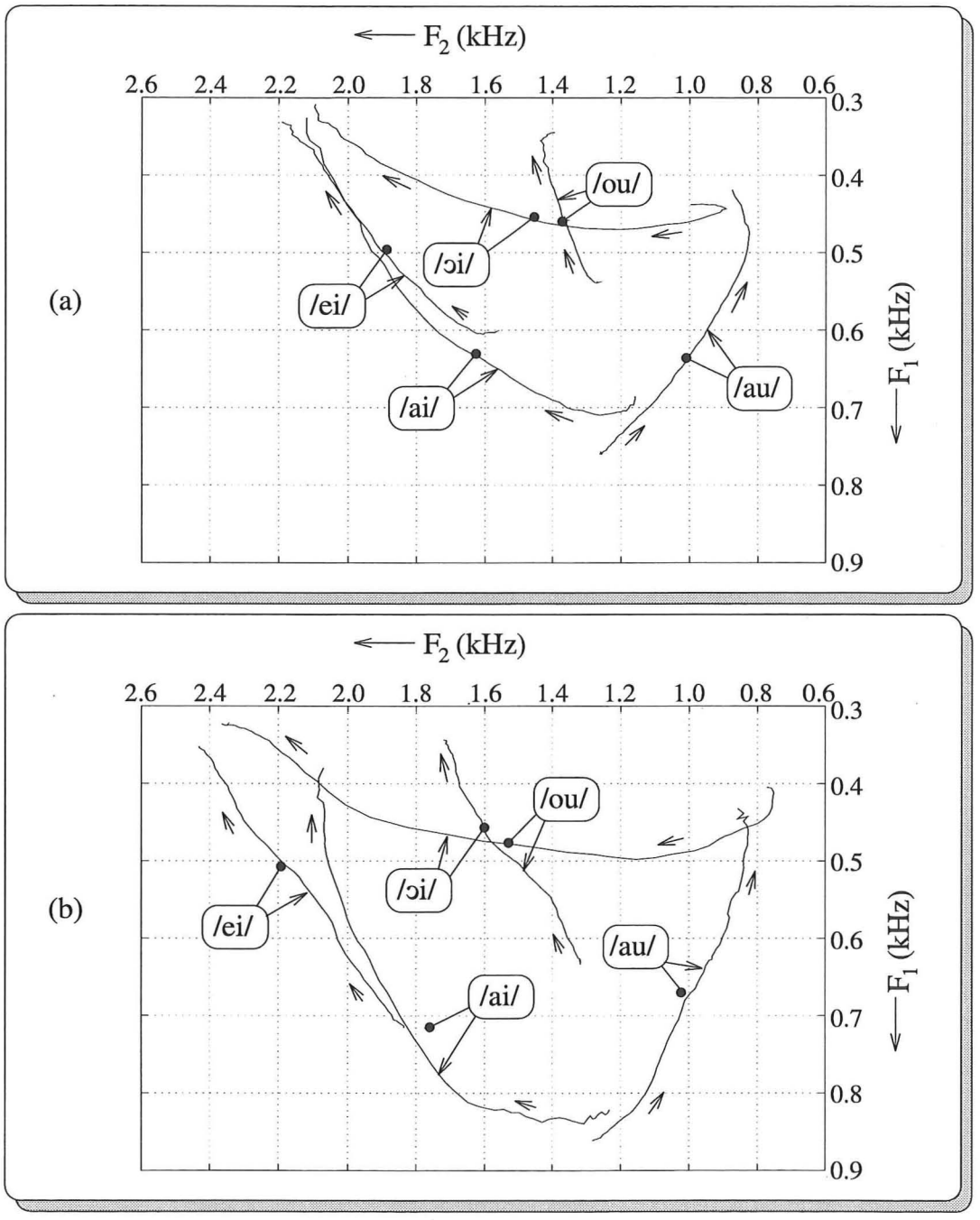

Figure 4.2.3.1-3. (F_1,F_2) coordinates (labelled black dots) used in conjunction with *STEP* to select "centred" speech portions from (a) speaker JK's and (b) speaker HD's closing diphthong realizations. For each diphthong, the F_1-F_2 trajectory shown is an average computed from the speech portions selected to suit *extended-token TDNN* (3206 speech samples; note the short arrows lying next to these trajectories indicate the directions of their transitions with time).

then used to generate the tokens discussed in §5.1.1.

As well as showing the (F_1,F_2) coordinates used to determine the time-domain "centres" of each speaker's closing diphthong realizations, Figure 4.2.3.1-3 also shows the *average F_1-F_2 trajectories* of the speech portions selected to suit *extended-token TDNN* (3206

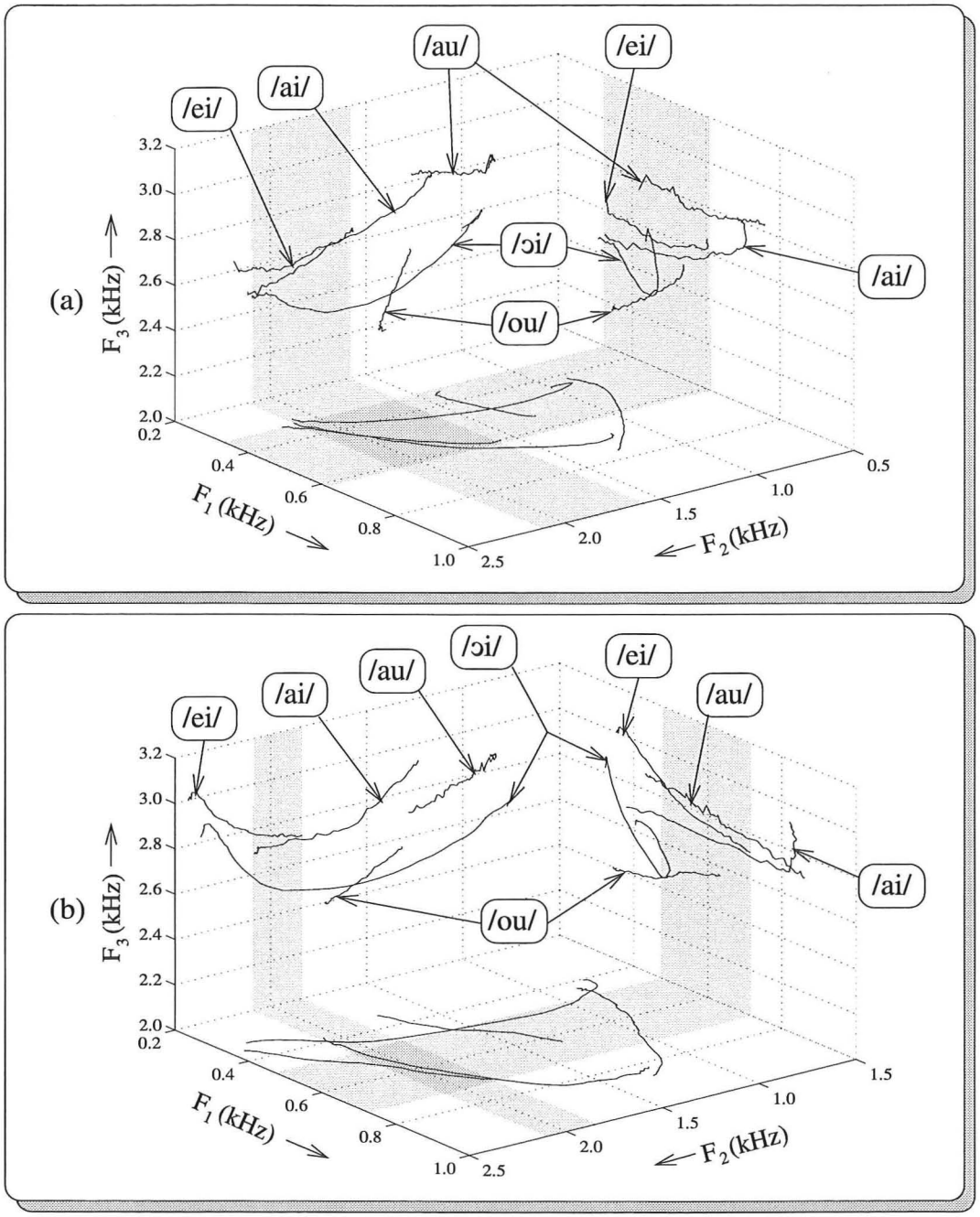

Figure 4.2.3.1-4 Three dimensional projection plots of the average diphthong-glides for (a) speaker JK's and (b) speaker HD's closing diphthong realizations (formant trajectories are projected onto the F_1-F_2, F_1-F_3 and F_2-F_3 planes). The shaded areas depicted contain the regions where the trajectories associated with realizations of /ai/ and /ei/ exhibit near overlap for each speaker.

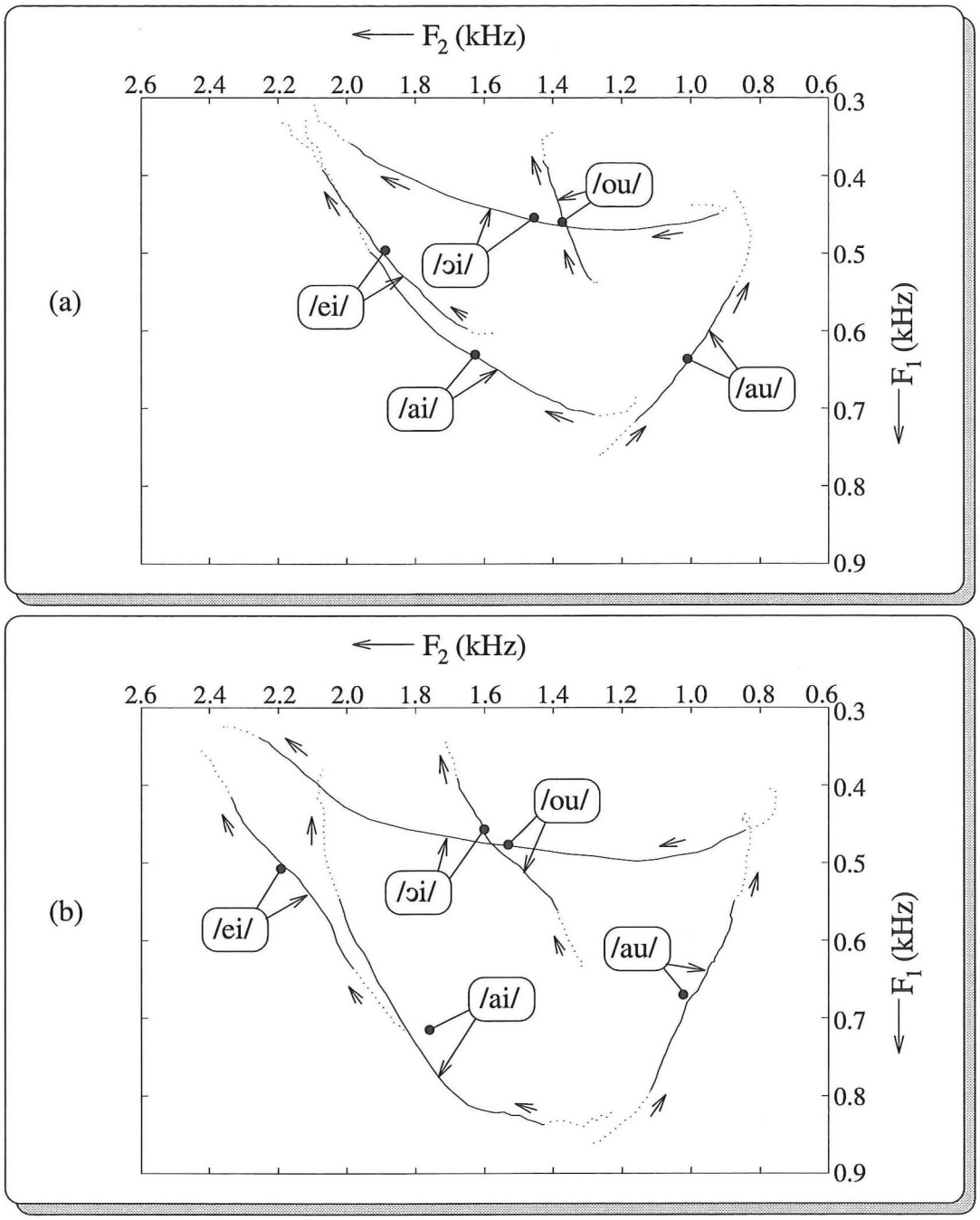

Figure 4.2.3.1-5. Same as Figure 4.2.3.1-3 except for basic-token TDNN. The *solid-line* F_1-F_2 trajectories shown are averages computed from the speech portions selected (1706 samples) to suit *basic-token TDNN*, while the *dotted-line* trajectories are the same as those in Figure 4.2.3.1-3.

speech samples). Notably, these trajectories are all quite distinct for both speakers, *except* for those corresponding to /ai/ and /ei/ which exhibit extended regions of near overlap. As the shaded regions in Figure 4.2.3.1-4 highlight, this near overlap remains evident when F_3 is also considered. This figure shows the *projections* of the average diphthong-glides of speaker JK's and speaker HD's closing diphthong realizations onto the F_1-F_2, F_1-F_3 and F_2-F_3 planes. In all

three planes, the average trajectories corresponding to /ai/ and /ei/ exhibit near overlap within the shaded regions shown. From more detailed plots produced by *STEP* resembling that in Figure 4.2.3.1-3, it is also apparent that the rates of F_1, F_2 and F_3 transition within the regions of *trajectory overlap* in Figures 4.2.3.1-3 and 4.2.3.1-4 are similar for realizations of /ai/ and /ei/. Consequently, for each speaker, in is evident that realizations of /ai/ and /ei/ are not readily distinguished by the transitions of their first three formants within the region of trajectory overlap. This fact hinders extended-token TDNN training (see the epoch counts in Tables A1.1.1.1-1 and A1.1.1.2-1) and strongly influences the *states* defined to permit closing diphthong recognition using sequence-token TDNNs (see the next section).

Figure 4.2.3.1-5 shows the average F_1-F_2 trajectories (solid lines) measured from the speech portions selected to suit *basic-token TDNN* (1706 speech samples). These trajectories are shorter than those for extended-token TDNNs (the dotted lines), since they correspond to 15 instead of 30 slice tokens. The average glides for /ai/ and /ei/ in Figure 4.2.3.1-5 overlap less than their counterparts in Figure 4.2.3.1-3, making basic-token TDNNs somewhat easier to train than extended-token TDNNs (see Tables A1.1.1.1-1 and A1.1.1.2-1).

4.2.3.2 Sequence-token TDNN

Figure 4.2.3.2-1 depicts the architecture of the sequence-token TDNNs used to process closing diphthong realizations in this work. This architecture is identical to that of basic-token TDNN (see Figure 4.2-1) with the exception that the top two layers have six rather than five (unique) nodes. Unlike the output nodes of basic- and extended-token TDNNs which have phonemic symbols assigned (see Figures 4.2-1 and 4.2.3.1-1), each output node (and associated second hidden layer node) of a sequence-token TDNN is assigned a *state* (0 to 5 in Figure 4.2.3.2-1). These states constitute the elements of sequences that a sequence-token TDNN is intended to produce in response to closing diphthong realizations. Consequently, each state corresponds to an object whose realizations are equivalent to specific *portions* of one or more closing diphthong's realizations.

Figure 4.2.3.2-2 illustrates the *anticipated* relationships between the six states associated with the sequence-token TDNNs discussed in this thesis and a set of coarse F_1-F_2 sub-regions containing portions of closing diphthong realizations (the relationships shown are for the multi-speaker sequence-token TDNNs discussed in §5.2). These sub-regions were selected to contain "similar" features (F_1-F_2 trajectory portions), while *minimizing* the number of states required to successfully distinguish realizations of the five closing diphthongs of New Zealand English. For example, state *1* is intended to contain the initial portions of the overlapping trajectories associated with /ai/'s and /ei/'s realizations. Similarly, state *2*, is intended to contain the final portions of these overlapping trajectories, as well as the final

portions of the diphthong-glides associated with /ɔi/'s realizations.

Given the assignment of states shown in Figure 4.2.3.2-2, realizations of /ai/ are typically distinguished from those of /ei/ by virtue of the additional state (state *0*) produced by sequence-token TDNNs in response to such realizations. In particular, the sequence *1-2* is the intended response to realizations of /ei/, whereas the sequence *0-1-2* is the intend response to realizations of /ai/. Since the *reference sequence*, *0-1-2,* for /ai/ incorporates the

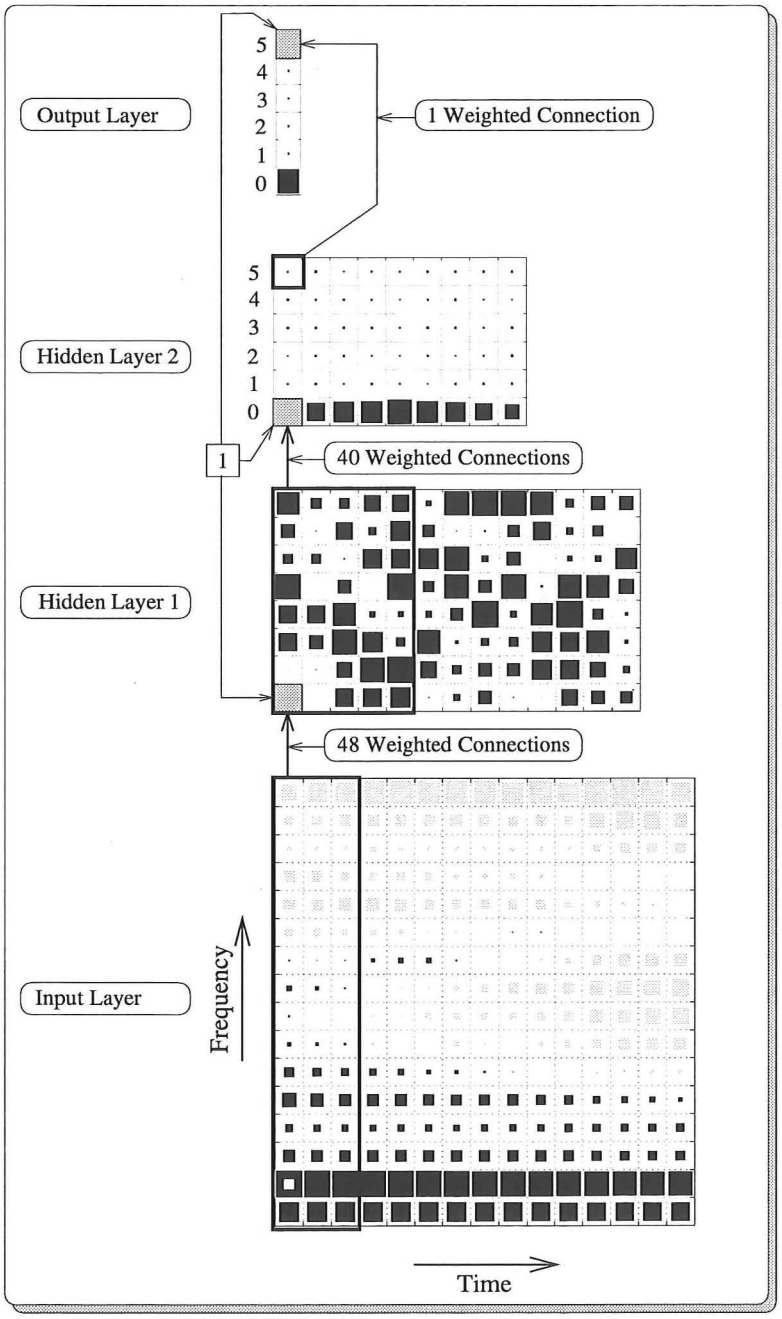

Figure 4.2.3.2-1 The architecture of the *sequence-token TDNNs* used to process closing diphthong realizations in this work.

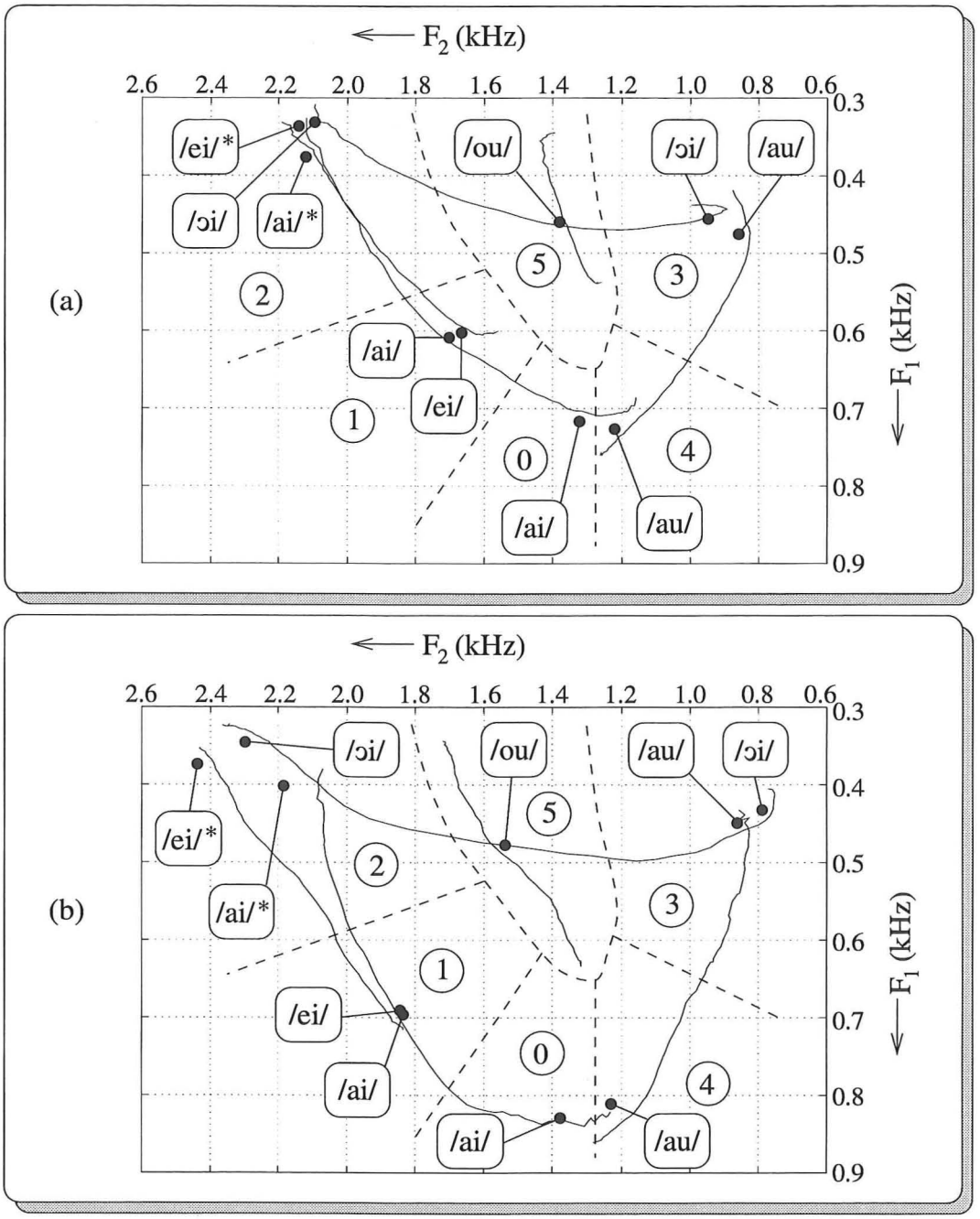

Figure 4.2.3.2-2 The anticipated relationships between F_1-F_2 regions (the regions partitioned by the dashed lines) and the six states (the circled numbers) associated with the sequence-token TDNNs discussed in this thesis, for (a) speaker JK and (b) speaker HD. The (F_1,F_2) coordinates used to select "centred" speech portions suiting sequence-token TDNNs (the labelled black dots) are also shown.
*Note that only speech portions from selected realizations of this phoneme were used to generate tokens representing state *2*.

reference sequence, *1-2*, for /ei/, it is assumed that *longer* reference sequences are matched with those produced by a sequence-token TDNN in preference to shorter reference sequences. Thus, a response sequence like *4-0-1-2-4* is eventually matched with *0-1-2* signifying /ai/ in

preference to *1-2* signifying /ei/, as shown in Figure 4.2.3.2-3.

As Figure 4.2.3-1 depicts, expert modules comprising sequence-token TDNNs also contain *matchers*. Figure 4.2.3.2-3 depicts the simple algorithm implemented by the *matchers* used in this work to match the sequences produced by a sequence-token TDNN with the reference sequences for the five closing diphthongs listed in Table 4.2.3.2-1. A matching frame slides along the *collapsed response-sequence* produced by such a TDNN (see Figure 5.1.2-3), selecting groups of three states (the number of states in the longest reference sequence used in this work). For each frame positioning, attempts are made to match progressively shorter sequences that each start with the first state in the frame. If no match is found (part (a)) then the matching frame advances by one state and the new sequences contained are tested for a match (see part (b)). If an expert module's matcher finds a match

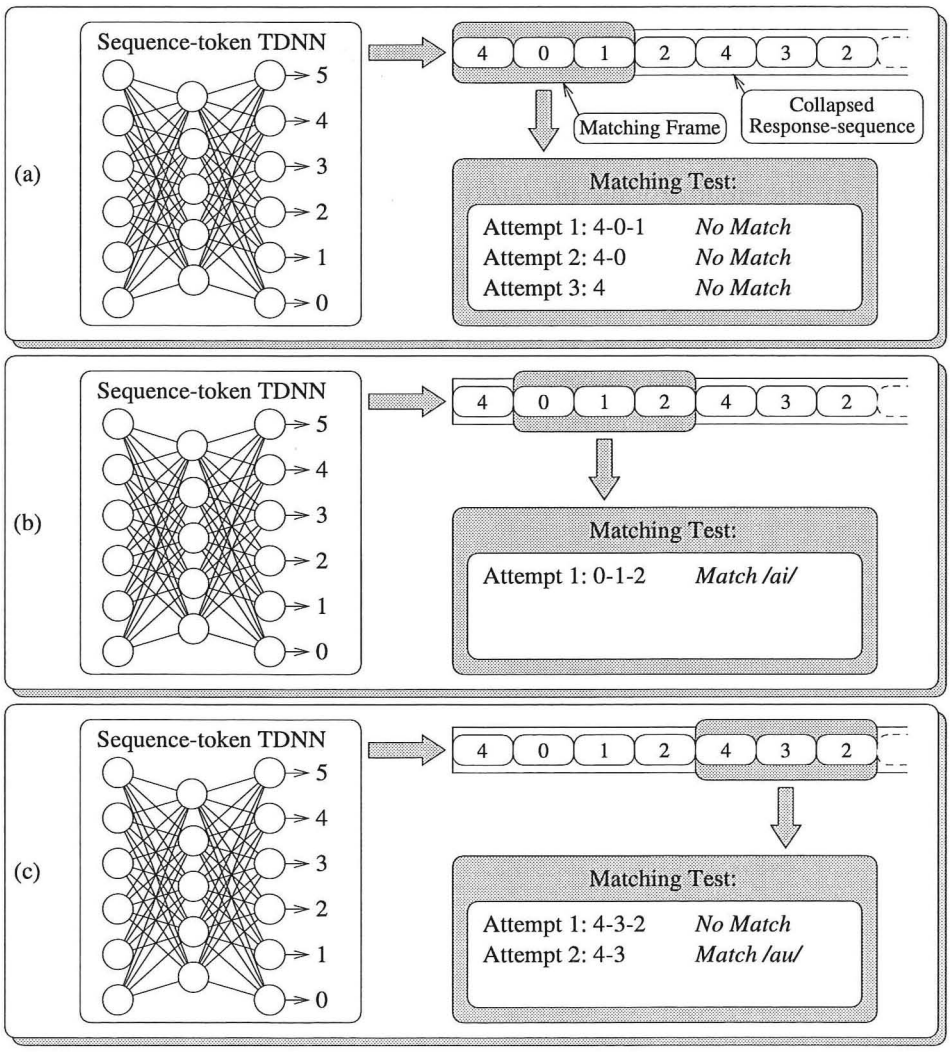

Figure 4.2.3.2-3. A depiction of the algorithm used to match the collapsed response-sequences produced by a sequence-token TDNN with reference sequences corresponding to the five closing diphthongs of New Zealand English. This algorithm is implemented by the *matcher* in expert modules comprising sequence-token TDNNs.

(part (b)), the corresponding phonemic symbol is entered in its output phoneme sequence (see Figure 5.1.2-3). Following this, the matching frame is advanced by *the number of states in the matching reference sequence* (see part (c)).

More sophisticated matching algorithms than that depicted in Figure 4.2.3.2-3 are conceivable for matching the sequences produced by sequence-token TDNNs with reference sequences. For example, one might consider an algorithm incorporating dynamic time warping or hidden Markov models. However, as the experimental results discussed in §5.1.4 and §5.2 demonstrate, the simple matching algorithm depicted in Figure 4.2.3.2-3 is sufficient when used in conjunction with the *collective response-sequences* produced by a *squad* of sequence-token TDNNs.

As Table 4.2.3.2-1 indicates, not all the reference sequences signifying a closing diphthong are the same length and not all these phonemes are represented by multiple sequences. The top sequence for each diphthong is that originally intended to signify it. However, the additional sequences listed for /ai/ and /ɔi/, were added after observing the behaviours of sequence-token TDNNs processing closing diphthong realizations (including their behaviours when processing noise corrupted realizations). These additional sequences permit variation in the realizations of /ai/ and /ɔi/, while still enabling them to be distinguished from one another and the other closing diphthongs. Unlike the other closing diphthongs listed in Table 4.2.3.2-1, /ou/ is signified by a reference sequence containing only one state (state *5*). This representation was adopted for simplicity and in hindsight is the principal weakness of the sequence-token TDNNs discussed in this work. The use of such a short reference sequence is a major cause of the (few) false-positive errors produced by expert modules comprising these networks (see §5.2.2).

Diphthong	Sequences
/ai/	*0 - 1 - 2*
	0 - 1
	0 - 2
/au/	*4 - 3*
/ɔi/	*3 - 2*
	3 - 5 - 2
/ei/	*1 - 2*
/ou/	*5*

Table 4.2.3.2-1. The reference sequences used in conjunction with sequence-token TDNNs to signify closing diphthongs in this work.

To train sequence-token TDNNs to produce the reference sequences listed in Table 4.2.3.2-1, tokens were generated from speech portions whose time-domain "centres" were selected by *STEP* in conjunction with the (F_1,F_2) coordinates shown in Figure 4.2.3.2-2. For each closing diphthong, except /ou/, these coordinates were selected to *fully* capture the diphthong-glides associated with its realizations using one or more 15 slice tokens. In

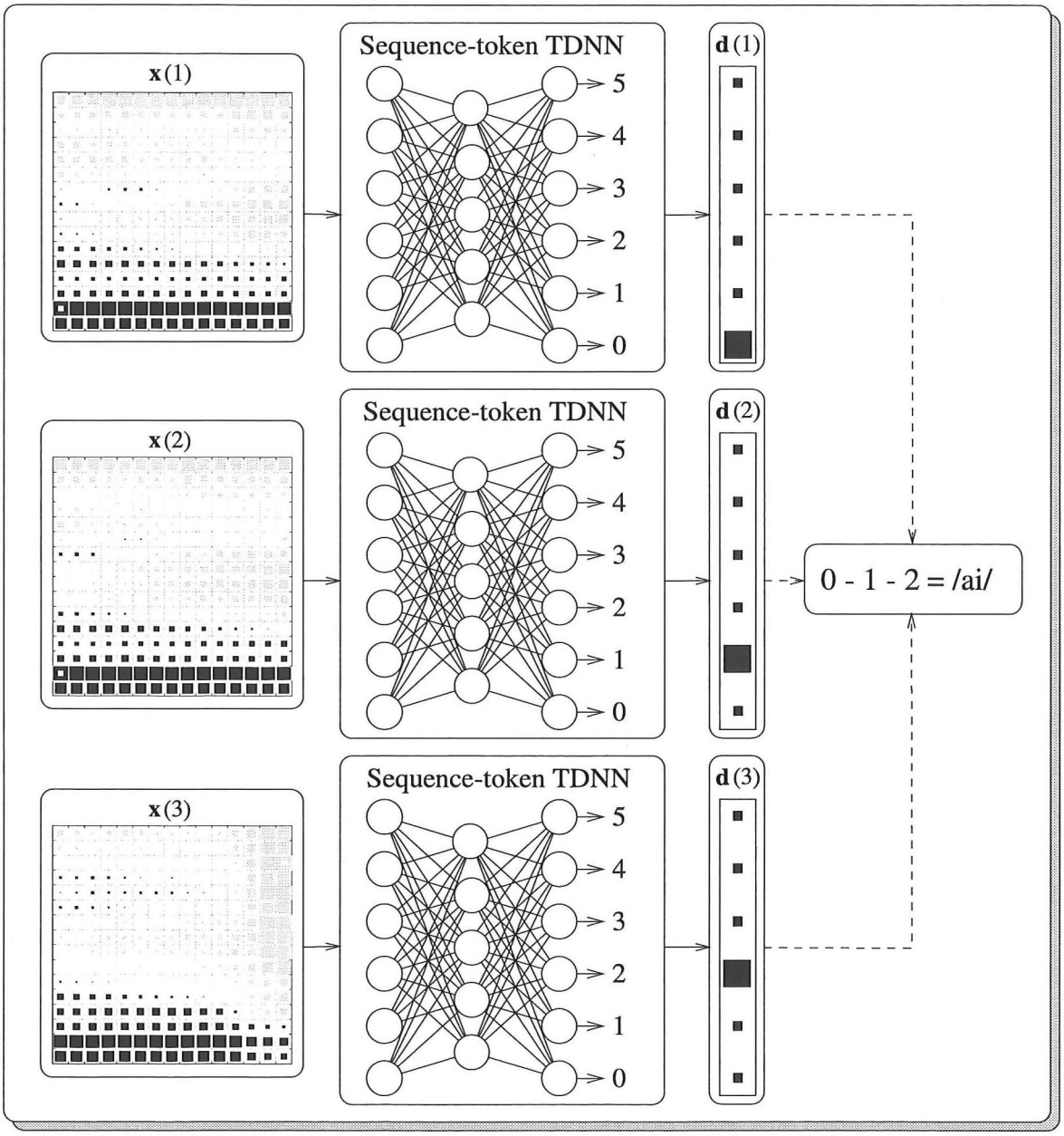

Figure 4.2.3.2-4. Training tokens, ***x(1)***, ***x(2)*** and ***x(3)*** and their associated desired responses, ***d(1)***, ***d(2)*** and ***d(3)*** for a sequence-token TDNN. The three tokens shown are derived from *one* realization of /ai/ and (from top to bottom) represent the states 0, 1 and 2. Consequently, the desired responses shown have the output nodes associated with these states most act. During the operation of the sequence-token TDNN shown, the time-order sequence *0-1-2* is desired in response to realizations of /ai/.

addition, they were also selected to capture portions of the on- and off-glides to provide contextual information concerning closing diphthong realizations. Consequently, many of (F_1,F_2) coordinates depicted in Figure 4.2.3.2-2 are close to the ends of the average F_1-F_2 trajectories depicted in this figure.

Figure 4.2.3.2-4 illustrates the process of training a sequence-token TDNN using tokens derived from a realization of /ai/ (this figure depicts *one* network, *three* training tokens and *three* desired responses). Associated with the group of training tokens shown (***x****(1)*, ***x****(2)* and ***x****(3)*) is a group of desired responses (***d****(1)*, ***d****(2)* and ***d****(3)*) that signify the elements of the time-ordered sequence *0-1-2,* which in turn signifies /ai/ (see Table 4.2.3.2-1). Ideally, during the operation of a sequence-token TDNN, *0-1-2* is produced in response to a realization of /ai/, as shown in Figure 5.1.2-3.

An important feature of the reference sequences listed in Table 4.2.3.2-1 is that only a small fraction of those possible are used to signify closing diphthongs. In particular, given that the reference sequences contain a maximum of three states and that attempts are made to match longer reference sequences first, there exist 150 possible reference sequences of which 8 are used (5.33%). The number of possible reference sequences may be found as follows. Considering the reference sequences containing three states first, these may be written generally as *a-b-c* under the constraints that $a \neq b$ and $b \neq c$, where a, b and $c \in \{0,1,...,5\}$. The constraints $a \neq b$ and $b \neq c$ reflect the fact that consecutive occurrences of the same state in a sequence-token TDNN's response-sequence cannot arise, since these would be collapsed to single occurrences prior to matching. Assuming six possible states, {0,1,...,5}, there are $6 \times 5 \times 5 = 150$ possible three state sequences, *a-b-c*, that satisfy the conditions above. For a matcher to "find" examples of the shorter reference sequences listed in Table 4.2.3.2-1 when processing matching frames containing three elements, *at least one* of the three state reference sequences, having the shorter reference sequence as its *initial elements*, must not be used as a reference sequence. For example, to find *0-1*, at least one of the reference sequences *0-1-c* must not be used (*0-1-5* for example). Similarly, to find *5*, at least one of the reference sequences *5-b-c* must not be used.[5] Consequently, each shorter reference sequence in Table 4.2.3.2-1 *replaces at least one* of the three state sequences, implying a maximum of 150 reference sequences are possible.

Using only 8 of the 150 reference sequences possible to signify closing diphthongs is advantageous since the response-sequences produced by sequence-token TDNNs in response to "noise" (sounds other than closing diphthong realizations) are less likely to contain closing

---

[5]Note in this case that at least one of the reference sequences *5-b* must not be used either. In this example, the two-state reference sequences desired (see table 4.2.3.2-1) satisfy this condition, since *none* are of the form *5-b*.

diphthong reference sequences (assuming all 150 possible reference sequences are equally likely in response to noise). Consequently, as demonstrated by the results presented in §5.1, expert modules comprising sequence-token TDNNs are less likely to produce false-positive errors compared to expert modules that use individual TDNN responses to signify closing diphthongs (see Figure 4.2.3-1 (a), (b), (d) and (e)). The use of sequences to signify the closing diphthongs does incur one minor penalty, however. Such sequences require time to observe, implying recognition is delayed until near the ends of closing diphthong realizations (until the onsets of their *off-glides* approximately). This delay does not pose a problem provided the arbitration module used can accommodate it. Information concerning the temporal positioning of a closing diphthong realization is available from the matcher, once it has been identified. This information may be used by an arbitration module to realign the phoneme sequences produced by expert modules comprising sequence-token TDNNs with those of expert modules whose phoneme sequences are undelayed (such as those comprising basic- or extended-token TDNNs). Note the depictions of the phoneme sequences in Figures 5.1.2-3 and 5.1.4.5 assume such realignment.

4.3 ANN Squads

The use of phoneme class partitioning in the formation of a modular TDNN for phoneme recognition (see §4.2.2) poses a significant dilemma. On the one hand, such partitioning enables modular TDNNs to be trained tractably by splitting the task of phoneme recognition into simpler sub-tasks. On the other hand, however, this simplification results in expert modules whose under-constrained behaviours evoke large numbers of false-positive errors.

Several methods of overcoming the dilemma posed by phoneme class partitioning have been proposed, however, these have either proven ineffectual or costly. Miyatake *et al* (1990) proposes training modular TDNNs (LP-TDNNs, see §4.2.1) as integrated networks to overcome false-positive errors by "lateral inhibition". Despite training in this manner, and with aligned and misaligned tokens (see §4.2.1), their LP-TDNNs still produced large numbers of these errors. Several authors, such as Sawai *et al* (1989) and Hataoka and Waibel (1990) suggest training expert modules using tokens representing realizations of the phonemes they are intended to recognize, as well as tokens representing realizations of *counter-example* phonemes (phoneme from other classes). Unlike the expert modules discussed in this thesis, the expert modules proposed by Sawai *et al* (1989) and Hataoka and Waibel (1990) have a special output node (or nodes) assigned to counter-examples, as well as the output nodes assigned to the phonemes being recognized. Although the use of tokens representing counter-examples during training better constrains the behaviour of expert modules as desired, it

dilutes the benefits of phoneme class partitioning. Typically, the number of counter-example phonemes to be "learned" by an expert module greatly exceeds the number of phonemes to be recognized (for example, an expert module trained to recognize the five closing diphthongs of New Zealand English must learn to recognize 38 counter-example phonemes also). Inclussion of tokens representing all possible counter-example phonemes greatly complicates the training of an expert module, since such tokens will exhibit enormous variation, making the learning of effective decision boundaries very difficult. Sawai *et al* (1989) attempts to overcome this problem by limiting the number of counter-example phonemes to those whose realizations are readily confused with the realizations of the phonemes to be recognized. Regrettably, this compromise is not entirely successful, since the resulting expert modules may still respond actively to realizations of the phonemes not chosen as counter examples (Sawai *et al* 1989 gives and example of this).

In this thesis, it is proposed that the dilemma of phoneme class partitioning be overcome by using expert modules that comprise ensembles of TDNNs rather than individual networks. This approach produces better constrained expert modules, thereby minimizing potential false-positive errors, while enabling the TDNNs they comprise to be trained on simple classification sub-tasks, as intended under phoneme class partitioning. In this work, ensembles of TDNNs were initially trained to permit the performance variations due to different weight solutions ($\boldsymbol{w}^*$) to be studied. These ensembles are referred to as *squads*, to signify that each network contained has the *same architecture* and is trained using the *same training tokens*, but different initial weights $\boldsymbol{w}(0)$.[6] In this work, the weights were initialized with random *real numbers* lying in the range [-0.5, 0.5], following Pao (1989) (as usual, these values were randomly sampled from a *uniform* population). By current standards this range is large (see Haykin 1994, section 6.7), however, it ensures that the component networks of a squad have a good chance of acquiring different weights solutions, $\boldsymbol{w}^*$, during training, since satisfactory convergence to different local minima (or different points within the depressed region surrounding a local minima) is more likely.

The rational for using squads of networks to form expert modules is now explained in conjunction with the *simplified* classification task depicted in Figure 4.3-1. This task involves classifying the realizations of two objects, O_1 and O_2 which are represented by normalized tokens containing the elements I_1 and I_2 ($I_1, I_2 \in [-1,1]$). Figure 4.3-1 (a) depicts the *pattern-space* spanned by the possible values of I_1 and I_2. The labelled symbols ("dingbats") within this space indicate the values of I_1 and I_2 associated with *training tokens* representing realizations of O_1 and O_2. The dashed line in Figure 4.3-1 (a) represents a *decision boundary* which might be learned by the network, Net_1, depicted in Figure 4.3-1 (b).

---

[6]The term *squad* seems appropriate since, according to one dictionary definition, it refers to a "small group of soldiers working or being trained together" (Cowie 1989).

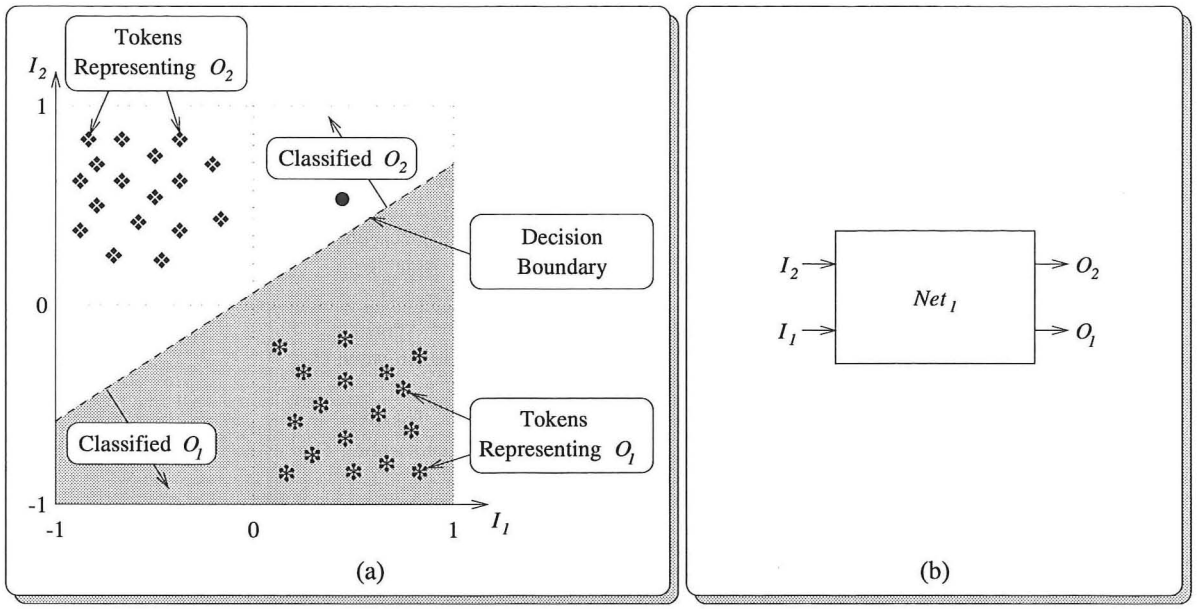

Figure 4.3-1. The pattern-space (part (a)) for a simple classification task involving two objects, O_1 and O_2, whose realizations are represented by normalized tokens comprising two elements, I_1 and I_2. Part (b) shows a block diagram of a simple ANN trained to classify realizations of O_1 and O_2, whose learned *decision boundary* appears in part (a).

This boundary splits the pattern-space into two regions whose elements are classified as O_1 or O_2 when processed by Net_1 in conjunction with the most-active rule. From the positioning of this boundary, it is clear that all the training tokens represented in this example are correctly classified.

As well as correctly identifying the tokens representing O_1 and O_2 in Figure 4.3-1, Net_1 also classifies the remaining elements of pattern-space, like that indicated by the ● in Figure 4.3-1 (a) which is classified as a realization of O_2. While this behaviour may be useful in some circumstances, it is not desirable if Net_1 is to form an expert module in a modular network. Ideally, as discussed in §4.2.2, an expert module should only classify tokens corresponding to its classification sub-task and "ignore" all other input (the token represented by ●, being distant from the training tokens representing O_1 and O_2, is unlikely to be a realization of either object and should, therefore, be "ignored"). Unfortunately, Net_1 (like individual TDNNs, Waibel *et al* 1989b) is incapable of achieving this ideal and is likely cause false-positive errors by responding highly actively to inappropriate tokens, like that represented by ● in Figure 4.3-1(a).

Due to the simplicity of the classification task posed for Net_1, there exist other similar networks whose decision boundaries may also split the pattern-space depicted in Figure 4.3-1 (a) to permit correct classification of the training tokens representing O_1 and O_2 shown. Possible examples of such decision boundaries are depicted in Figure 4.3-2. From these boundaries it is evident that their associated networks *agree* concerning the classifications of the tokens corresponding (*predominantly*) to O_1 and O_2, and *disagree* concerning all other

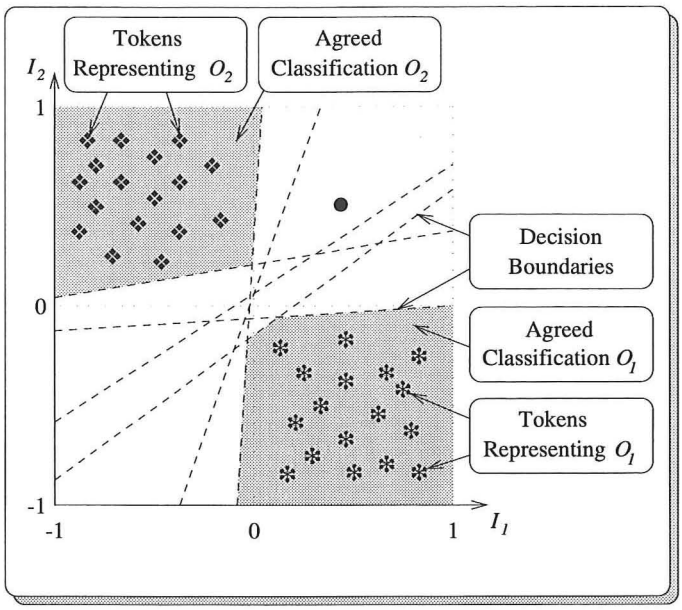

Figure 4.3-2. Examples of several possible decision boundaries corresponding to networks which may correctly classify the tokens shown representing objects O_1 and O_2. Those regions where all these networks agree about the classification of the tokens contained are shaded.

tokens (like that represented by ● in Figure 4.3-2). By combining the responses of these networks, the level of agreement they exhibit concerning a given token may be used to determine whether it should be classified or "ignored". In this way several networks may be combined to form a system that is selective about the tokens it attempts to classify. For convenience, such systems are referred to as *selective-systems.*

In this thesis, the responses of the individual networks forming a squad are combined using a generalized version of the *majority voting rule* (Hansen and Salamon 1990), referred to as the *selective-system voting rule.* Under this rule, the classification of a given token produced collectively by a squad is that reached by a fraction, A, of its networks, where $\frac{1}{2} < A \leq 1$ is referred to as the *agreement threshold.* If a smaller proportion than A of these networks agree concerning the identity of a token, then a *null classification* is produced to signify that the squad cannot classify this token.[7] For example, if $A=1$, then the token represented by ● in Figure 4.3-2 would lead the squad of networks whose decision boundaries are also depicted in this figure to produce a *null* classification under the selective-system voting rule, since not all these networks agree concerning the classification of this token.

The selective-system voting rule may be stated mathematically as follows. Assuming R networks, each having K output nodes, are combined, the activation of the i^{th} *collective*

[7]In practice this *null classification* could be signalled in a number of different ways. In this thesis, it is envisaged that a null classification be signalled by setting *all* the collective or *squad output nodes* to *0*, implying complete inactivity.

output node, ϕ_i, is given by

$$\phi_i = \begin{cases} \overline{o_i}, & \text{if } S(i) \geq A \\ 0, & \text{otherwise} \end{cases} \tag{4.3-1}$$

where $\bar{o}_i$ is the average activation of i^{th} output node for all R networks given by

$$\overline{o_i} = \frac{\sum_{r=1}^{R} o_{i,r}}{R}, \tag{4.3-2}$$

A is the agreement threshold and $S(i)$ is given by

$$S(i) = \frac{\sum_{r=1}^{R} M_i(r)}{R} \tag{4.3-3}$$

In this expression, $M_i(r)$ is given by

$$M_i(r) = \begin{cases} 1, & \text{if } o_{i,r} \text{ is the most active} \\ 0, & \text{otherwise} \end{cases} \tag{4.3-4}$$

and $o_{i,r}$ is the i^{th} output of the r^{th} network combined. If the i^{th} output node of all R networks combined is the *most active*, then all $M_i(r)$ will be equal to *1*, $S(i)=1 \geq A$ and $\phi_i=\bar{o}_i$, implying the object index associated with the i^{th} collective output node is the collective classification. If no $S(i) \geq A$, $i=1,2,...K$, then all the squad output node activations are zero, implying a *null* classification.[6] In the event of a tie such that the largest values of $S(i)$, $i=1,2,...,K$ are equal, a *null* classification is also produced, since no $S(i) > 0.5$ and, therefore, all are less than A.

Figure 4.3-3 (a) depicts a modular network incorporating one expert module to classify the objects O_1 and O_2 and another to classify the objects O_3 and O_4. If these modules are formed from the squads of networks whose decision boundaries are shown in Figures 4.3-3 (b) and (c), and A=1, then inappropriate tokens for each module, like those represented by the ●s shown, are (predominantly) given *null* classifications. Consequently, unhindered by conflicting classifications from the two expert modules shown in Figure 4.3-3 (a), the arbitration module is *more likely* to select the classifications produced by the appropriate expert module for each token processed and is, therefore, less likely to produce false-positive errors.

Interestingly, the task of classifying the objects represented by the tokens depicted in

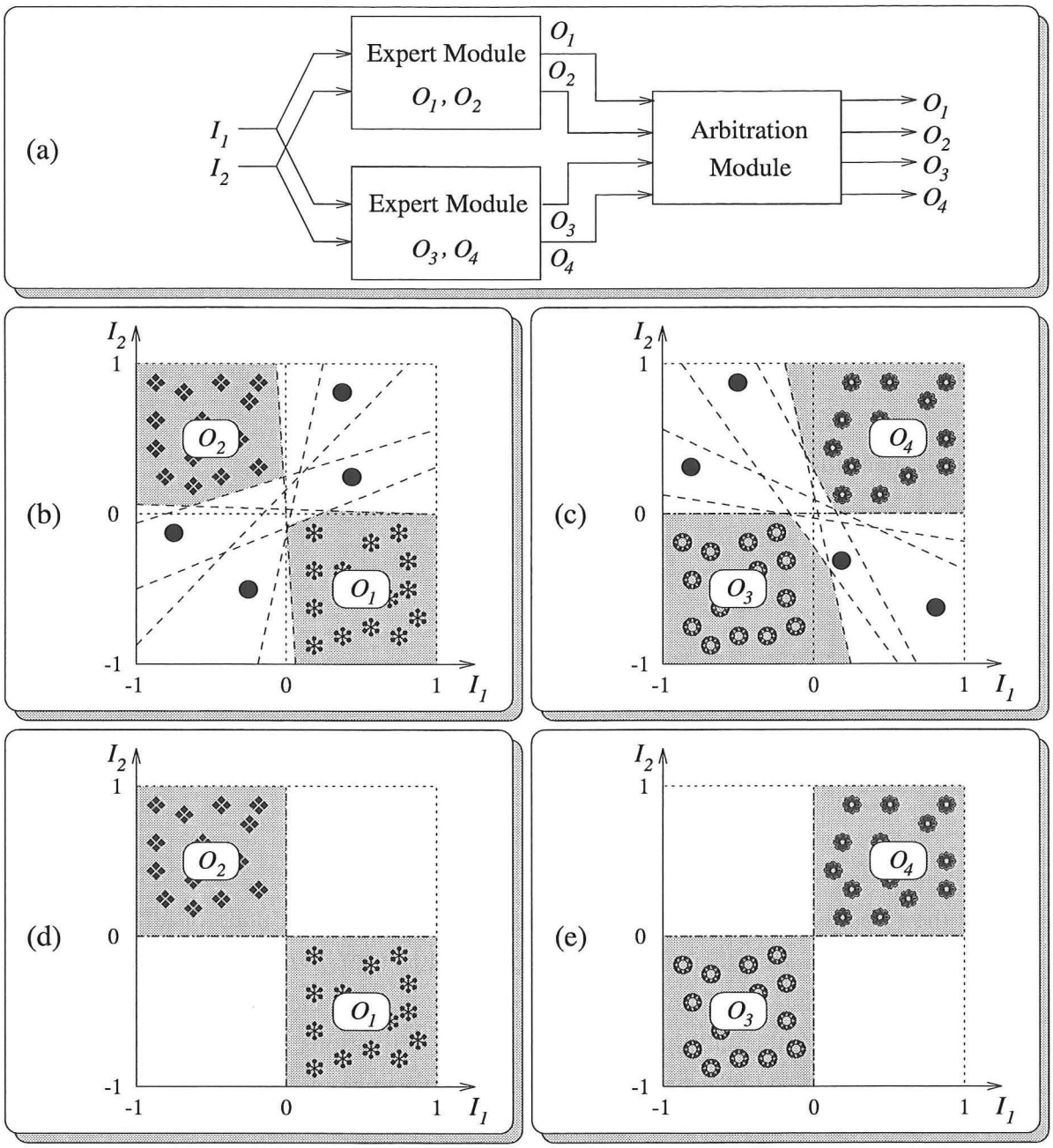

Figure 4.3-3. A modular network (part (a)) to solve a classification task involving four objects, O_1, O_2, O_3 and O_4, whose realizations are represented by normalized tokens containing the elements I_1 and I_2. The pattern-spaces shown in parts (b) and (c) contain the decision boundaries (dashed lines) associated with the component networks of the squad used to form each expert module in part (a). Parts (d) and (e) show decision boundaries associated with the component networks required to form an optimal squad for each expert module.

Figure 4.3-3 (b) and (c) might be achieved optimally using squads comprising just *two* networks each. The decision boundaries associated with these networks are depicted in parts (d) and (e) of Figure 4.3-3. Unfortunately, these networks are less likely to be produced by training algorithms such as the back-propagation algorithm (see §4.1), since they are likely to be associated with larger values of $\mathscr{E}_{av}$ (see equation 4.1-6). However, as yet there is no

better way of designing component networks for squads, implying larger, sub-optimal, squads must be used. Despite being more computationally intensive, sub-optimal squad-based expert modules may better approximate ideal expert modules for phoneme class recognition than traditional expert modules comprising individual networks, as is demonstrated by the results presented in §5.1.4.

An ideal expert module for phoneme class recognition is an example of a *selective-system* that applies phoneme classifications to selected tokens, while "ignoring" (applying *null* classifications to) others. This behaviour is viewed as a form of *selective attention* in this thesis, since such modules give the appearance of attending to (applying non-*null* classifications to) only selected tokens. ANN mechanisms for selective attention have also been demonstrated by Fukushima (Fukushima and Imagawa 1993; Fukushima 1987). His networks are intended to both segment and classify hand written characters and are, therefore, significantly more complicated than the squads proposed in this thesis (these must only classify or "ignore" their input).

The use of network ensembles in various applications has been proposed by a number of authors, such as Benediktsson *et al* (1993), Benediktsson and Swain (1992), Jordan and Jacobs (1992) and Hansen and Salamon (1990). Regrettably, these network ensembles are of limited use for automated phoneme recognition, since their architectures do not permit temporal information processing. Benediktsson *et al* (1993) discusses using network ensembles (so called *parallel consensual neural networks*) to solve classification tasks involving data obtained from several sources (remote sensing and geographic data), and analyses these in terms of statistical consensus theories (similar work is also described in Benediktsson and Swain 1992). Hansen and Salamon (1990) discuss using ensembles to improve upon the performances of individual networks when solving *non-modular* classification tasks (like classifying realizations of O_1 and O_2). Their statistical analyses of such ensembles attempt to treat the difficult problem of correlation between component network outputs. Unfortunately, their findings are not applicable to the squads of TDNNs discussed in this thesis, since the component networks of these are trained using *identical* tokens.[8]

The modular network comprising ensembles of *local experts* discussed by Jordan and Jacobs (1992), is currently the closest network architecture to the modular TDNNs discussed in this thesis. As with the expert modules of a modular TDNN, local experts are trained to classify object realizations belonging to one sub-task of a recognition problem that has been partitioned into several sub-tasks. As with expert modules, local experts also derive input

[8]This training practice invalidates Hansen and Salamon's assumption that the component networks make independent errors, which (as they discuss) is only valid approximately, even when independent token sets are used for training.

from a common input token. However, in contrast to modular TDNNs, arbitration between local experts is conducted by *gating networks* that use the *input tokens* fed to local experts, rather than their *outputs*, to decide which to "believe" in response to each input token processed. Consequently, current module networks compising local experts cannot make use of *squad-based* local expert architectures, since information concerning network agreement cannot influence arbitration decisions.

In this thesis, the results of experiments with traditional and squad-based expert modules for closing diphthong recognition are reported (see §5.1). These results provide empirical evidence to suggest that the latter are better approximations to ideal expert modules for closing diphthong recognition than the former. This evidence provides a strong motivation for further theoretical investigation of network ensembles in the context of modular network approaches to automated phoneme recognition.

Chapter 5

Closing Diphthong Recognition Using TDNNs

This chapter presents the results of experiments using traditional and squad-based expert modules of the form discussed in §4.2.3 to recognize New Zealand English closing diphthong realizations. The next section compares the performances of *speaker-dependent* expert modules comprising basic-, extended- and sequence-token TDNNs created for the two New Zealand speakers discussed in §3.1. §5.2 presents the results of further experiments to discover the properties of expert modules comprising squads of sequence-token TDNNs. These experiments attempt *multi-speaker* recognition of the closing diphthong realizations produced by both New Zealand speakers. Finally, §5.3 summarises the main results presented in this chapter.

5.1 Speaker-Dependent Experiments

This section discusses the creation and testing of traditional and squad-based expert modules comprising basic-, extended- and sequence-token TDNNs. §5.1.1 discusses the training of the TDNNs used to form these modules. This is followed in §5.1.2 by a discussion of the methodology adopted in this work to compare expert modules. §5.1.3 presents test results for the traditional expert modules comprising individual TDNNs. These results are used as a reference for comparison with the squad-based expert module test results presented in §5.1.4.

5.1.1 Training

For speakers JK and HD (see §3.1.1), 50 examples of basic–, extended- and sequence-token TDNN were trained successfully using the modified back-propagation algorithm discussed in §4.2 (150 networks in total). Speaker-dependent training (and subsequent testing) was adopted following the experimental approach used by Waibel and his colleagues for Japanese phoneme recognition (see §4.2.1). This approach establishes whether *candidate expert modules* for a given phoneme class can accommodate intra-speaker variation *before*

progressing to more difficult multi-speaker and speaker-independent recognition tasks incorporating inter-speaker variation as well.

To determine the best method of training basic-, extended- and sequence-token TDNNs to process speaker JK's and speaker HD's closing diphthong realizations, initial training experiments were conducted. For these experiments, tokens representing the closing diphthong realizations in each speaker's 320 closing diphthong syllables (see §3.1.1) were generated to suit each type of TDNN. For the basic- and extended-token TDNNs, these tokens were generated from "centred" speech portions selected using *STEP* in conjunction with each speaker's (F_1,F_2) coordinates, as discussed in §4.2.3.1. For each speaker, a total of 320 tokens were generated to suit both these types of TDNN. For the sequence-token TDNNs, tokens were generated from speech portions selected using *STEP* in conjunction with each speaker's (F_1,F_2) coordinates discussed in §4.2.3.2. For each speaker, a total of 580 tokens were generated to suit this type of TDNN, with many tokens representing different portions of the same closing diphthong realizations (see Figure 4.2.3.2-2).

As discussed in §3.1.1, four realizations of each closing diphthong were recorded for speakers JK and HD in each of the sixteen contexts listed in Table 3.1.1-1. Consequently, for each speaker, the tokens suiting each type of TDNN were split into four groups containing one token representing each closing diphthong in each context. Examples of each type of TDNN were trained (speaker-dependently) using one, two and three of their associated groups of tokens. From these trials it was found that examples of basic-, extended- and sequence-token TDNNs, when trained with *one group of tokens*, could correctly classify all the tokens in their other three groups. Consequently, *all* the TDNNs created for the speaker-dependent experiments discussed in this section were trained using *one group of tokens* representing one quarter of their associated speaker's available corpus of closing diphthong realizations (80 realizations). The remaining tokens (representing 240 realizations) were used to test the *classification performances* of their associated TDNNs at the completion of training. For convenience, these tokens are referred to as *aligned test tokens* in this thesis.

For speakers JK and HD, §A1.1.1.1 and §A1.1.1.2 give training details for the 50 examples of basic-, extended- and sequence-token TDNNs created for the experiments discussed in this section. For each speaker, the examples of each type of TDNN were trained using *identical* training tokens, but with different initial random weights, $\boldsymbol{w(0)}$, given by *real numbers* chosen randomly from the range [-0.5, 0.5] (all the weights for each TDNN trained were initialized with such values). Initially, this was done to permit the variation in performance due to different weight solutions to be observed, however, subsequently it enabled the formation of expert modules comprising *squads* of TDNNs. During the training of each TDNN created, its weights were updated until the average McClelland error (see equation (4.1-6)) fell below a predetermined threshold; $\mathscr{E}_{av} < 0.015$ for basic- and sequence-token TDNNs and $\mathscr{E}_{av} < 0.05$ for extended-token TDNNs. These target errors were chosen

following the preliminary experiments with each type of TDNN discussed above and led to the *perfect classification performances* listed in Tables A1.1.1.1-1 and A1.1.1.2-1, for *all* examples of the three types of TDNN created.

5.1.2 A Methodology for Comparing Expert Modules

Given a set of *candidate expert modules* for a particular phoneme class, such as the closing diphthongs, a method of comparing these modules is required to permit one to be selected for use in a modular TDNN. Ideally, the best candidate might be selected by incorporating each in an existing modular TDNN, one at a time, and comparing the resulting performances. Unfortunately, however, a modular TDNN for New Zealand English does not yet exist, implying candidate expert modules for such a system must be compared in some other manner.

One method of comparing candidate expert modules in the absence of a modular TDNN is to compare the *classification performances* afforded by their component TDNNs (Hataoka and Waibel 1990 use this approach). Using this method, a set of *aligned test tokens* not observed during training are processed by a trained TDNN to determine the proportion of these it can correctly classify. This proportion (often quoted as a percentage) is then used as an *estimate* of the network's classification performance. Unfortunately, this performance measure does not completely describe the behaviour of an expert module when operating within a modular TDNN. In particular, classification performance only measures the ability of such a module to correctly process *appropriate input* (tokens representing phoneme realizations from its phoneme class), while ignoring the errors it may *potentially* cause by responding to *inappropriate input* (tokens associated with phoneme realizations from other classes).

Classification performance is also of limited use when comparing the various types of speaker-dependent expert modules discussed in this chapter for two further reasons. First, all of the TDNNs from which these modules are formed afford perfect classification performances (see Tables A1.1.1.1-1 and A1.1.1.2-1). Consequently, this performance measure provides no guidance concerning which candidate expert module to select. Second, classification performance does not equate to phoneme recognition performance (*recognition performance*, henceforth) for expert modules comprising sequence-token TDNNs. For these expert modules, classification performance only indicates that the elements of a desired reference sequence may be identified from aligned test tokens, not that such elements may be identified in the correct sequence to enable phoneme recognition.

Given the problems with classification performance described above, it is desirable to establish a new method of comparing expert modules that considers more fully their *potential*

effects on the performance of a modular TDNN. The method proposed in this thesis compares the responses of candidate expert modules to *entire utterances*, rather than to aligned test tokens. Input derived from such utterances better resembles that which expert modules would receive if present in an operational modular TDNN, permitting their behaviours to be observed more fully. For example, Figure 5.1.2-1 depicts the response of a traditional expert module comprising a basic-token TDNN to an utterance of the word *bide* (/baid/). Part (a) shows the speech signal associated with this utterance, while part (b) shows estimates of the formant frequencies associated with the *centre slice* of each token processed (these tokens contain 15 slices). Part (c) shows a Hinton diagram representing the expert module's response to each token (a column of squares in this diagram represents one response). Part (d) shows the response-sequence (for convenience, plotted rather than listed as in Figure 2.5.2-1) derived from the Hinton diagram in part (c) using the most-active rule (see §4.1). Part (e) traces the activation associated with the *most active* module output. The dashed lines in parts (a) delimit the speech portion used to create an aligned test token from the utterance of *bide* shown. The corresponding lines in part (b) delimit the formant transitions within this speech portion.

Apart from the *desired response* corresponding to a realization of /ai/ in the utterance depicted in Figure 5.1.2-1, *highly active* and *incorrect* responses are also produced while processing other portions of this utterance. *Potentially*, these incorrect responses may lead a modular TDNN to produce false-positive errors, particularly those responses signifying /au/ and /ei/ in the vicinity of /ai/'s realization.

Without observing a modular TDNN in operation, the degree to which a candidate expert module must be active to cause a false-positive error at a given instant, is unknown. One method of overcoming this lack of knowledge is to take the pessimistic view that *all* incorrect responses lead to false-positive errors. Figure 5.1.2-2 shows the effect of applying this view to the response-sequence depicted in Figure 5.1.2-1. In this case, groups of like responses within the sequence depicted in part (c) (the horizontal line-segments) are collapsed to form single elements of the phoneme sequence depicted in part (d). This sequence is assumed (pessimistically) to be the output of a modular TDNN in response to the utterance of the word *bide* processed and contains five false-positive errors, as shown.

Given phoneme sequences concerning entire utterances, like that in Figure 5.1.2-2 (d), the performances of their associated expert modules may be compared. For convenience, this method of comparing expert modules is referred to as the *isolated-test method*, since the modules compared are isolated from a modular TDNN. The isolated test method permits candidate expert modules to be compared in terms of *recognition performance* and *false-positive error performance*. Recognition performance measures the ability of an expert module to correctly recognize phoneme realizations represented by tokens that it would receive if present in an operational modular TDNN. Importantly, these tokens are *not* aligned in any way with the phoneme realizations present in test utterances. False-positive error performance

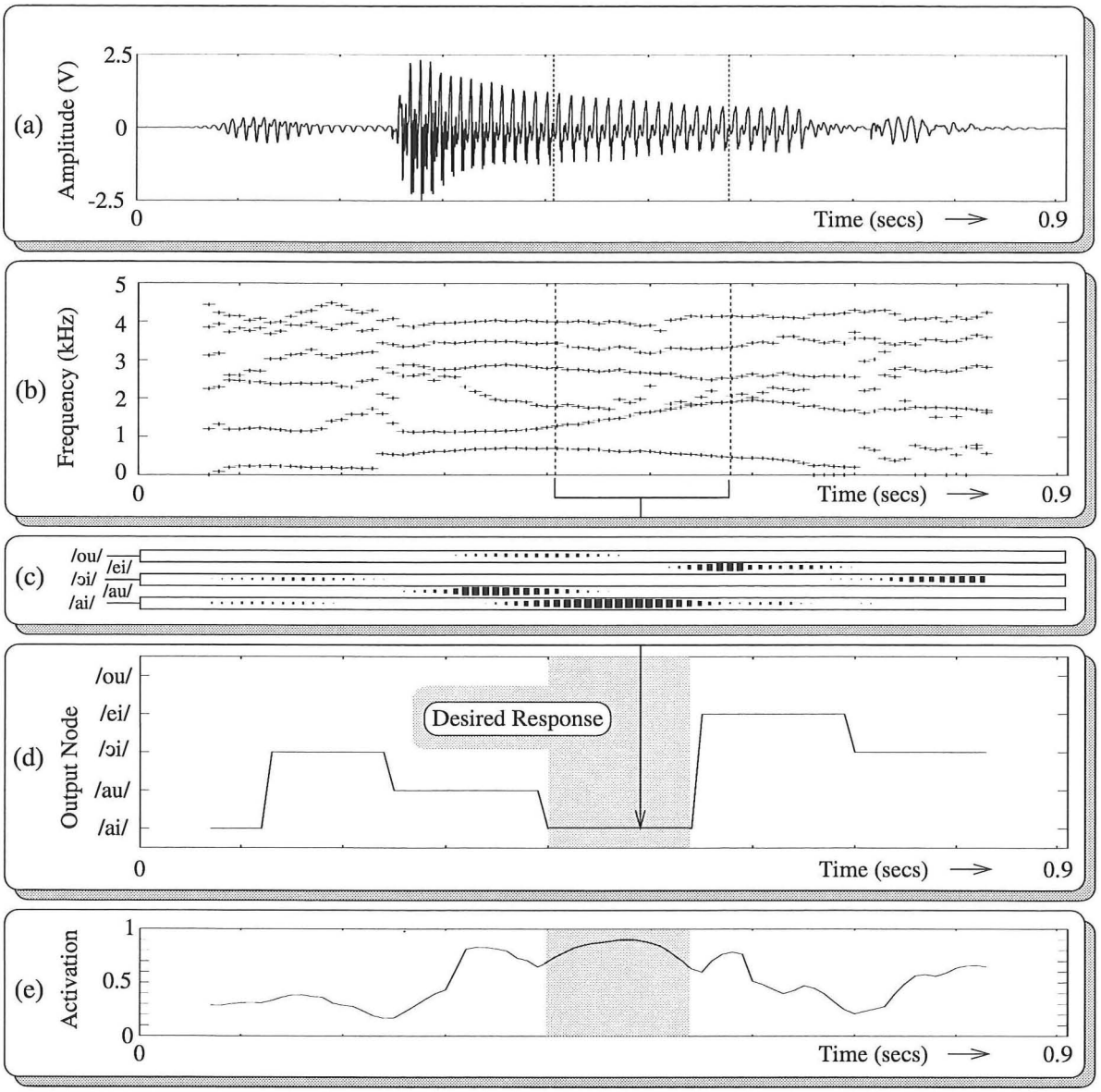

Figure 5.1.2-1. The response of a traditional expert module comprising a basic-token TDNN to an utterance of the word *bide* (/baid/) (speaker JK). Parts (a) and (b) show the speech signal and raw formant tracks for this utterance respectively, while part (c) is a Hinton diagram of the module's response. The elements of this response are aligned with the *centre* slices of their associated tokens and span the full temporal extent of the utterance shown (they do not extend over all the "silence" at each end of this utterance, since tokens for basic-token TDNN must contain 15 slices). The *response-sequence* depicted in part (d) indicates which module output node is most active at each instant of time. The activation of this most active output node is given in part (e).

measures the *potential* of an expert module to cause false-positive errors when operating in a modular TDNN. Comparing expert modules using this performance measure is desirable since false-positive errors are a major problem with current modular TDNNs (see 4.2.1).

For traditional expert modules comprising individual basic- or extended-token TDNNs, phoneme sequences for use with the isolated-test method are derived directly from the response-sequences of their component networks, as depicted in Figure 5.1.2-2. For squad-

based expert modules comprising these types of TDNNs, phoneme sequences are derived directly from their *collective response-sequences*, as discussed in §5.1.4. For traditional expert modules comprising individual sequence-token TDNNs, phoneme sequences are derived as depicted in Figure 5.1.2-3. The response-sequence of the component sequence-token TDNN within such a module (see part (c)) is first collapsed by replacing groups of like responses by single elements (see part (d)). The algorithm depicted in Figure 4.2.3.2-3 is then applied by

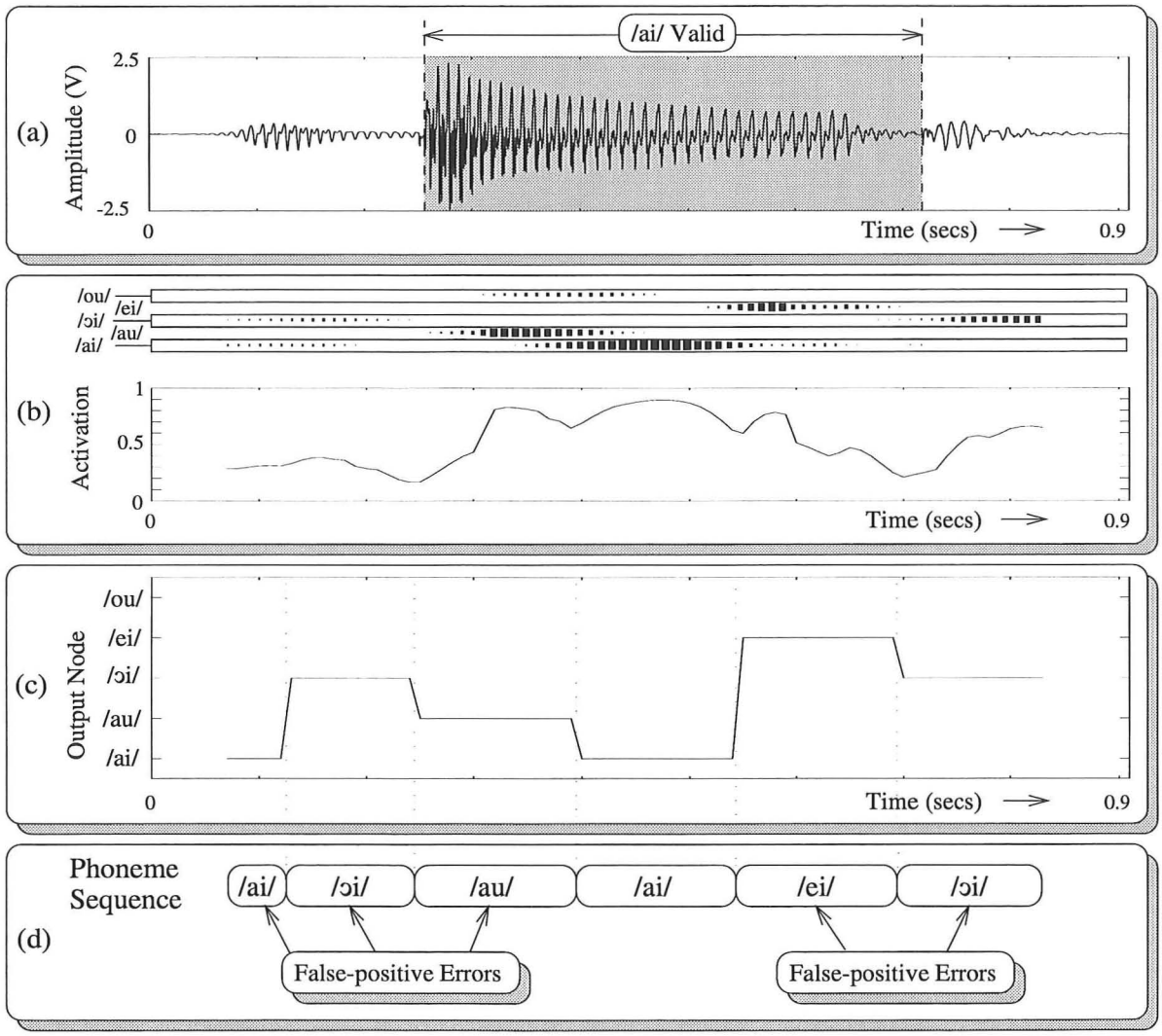

Figure 5.1.2-2. The method used to derive phoneme sequences for use with the isolated-test method from the responses of a traditional expert module comprising a basic-token TDNN (phoneme sequences are derived similarly for expert modules comprising extended-token TDNNs). Parts (a) and (b) show an utterance of the word *bide* and the expert module's response to it, respectively. Part (c) shows the response-sequence derived from part (b) using the *most-active* rule. From this sequence, the required phoneme sequence shown in part (d) is derived *directly*. As indicated, this phoneme sequence contains one correct response and five false-positive errors. Detections of the desired diphthong /ai/ are only regarded as valid within the extent marked by the dashed lines shown in part (a), which correspond to the hand labelled instances of plosive release (or "silence" for diphthong syllables not beginning or ending with a plosive). Speech portions outside this extent are assumed to have a low correlation with the diphthong /ai/ (see Fant's model, Figure 2.2.3-1), an assumption supported by the low activation of the /ai/ output for these portions (see part (b)).

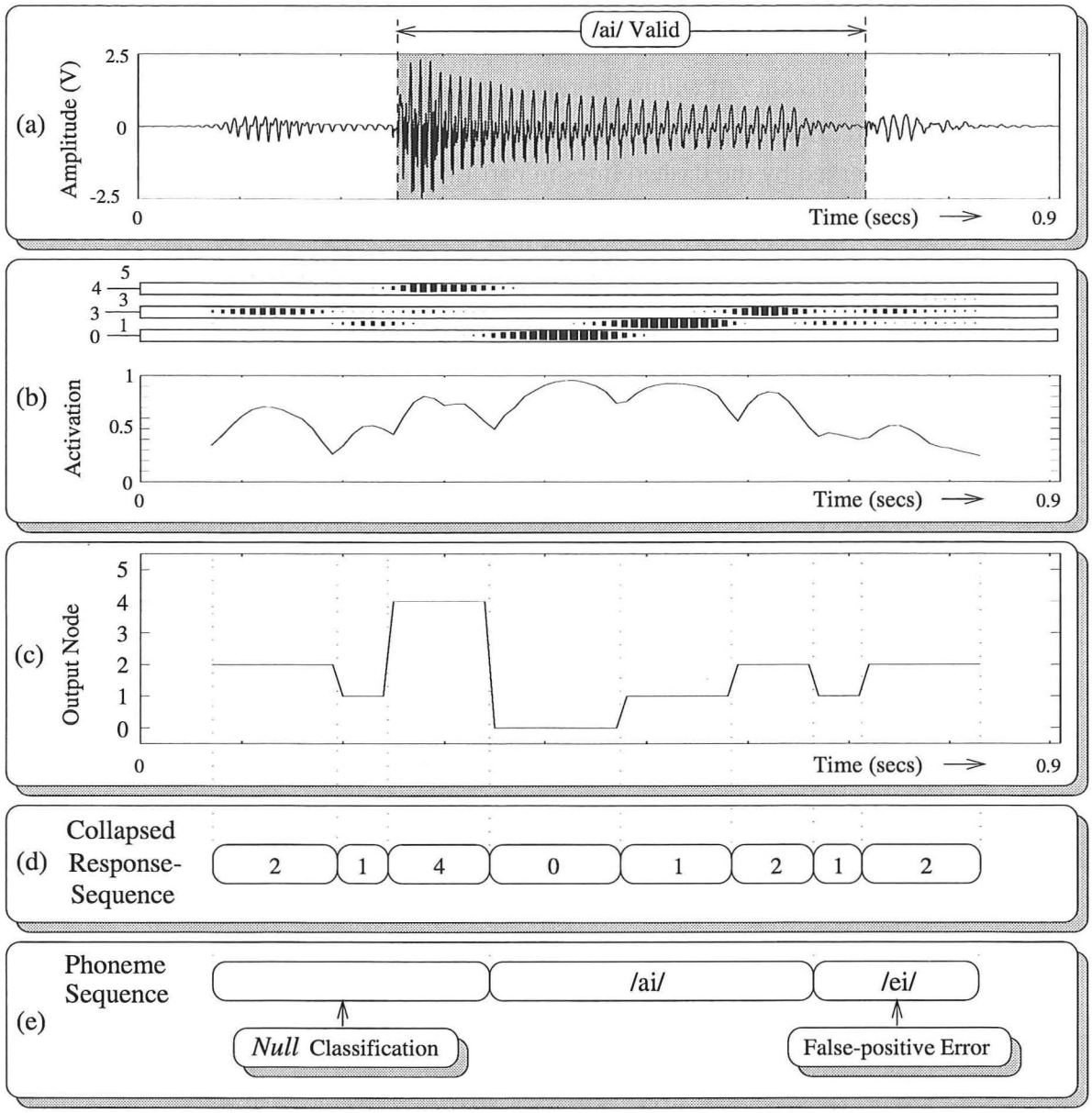

Figure 5.1.2-3. The method used to derive phoneme sequences for use with the isolated-test method from the responses of a traditional expert module comprising a sequence-token TDNN. Parts (a) and (b) show an utterance of the word *bide* and the expert module's response to it, respectively. Part (c) shows the response-sequence derived from part (b) using the *most-active* rule. This sequence is collapsed (see part (d)) and then processed by the module's *matcher* to produce the required phoneme sequence shown in part (e) (note the blank element of this sequence implies a *null* classification).

the module's *matcher* to determine if any closing diphthong realizations are present. If found, information concerning the extents of such realizations is obtained from the matcher and entered in the module's phoneme sequence (see part (e)). Note that the blank elements depicted in such sequences signify *null* classifications, implying no closing diphthong realization is present. The derivation of phoneme sequences from squad-based expert modules comprising sequence-token TDNNs follows a similar process, but once again uses collective

response-sequences as discussed in §5.1.4.

When comparing expert modules using the isolated test method, detections of the "correct" phoneme that do not fall within the region *anticipated* for its realization are regarded as false-positive errors. For example, as shown in Figure 5.1.2-2, detections of /ai/ that do not fall within bounds marked by the dashed lines in part (a), are regarded as false-positive errors (these bounds were hand labelled and correspond to the instances of plosive release in the example shown). It must be noted that false-positive error performances obtained using the isolated-test method verge on *worst case* estimates, since many of the "errors" observed correspond to marginally active module responses that may not normally cause false-positive errors. However, given the similar propensity for basic- extended- and sequence-token TDNNs to be highly active in response to inappropriate input (see, for example, the mean activations in Figures 5.1.4-1 through 5.1.4-3), it is assumed that false-positive error performances compared in this work are representative of the *relative potential* for expert modules comprising these TDNNs to cause false-positive errors.

5.1.3 The Performances of Traditional Expert Modules

This section discusses the performances of traditional expert modules for closing diphthong recognition comprising individual basic-, extended- and sequence-token TDNNs. These performances were estimated using the isolated-test method and provide a reference for comparison with those observed for the squad-based expert modules discussed in §5.1.4

5.1.3.1 Performances on Closing Diphthong Syllables

For speakers JK and HD, Table 5.1.3.1-1 lists statistics concerning the recognition performances of traditional expert modules comprising individual basic-, extended- and sequence-token TDNNs. For convenience, these modules are referred to as BT_1s, ET_1s and ST_1s, respectively.[1] For each speaker, these statistics were determined by processing the closing diphthong syllable utterances not used for network training (240 utterances per speaker) using the 50 expert modules of each type available. For both speakers, the recognition performances of the BT_1s and ET_1s are uniformly perfect, like the classification performances of their component networks (see §A1.1.1.1 and §A1.1.1.2). In contrast, the

[1]The subscript *1* indicates an expert module containing an individual network. In §5.1.4, expert modules comprising squads of TDNNs are similarly named with the subscripted numeral indicating the number of networks combined within each squad.

recognition performances of the ST_1s are imperfect and variable, unlike the perfect

(a) Speaker JK

Module Type	% Correct			
	min.	mean	max.	standard deviation
Basic-Token (BT_1)	100	100	100	0.0
Extended-Token (ET_1)	100	100	100	0.0
Sequence-Token (ST_1)	67	78	90	5.1

(b) Speaker HD

Module Type	% Correct			
	min.	mean	max.	standard deviation
Basic-Token (BT_1)	100	100	100	0.0
Extended-Token (ET_1)	100	100	100	0.0
Sequence-Token (ST_1)	67	85	94	6.5

Table 5.1.3.1-1 Statistics concerning the recognition performances of traditional expert modules comprising individual basic-, extended- and sequence-token TDNNs when processing (a) speaker JK's and (b) speaker HD's closing diphthong syllables (syllables *not* used for network training).

classification performances of their component networks (see §A1.1.1.1 and §A1.1.1.2).

From Table 5.1.3.1-1, it is apparent that the recognition performances for the ST_1s differ from those for the BT_1s and ET_1s. The significance of these differences may be analyzed further using *Cochran's generalized Q-test* for correlated proportions (see §A3.1). This test examines the *null hypothesis* that the recognition performances (expressed as proportions) are equal, while accounting for the correlations that may exist between them. Such correlations are likely, since each speaker's expert modules have been trained and tested using the same sets of speech utterances (one set for training and one set for testing).

Applying Cochran's generalized Q-test to the recognition performances for speaker JK's expert modules (M=150 modules in total) gives Q=7429, which exceeds the *critical value* $Q_{\alpha=0.01}(\varepsilon=0.0154,\nu=149)$=640, where α is the *level of significance* (Neter *et al* 1988; Daniel 1990), ε is a variable computed during testing (see §A3.1) and ν is the degrees of freedom. Consequently, the null hypothesis that speaker JK's traditional expert modules all perform equally is *rejected* at a significance level of 0.01. From this outcome it is inferred that the recognition performances of *at least some* of speaker JK's ST_1s differ significantly from 100%, the performance afforded by *all* of this speaker's BT_1s and ET_1s.

To discover if all speaker JK's ST_1s perform differently from this speaker's BT_1s and

ET_1s, a second Q-test may be conducted in conjunction with *worst* examples of the latter modules and the *best* ST_1.[2] This test gives $Q=48 > Q_{\alpha=0.01}(\varepsilon=0.5,\nu=2)=14.2$, implying the performance of the best ST_1 differs significantly from 100%, the performance afforded by the worst BT_1 and ET_1. From this outcome, and that of the first Q-test, it is inferred that the performances of *all* speaker JK's ST_1s differ significantly from 100%. Since the recognition performances of these modules are *worse* than those of the BT_1s and ET_1s (worse than 100%), it is concluded that speaker JK's BT_1s and ET_1s afford significantly better recognition performances than his ST_1s when used for *speaker-dependent closing diphthong recognition.*

The same conclusion is also reached for speaker HD's traditional expert modules. In this case, the first (full) Q-test gives $Q=5830 > Q_{\alpha=0.01}(\varepsilon=0.019,\nu=149)=579$, while the second (partial) Q-test gives $Q=28 > Q_{\alpha=0.01}(\varepsilon=0.5,\nu=2)=14.2$. Notably, the poorer recognition performances of the ST_1s tested are caused by *interposed errors* in the response-sequences produced by their component sequence-token TDNNs, as depicted in Figure 5.1.3.1-1. Ideally, an ST_1's component sequence-token TDNN should produce a response-sequence (part (a)) that, when collapsed (part (b)), may be matched to a reference sequence corresponding to a closing diphthong (in this example *0-1-2* corresponding to /ai/). If an interposed error occurs (part (d)), the collapsed response-sequence resulting (part (e)) may no longer contain a sequence corresponding to the desired phoneme. This in turn causes the matcher to produce an omission error (implying recognition failure) and may also lead to a false-positive error as depicted in Figure 5.1.3.1-1 (f). As §5.1.4 discusses, the detrimental effects of interposed errors like that in Figure 5.1.3.1-1 (d), may be alleviated by using squads of sequence-token TDNNs.

For speakers JK and HD, Table 5.1.3.1-2 lists statistics concerning the false-positive error performances of their BT_1s, ET_1s and ST_1s. Once again, these statistics were measured using the closing diphthong syllable utterances not used for network training (240 utterances per speaker containing 600 phoneme realizations, of which 360 are not closing diphthong realizations). The false-positive error performances listed in Table 5.1.3.1-2 indicate the abilities of the various traditional expert modules tested to "ignore" phoneme realizations corresponding to the voiced plosives /b/, /d/ and /g/ and intervals of "silence" also present in the test utterances (see §3.1.1).

---

[2]Using the original version of Cochran's Q-test, the proportions (recognition performances) that are significantly different may be identified by *partitioning* Q into separate components, each of which measures a specified source of variability (Fleiss 1973). Unfortunately, however, the same technique has not yet been extended to the *generalized* version of Cochran's Q-test discussed in §A3.1. The alternative to partitioning used in this thesis is *biased* in favour of finding no significant difference, since proportions exhibiting the *least* differences are compared. Such biasing is acceptable when the differences being compared are large, leading to the rejection of the null hypothesis of equal proportions.

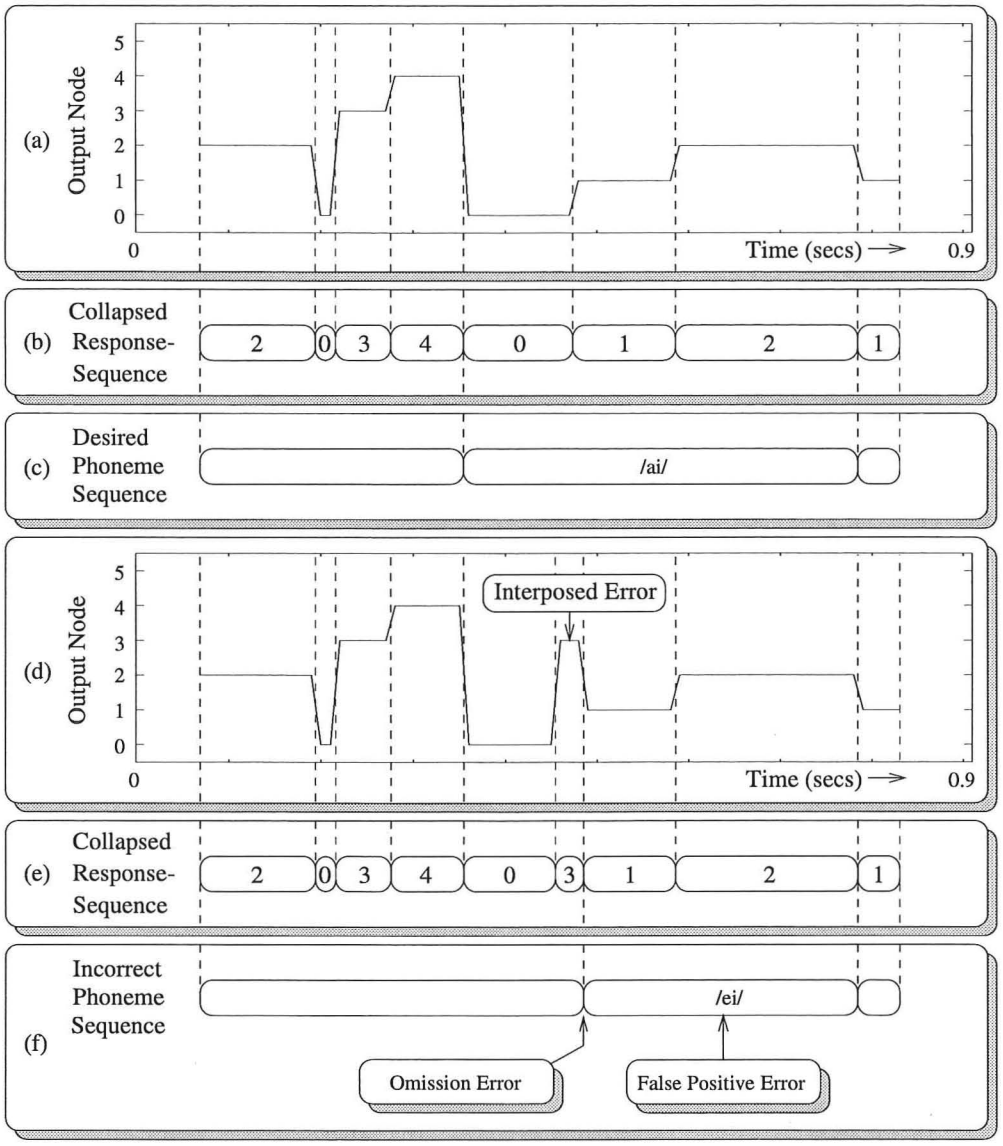

Figure 5.1.3.1-1. The manner by which interposed errors may cause traditional expert modules comprising individual sequence-token TDNNs to fail. Parts (a) through (c) show the desired behaviour of such a module and its component sequence-token TDNN. When the latter produces an interposed error, as in part (d), the phoneme sequence produced by the former (part (f)) may not resemble that desired (part (c)).

To permit a fair comparison of the false-positive error performances listed in Table 5.1.3.1-2, it is necessary to first normalize each performance with respect to the number of tokens processed by its associated expert module. Due to the short, isolated test utterances used to estimate these performances, the BT_1s and ST_1s process considerably more input tokens than the ET_1s, since the latter uses 30 instead of 15 slice tokens (see §4.2.3.1). Consequently, the BT_1s and ST_1s produce more responses per test utterance (see Figures 5.1.4-1 through 5.1.4-3), implying they have a greater likelihood of producing potential false-positive errors. To counter this imbalance (which would be less pronounced if long continuous utterances had been used for testing), the false-positive error performances of the BT_1s, ET_1s

and ST_1s observed in this work are scaled by $1/N_{BT}$, $1/N_{ET}$ and $1/N_{ST}$, respectively, where N_{BT}, N_{ET} and N_{ST} are the numbers of tokens processed (and responses made) by each type of module. For speaker JK, N_{BT}=N_{ST}=17 200 and N_{ET}=13 600, while for speaker HD, N_{BT}=N_{ST}=17_312 and N_{ET}=13 712. Using these values, Figure 5.1.3.1-2 depicts the *normalized* sampling distributions of false-positive error performance associated with the 50 examples of each speakers three types of traditional expert module.

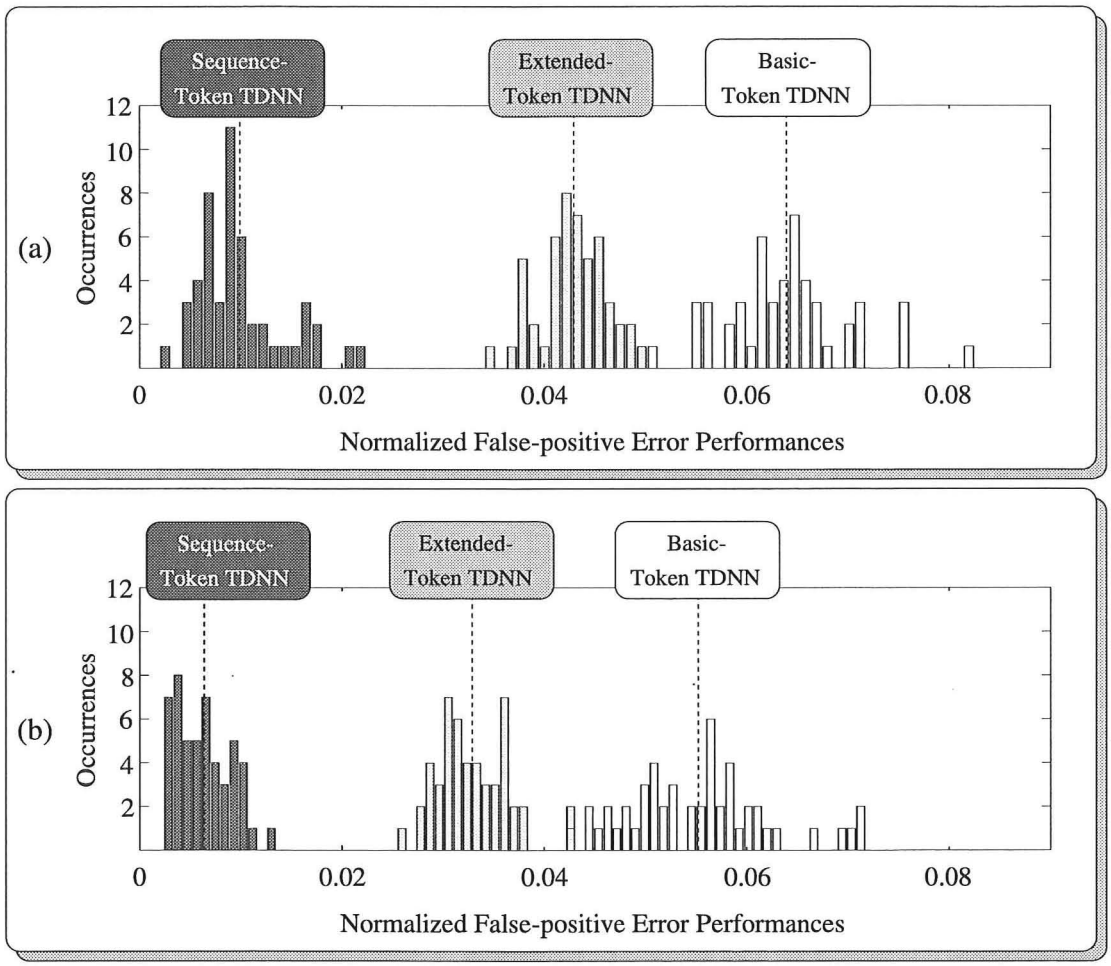

Figure 5.1.3.1-2. The sampling distributions of *normalized* false-positive error performance for (a) speaker JK's and (b) speaker HD's traditional expert modules comprising basic-, extended- and sequence-token TDNNs. The dashed vertical lines in this figure indicate mean false-positive error performance for each distribution.

Since each speaker's false-positive error performances displayed in Figure 5.1.3.1-2 do not arise from dichotomous expert module responses (more than one false-positive error may occur per test utterance), Cochran's generalized Q-test may not be used to compare them. Consequently, an alternative statistical test that does not assume dichotomous responses is required. In this thesis, the *Games-Howell test* discussed in §A3.2 is used to compare false-positive error performances. This test permits *multiple* pair-wise comparisons between an

arbitrary number of means, provided estimates of their variances are also available. The means compared may be estimated from different sized samples and their populations need not have the same variances. In this thesis, it is assumed that the mean false-positive error performances of expert modules tested are *positively correlated.* Evidence for this assumption is presented in §A3.2. Though intended for comparing independent means, the Games-Howell test is *conservative* when used in conjunction with positively correlated means (see §A3.2), implying only large differences in mean false-positive error performance are *likely* to be found significant. Since a more suitable (and powerful) statistical test does not yet exist to compare multiple correlated means, while assuming different sample sizes and population variances, the Games-Howell test must suffice.

(a) Speaker JK

Module Type	False-Positive Errors Per Expert Module			
	min.	mean	max.	standard deviation
Basic-Token (BT_1)	875	1100	1419	105
Extended-Token (ET_1)	478	584	681	44
Sequence-Token (ST_1)	34	170	368	73

(b) Speaker HD

Module Type	False-Positive Errors Per Expert Module			
	min.	mean	max.	standard deviation
Basic-Token (BT_1)	745	957	1242	122
Extended-Token (ET_1)	352	451	582	49
Sequence-Token (ST_1)	41	110	223	49

Table 5.1.3.1-2. Statistics concerning the false-positive error performances of traditional expert modules comprising individual basic-, extended- and sequence-token TDNNs when processing (a) speaker JK's and (b) speaker HD's closing diphthong syllables (syllables *not* used for network training). For each speaker, the utterances processed contain 600 phoneme realizations, of which 360 are realizations of /b/, /d/ and /g/.

Table 5.1.3.1-3 lists the results of Games-Howell tests comparing the normalized mean false-positive error performances of each speaker's traditional expert modules (these are presented following the method used in Sokal and Rolhf 1981). Each number below the diagonal is associated with one *pairing* of the three expert modules being compared and indicates the difference between their normalized mean false-positive error performances. The

numbers above the diagonal correspond to the *minimum significant differences* (*MSD*s, see equation (A3.2-5)) evaluated by the Games-Howell tests. Each *MSD* corresponds to the pair-wise mean difference diagonally opposite it, and indicates the minimum difference considered statistically significant when $\alpha=0.01$. For convenience, the pair-wise mean differences which exceed their associated *MSD* are marked with an asterisk. For differences so marked, the null hypothesis that the pair of normalized mean false-positive error performances compared are the equal, is *rejected* with a significance level of $\alpha=0.01$.

(a) Speaker JK

	BT_1	ET_1	ST_1
BT_1	-	0.005	0.005
ET_1	0.021*	-	0.004
ST_1	0.054*	0.033*	-

(a) Speaker HD

	BT_1	ET_1	ST_1
BT_1	-	0.006	0.005
ET_1	0.022*	-	0.003
ST_1	0.049*	0.027*	-

Table 5.1.3.1-3. Results of Games-Howell tests comparing the *normalized* mean false-positive error performances of each speaker's traditional expert modules for closing diphthong recognition. The numbers below the diagonal are the *normalized* pair-wise mean differences observed, while the numbers above the diagonal are the *minimum significant differences* (*MSD*s, see equation (A3.2-5)) evaluated by the Games-Howell test with $\alpha=0.01$. The pair-wise differences that exceed their associated *MSD* (diagonally opposite) are marked with an asterisk to indicate that they are statistically significant.

For each speaker, the results listed in Table 5.1.3.1-3 indicate that the mean false-positive error performances of the three types of expert module compared differ significantly from one another. From the relative positions of the distributions in Figure 5.1.3.1-2, it is, therefore, concluded that each speaker's ST_1s afford significantly better false-positive error performances than their ET_1s, which in turn afford significantly better performances than their BT_1s (in the context of processing closing diphthong syllable utterances like those discussed in §3.1.1).

The superior false-positive error performances of each speaker's ST_1s are attributed to their use of a limited set of *valid* reference sequences to represent closing diphthong phonemes, as discussed in §4.2.3.2. Consequently, for example, a speech portion that causes the component sequence-token TDNN of an ST_1 to produce the sequence 2-0-3-4 (see the first 0.35 seconds of Figure 5.1.3.1-1 (b)), does not cause any false-positive errors, since this

sequence, nor any of its sub-sequences, matches any of the reference sequences listed in Table 4.2.3.2-1. By contrast, this same speech portion may cause a BT_1 or ET_1 to produce one or more false-positive errors.

Comparing the performances of the traditional expert modules created for each speaker, those for speaker HD are slightly better in terms of recognition and/or false-positive error performances than those for speaker JK. In this thesis, no particular significance is attached to this finding since a sample of two speakers (one of each sex) is insufficient to analyze trends, such as those based on speaker sex. It is, however, reassuring to know that traditional expert modules for closing diphthong recognition function similarly for New Zealand English speakers of both sexes.

To summarise, the results presented in this section demonstrate that traditional expert modules comprising individual basic- and extended-token TDNNs afford significantly better recognition performances than their counterparts comprising individual sequence-token TDNNs, but at the cost of significantly worse false-positive error performances. Of the three types of traditional expert module tested, those comprising individual extended-token TDNNs afford the best performance *compromises*. These modules share the high recognition performances afforded by modules comprising basic-token TDNNs, but make significantly fewer false-positive errors than the latter modules. Interestingly, the traditional expert modules comprising basic-token TDNNs tested afford excellent recognition performances despite using short 15 slice tokens. Consequently, in contrast to Hataoka and Waibel's suggestion (see §4.2.1), improving false-positive error performance provides are stronger motive for using extended tokens to represent closing diphthong realizations, than improving recognition performance does.

5.1.4 The Performances of Squad-based Expert Modules

This section presents the performances obtained using expert modules comprising *squads* of basic-, extended- and sequence-token TDNNs for closing diphthong recognition. Such squads were originally motivated by observations of TDNN responses like those shown in Figures 5.1.4-1, 5.1.4-2 and 5.1.4-3. These figures depict the responses of *all* speaker JK's basic-, extended- and sequence-token TDNNs, respectively, to one of his utterances of the word *bide* (/baid/). Part (a) of each figure shows the raw formant tracks for this utterance. Part (b) of each figure shows the response-sequences for all 50 examples of the TDNNs associated with it (these are superimposed upon one another). Finally, part (c) of each figure shows the mean and range of activations for the most active output nodes of all 50 TDNN examples. In each figure, the dashed vertical lines in part (a) again delimit the extent of an aligned test token(s) representing the realization of /ai/ in speaker JK's utterance of *bide*.

From Figures 5.1.4-1 through 5.1.4-3, the following may be observed. First, for each type of TDNN, the 50 response-sequences depicted coincide for tokens *resembling* the aligned test tokens, implying *complete agreement* about the classifications of these tokens. With the exception of the second interval of such agreement for the basic-token TDNNs (that for /ei/ in Figure 5.1.4-1 (b)), the intervals of complete agreement depicted correspond to *desired* responses for the closing diphthong realization processed. Specifically, the basic- and extended-token TDNNs all respond most actively with the output node signifying /ai/, while the sequence-token TDNNs all produce the *sequence* of most active outputs *0-1-2*, which also signifies /ai/ (see Table 4.2.3.2-1). Second, during the intervals of complete agreement depicted in Figures 5.1.4-1 through 5.1.4-3, the activations associated with the most-active outputs for all 50 TDNN examples shown, tend to 0.9, the activation desired for tokens representing /ai/ during training. This behaviour is apparent for all three TDNN approaches, however, as indicated by the traces for the extended-token TDNNs (Figure 5.1.4-2 (c)), may be less pronounced than the coincidence of TDNN response-sequences.

Third, during intervals where network agreement is not complete, the activations of

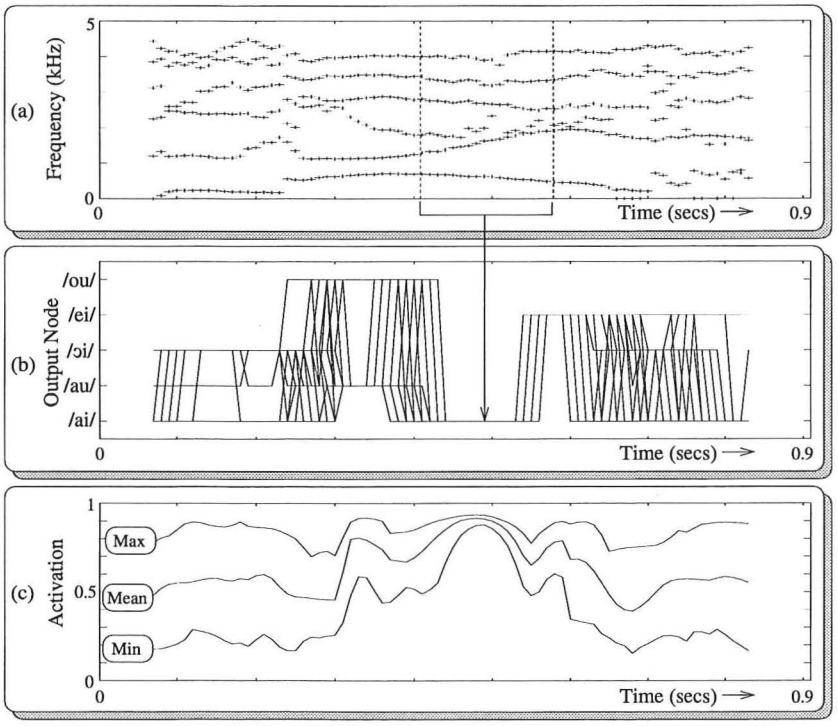

Figure 5.1.4-1. The responses of all speaker JK's 50 basic-token TDNNs to an utterance of the word *bide* (/baid/). For reference, part (a) shows the raw formant tracks for this utterance whose signal is depicted in part (a) of Figure 5.1.2-1. Part (b) shows the response-sequences for all 50 TDNNs. These are coincident (*agree completely*) in the vicinity of the aligned test token (the region delimited by the dashed lines in part (a)) whose centre is indicated by the arrow joining parts (a) and (b). During the interval of complete agreement, the activation of the most-active output (part (c)) approaches 0.9, the magnitude desired for tokens representing /ai/ during training. During the remainder of the utterance, the 50 networks often *disagree* and their most active output nodes may be highly active, as indicated by the *mean* and *max* curves in part (c).

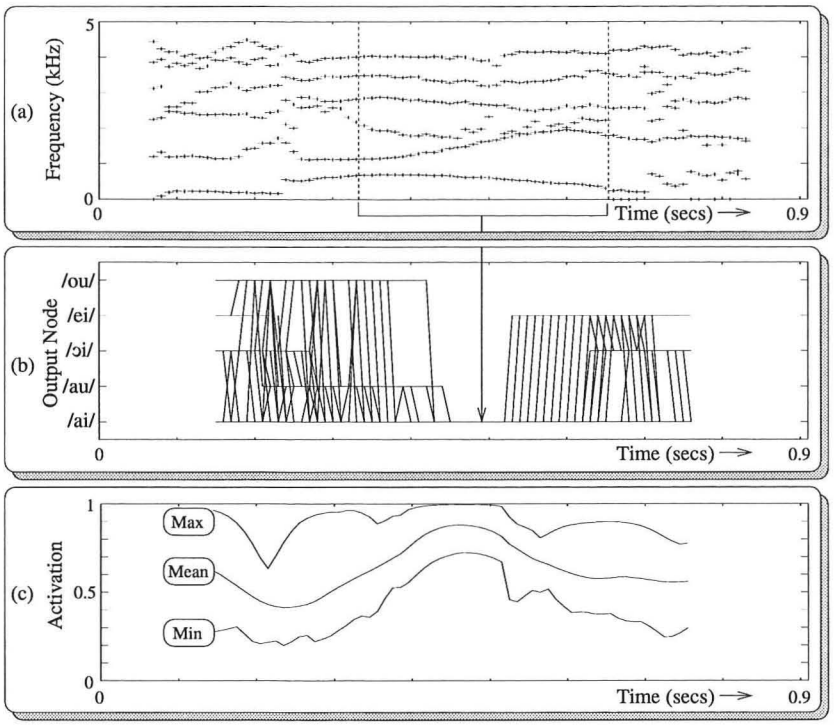

Figure 5.1.4-2. The same as Figure 5.1.4-1 except the traces are for 50 extended-token TDNNs.

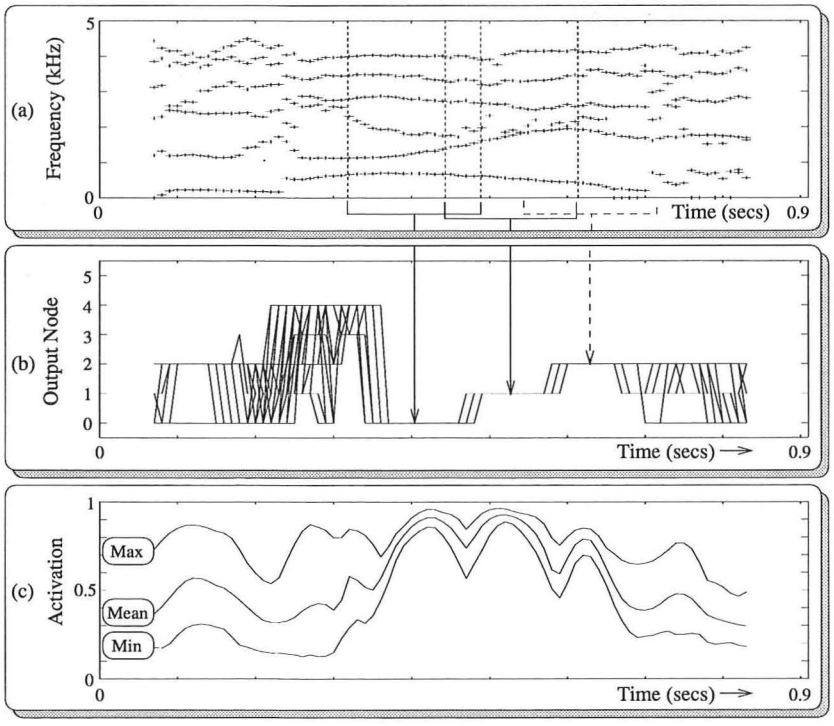

Figure 5.1.4-3. The same as Figure 5.1.4-1 except traces are for 50 sequence-token TDNNs. Note that only the solid line arrows joining parts (a) and (b) correspond to aligned test tokens extracted from the utterance of *bide* processed for this example, while the dashed-line arrow indicates the approximate spectral content of the tokens associated with the final region of complete network agreement.

the most active output nodes tend to be more variable. In particular, the most active output nodes of certain TDNN examples are highly active during such intervals, which often correspond to realizations of phonemes other than the closing diphthongs. Such activity is undesirable when these examples are used individually to form traditional expert modules, since it is likely to cause false-positive errors (see §4.2.2). Unfortunately, high activations in response to inappropriate input are exhibited by all 50 of each speaker's basic-, extended- and sequence-token TDNNs, though the input eliciting such activations differs from one TDNN example to the next. Consequently, traditional expert modules comprising such TDNNs are unlikely to perform like an ideal expert module for closing diphthong recognition (see §4.2.2).

From Figures 5.1.4-1 through 5.1.4-3, it is apparent that basic-, extended- and sequence-token TDNN examples *might* be combined to form squads of like networks to correctly classify input tokens representing closing diphthong realizations, while "ignoring" tokens representing other phoneme realizations or "silence". In this work, the response-

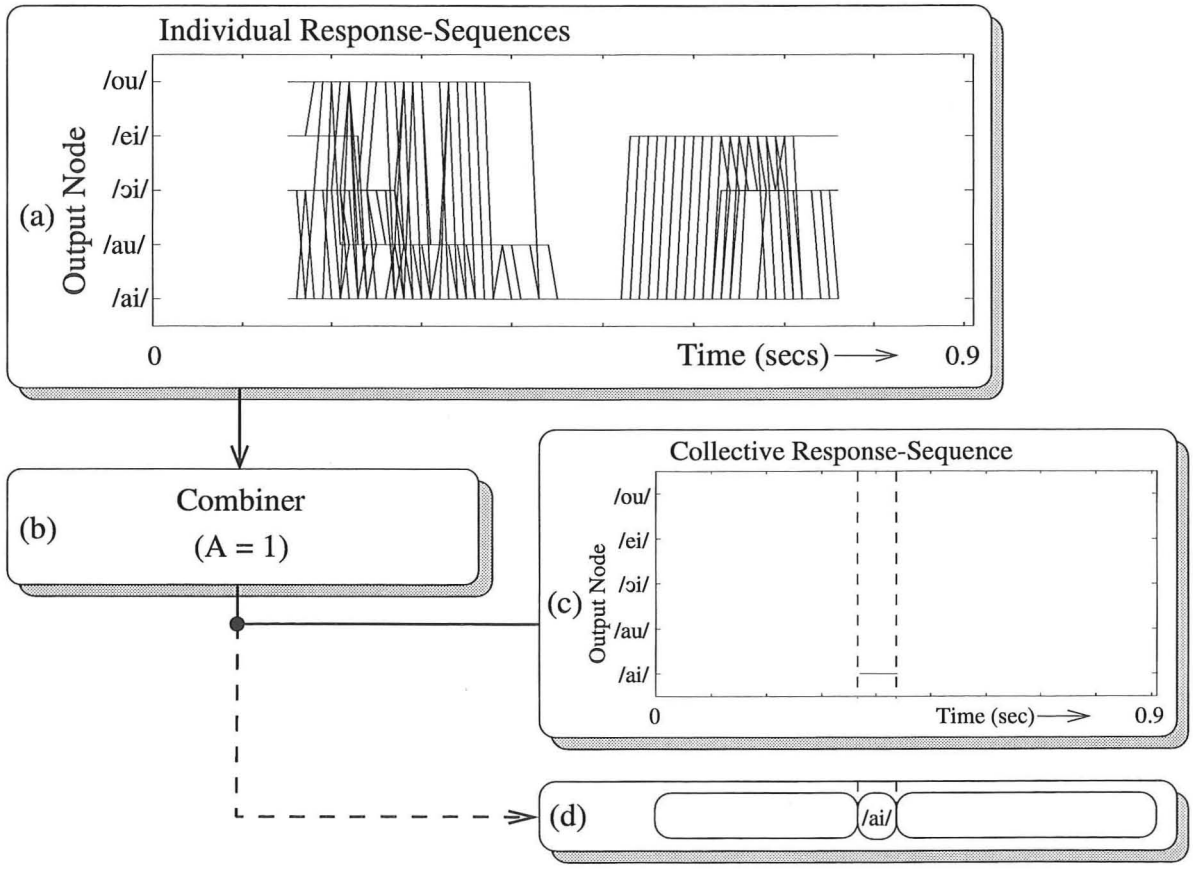

Figure 5.1.4-4. The method used to derive phoneme sequences for use with the isolated-test method from the individual response-sequences of component extended token TDNNs forming a squad-based expert module (phoneme sequences are derived similarly for squad-based expert modules comprising basic-token TDNNs). Part (a) shows the individual TDNN response-sequences, which when processed by the combiner (part (b)), are transformed into the collective response-sequence shown in part (c). The latter response is transformed into a phoneme sequence (part (d)) using the same method as for traditional expert modules comprising basic- or extended-token TDNNs (see Figure 5.1.2-2).

sequences produced by the component TDNNs of a squad are combined to form *collective response-sequences* by a *combiner* (see Figure 4.2.3-1) that utilizes the *selective-system voting rule* discussed in §4.3. Figures 5.1.4-4 (c) and 5.1.4-5 (c) show the collective response-sequences that result when such a combiner is applied to the individual extended- and sequence-token TDNN response-sequences shown in parts (a) of these figures, respectively (collective response-sequences similar to that in Figure 5.1.4-4 (c) are observed for basic-token TDNN squads also). In these examples, the agreement threshold, A, is set to *1*, implying *complete agreement* is required to produce a non-*null* collective response. As shown in Figures 5.1.4-4 (c) and 5.1.4-5 (c), this setting rejects all but the intervals of complete agreement between the response-sequences combined.

Apart from showing the formation of collective response-sequences, Figures 5.1.4-4 and 5.1.4-5 demonstrate the method used to derive phoneme sequences from squad-based

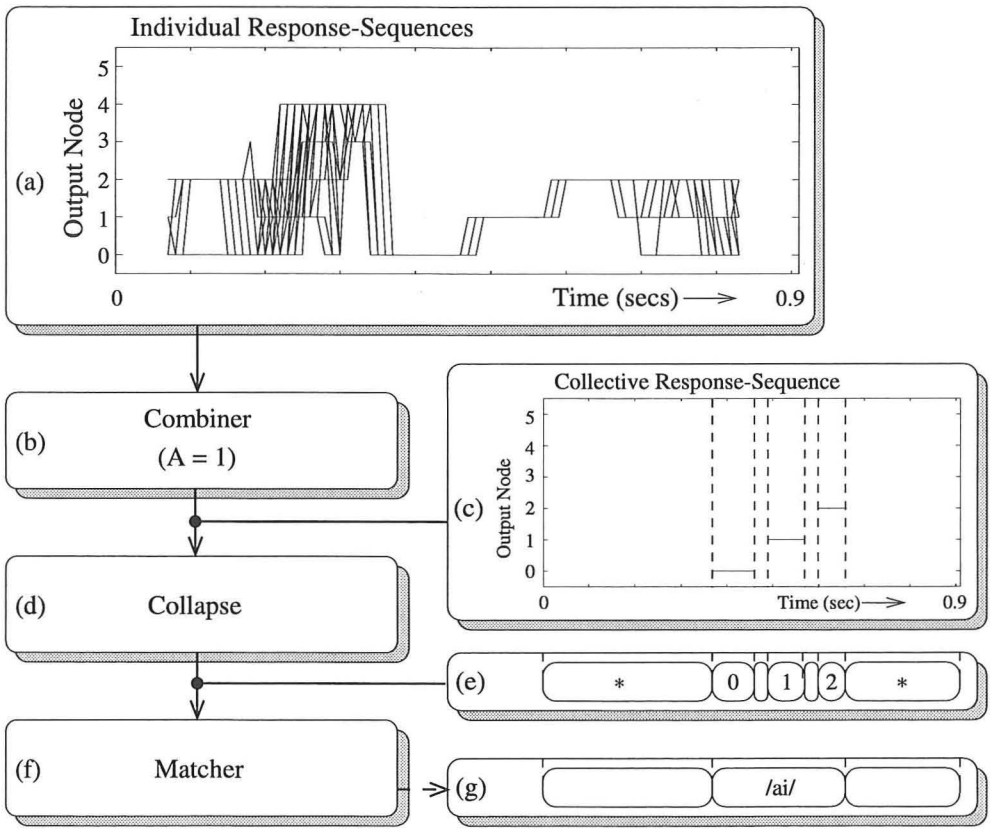

Figure 5.1.4-5. The method used to derive phoneme sequences for use with the isolated-test method from the individual response-sequences of component sequence-token TDNNs forming a squad-based expert module. Part (a) shows the individual TDNN response-sequences, which when processed by the combiner (part (b)), are transformed into the collective response-sequence shown in part (c). The latter response is collapsed (part (d)) and the resulting sequence (part (e)) transformed by the module's matcher (part (f)) to produce its phoneme sequence (part (g)). During matching, short intervals of *null* classification in a collective response-sequence are ignored, while the longer intervals of *null* classification (marked by the asterisks) act as boundaries over which matching sequences may not straddle.

expert modules for use with the isolated-test method (these figures were produced using speaker JK's TDNNs and the same utterance of the word *bide* used to produce Figures 5.1.2-2 and 5.1.2-3). For squad-based expert modules comprising basic- or extended-token TDNNs, phoneme sequences are derived from their collective response-sequences using the same method of collapsing like responses into single phonemic symbols, as is used with their traditional expert module counterparts (as before, the blank elements of the phoneme sequence shown in Figure 5.1.4-4 (d) correspond to *null* classifications). For squad-based expert modules comprising sequence-token TDNNs, phoneme sequences are derived using the same approach as for their traditional expert module counterparts, with one exception; during sequence matching, short intervals of *null* classifications (≤5 responses) are ignored, while longer intervals (>5 responses) are treated as boundaries over which matching sequences may not straddle. In Figure 5.1.4-5 (e), the longer intervals of *null* classification in the collapsed response-sequence shown, are marked with asterisks to indicate their special status.

As Figures 5.1.4-4 and 5.1.4-5 show, combining TDNNs into squads may be used to suppress many of the highly active and inappropriate responses made by such networks individually (see Figures 5.1.4-1 through 5.1.4-3). Consequently, expert modules formed from squads of TDNNs may produce phoneme sequences that better resemble those of an ideal expert module, than those produced by traditional expert modules. In particular, squad-based expert modules for closing diphthong recognition may be trained to produce phoneme sequences that contain non-*null* elements (non-blank elements) only in the vicinities of closing diphthong realizations, as shown in Figures 5.1.4-4 and 5.1.4-5.

In this work, the agreement threshold, A, is set to *1* for all squad-based expert modules discussed. As a consequence of the perfect classification performances of the component basic-, extended- and sequence-tokens TDNNs forming these squads (see A1.1.1.1 and A1.1.1.2), this setting permits good rejection of inappropriate input, *without* reducing their recognition performances. Selecting an optimal value of A for use with squads of TDNNs generally is left as a subject for further research.

The next section discusses the performances of speaker-dependent squad-based expert modules when processing speaker JK's and speaker HD's closing diphthong syllable utterances. This is followed in §5.1.4.2 by a discussion of the false-positive error performances of these modules when processing the monophthong syllables uttered by speakers JK and HD (see §3.1.2).

5.1.4.1 Performance on Closing Diphthong Syllables

For speaker's JK and HD, Table 5.1.3.1-1 lists statistics concerning the recognition and false-positive error performances of expert modules comprising squads of basic-,

extended- and sequence-token TDNNs, referred to as BT_N, ET_N and ST_N, respectively (where N is the number of networks combined to form a squad). The performances of five expert modules comprising squads of 10 (randomly chosen) TDNN examples and one expert module comprising a squad of 50 examples, are listed for each speaker and TDNN approach. As in §5.1.3.1, these performances were measured using the closing diphthong syllable utterances *not* used in conjunction with network training (240 syllables per speaker).

(a) Speaker JK

Module Type	% Correct	False-Positive Errors Per Module			
		min.	mean	max.	standard deviation
Basic-Token (BT_{10})	100	357	407	437	33.1
Extended-Token (ET_{10})	100	53	93	134	33.8
Sequence-Token (ST_{10})	98.8	2	4	8	2.4
Basic-Token (BT_{50})	100	-	201	-	-
Extended-Token (ET_{50})	100	-	18	-	-
Sequence-Token (ST_{50})	99.7	-	1	-	-

(b) Speaker HD

Module Type	% Correct	False-Positive Errors Per Module			
		min.	mean	max.	standard deviation
Basic-Token (BT_{10})	100	279	390	496	99.3
Extended-Token (ET_{10})	100	158	184	216	30.0
Sequence-Token (ST_{10})	100	0	2	4	1.5
Basic-Token (BT_{50})	100	-	208	-	-
Extended-Token (ET_{50})	100	-	84	-	-
Sequence-Token (ST_{50})	100	-	0	-	-

Table 5.1.4.1-1. The recognition (% correct) and false-positive error performances of expert modules comprising squads of basic-, extended- and sequence-token TDNNs containing 10 TDNNs (five modules tested) and 50 TDNNs (one module tested) when processing each speaker's closing diphthong syllables (syllables not used for network training).

With the exception of speaker JK's ST_{10}s and ST_{50}, all the expert modules tested recognized 100% of the closing diphthong realizations present in the utterances processed.

Speaker JK's ST_{10}s recognized 98.8% of these realizations on average[3], while this speaker's ST_{50} recognized 99.7%. Comparing these performances with those of speaker JK's other squad-based expert modules using Cochran's generalized Q-test, reveals that the recognition performances of all these modules are not significantly different. In particular, comparing the ST_{10}s with the BT_{10}s and ET_{10}s (15 modules in total), a Q-test gives Q=46.1 < $Q_{\alpha=0.01}(\varepsilon=0.1132,\nu=14)$=72.5, while comparing the ST_{50} with the BT_{50} and ET_{50} (3 modules in total), Q=2 < $Q_{\alpha=0.01}(\varepsilon=0.1132,\nu=2)$=13.3. Consequently, it is concluded that *all* the squad-based expert modules tested afford the same recognition performance.

Comparing the recognition performances of each speaker's squad-based expert modules (Table 5.1.4.1-1) with their traditional expert module counterparts (Table 5.1.3.1-1), it is apparent that only the performances of their ST_{10}s and ST_{50} differ from those of their ST_1s. The significance of these differences may be tested for each speaker by comparing the *best* ST_1 with the worst ST_{10} using a Q-test. Comparing these modules for speaker JK gives Q=12.5 > $Q_{\alpha=0.01}(\varepsilon=1,\nu=1)$=6.6. From this outcome, it is inferred that the recognition performances of *all* speaker JK's ST_{10}s and ST_1s are significantly different. The same inference may also be made for this speaker's ST_{50}, since its recognition performance exceeds that of his worst ST_{10}. Similarly, comparing speaker HD's worst ST_{10} with this speaker's best ST_1 gives Q=14 > $Q_{\alpha=0.01}(\varepsilon=1,\nu=1)$=6.6, implying the inferences made for speaker JK also apply to speaker HD. Consequently, considering the relative recognition performances of each speaker's ST_1s, ST_{10}s and ST_{50}, it is concluded that their ST_{10}s and ST_{50}s afford significantly better recognition performances than their ST_1s, when attempting to recognize their closing diphthong realizations. The high (near perfect) recognition performances of the ST_{10}s and ST_{50}s tested, indicate that the use of squads in conjunction with the selective-system voting rule (see §4.3) may effectively eliminate the interposed errors produced by their component sequence-token TDNNs (see Figure §5.1.3.1-1). Such errors are not present within the *collective response-sequences* of these squad-based expert modules, since not all component sequence-token TDNNs make the same interposed errors at the same times. Consequently, the interposed errors produced by the individual component sequence-token TDNNs of a squad are "ignored" as a result of network disagreement concerning these responses.

As in §5.1.3.1-1, the mean false-positive error performances of the various squad-based expert modules must also be normalized to permit fair comparisons of these performances. The scale factors used for this normalization are identical to those discussed in §5.1.3.1-1, since (in the worst case) the component networks of a squad may produce unanimous responses that alternate between incorrect object indices when processing a series of tokens.

Table 5.1.4.1-2 presents a full list of Games-Howell test results obtained when

---

[3]For these expert modules, the minimum performance was 97.9%, the maximum performance was 99.2% and the standard deviation was 0.51%.

comparing the *normalized* mean false-positive error performances of each speaker's traditional and squad based expert modules (only the modules comprising squads of 10 TDNN examples are considered, since estimates of the performance variation associated with each speaker's BT_{50}, ET_{50} and ST_{50} are unavailable). Comparing the BT_{10}s, ET_{10}s and ST_{10}s amongst themselves first, the normalized mean false-positive error performances of speaker JK's ST_{10}s and ET_{10}s are found to be significantly different from this speaker's BT_{10}s, but not significantly different from one another. In contrast, the exact opposite results are found for speaker HD's BT_{10}s, ET_{10}s and ST_{10}s, due mainly to the largish variation associated with the false-positive error performances of this speaker's BT_{10}s (see Table §5.1.4.1-1). In particular, only her ST_{10}s and ET_{10}s are found to afford significantly different false-positive error performances. From Table 5.1.4.1-1, it is apparent that both speaker's ST_{10}s afford the best false-positive error performances of the expert modules comprising squads of 10 TDNN examples tested. Regrettably, however, there is only partial statistical evidence to support the conclusion that the ST_{10}s afford significantly better false-positive error performances than the other two types of expert modules.

Comparing the normalized mean false-positive errors of each speaker's traditional

(a) Speaker JK

	BT_1	ET_1	ST_1	BT_{10}	ET_{10}	ST_{10}
BT_1	-	0.005	0.005	0.007	0.009	0.004
ET_1	0.021*	-	0.004	0.008	0.011	0.002
ST_1	0.054*	0.033*	-	0.007	0.010	0.003
BT_{10}	*0.040**	0.019*	0.014*	-	0.016	0.009
ET_{10}	0.057*	*0.036**	0.003	0.017*	-	0.012
ST_{10}	0.064*	0.043*	*0.010**	0.023*	0.007	-

(b) Speaker HD

	BT_1	ET_1	ST_1	BT_{10}	ET_{10}	ST_{10}
BT_1	-	0.006	0.005	0.025	0.007	0.005
ET_1	0.022*	-	0.003	0.028	0.007	0.003
ST_1	0.049*	0.027*	-	0.027	0.008	0.002
BT_{10}	*0.033**	0.010	0.016	-	0.029	0.027
ET_{10}	0.042*	*0.020**	0.007	0.009	-	0.009
ST_{10}	0.055*	0.033*	*0.006**	0.022	0.013*	-

Table 5.1.4.1-2. Results of Games-Howell tests comparing the *normalized* mean false-positive error performances of (a) speaker JK's and (b) speaker HD's traditional and (small) squad-based expert modules when processing their associated speaker's closing diphthong syllable utterances.

expert modules with their counterparts comprising squads of 10 TDNNs (see the italicized entries in Table 5.1.4.1-2), it is evident that these are significantly different, *irrespective of component TDNN type.* Consequently, from the relative performances of these modules, it is concluded (for both speakers) that the BT_{10}s, ET_{10}s and ST_{10}s tested afford significantly better false-positive error performances than their traditional expert module counterparts, when processing closing diphthong syllable utterances like those discussed in §3.1.1. Since the false-positive error performances of each speaker's BT_{50}, ET_{50} and ST_{50} are better than *any* of their BT_{10}s, ET_{10}s and ST_{10}s, respectively, the same conclusion is also likely when comparing the larger squad-based and traditional expert modules, though there is insufficient evidence to confirm this statistically.

The superior false-positive error performances of the expert modules comprising the larger squads of 50 TDNN examples, is attributed to the greater probability of combining (randomly chosen) networks that disagree about inappropriate input (see §4.3). The false-positive error performances associated with each speaker's BT_{50}, ET_{50} and ST_{50} provide an indication of what might be achieved by smaller squads whose component networks are optimally trained for use in a squad. §A1.1.2.1 and §A1.1.2.2 give detailed breakdowns of the potential false-positive errors produced by speaker JK's and speaker HD's BT_{50}, ET_{50} and ST_{50}, respectively (similar patterns were observed for the modules comprising 10 TDNN examples also). Of greatest interest are the false-positive errors associated with diphthongs /ai/ and /ei/. Each speaker's BT_{50} frequently produced the phoneme sequence */ai/-/ei/* in response to realizations of *both* these phonemes, implying one correct response and one false-positive error. Unfortunately, on occasion, both elements of this *ambiguous* phoneme sequence were produced in conjunction with highly active (expert module) output nodes, providing no reliable cue as to which element was false. Consequently, it is likely that not all ambiguous /ai/-/ei/ sequences produced by a BT_{50} in response to realizations of /ai/ or /ei/, may be corrected by an arbitration module. Unfortunately, the ambiguity created by this sequence may, therefore, be conveyed to subsequent levels of linguistic processing, like morpheme recognition where, for example, it could make distinguishing between morphemes such as *bide* and *bade* difficult.[4]

Since /ai/ and /ei/ are perhaps the most frequently used closing diphthongs in New Zealand English (see §2.4), the ambiguity created by each speaker's BT_{50} is extremely undesirable. Though such ambiguity might be corrected during subsequent processing, the performances of each speaker's ST_{50} indicate that it need not arise in the first place. In comparison to their BT_{50}, each speaker's ET_{50} produces far fewer ambiguous /ai/-/ei/ phoneme sequences, however, these are not entirely eliminated. Only each speaker's ST_{50}, like their

[4]In the case of *bade* and *bide*, ambiguity caused by the phoneme sequence */b/-/ai/-/ei/-/d/* would have to be corrected at the semantic level, since both words are verbs.

ST_{10}s, were observed to produce no ambiguous /ai/-/ei/ sequences.

To aid summarizing the performance results presented in this section and §5.1.3.1, Figure 5.1.4.1-1 depicts these performances graphically with separate axes corresponding to recognition and normalized false-positive error performances. The lines associated with the various types of expert module compared (where visible) correspond to the range of performances observed when processing their associated speaker's closing diphthong syllable utterances. Comparing the traditional expert modules with squad-based modules of the same type, the squad-based modules afford identical or significantly better recognition performances compared to the traditional expert modules, while affording significantly better false-positive error performances. Consequently, it is concluded that squad-based expert modules are more

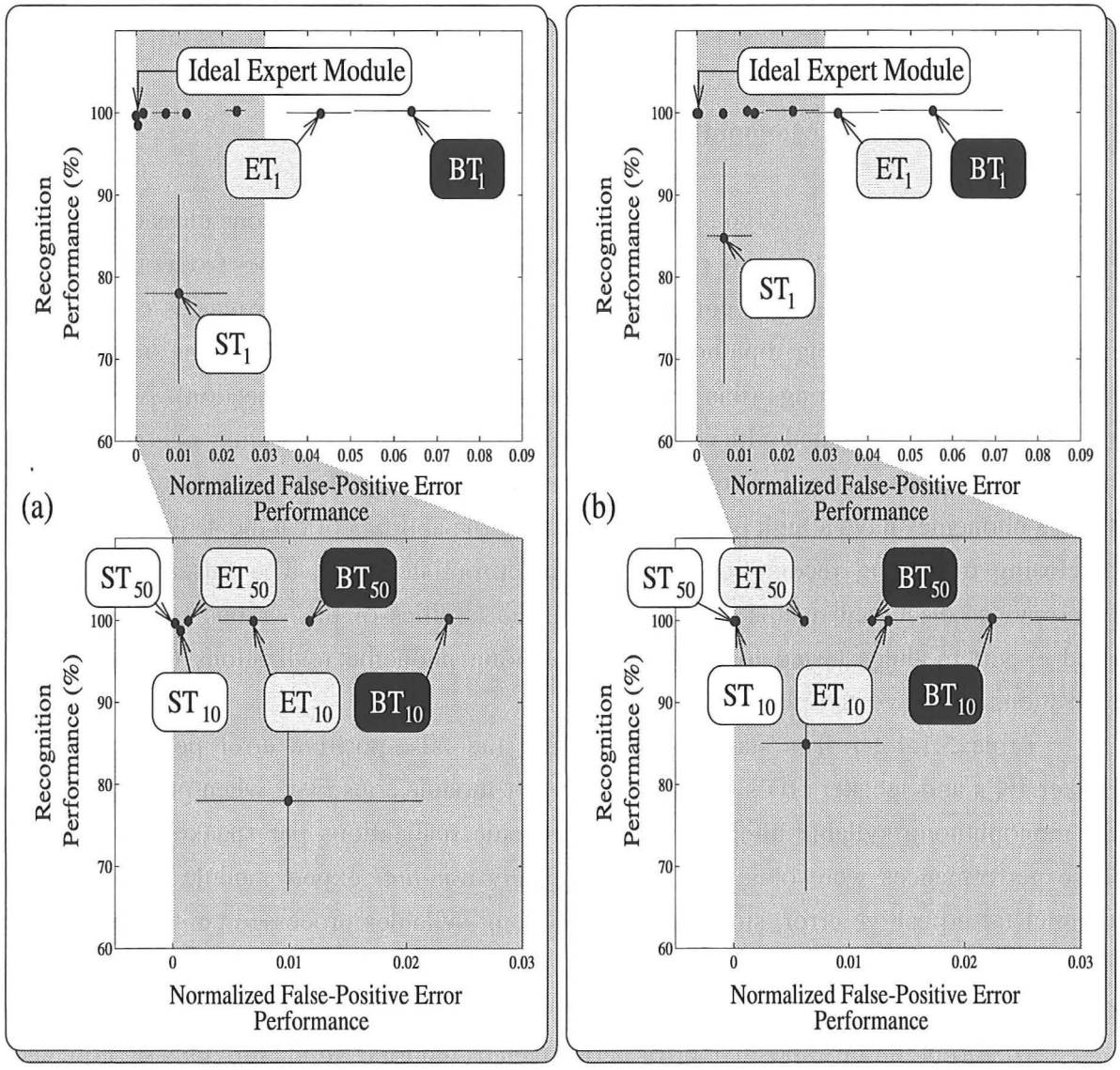

Figure 5.1.4.1-1. A summary of the performance results presented in this section and §5.1.3.1 for (a) speaker JK and (b) speaker HD.

suitable for closing diphthong recognition, since their performances better approximate those of an ideal expert module.

Comparing the various types of squad-based expert modules tested, it is apparent that these differ mainly in their false-positive error performances, since all afford very similar recognition performances. The *consistently* better false-positive error performances of the squad-based expert modules comprising sequence-token TDNNs provides a strong motive for preferring these modules for closing diphthong recognition. Of all the modules tested, the ST_{50}s afford the best approximations to ideal expert modules for closing diphthong recognition, recognizing all (or nearly all) of the closing diphthong realizations they processed, while making nearly no potential false positive errors. Each speaker's ST_{10}s afforded performances only marginally worse than their ST_{50}, while requiring only a *fifth* of the computational effort.

5.1.4.2 Performance on Monophthong-Syllables

As discussed in §4.2.2, an expert module designed to recognize one class of phoneme realizations should, ideally, ignore realizations corresponding to phonemes from other classes. Consequently, it is desirable to compare the false-positive error performances of candidate expert modules for closing diphthong recognition in response to realizations of phonemes other than closing diphthongs (this was done partially in the previous section). As discussed in §2.3, the glides associated with certain closing diphthong realizations may contain qualities characteristic of the realizations of one or more monophthongs. For this reason, realizations of these phonemes have a high priority when testing the abilities of candidate expert modules for closing diphthong recognition to ignore inappropriate input. The false-positive error performances discussed in this section indicate the abilities of the various types of squad-based expert modules tested to ignore monophthong phoneme realizations (in addition to realizations of the voiced plosives).

Table 5.1.4.2-1 lists statistics concerning the false-positive error performances of speaker JK's and speaker HD's squad-based expert modules observed when processing their 160 monophthong syllable utterances (400 phoneme realizations per speaker; see §3.1.2). These performances were obtained by treating *any* non-*null* expert module response as a potential false-positive error, since the monophthong syllables processed contain no closing diphthong realizations (§3.1.2). As in §5.1.2, groups of identical non-*null* responses were collapsed to form *single* false-positive errors.

Mirroring the false-positive error performances reported in the previous section, each speaker's ST_{10}s and ST_{50} afford the best performances (for their respective squad sizes) when processing their associated speaker's monophthong syllable utterances. Table 5.1.4.2-2 lists

the Games-Howell test results obtained by comparing the normalized mean false-positive error performances afforded by each speaker's BT_{10}s, ET_{10}s and ST_{10}s. Once again, these performances were normalized using the scale factors discussed in §5.1.3.1 with N_{BT}=N_{ST}=9_146 and N_{ET}=6 746 for speaker JK and N_{BT}=N_{ST}=8 132 and N_{ET}=5 732 for speaker HD.

As Table 5.1.4.2-2 indicates, only the mean false-positive error performances of speaker JK's ST_{10}s and BT_{10}s differ significantly from one another. By contrast, the mean performance

(a) Speaker JK

Module Type	False-Positive Errors Per Module			
	min.	mean	max.	standard deviation
Basic-Token (BT_{10})	229	263	284	20.3
Extended-Token (ET_{10})	68	110	131	29.5
Sequence-Token (ST_{10})	36	46	61	11.8
Basic-Token (BT_{50})	-	165	-	-
Extended-Token (ET_{50})	-	36	-	-
Sequence-Token (ST_{50})	-	26	-	-

(b) Speaker HD

Module Type	False-Positive Errors Per Module			
	min.	mean	max.	standard deviation
Basic-Token (BT_{10})	200	246	284	36.1
Extended-Token (ET_{10})	108	136	156	21.9
Sequence-Token (ST_{10})	28	33	35	2.8
Basic-Token (BT_{50})	-	150	-	-
Extended-Token (ET_{50})	-	64	-	-
Sequence-Token (ST_{50})	-	20	-	-

Table 5.1.4.2-1. False-positive error performances for expert modules comprising squads of 10 or 50 basic-, extended- and sequence- TDNNs when processing (a) speaker JK's and (b) speaker HD's monophthong syllable utterances (160 utterances per speaker).

of speaker HD's ST_{10}s are significantly different from those of this speaker's BT_{10}s and ET_{10}s.

Consequently, from the relative performances of speaker HD's BT_{10}s, ET_{10}s and ST_{10}s, it is concluded that this speaker's ST_{10}s afford better false-positive error performance than her BT_{10}s and ET_{10}s, when processing her monophthong syllable utterances (see §3.1.2). For speaker JK, it may only be concluded that this speaker's ST_{10}s afford significantly better false-positive error performances than his BT_{10}s.

As observed when processing each speaker's closing diphthong syllable utterances (see Table 5.1.4.1-1), the squad-based expert modules comprising 50 TDNNs also afford the best false-positive error performances observed when processing each speaker's monophthong syllable utterances. In particular, each speaker's ST_{50} makes fewer than 30 potential false-

(a) Speaker JK

	BT_{10}	ET_{10}	ST_{10}
BT_{10}	-	0.015	0.007
ET_{10}	0.012	-	0.017
ST_{10}	0.024*	0.011	-

(a) Speaker HD

	BT_{10}	ET_{10}	ST_{10}
BT_{10}	-	0.016	0.016
ET_{10}	0.007	-	0.014
ST_{10}	0.026*	0.020*	-

Table 5.1.4.2-2. Results of Games-Howell tests comparing the normalized mean false-positive error performances of (a) speaker JK's and (b) speaker HD's various expert modules when processing their monophthong syllable utterances.

positive errors in response to the 400 phoneme realizations present in their monophthong utterances (160 of which are monophthong realizations). §A1.1.2.3 and §A1.1.2.4 give detailed breakdowns of the false-positive errors produced by the BT_{50}, ET_{50} and ST_{50} associated with speakers JK and HD, respectively. The majority of potential false-positive errors for speaker JK's BT_{50} and ET_{50} are caused by front or central monophthongs being detected as /ai/ or /ei/. For this speaker's ST_{50}, potential false-positive errors are predominantly detections of /ou/, due to the single element reference sequence used to represent this diphthong (see Table 4.2.3.2-1). By contrast, the majority of potential false-positive errors for speaker HD's BT_{50} and ET_{50} are caused by high front monophthong, or "silence", being detected as /ci/. This closing diphthong is also falsely detected by speaker HD's ST_{50}, mainly in response to realizations of the back rounded monophthongs. For both

speakers, *no one monophthong is primarily responsible for the false-positive errors observed.* This finding is attributed to the relatively even spread of diphthong-glides within the frequency space corresponding to monophthong realization qualities (see Figure 2.3-4 for example).

5.2 Multi-Speaker Experiments

As a consequence of the good results obtained with squad-based expert modules comprising sequence-token TDNNs, the properties of such modules for closing diphthong recognition were investigated further. In particular, important questions concerning the ability of such expert modules to handle utterances produced by multiple speakers and to handle noise corruption were examined. This section presents the results arising from experiments conducted to answer these questions. Since these results are exploratory, rather than comparative as in §5.1, they are not analyzed statistically.

The next section discusses the performances of *multi-speaker* expert modules comprising squads of sequence-token TDNNs observed when processing speaker JK's and speaker HD's closing diphthong syllables. This is followed in §5.2.2 by a discussion of similar performances observed when processing each speaker's monophthong syllables. Finally, §5.2.3 discusses the robustness of squad-based expert modules comprising sequence-token TDNNs when processing corrupted versions of each speaker's closing diphthong syllables.

5.2.1 Multi-Speaker Performance on Closing Diphthong Syllables

To test the ability of squad-based expert modules comprising sequence-token TDNNs to function with multiple speakers, 50 sequence-token TDNNs were trained using a combination of the tokens used previously for speaker-dependent training (a total of 290 training tokens derived from 80 of speaker JK's and 80 of speaker HD's closing diphthong syllable utterances, see §5.1.1). For convenience later in §5.2.3, these networks are referred to as the *CL* sequence-token TDNNs, where *CL* implies training in conjunction with "clean" speech utterances. Training details for the *CL* sequence-token TDNNs are given in §A1.2.1.1. These networks were trained using identical training tokens, but different initial weights. As for the speaker-dependent TDNNs trained in this work (see section 5.1.1), these weights were initialize with random *real numbers* lying in the range [-0.5, 0.5]. Compared to the speaker-dependent sequence-token TDNNs, the *CL* sequence-token TDNNs were slightly more difficult to train, requiring 177.2 epochs on average to train compared to 145 and 153.4 for

the speaker-dependent networks. In contrast to the speaker-dependent sequence-token TDNNs, not all of the *CL* sequence-token TDNNs achieved perfect classification performances during training. However, all the misclassifications made consisted of tokens representing state *2* associated with realizations of /ai/ (by speaker HD) being misclassified as state *1*, which does not hinder correct /ai/ recognition, since *0-1* is sufficient to signify this phoneme (see Table 4.2.3.2-1).

To permit comparison with the speaker-dependent results discussed in §5.1.4.1, five multi-speaker expert modules comprising squads of 10 sequence-token TDNNs (ST_{10}s) and one such module comprising a squad of 50 sequence-token TDNNs (ST_{50}) were tested. Table 5.2.1-1 list statistics concerning the recognition and false-positive error performances observed when processing the 480 closing diphthong syllable utterances (240 per speaker) not used for network training. As indicated by this table, the ST_{50} affords near perfect performances, while those of the ST_{10}s are only fractionally worse. Comparing the multi-speaker ST_{10} and ST_{50} performances with those of their speaker-dependent counterparts (see Table 5.1.4.1-1), they are practically identical. Note that the numbers of false-positive errors made by the multi-speaker ST_{50} and ST_{10}s (processing 480 syllables) are equal, or approximately equal on average, to the cumulative numbers of such errors made by their speaker-dependent counterparts (each processing 240 syllables).

(a) Squads of 10 Sequence-Token TDNNs (ST_{10})

Quantity	min.	mean	max.	standard deviation
% Correct	98.5	99.4	100	0.56
False-positive Errors	2	7.2	10	3.6

(b) Squad of 50 Sequence-Token TDNNs (ST_{50})

Quantity	Value
% Correct	100
False-positive Errors	1

Table 5.2.1-1. Recognition (% correct) and false-positive error performances for multi-speaker expert modules comprising (a) squads of 10 sequence-token TDNNs (five modules tested) and (b) a squad of 50 sequence-token TDNNs (one module tested) when processing both speaker JK's and speaker HD's 240 closing diphthong syllables (syllables not used for network training containing 1200 phoneme realizations in total, of which 720 are not closing diphthong realizations).

The results presented in this section demonstrate that multi-speaker expert modules comprising squads of sequence-token TDNNs may be created which perform comparably with

similar speaker-dependent modules (at least for two speakers). In particular, they show that such modules may be successfully trained to accommodate the large differences in average diphthong-glides evident between adult male and female speakers (particularly those associated with /ai/ and /ei/, see Figure 4.2.3.1-4).

5.2.2 Multi-Speaker Performance on Monophthong Syllables

Table 5.2.2-1 list statistics concerning the false-positive error performances observed while processing both speaker JK's and speaker HD's monophthong syllable utterances (160 utterances per speaker), using the ST_{10}s and ST_{50} discussed in the previous section. The ST_{50} makes 40 potential false-positive errors while processing the 800 phoneme realizations present in these utterances. This error count is slightly better than the 46 errors made cumulatively by the speaker-dependent ST_{50}s (see Table 5.1.4.2-1). By contrast, the multi-speaker ST_{10}s perform marginally worse than their speaker-dependent counterparts, producing 85 potential false-positive errors (on average) compared to a cumulative total of 79 errors (on average) for the latter (see Table 5.1.4.2-1).

(a) Squads of 10 Sequence-Token TDNNs (ST_{10})

Quantity	min.	mean	max.	standard deviation
False-positive Errors	73	85	99	9.3

(b) Squad of 50 Sequence-Token TDNNs (ST_{50})

Quantity	Value
False-positive Errors	40

Table 5.2.2-1. False-positive error performances for multi-speaker expert modules comprising (a) squads of 10 sequence-token TDNNs (five modules tested) and (b) a squad of 50 sequence-token TDNNs (one module tested) when processing both speaker JK's and speaker HD's 240 monophthong syllables (800 phoneme realizations in total, of which 320 are monophthong realizations).

Details of the false-positive errors made by the multi-speaker ST_{50} in response to speaker JK's and speaker HD's monophthong syllable utterances are given in §A1.2.2.1. Notably, false-positive detections of /ou/ constitute over half of the potential false-positive errors made, indicating that the sequence representing this phoneme (currently *-5-*, see Table 4.2.3.2-1) should be lengthened to reduce the likelihood of its "chance" occurrence in response to inappropriate input.

The results presented in this section demonstrate that training component sequence-token TDNNs for multiple speakers may not unduly hinder the ability of ST_{10}s and ST_{50}s for closing diphthong recognition to ignore monophthong (and voiced plosive) realizations.

5.2.3 Robustness to Noise Corrupted Closing Diphthong Syllables

Using the sequence matching approach depicted in Figure 4.2.3.2-3, the correct operation of the matcher within a squad-based expert module comprising a squad of sequence-token TDNNs relies on the production of uncorrupted sequences corresponding to a set of reference sequences (see Table 4.2.3.2-1). Consequently, it is desirable to know how the recognition and false-positive error performances of such modules are affected by noise corrupted speech utterances. This section discusses the performances of three multi-speaker ST_{50}s in response to speech utterances corrupted with band-limited *white noise*. Larger squads of 50 sequence-token TDNNs were tested because they provide the best indication of the performances that may be achieved by smaller squads whose component sequence-token TDNNs are optimized for use in a squad. White noise was selected as the source of corruption following the experimental approach used by Dawson and Sridharan (1992) to test TDNNs for speech enhancement.

The solid lines in Figure 5.2.3-1 indicate the recognition and false-positive error performances of the ST_{50} discussed in §5.2.1 (denoted $ST_{50(CL)}$), verses signal-to-noise ratio (SNR). Once again, these performances were estimated using the 480 closing diphthong syllables (240 utterances per speaker) not used for network training. Estimates of the recognition and false-positive error performances of the $ST_{50(CL)}$ were obtained using these utterances as they were recorded (denoted "CL" in Figure 5.2.3-1, implying "clean" speech utterances) and with additional white noise to give approximate SNRs of 0, 5, 10, 15, 20 and 30 dB. SNR was estimated for the M samples representing each utterance using

$$[SNR]_{dB} \approx 10\log_{10}\left[\frac{\sum_{m=1}^{M}\left(s(m)-\bar{s}\right)^2}{\sum_{m=1}^{M} n^2(m)}\right] \qquad (5.2.1\text{-}1)$$

(Elder 1992), where $s(m)$ and $n(m)$, m=1, 2,..., M, are speech and noise samples, respectively, and the mean speech level, $\bar{s}$, is given by

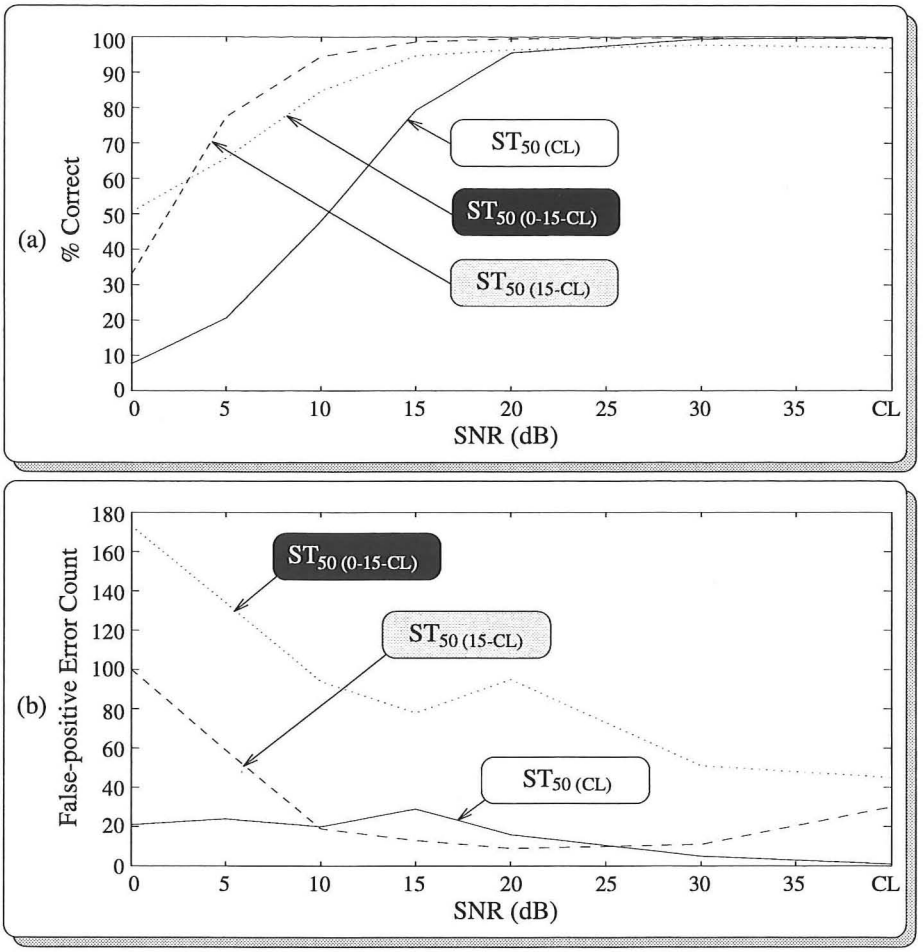

Figure 5.2.3-1. (a) Recognition (% correct) and (b) false-positive error performances of three expert modules comprising squads of 50 sequence-token TDNNs when processing speaker JK's and speaker HD's closing diphthong syllables corrupted with varying levels of white noise (varying SNR).

$$\bar{s} = \frac{\sum_{m=1}^{M} s(m)}{M} \qquad (5.2.1\text{-}2)$$

As Figure 5.2.3-1 (a) indicates, the recognition performance of the $ST_{50(CL)}$ decays gradually with decreasing SNR, until SNR falls below 20 dB, at which point it decays rapidly. In contrast, the number of potential false-positive errors made by $ST_{50(CL)}$ rises gradually with decreasing SNR and peaks at less than 30 errors. These performances, particularly the recognition performances, indicate that an ST_{50} trained to accommodate the variation in uncorrupted closing diphthong realizations produced by speakers JK and HD (a male and a female speaker; see Figure 4.2.3.2-2), may "tolerate" variation caused by white noise corruption provided SNR≥20 dB. When SNR falls below 20 dB, the variation between corrupted and uncorrupted closing diphthongs becomes too extreme and uncorrupted

realizations are treated as inappropriate input. Importantly, such realizations *are not* misclassified, but rather are "ignored" as indicated by the low number of potential false-positive errors made by the $ST_{50(CL)}$ for small values of SNR.

In an attempt to obtain improved recognition performance from a multi-speaker ST_{50} at low SNR, two further expert modules, $ST_{50(15\text{-}CL)}$ and $ST_{50(0\text{-}15\text{-}CL)}$, were created and tested. These modules were formed from separate sets of 50 sequence-token TDNNs referred to as the *15-CL* sequence-token TDNNs and the *0-15-CL* sequence-token TDNNs, respectively. All of these networks were trained in conjunction with the same closing diphthong realizations used to train the *CL* sequence-token TDNNs (160 realizations, see §5.2.1), however, tokens representing corrupted and uncorrupted versions of these realizations were used. In particular, the *15-CL* sequence-token TDNNs were trained in conjunction with "clean" versions and versions corrupted with white noise to give SNRs approximating 15 dB (implying 580 training tokens). Likewise, the *0-15-CL* sequence-token TDNNs were trained in conjunction with "clean" versions and versions corrupted to give SNRs approximating 0 and 15 dB (implying 870 training tokens). §A1.2.1.2 and §A1.2.1.3 give training details for the *15-CL* and *0-15-CL* sequence-token TDNNs, respectively. These networks were trained using fixed numbers of back-propagation iterations (100 340 for the *15-CL* networks, 200 100 for the *0-15-CL* networks), since establishing satisfactory target errors, *a priori*, proved difficult.

The dashed and dotted lines in Figure 5.2.3-1 show the recognition and false-positive error performances of the $ST_{50(15\text{-}CL)}$ and the $ST_{50(0\text{-}15\text{-}CL)}$, respectively (§A1.2.2.2 presents the data for all the curves plotted in Figure 5.2.3-1). Both these expert modules afford better recognition performances than the $ST_{50(CL)}$ at low SNRs (particularly between 0 and 20 dB), but are marginally worse for "clean" speech utterances. Unfortunately, as Figure 5.2.3-1 (b) shows, the improved recognition performances of the $ST_{50(15\text{-}CL)}$ and $ST_{50(0\text{-}15\text{-}CL)}$ at low SNRs come at the cost of poorer false-positive error performances typically, particularly for the $ST_{50(0\text{-}15\text{-}CL)}$. This behaviour is attributed to the need to *enlarge* the agreement regions in pattern-space (see Figure 4.3-2), in order to accommodate the *greater variation* present in the training tokens representing varying levels of noise corruption. By enlarging these regions, the probability of component network disagreement concerning inappropriate input is diminished, leading to a rise in the number of potential false-positive errors made.

The results associated with the $ST_{50(15\text{-}CL)}$ and $ST_{50(0\text{-}15\text{-}CL)}$ indicate that using a single expert module to recognize closing diphthong realizations corrupted by an arbitrary amount of noise is undesirable, since false-positive error performance may be poor. A more desirable alternative to this approach might be to use several expert modules for closing diphthong recognition, each suited to a specific range of SNR. This alternative has the advantage that the individual modules trained must only accommodate part of the variation in closing diphthong realizations caused by varying levels of noise corruption. Provided the number of potential false-positive errors made by each module is small in response to realizations whose

level of corruption suits another module, this approach may be successful.

The results presented in this section demonstrate that an ST_{50} trained in conjunction with "clean" speech utterances may be expected to perform reasonably robustly, provided SNR $\geq$ 20 dB. They also indicate that attempting to recognize arbitrarily corrupted closing diphthong realizations using a single expert module is perhaps undesirable.

5.3 Summary

From the speaker-dependent results reported in §5.1, the following results are important. First and foremost, expert modules comprising squads of basic-, extended- or sequence-token TDNNs afford significantly better recognition and/or false-positive error performances than their traditional expert module counterparts. This implies the former are better approximations to an ideal expert module for closing diphthong recognition. Second, there is partial evidence to suggest that squad-based expert modules comprising sequence-token TDNNs are better for closing diphthong recognition than similar modules comprising basic- or extended-token TDNNs. Such modules constitute the best approximations to an ideal expert module for closing diphthong recognition *observed* and, unlike the other two module types, do not produce ambiguous phoneme sequences in response to realizations of /ai/ and /ei/.

The results presented in §5.2 demonstrate that multi-speaker expert modules comprising squads of sequence-token TDNNs afford comparable recognition and false-positive error performances to their speaker-dependent counterparts. In particular, it is demonstrated that such modules may successfully recognise male and female closing diphthong realizations, despite the large differences that may exist between their average diphthong-glides. The results presented in this section also demonstrate that squad-based expert modules comprising sequence-token TDNNs may be trained to recognize "clean" and highly corrupted closing diphthong realizations, though at the cost of degraded false-positive error performances.

Chapter 6

Conclusion and Suggestions for Further Research

6.1 Conclusion

As a step towards the development of a *modular TDNN* for recognizing phonemes realized with a New Zealand accent, this thesis focuses on the development of an *expert module* for closing diphthong recognition. Realizations of these phonemes pose problems for traditional recognition approaches based on TDNNs, due to their extended durations (Hataoka and Waibel 1990). In addition, when realized with a New Zealand accent, the diphthong-glides associated with realizations of /ai/ and /ei/ may overlap significantly, making them difficult to recognize unambiguously. This thesis presents and compares two kinds of expert modules for closing diphthong recognition, referred to as *traditional* and *squad-based expert modules*. Traditional expert modules comprise individual TDNNs and have been used extensively for Japanese phoneme recognition (see Waibel *et al* 1989a; Waibel *et al* 1989b; Miyatake *et al* 1990). In contrast, the squad-based expert modules proposed in this thesis comprise ensembles of similarly trained TDNNs, referred to as *squads*. This thesis discusses examples of both kinds of expert module formed from one of three types of TDNN, referred to as *basic-*, *extended-* and *sequence-token TDNNs*. Unlike basic-token TDNN, extended- and sequence-token TDNNs are intended specifically for diphthong recognition, the latter being developed in this work to recognize New Zealand English closing diphthongs.

Following the approach used by Waibel and his colleagues to develop expert modules for Japanese phoneme recognition (Waibel *et al* 1989a; Waibel *et al* 1989b), traditional and squad-based expert modules for closing diphthong recognition are trained, tested and compared *speaker-dependently* in this work. Expert modules are compared in terms of *recognition* and *false-positive error performances*, the latter being a measure of a module's *potential* to cause *false-positive errors* when used within a modular TDNN. This thesis discusses traditional and squad-based expert modules created and compared for two adult speakers of New Zealand English (one of each sex), both of whom have *general* New Zealand English accents.

Comparing the performances of the various traditional and squad-based expert modules tested, the following results are observed for *both* speakers. First, of the traditional expert modules tested, those comprising extended-token TDNNs afford the best performance

compromises. In particular, these modules exhibit significantly better recognition performances than traditional expert modules comprising sequence-token TDNNs (though worse false-positive error performances) and significantly better false-positive error performances than those comprising basic-token TDNNs (though identical recognition performances). Consequently, if traditional expert modules are to be used for closing diphthong recognition, those comprising extended-token TDNNs are to be preferred. A similar preference is reported by Hataoka and Waibel (1990) also, who test TDNNs resembling basic- and extended-token TDNNs for American English diphthong recognition. However, in contrast to the findings of Hataoka and Waibel (1990), the use of extended tokens when recognizing New Zealand English closing diphthongs is principally motivated by the desire to improve false-positive error performances, rather than recognition performances.

Second, squad-based expert modules for closing diphthong recognition afford significantly better recognition and/or false-positive error performances than their traditional expert module counterparts, irrespective of whether basic-, extended- or sequence-token TDNNs are used. In particular, their false-positive error performances are *consistently* better as a consequence of their ability to "ignore" phoneme realizations from classes other than their own. This ability is viewed as a form of *selective attention* in this thesis, since squad-based expert modules may "attend" (respond with phonemic symbols) to selected input, while "ignoring" (responding with a *null classification* to) other inappropriate input.

Importantly, selective attention is achieved (to varying degrees) by the squad-based expert modules discussed in this thesis, by training with tokens that *only* represent phoneme realizations from their own class - the closing diphthongs. These modules *do not* need to experience examples of tokens representing phoneme realizations from other classes during training, to be able to "ignore" them during operation. For modular approaches to automated phoneme recognition, this ability is advantageous since the pool of phonemes whose realizations must be "ignored" by a given expert module is generally much larger than the pool of phonemes whose realizations must be recognized. Consequently, the need to train only with tokens representing the latter greatly simplifies training. Based on the results presented in this thesis, it is concluded that squad-based expert modules are preferable to traditional expert modules for closing diphthong recognition.

Third, of the squad-based expert modules tested, those comprising sequence-token TDNNs afford *consistently* better false-positive error performances than those comprising basic- or extended-token TDNNs, while the recognition performances of all three types of modules are very similar. In particular, it is demonstrated that squad-based expert modules comprising sequences-token TDNNs are better at "ignoring" voiced plosive and monophthong realizations than squad-based expert modules comprising the other two types of TDNN. In addition, the former do not produce the ambiguous phoneme sequence /ai/-/ei/ in response to realizations of /ai/ and /ei/ that squad-based expert modules comprising basic- or extended-

token TDNNs do. Consequently, squad-based expert modules comprising sequence-token TDNNs are recommended as the preferred method of recognizing closing diphthongs realized with a New Zealand accent.

The speaker-dependent results reported in this thesis also demonstrate that traditional and squad-based expert modules for closing diphthong recognition may be trained to perform equally well for New Zealand English speakers of either sex. Given the large differences that may exist between the average diphthong-glides of closing diphthongs realized by adult male and female speakers, it is desirable to know whether such modules may be trained to accommodate these differences. The additional experiments with squad-based expert modules comprising sequence-token TDNNs discussed in this thesis, demonstrate that multi-speaker modules of this type may perform very similarly to their speaker-dependent counterparts. They also demonstrate that such modules may be trained to recognize highly corrupted and uncorrupted closing diphthong realizations, though at the expense of false-positive error performance.

6.2 Suggestions for Further Research

Based on the findings presented in this thesis, the following areas of research are suggested for further investigation.

First, a new method of training TDNNs for use in squads is required to produce better and more computationally efficient squad-based expert modules. The TDNNs used to form the squad-based expert modules discussed in this thesis were trained *sub-optimally* using a variant of the traditional back-propagation algorithm to minimize the errors associated with their *individual* performances. This algorithm may perhaps be modified to train several TDNNs simultaneously to allow the errors associated with their performance *as a squad* to be minimized. Ideally, an algorithm developed to train component TDNNs for squads should only use training tokens representing appropriate input in order to keep training as simple as possible.

Second, to permit the construction of a modular TDNN for New Zealand English phonemes, research to find suitable expert module architectures for this accent's phoneme classes, other than the closing diphthongs, is required. Following the results reported by Waibel and his colleagues concerning Japanese phoneme recognition (Waibel *et al* 1989a; Waibel *et al* 1989b; Miyatake *et al* 1990), it is likely that expert modules comprising basic-token TDNNs may suffice for most of these classes. From the results presented in this thesis, it is also anticipated that these expert modules should be formed from squads of TDNNs to ensure good false-positive error performances.

Third, using knowledge of the expert module architectures best suited to recognizing

all the phonemes of New Zealand English, an arbitration module capable of integrating their responses is required. This module must be able to process delayed phoneme sequences if expert modules comprising sequence-token TDNNs are used for closing diphthong recognition. Ideally, it should also produce phoneme sequences containing ranked phonemic alternatives to permit subsequent processing of ambiguous phoneme realizations in conjunction with feed-back from processing at higher linguistic levels.

Appendix 1

TDNN Training and Test Results

A1.1 Speaker-Dependent Experiments

A1.1.1 Training

A1.1.1.1 Speaker JK

(a) Basic-Token TDNN

Quantity	min.	mean	max.	standard deviation
Classification Performance	100%	100%	100%	0
$\mathscr{E}_{av}$	0.0136	0.0146	0.015	0.0003
Iterations	6240	7652	9840	778
Epochs	78	96	123	9.7
Target Error	-	0.015	-	-

(b) Extended-Token TDNN

Quantity	min.	mean	max.	standard deviation
Classification Performance	100%	100%	100%	0
$\mathscr{E}_{av}$	0.035	0.048	0.050	0.004
Iterations	7280	12494	20000	3675
Epochs	91	156.2	250	45.9
Target error	-	0.05	-	-

(c) Sequence-Token TDNN

Quantity	min.	mean	max.	standard deviation
Classification Performance	100%	100%	100%	0
$\mathscr{E}_{av}$	0.0145	0.0149	0.015	0.002
Iterations	16820	24032	48285	5191
Epochs	116	145	333	36
Target Error	-	0.015	-	-

Table A1.1.1.1-1. Statistics concerning the training of 50 basic-, extended- and sequence-token TDNNs for speaker JK (see §5.1.1). *Classification performance* measures the ability of these TDNNs to correctly classify the *aligned test tokens* not used for training. $\mathscr{E}_{av}$ is the average McClelland error at the completion of training (see equation (4.1–6)). *Iterations* is the number of (modified) back-propagation algorithm iterations used during training and *epochs* is the number of weight updates used (batch mode weight update). Finally, *target error* indicates the maximum value of $\mathscr{E}_{av}$ for successfully trained networks.

A1.1.1.2 Speaker HD

(a) Basic-Token TDNN

Quantity	min.	mean	max.	standard deviation
Classification Performance	100%	100%	100%	0
$\mathscr{E}_{av}$	0.0137	0.0146	0.0150	0.0003
Iterations	5040	10294	15920	2251
Epochs	63	126.3	250	28.1
Target Error	-	0.015	-	-

(b) Extended-Token TDNN

Quantity	min.	mean	max.	standard deviation
Classification Performance	100%	100%	100%	0
$\mathscr{E}_{av}$	0.041	0.047	0.050	0.001
Iterations	5120	12908	20000	3852
Epochs	64	161.3	250	48
Target error	-	0.05	-	-

(c) Sequence-Token TDNN

Quantity	min.	mean	max.	standard deviation
Classification Performance	100%	100%	100%	0
$\mathscr{E}_{av}$	0.0143	0.0148	0.0150	0.0002
Iterations	17110	22243	45675	4492
Epochs	118	153.4	315	31
Target Error	-	0.015	-	-

Table A1.1.1.2-1. Statistics concerning the training of 50 basic-, extended- and sequence-token TDNNs for speaker HD (see §5.1.1). See Table A1.1.1.1-1 for descriptions of the quantities listed in this table.

A1.1.2 Test Results for Squad-Based Expert Modules

A1.1.2.1 Diphthong Syllables: Speaker JK - False-Positive Errors

(a) Basic-Token (BT_{50})

Diphthong in Syllable	False-Positive Error Diphthong /ai/	/au/	/ɔi/	/ei/	/ou/
/ai/	-	17	8	39	11
/au/	12	-	21	0	0
/ɔi/	0	3	-	7	3
/ei/	32	0	22	-	0
/ou/	1	1	24	0	-

(b) Extended-Token (ET_{50})

Diphthong in Syllable	False-Positive Error Diphthong /ai/	/au/	/ɔi/	/ei/	/ou/
/ai/	-	4	0	10	0
/au/	0	-	4	0	0
/ɔi/	0	0	-	0	0
/ei/	0	0	0	-	0
/ou/	0	0	0	0	-

(c) Sequence-Token (ST_{50})

One realization of /ei/ (the one realization not correctly recognized) leads to a /ai/ false-positive error.

Table A1.1.2.1-1. A breakdown of the false-positive errors for speaker JK's BT_{50}, ET_{50} and ST_{50} when processing his 240 closing diphthong syllables not used for network training. *Diphthong in syllable* refers to the diphthong realization present within a syllable processed, while *false-positive error diphthong* refers to the diphthong falsely detected during this processing.

A1.1.2.2 Diphthong Syllables: Speaker HD - False-Positive Errors

(a) Basic-Token (BT_{50})

Diphthong in Syllable	False-Positive Error Diphthong /ai/	/au/	/ɔi/	/ei/	/ou/
/ai/	-	35	15	27	0
/au/	18	-	7	2	0
/ɔi/	0	23	-	1	0
/ei/	46	0	26	-	0
/ou/	0	2	6	0	-

(b) Extended-Token (ET_{50})

Diphthong in Syllable	False-Positive Error Diphthong /ai/	/au/	/ɔi/	/ei/	/ou/
/ai/	-	0	5	1	0
/au/	14	-	6	0	0
/ɔi/	0	0	-	0	0
/ei/	16	0	42	-	0
/ou/	0	0	0	0	-

(c) Sequence-Token (ST_{50})

No false-positive errors.

Table A1.1.2.2-1. A breakdown of the false-positive errors for speaker HD's BT_{50}, ET_{50} and ST_{50} when processing her 240 closing diphthong syllables not used for network training.

A1.1.2.3 Monophthong Syllables: Speaker JK - False-Positive Errors

(a) Basic-Token (BT_{50})

Monophthong in Syllable	False-Positive Error Diphthong /ai/	/au/	/ɔi/	/ei/	/ou/	Total
/ɑ/	4	16	0	0	0	20
/æ/	12	0	0	11	0	23
/ɔ/	4	8	6	0	0	18
/e/	6	0	1	16	0	23
/ɜ/	5	0	0	0	7	12
/i/	9	0	0	4	0	13
/u/	7	0	4	0	4	15
/ʌ/	13	1	1	0	0	15
/ɪ/	7	0	0	7	0	14
/ʊ/	2	1	4	3	2	12
Total	69	26	16	41	13	165

(b) Extended-Token (ET_{50})

Monophthong in Syllable	False-Positive Error Diphthong /ai/	/au/	/ɔi/	/ei/	/ou/	Total
/ɑ/	0	2	0	0	0	2
/æ/	0	0	0	8	0	8
/ɔ/	0	0	6	0	0	6
/e/	0	0	0	1	0	1
/ɜ/	0	0	0	0	4	4
/i/	0	0	0	0	0	0
/u/	0	0	1	0	3	4
/ʌ/	0	1	0	0	0	1
/ɪ/	0	0	0	6	0	6
/ʊ/	0	0	2	0	2	4
Total	0	3	9	15	9	36

Table A1.1.2.3-1. See caption on opposite page.

(c) Sequence-Token (ST_{50})

Monophthong in Syllable	False-Positive Error Diphthong /ai/	/au/	/ɔi/	/ei/	/ou/	Total
/ɒ/	0	0	0	0	0	0
/æ/	0	0	0	6	0	6
/ɔ/	0	0	0	0	0	0
/e/	0	0	0	0	0	0
/ɜ/	0	0	0	0	10	10
/i/	0	0	0	0	0	0
/u/	0	0	0	0	6	6
/ʌ/	1	0	0	1	0	2
/ɪ/	0	0	0	0	0	0
/ʊ/	0	0	0	0	2	2
Total	1	0	0	7	18	26

Table A1.1.2.3-1. A breakdown of the false-positive errors for speaker JK's BT_{50}, ET_{50} and ST_{50} when processing his 160 monophthong syllables. *Monophthong in syllable* refers to the monophthong realization present within a syllable processed and *false-positive error diphthong* refers to the diphthong falsely detected during this processing.

A1.1.2.4 Monophthong Syllables: Speaker HD - False-positive Errors

(a) Basic-Token (BT_{50})

Monophthong in Syllable	False-Positive Error Diphthong /ai/	/au/	/ɔi/	/ei/	/ou/	Total
/ɒ/	0	16	5	0	0	21
/æ/	0	0	10	14	0	24
/ɔ/	0	9	4	0	0	13
/e/	0	0	14	5	0	19
/ɜ/	0	0	8	0	7	12
/i/	0	0	16	0	0	16
/u/	0	0	9	0	4	9
/ʌ/	5	0	8	0	0	13
/ɪ/	0	0	16	0	0	16
/ʊ/	0	1	6	0	2	7
Total	5	26	96	19	4	150

Table A1.1.2.4-1. See caption on next page.

(b) Extended-Token (ET_{50})

Monophthong in Syllable	False-Positive Error Diphthong /ai/	/au/	/ɔi/	/ei/	/ou/	Total
/ɑ/	0	4	2	0	0	6
/æ/	0	0	4	4	0	8
/ɔ/	0	0	6	0	0	6
/e/	0	0	6	0	0	6
/ɜ/	0	0	0	0	1	1
/i/	0	0	16	0	0	16
/u/	0	0	1	0	0	1
/ʌ/	0	0	0	0	0	0
/ɪ/	0	0	16	6	0	16
/ʊ/	0	0	4	0	0	4
Total	0	4	55	4	1	64

(c) Sequence-Token (ST_{50})

Monophthong in Syllable	False-Positive Error Diphthong /ai/	/au/	/ɔi/	/ei/	/ou/	Total
/ɑ/	0	0	5	0	0	5
/æ/	0	0	0	6	0	6
/ɔ/	0	0	2	0	0	2
/e/	0	0	0	0	0	0
/ɜ/	0	0	0	0	1	1
/i/	0	0	0	0	0	0
/u/	0	0	0	0	0	0
/ʌ/	0	0	0	1	0	1
/ɪ/	0	0	0	0	0	0
/ʊ/	0	0	5	0	0	5
Total	0	0	12	7	1	20

Table A1.1.2.4-1 *(continued).* A breakdown of the false-positive errors for speaker HD's BT_{50}, ET_{50} and ST_{50} when processing her 160 monophthong syllables.

A1.2 Multi-Speaker Experiments

A1.2.1 Training Results

A1.2.1.1 *CL* Sequence-Token TDNNs

Quantity	min.	mean	max.	standard deviation
Classification Performance	99.7%	99.9%	100%	0.005
$\mathscr{E}_{av}$	0.0110	0.0126	0.0139	0.00073
Iterations	35670	51382	78590	7086
Epochs	123	177.2	271	24.4
Target Error	-	0.015	-	-

Table A1.2.1.1-1. Statistics concerning the training of the 50 *CL sequence-token TDNNs* for speakers JK and HD. See Table A1.1.1.1-1 for a description of the quantities list in this table. The training tokens for these TDNNs were derived from speech portions as recorded in an anechoic chamber ("clean" speech portions).

A1.2.1.2 *15-CL* Sequence-Token TDNNs

Quantity	min.	mean	max.	standard deviation
Classification Performance	99.4	99.7	99.9	0.094
$\mathscr{E}_{av}$	0.0192	0.0257	0.0414	0.0038
Iterations	-	100 340	-	-
Epochs	-	173	-	-

Table A1.2.1.2-1. Statistics concerning the training of the 50 *15-CL sequence-token TDNNs* for speakers JK and HD. The training tokens for these TDNNs were derived from "clean" speech portions and from speech portions corrupted with band-limited white noise to have approximate SNRs of 15 dB.

A1.2.1.3 *0-15-CL* Sequence-Token TDNNs

Quantity	min.	mean	max.	standard deviation
Classification Performances	94.2	94.9	95.6	0.32
$\mathscr{E}_{av}$	0.0734	0.0823	0.1097	0.00632
Iterations	-	200 100	-	-
Epochs	-	230	-	-

Table A1.2.1.3-1. Statistics concerning the training of the 50 *0-15-CL sequence-token TDNNs* for speakers JK and HD. The training tokens for these TDNNs were derived from "clean" speech portions and from speech portions corrupted with band-limited white noise to have approximate SNRs of 0 dB and 15 dB.

A1.2.2 Results

A1.2.2.1 Monophthong Syllables: False-Positive Errors (CL Sequence-Token TDNNs)

Monophthong in Syllable	False-Positive Error Diphthong /ai/	/au/	/ɔi/	/ei/	/ou/	Total
/ɑ/	0	0	3	0	0	3
/æ/	0	0	0	5	0	5
/ɔ/	0	0	3	0	0	3
/e/	0	0	0	0	0	0
/ɜ/	0	0	0	0	12	12
/i/	0	0	0	0	0	0
/u/	0	0	0	0	9	9
/ʌ/	1	0	0	0	0	1
/ɪ/	0	0	0	0	0	0
/ʊ/	0	0	7	0	0	7
Total	1	0	13	5	21	40

Table A1.2.2.1-1. A breakdown of the false-positive errors for the multi-speaker ST_{50} trained for speakers JK and HD (comprising CL networks) when processing their monophthong syllables (320 utterances).

A1.2.2.2 Robustness to Noise Corruption

(a) $ST_{50(CL)}$

SNR (dB)	% Correct	Number of False-Positive Errors
0	7.7	21
5	21.3	24
10	48.1	20
15	79.4	29
20	95.5	16
30	99.4	5
Clean	100.0	1

(b) $ST_{50(15\text{-}CL)}$

SNR (dB)	% Correct	Number of False-positive Errors
0	33.0	100
5	77.5	59
10	94.5	19
15	98.6	13
20	99.5	9
30	99.8	11
Clean	99.6	30

(c) $ST_{50(0\text{-}15\text{-}CL)}$

SNR (dB)	% Correct	Number of False-positive Errors
0	50.6	173
5	65.8	134
10	84.8	94
15	94.7	78
20	96.4	95
30	97.7	51
Clean	96.9	45

Table A1.2.2.2-1 The recognition (% correct) and false-positive error performances for (a) $ST_{50(CL)}$, (b) $ST_{50(15\text{-}CL)}$, and (c) $ST_{50(0\text{-}15\text{-}CL)}$ when processing speaker JK's and speaker HD's closing diphthong syllables (480 utterances) corrupted by white noise.

Appendix 2

Weight Changes for TDNNs

This appendix presents a derivation of the expressions required to determine the weight changes for the *unique weights* of a TDNN like those used in this work. These networks are assumed to have an input layer, two hidden layers and an output layer (see Figure 4.2-1), referred to by the letters *h*, *i*, *j* and *k*, respectively. Due to the complex pattern of connectivity within TDNNs, expressions for the weight changes desired during training must be formulated for each layer individually. Consequently, these expressions are derived in the next three sections, commencing with the weights feeding the output layer (layer *k*) and working backwards to those feeding the first hidden layer (layer *i*). Within this derivation the following notation is adopted.

$o^{x}_{n,m}(p)$ the output of the $(n,m)^{th}$ node of layer x in response to token $\boldsymbol{x}(p)$ (a matrix of nodes is assumed following the "unfolded" representation of a TDNN depicted in Figure 4.2-1 (b)).

$w^{y(n,m)}_{x(a,b)}$ the current weight associated with the weighted connection joining the $(a,b)^{th}$ node of layer x to the $(n,m)^{th}$ node of layer y .

$v^{x}_{n,m}(p)$ the sum of the inputs into the $(n,m)^{th}$ node of layer x when processing $\boldsymbol{x}(p)$.

$\Delta w^{y(n,m)}_{x(a,b)}$ the required weight change for the weighted connection joining the $(a,b)^{th}$ node of layer x to the $(n,m)^{th}$ node of layer y.

$\eta^{y(n,m)}_{x(a,b)}$ the current learning rate for $w^{y(n,m)}_{x(a,b)}$

A2.1 Weight Changes for Connections Feeding the Output Layer

As depicted in Figure A2.1-1, each node in the output layer (layer k) is connected to a row of M^j nodes in the second hidden layer (layer j) by weight connections with the same weight and to a bias node by another connection with a unique weight. Consequently, only two unique weights need be stored for each of the N^k output nodes. This section derives expressions for the weight changes required to update these unique weights using equation (4.1-4).

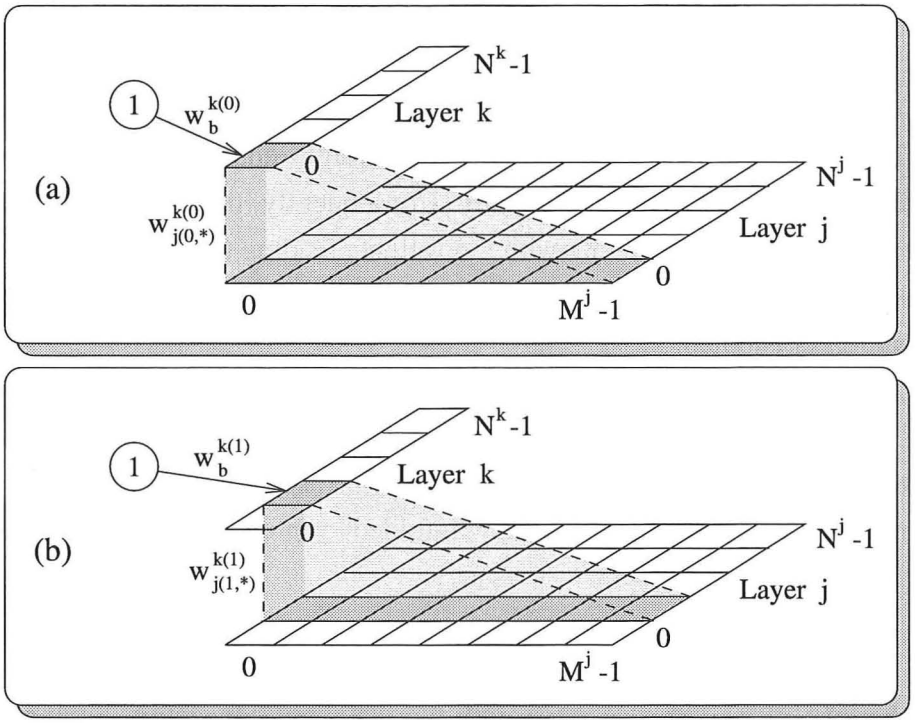

Figure A2.1-1. Shows examples of the weighted connections feeding the output layer nodes of a TDNN. Part (a) shows the connections feeding the first node in layer k (node 0), which is fed by M^j weighted connections (all with weights equalling $w_{j(0,*)}^{k(0)}$) from layer j and one weighted connection (weight equal to $w_b^{k(0)}$) from a bias node. Similar weighted connections feed the second node in layer k (node 1), as shown in part (b). Note that $N^k=N^j$ and M^j is the number of node replicas in layer j.

For the n^{th} node in layer k, the required change to the common weight shared by the weighted connections linking this node to the n^{th} row of layer j, is given by

$$\Delta w_{j(n,*)}^{k(n)} = \frac{\sum_{m=0}^{M^j-1} \Delta w_{j(n,m)}^{k(n)}}{M^j} \qquad \text{(A2.1-1)}$$

where n=0,1,...,N^k ($N^k=N^j$) and M^j is the number of node replicas in layer j. The notation for

this weight change includes an asterisk, instead of an index m, to indicate it is identical for all the weighted connection replicas corresponding to m=0,1,...,M^j-1.

The weight changes summed on the right hand side of equation (A2.1-1) must be evaluated separately, since they are typically different for each weighted connection replica. The weight change required for each connection replica is given by

$$\Delta w_{j(n,m)}^{k(n)} = -\eta_{j(n,*)}^{k(n)} \frac{\partial \mathscr{E}_{av}}{\partial w_{j(n,m)}^{k(n)}} \tag{A2.1-2}$$

which becomes

$$\Delta w_{j(n,m)}^{k(n)} = -\frac{\eta_{j(n,*)}^{k(n)}}{P} \sum_{p=0}^{P-1} \frac{\partial \mathscr{E}(p)}{\partial w_{j(n,m)}^{k(n)}} \tag{A2.1-3}$$

using equation (4.1-6) (note η in this expression is associated with the *common weight change* given by equation (A2.1-1)). Expanding the partial derivative on the right of this expression gives

$$\frac{\partial \mathscr{E}(p)}{\partial w_{j(n,m)}^{k(n)}} = \frac{\partial \mathscr{E}(p)}{\partial e_n^k(p)} \cdot \frac{\partial e_n^k(p)}{\partial o_n^k(p)} \cdot \frac{\partial o_n^k(p)}{\partial v_n^k(p)} \cdot \frac{\partial v_n^k(p)}{\partial w_{j(n,m)}^{k(n)}} \tag{A2.1-4}$$

where

$$\begin{aligned} \frac{\partial \mathscr{E}(p)}{\partial e_n^k(p)} &= \frac{\partial}{\partial e_n^k(p)} \left[-\sum_{n=0}^{N^k-1} \ln\left(1 - e_n^k(p)^2\right) \right] \\ &= \frac{\partial}{\partial e_n^k(p)} \left[-\ln\left(1 - e_n^k(p)^2\right) \right] \\ &= \frac{2 e_n^k(p)}{1 - e_n^k(p)^2} \end{aligned} \tag{A2.1-5}$$

using equation (4.1-7) (note $e_n^k(p)$ is the same as $e_m(p)$, see §4.1),

$$\frac{\partial e_n^k(p)}{\partial o_n^k(p)} = \frac{\partial}{\partial o_n^k(p)} \left[d_n(p) - o_n^k(p) \right] = -1 \tag{A2.1-6}$$

using equation (4.1-8),

$$\frac{\partial o_n^{\,k}(p)}{\partial v_n^{\,k}(p)} = o_n^{\,k}(p)\left(1 - o_n^{\,k}(p)\right) \tag{A2.1-7}$$

assuming a sigmoidal non-linearity (see equation (4.1-1)) and

$$\frac{\partial v_n^{\,k}(p)}{\partial w_{j(n,m)}^{k(n)}} = o_{n,m}^{\,j}(p) \tag{A2.1-8}$$

since $v_n^{\,k}(p)$, the summed input to the n^{th} output node prior to non-linear transformation, is given by

$$v_n^{\,k}(p) = \left[\sum_{m=0}^{M^j-1} w_{j(n,m)}^{k(n)} o_{n,m}^{\,j}(p)\right] + w_b^{k(n)}.1 \tag{A2.1-9}$$

For convenience later, the *recursion* term $\delta_n^{\,k}(p)$ is defined as

$$\begin{aligned}\delta_n^k(p) &= \frac{\partial \mathscr{E}(p)}{\partial v_n^{\,k}(p)} \\ &= \frac{\partial \mathscr{E}(p)}{\partial e_n^{\,k}(p)} \cdot \frac{\partial e_n^{\,k}(p)}{\partial o_n^{\,k}(p)} \cdot \frac{\partial o_n^{\,k}(p)}{\partial v_n^{\,k}(p)} \\ &= \frac{-2e_n^{\,k}(p)}{1 - e_n^{\,k}(p)^2} o_n^{\,k}(p)\left(1 - o_n^{\,k}(p)\right)\end{aligned} \tag{A2.1-10}$$

Rewriting equation (A2.1-4) using this expression and equation (A2.1-8), it may be used in conjunction with equations (A2.1-2) and (A2.1-3) to rewrite equation (A2.1-1) as

$$\Delta w_{j(n,*)}^{k(n)} = \frac{1}{M^j} \sum_{m=0}^{M^j-1} \left[\frac{-\eta_{j(n,*)}^{k(n)}}{P} \sum_{p=0}^{P-1} \delta_n^k(p) o_{n,m}^{\,j}(p)\right] \tag{A2.1-11}$$

where n=0,1,...,N^k. This may be simplified to

$$\Delta w_{j(n,*)}^{k(n)} = \frac{-\eta_{j(n,*)}^{k(n)}}{M^j P} \sum_{m=0}^{M^j-1} \sum_{p=0}^{P-1} \delta_n^k(p) o_{n,m}^{\,j}(p) \tag{A2.1-12}$$

Note that if $e_n^k(p)>0$ when processing $\boldsymbol{x}(p)$ (the desired value, $d_n(p)$ exceeds the observed output, $o_n^k(p)$) then $\Delta w_{j(n,*)}^{k(n)} > 0$ *in response to this training token*, ensuring $o_n^k(p)$ is *increased* as required.

Similarly, the weight change associated with each output node's weighted connection to the bias node is given by

$$\Delta w_b^{k(n)} = \frac{-\eta_b^{k(n)}}{P} \sum_{p=0}^{P-1} \delta_n^k(p).1 \qquad \text{(A2.1-13)}$$

assuming this connection originates from a fully active node (o_b=1). Unlike equation (A2.1–12), this change is derived from an *average* over the P recursion terms alone, since there is only *one* time replica of this weighted connection feeding the n^{th} output node. Note that if $e_n^k(p)>0$ when processing $\boldsymbol{x}(p)$, then $\Delta w_b^k>0$ *in response to this training token*, implying the associated sigmoidal non-linearity is "displaced" towards *smaller* values of the net input $v_n^k(p)$, ensuring $o_n^k(p)$ is *increased* as required.

Equations (A2.1-12) and (A2.1-13) give the required weight changes for the *unique weights* associated with the weighted connections feeding the nodes in layer k (including all the replicas of these connections). Each of these unique weights has a unique learning rate, as required for delta-bar-delta learning (see §4.1.1) (for simplicity, this learning rate is denoted using the same indices as the weight change).

A2.2 Weight Changes for Connections Feeding the Second Hidden Layer

As depicted in Figure A2.2-1, the M^j replicas of each node in the second hidden layer (one row of nodes in layer j) are each connected to a "window" of $N^i \times L^i$ nodes in the first hidden layer (layer i) by weighted connections sharing the same set of common weights. These M^j node replicas are also connected to the bias node by weighted connections sharing the same weight. Consequently, only $N^i \times L^i + 1$ unique weights need be stored for each of the N^j *unique* nodes in the second hidden layer. This section derives expressions for the weight changes required to update these unique weights using equation (4.1-4).

For each node replica in the n^{th} row of layer j, the required weight changes for the set of weighted connections linking it to a "window" of nodes in layer i is given by

$$\Delta w_{i(a,l)}^{j(n,*)} = \frac{\sum_{m=0}^{M^j-1} \Delta w_{i(a,m+l)}^{j(n,m)}}{M^j} \qquad \text{(A2.2-1)}$$

where a=0,1,...,N^i-1 and l=0,1,...,L^i-1 index the set of unique weights shared by *all* M^j node

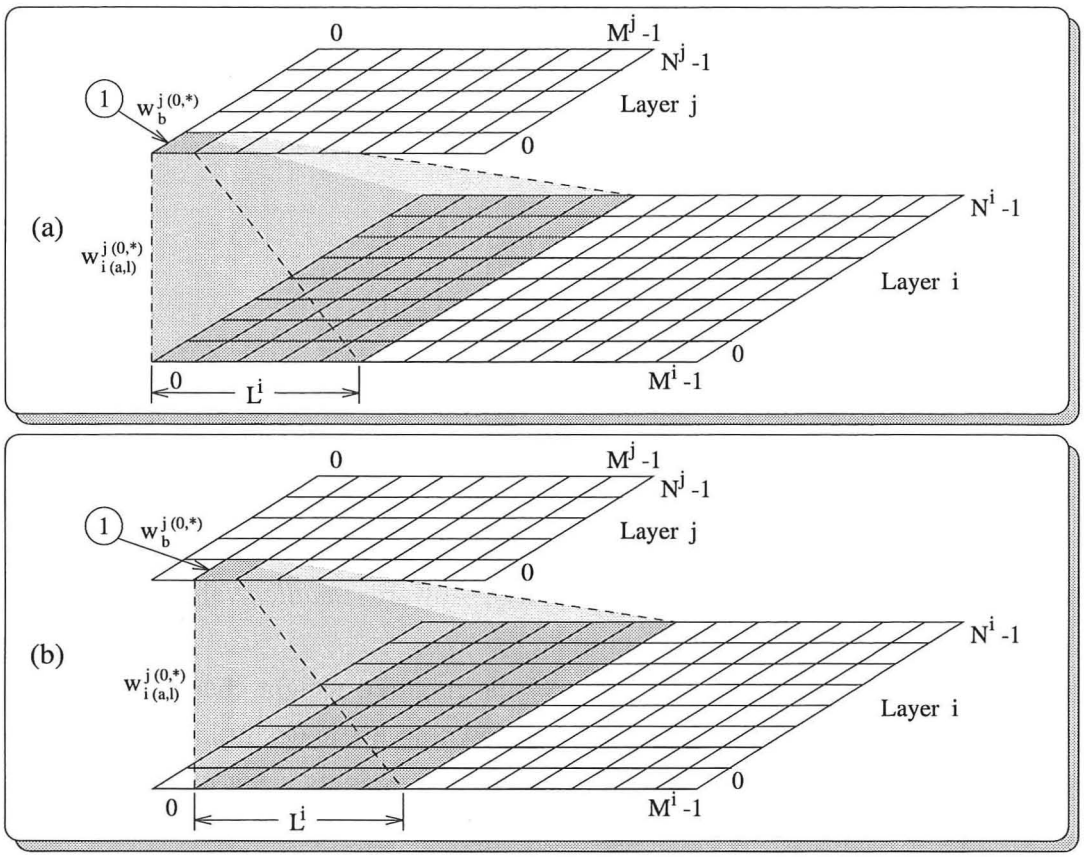

Figure A2.2-1. Shows examples of the weighted connections which join each node in layer *j* to a window of nodes in layer *i*. The weights feeding each replica of a node (for example, the shaded replicas in row *0* of layer *j*, shown in parts (a) and (b)) are fed by weighted connections sharing the same weights for (positionally) equivalent pairs of nodes joined.

replicas and n=0,1,...,N^j-*1*. Once again the index m is replaced by an asterisk to indicate that each weight change is identical for all m=0,1,...,M^j. [1]

The weight changes summed on the right hand side of equation (A2.2-1) must be evaluated separately, since they are typically different for each value of m. These changes are given by

$$\Delta w_{i(a,m+l)}^{j(n,m)} = -\eta_{i(a,l)}^{j(n,*)} \frac{\partial \mathscr{E}_{av}}{\partial w_{i(a,m+l)}^{j(n,m)}} = \frac{-\eta_{i(a,l)}^{j(n,*)}}{P} \sum_{p=0}^{P-1} \frac{\partial \mathscr{E}(p)}{\partial w_{i(a,m+l)}^{j(n,m)}} \qquad \text{(A2.2-2)}$$

[1]Note, however, that this replication is viewed with respect to the receiving layer (layer *j*) instead of the originating layer, as in the previous section.

Expanding the partial derivative incorporating $\mathscr{E}(p)$ gives

$$\frac{\partial \mathscr{E}(p)}{\partial w_{i(a,m+l)}^{j(n,m)}} = \frac{\partial \mathscr{E}(p)}{\partial o_{n,m}^{j}(p)} \cdot \frac{\partial o_{n,m}^{j}(p)}{\partial v_{n,m}^{j}(p)} \cdot \frac{\partial v_{n,m}^{j}(p)}{\partial w_{i(a,m+l)}^{j(n,m)}} \tag{A2.2-3}$$

where

$$\frac{\partial o_{n,m}^{j}(p)}{\partial v_{n,m}^{j}(p)} = o_{n,m}^{j}(p)\left(1 - o_{n,m}^{j}(p)\right) \tag{A2.2-4}$$

assuming a sigmoidal non-linearity (see equation (4.1-1)) and

$$\frac{\partial v_{n,m}^{j}(p)}{\partial w_{i(a,m+l)}^{j(n,m)}} = o_{a,m+l}^{i}(p), \tag{A2.2-5}$$

since

$$v_{n,m}^{j}(p) = \left[\sum_{a=0}^{N^i-1}\sum_{l=0}^{L^i-1} w_{i(a,m+l)}^{j(n,m)} o_{a,m+l}^{i}(p)\right] + w_{b}^{j(n,m)}.1 \tag{A2.2-6}$$

The remaining partial differential term in equation (A2.2-3) (the first term on the right) is evaluated by considering which nodes in layer k are affected by $o_{n,m}^{j}(p)$, the output of the $(n,m)^{th}$ node in layer j. In the case of a TDNN, the usual expression for a fully connected network

$$\frac{\partial \mathscr{E}(p)}{\partial o_{n,m}^{j}(p)} = \sum_{c=0}^{N^k-1} \frac{\partial \mathscr{E}(p)}{\partial v_{c}^{k}(p)} \cdot \frac{\partial v_{c}^{k}(p)}{\partial o_{n,m}^{j}(p)} \tag{A2.2-7}$$

simplifies to

$$\begin{aligned} \frac{\partial \mathscr{E}(p)}{\partial o_{n,m}^{j}(p)} &= \frac{\partial \mathscr{E}(p)}{\partial v_{n}^{k}(p)} \cdot \frac{\partial v_{n}^{k}(p)}{\partial o_{n,m}^{j}(p)} \\ &= \delta_{n}^{k}(p) w_{j(n,m)}^{k(n)}, \end{aligned} \tag{A2.2-8}$$

since only the n^{th} node of layer k is affected by $o_{n,m}^{j}(p)$, regardless of the value of m=0,1,...,M^j-1 (the terms replacing the two partial derivative terms on the right hand side of this expression are derived from equations (A2.1-10) and (A2.1-9), respectively).

For convenience later, the *recursion* term $\delta_{n,m}^{k}(p)$ is defined as

$$\delta_{n,m}^{j}(p)=\frac{\partial \mathscr{E}(p)}{\partial v_{n,m}^{j}(p)}=\frac{\partial \mathscr{E}(p)}{\partial o_{n,m}^{j}(p)}\cdot\frac{\partial o_{n,m}^{j}(p)}{\partial v_{n,m}^{j}(p)} \qquad \text{(A2.2-9)}$$

$$=\delta_{n}^{k}(p)\; w_{j(n,m)}^{k(n)}\; o_{n,m}^{j}(p)\left(1-o_{n,m}^{j}(p)\right)$$

where the expressions replacing the two partial differential terms (right most on first line) are obtained from equations (A2.2-8) and (A2.2-4), respectively. Rewriting equation (A2.2-3) using this expression and equation (A2.2-5), equation (A2.2-1) may be rewritten as

$$\Delta w_{i(a,l)}^{j(n,*)}=\frac{1}{M^{j}}\sum_{m=0}^{M^{j}-1}\left[\frac{-\eta_{i(a,l)}^{j(n,*)}}{P}\sum_{p=0}^{P-1}\delta_{n,m}^{j}(p)\,o_{a,m+l}^{i}(p)\right] \qquad \text{(A2.2-10)}$$

$$=\frac{-\eta_{i(a,l)}^{j(n,*)}}{M^{j}P}\left[\sum_{m=0}^{M^{j}-1}\sum_{p=0}^{P-1}\delta_{n,m}^{j}(p)\,o_{a,m+l}^{i}(p)\right]$$

where a=0,1,...,N^{i}-1, l=0,1,...,L^{i}-1 and n=0,1,...,N^{j}-1.

Similarly, the weight change required for the weighted connection joining each unique node in layer j to the bias node, may be deduced by averaging the weight changes evaluated for each for each of its replicas thus

$$\Delta w_{b}^{j(n,*)}=\frac{\sum_{m=0}^{M^{j}-1}\Delta w_{b}^{j(n,m)}}{M^{j}}$$

$$=\frac{1}{M^{j}}\sum_{m=0}^{M^{j}-1}\left[\frac{-\eta_{b}^{j(n,*)}}{P}\sum_{p=0}^{P-1}\delta_{n,m}^{j}(p).1\right] \qquad \text{(A2.2-11)}$$

$$=\frac{-\eta_{b}^{j(n,*)}}{M^{j}P}\sum_{m=0}^{M^{j}-1}\sum_{p=0}^{P-1}\delta_{n,m}^{j}(p)$$

A2.3 Weights Changes for Connections Feeding the First Hidden Layer

As depicted in Figure A2.3-1, the M^{i} replicas of each node in the first hidden layer

(one row of nodes in layer i) are each connected to a "window" of $N^h \times L^h$ nodes in the input layer (layer h) by weighted connections sharing the same set of common weights. These M^i node replicas are also connected to the bias node by weighted connections sharing the same weight. Consequently, only $N^h \times L^h + 1$ unique weights need be stored for each of the N^i *unique* nodes in the first hidden layer. This section derives expressions for the weight changes required to update these unique weights using equation (4.1-4).

For each node replica in the n^{th} row of layer i, the required weight changes for the set of weighted connections linking it to a "window" of nodes in layer h is given by

$$\Delta w_{h(a,l)}^{i(n,*)} = \frac{\sum_{m=0}^{M^i-1} \Delta w_{h(a,m+l)}^{i(n,m)}}{M^i} \qquad \text{(A2.3-1)}$$

where a=0,1,...N^h-1 and l=0,1,...,L^h-1 index the set of unique weight values shared by *all* M^i node replicas and n=0,1,...,N^i-1. As for the second hidden layer, the index m is replaced by an asterisk to indicate that each weight change is identical for all m=0,1,...,M^i.

Once again, the weight changes summed on the right hand side of equation (A2.3-1) must be evaluated separately using

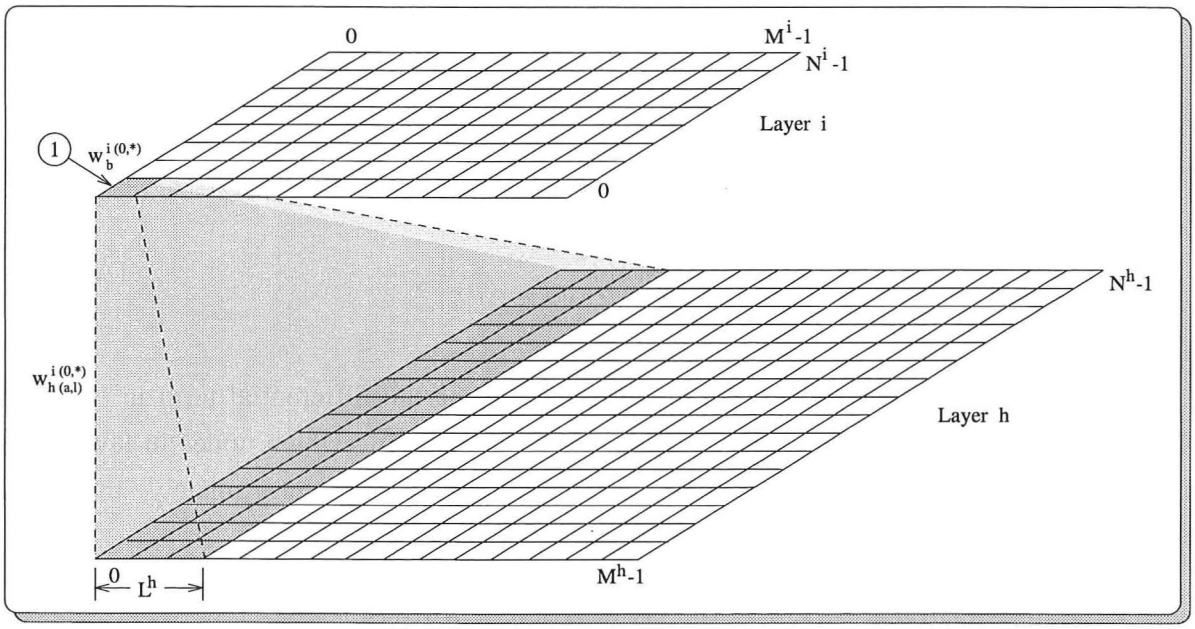

Figure A2.3-1. Shows an example of the weighted connections which join each node in layer i to a window of nodes in layer h. These connections, and those like them feeding the other N^i-1 unique nodes in layer i, are replicated in an identical fashion to the weighted connections feeding layer j (see Figure A2.2-1).

$$\Delta w_{h(a,m+l)}^{i(n,m)} = -\eta_{h(a,l)}^{i(n,*)} \frac{\partial \mathscr{E}_{av}}{\partial w_{h(a,m+l)}^{i(n,m)}} = \frac{-\eta_{h(a,l)}^{i(n,*)}}{P} \sum_{p=0}^{P-1} \frac{\partial \mathscr{E}(p)}{\partial w_{h(a,m+l)}^{i(n,m)}} \tag{A2.3-2}$$

Expanding the partial derivative incorporating $\mathscr{E}$(p) gives

$$\frac{\partial \mathscr{E}(p)}{\partial w_{h(a,m+l)}^{i(n,m)}} = \frac{\partial \mathscr{E}(p)}{\partial o_{n,m}^{i}(p)} \cdot \frac{\partial o_{n,m}^{i}(p)}{\partial v_{n,m}^{i}(p)} \cdot \frac{\partial v_{n,m}^{i}(p)}{\partial w_{h(a,m+l)}^{i(n,m)}} \tag{A2.3-3}$$

where

$$\frac{\partial o_{n,m}^{i}(p)}{\partial v_{n,m}^{i}(p)} = o_{n,m}^{i}(p)\left(1 - o_{n,m}^{i}(p)\right) \tag{A2.3-4}$$

assuming a sigmoidal non-linearity and

$$\frac{\partial v_{n,m}^{i}(p)}{\partial w_{h(a,m+l)}^{i(n,m)}} = o_{a,m+l}^{h}(p) \tag{A2.3-5}$$

since

$$v_{n,m}^{i}(p) = \left[\sum_{a=0}^{N^h-1} \sum_{l=0}^{L^h-1} w_{h(a,m+l)}^{i(n,m)} o_{a,m+l}^{h}(p)\right] + w_{b}^{i(n,m)}.1 \tag{A2.3-6}$$

As with the second hidden layer, the remaining partial differential term in equation (A2.3-3) (the first term on the right) is evaluated by considering which nodes in layer *j* are affected by $o_{n,m}^{i}(p)$, the output of the $(n,m)^{\text{th}}$ node in layer *i*. Unfortunately, the usual expression for a fully connected network

$$\frac{\partial \mathscr{E}(p)}{\partial o_{n,m}^{i}(p)} = \sum_{r=0}^{N^j-1} \sum_{s=0}^{M^j-1} \frac{\partial \mathscr{E}(p)}{\partial v_{r,s}^{j}(p)} \cdot \frac{\partial v_{r,s}^{j}(p)}{\partial o_{n,m}^{i}(p)} = \sum_{r=0}^{N^j-1} \sum_{s=0}^{M^j-1} \delta_{r,s}^{j}(p)\, w_{i(n,m)}^{j(r,s)} \tag{A2.3-7}$$

does not account for the pattern of connectivity within a TDNN (many of the weights in this expression are zero, implying no connection), or the replication of weights (note the terms replacing the two partial derivative terms on the right hand side of this expression are derived from equations (A2.2-9) and (A2.2–6), respectively).

The pattern of connections joining layers i and j is perhaps best understood by considering a simplified example in which $N^i=N^j=1$, such as that depicted in Figure A2.3-2. In this example, $M^i=8$, implying $M^j=6$, and $L^i=3$. Figure A2.3-2 (a) through (c) depict the connections between individual node replicas in layer j and their associated "windows" of node replicas in layer i, as normally presented when describing TDNNs (see for example, Figure 4.2-1 (b)). In this figure, the unique weights are each given a different line type to highlight the common set of weights joining node replicas in layers i and j. Figure A2.3-2 (d) and (e) show examples of the node replicas in layer j *affected* by a given node replica in layer i and the weights (indicated by the different line-types) associated with their interaction. Significantly, as shown in Figure A2.3-2 (f), not all node replicas in layer i affect the same number of node replicas in layer j. Only node replicas L^i-1 through M^i-L^i affect the maximum number of node replicas, L^i.

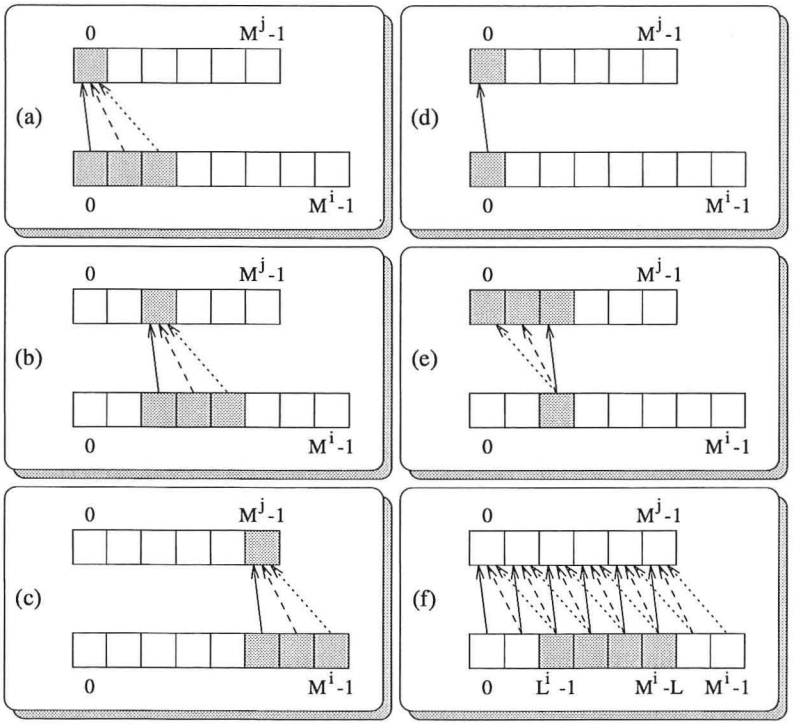

Figure A2.3-2. A simplified example ($N^i=N^j=1$, $M^i=8$, $M^j=6$ and $L^i=3$) showing the pattern of connectivity between layers i and j from the perspective of layer j (parts (a), (b) and (c); the traditional view as in Figure 4.2-1 (b)) and from the perspective of layer i (parts (d), (e) and (f)). The latter view is necessary to determine which nodes in layer i affect nodes in layer j, and through weighted connections with what weight? In this example, there are three unique weights represented by the solid, dashed and dotted arrows. Only the shaded nodes in part (f) affect the maximum number of nodes in layer j.

To account for the pattern of connectivity evident in Figure A2.3-2 (and more complicated TDNN examples), equation (A2.3-7) may be rewritten in terms of the unique weights as

$$\begin{aligned}\frac{\partial \mathscr{E}(p)}{\partial o^{i}_{n,m}(p)} &= \sum_{r=0}^{N^j-1}\sum_{s=\alpha}^{\beta}\frac{\partial \mathscr{E}(p)}{\partial v^{j}_{r,s}(p)}\cdot\frac{\partial v^{j}_{r,s}(p)}{\partial o^{i}_{n,m}(p)} \\ &= \sum_{r=0}^{N^j-1}\sum_{s=\alpha}^{\beta}\delta^{j}_{r,s}(p)\, w^{j(r,*)}_{i(n,m-s)}\end{aligned} \tag{A2.3-8}$$

where

$$\alpha = \begin{cases} 0 & m=0,1,\ldots,L^i-1 \\ m-L^i+1 & m=L^i,L^i+1,\ldots,M^j-1 \end{cases} \tag{A2.3-9}$$

and

$$\beta = \begin{cases} m & m=0,1,\ldots,M^i-L^i \\ M^j-1 & m=M^i-L^i+1,M^i-L^i+2,\ldots,M^j-1 \end{cases} \tag{A2.3-10}$$

For uniformity, the term $\delta^{i}_{n,m}(p)$ is defined as

$$\begin{aligned}\delta^{i}_{n,m}(p) &= \frac{\partial \mathscr{E}(p)}{\partial o^{i}_{n,m}(p)}\cdot\frac{\partial o^{i}_{n,m}(p)}{\partial v^{i}_{n,m}(p)} \\ &= \left[\sum_{r=0}^{N^j-1}\sum_{s=\alpha}^{\beta}\delta^{j}_{r,s}(p)\, w^{j(r,*)}_{i(n,m-s)}\right] o^{i}_{n,m}(p)\left(1-o^{i}_{n,m}(p)\right)\end{aligned} \tag{A2.3-11}$$

where the expressions replacing the two partial differential terms are obtained from equations (A2.3-8) and (A2.3-4), respectively. Rewriting (A2.3-3) using this expression and equation (A2.3-5), equation (A2.3-1) may be rewritten as

$$\Delta w_{h(a,l)}^{i(n,*)} = \frac{1}{M^i}\sum_{m=0}^{M^i-1}\left[\frac{-\eta_{h(a,l)}^{i(n,*)}}{P}\sum_{p=0}^{P-1}\delta_{n,m}^{i}(p)\,o_{a,m+l}^{h}(p)\right]$$
$$= \frac{-\eta_{h(a,l)}^{i(n,*)}}{M^i P}\left[\sum_{m=0}^{M^i-1}\sum_{p=0}^{P-1}\delta_{n,m}^{i}(p)\,o_{a,m+l}^{h}(p)\right] \qquad \text{(A2.3-12)}$$

where a=0,1,...,N^h-1, l=0,1,...,L^h-1 and n=0,1,...,N^i-1.

Similarly, the weight change required for the weighted connection joining each unique node in layer i to the bias node, may be deduced by averaging the weight changes evaluated for each of its replicas thus

$$\Delta w_b^{i(n,*)} = \frac{\sum_{m=0}^{M^i-1}\Delta w_b^{i(n,m)}}{M^i}$$
$$= \frac{1}{M^i}\sum_{m=0}^{M^i-1}\left[\frac{-\eta_b^{i(n,*)}}{P}\sum_{p=0}^{P-1}\delta_{n,m}^{i}(p).1\right] \qquad \text{(A2.3-13)}$$
$$= \frac{-\eta_b^{i(n,*)}}{M^i P}\sum_{m=0}^{M^i-1}\sum_{p=0}^{P-1}\delta_{n,m}^{i}(p)$$

Appendix 3

Statistical Tests

This appendix describes the statistical tests used in this work to compare the recognition and false-positive error performances of expert modules for closing diphthong recognition. When comparing various speech or phoneme recognition systems experimentally, two approaches may be taken to estimate such performances (be they recognition, false-positive error, or some other type of performances). The first is to use separate sets of speech utterances to train and test each system being compared so that their performances may be assumed to be *statistically independent.* This approach has the advantage that statistical tests for independent performances are simple are readily available within common statistical packages. However, given the usual limitations on the amount of speech data available, training and testing systems to be compared with separate sets of speech utterances necessitates that these tasks be conducted with less speech data. This practice may result in poorer, less realistic, recognition systems for comparison and/or less powerful statistical tests.

The second approach when estimating and comparing recognition system performances, is to use an identical set of utterances for training all the systems to be compared and another identical set for testing all these systems. This approach permits more speech data to be used when training and testing each system, but results in *correlated* performances. Given the small amount of speech data available for the experiments discussed in this thesis, the second approach to system training and testing has been adopted. The next section discusses *Cochran's generalized Q-test* which is used in this work to compare *correlated recognition performances.* This is followed in §A3.2 by a discussion of the *Games-Howell test* which is used to compare the *positively correlated false-positive error performances* observed in this work.

A3.1 Cochran's Generalized *Q*-test

Assuming the recognition performances of M recognition systems (*treatments*) are estimated and compared using N utterances (*blocks*), a *contingency table* like that shown in Table A3.1-1 may be composed (see Daniel 1990; Fleiss 1973). Each element X_{ij} in this table represents the response of the j^{th} recognition system to the i^{th} utterance and takes a value of

1 when this utterance is *correctly* recognized and *0* when is not (X_{ij} is a *dichotomous* random variable). These elements are assumed to be correlated within each *row*, since they represent the responses of different recognition systems to the *same* utterance. By contrast, the elements X_{ij} in each *column* are assumed to be independent, implying a recognition system's response to one utterance *must not* influence its response to another.[1] This condition is most easily satisfied by processing speech utterances (or portions thereof) which do not overlap in time, such as isolated words or syllables.

	Treatment					Block
Block	1	2	3	...	M	Totals
1	X_{11}	X_{12}	X_{13}	...	X_{1M}	S_1
2	X_{21}	X_{22}	X_{23}	...	X_{2M}	S_2
3	X_{31}	X_{32}	X_{33}	...	X_{3M}	S_3
.	.	.	.		.	.
.	.	.	.		.	.
.	.	.	.		.	.
N	X_{N1}	X_{N2}	X_{N3}	...	X_{NM}	S_N
Treatment Totals	T_1	T_2	T_3	...	T_M	T

Table A3.1-1. A *contingency table* for Cochran's Q-test. For experiments with speech recognition systems, the *treatments* are separate recognition systems trained and/or tested using the same sets of speech data, while the blocks are test utterances.

Cochran's Q-test examines the *null hypothesis* that the treatments (the recognition systems) are equally effective (Daniel 1990), implying a *two-sided* test (see Neter *et al* 1988). This hypothesis may be stated mathematically as H_0: $p_1 = p_2 = ... = p_M$; $p_{12} = p_{13} = ... = p_{M-1,M}$ (Berger and Gold 1973), where the *population* proportions, p_j, and joint proportions, p_{ij}, may be estimated using

$$\hat{p}_j = \frac{T_j}{N} \tag{A3.1-1}$$

and

[1]Due to this condition, speech or phoneme recognition systems that make use of prior or subsequent classifications when attempting to recognize their current input (perhaps using some form of language model) may not be compared using Cochran's Q-test (Gillick and Cox 1989).

$$\hat{p}_{ij} = \frac{\sum_{n=1}^{N} X_{ni} X_{nj}}{N} \qquad \text{(A3.1-2)}$$

respectively, where X_{ni} and X_{nj} are elements from the i^{th} and j^{th} columns of Table A3.1-1 and T_j is the *sum* of this table's column corresponding to the j^{th} treatment. Cochran's test statistic, Q, is given by

$$Q = \frac{M(M-1)\sum_{j=1}^{M} T_j^2 - T^2}{MT - \sum_{n=1}^{N} S_n^2} \qquad \text{(A3.1-3)}$$

(Fleiss 1973) where the S_n are the *block totals* given in Table A3.1-1 and T is their total. When H_0 is true, Q has a limiting chi-square distribution with $\nu=M-1$ degrees of freedom ($\chi^2(\nu)$) (Daniel 1990; Bhapkar and Somes 1977; Berger and Gold 1973).

In many experiments, including those involving speech recognition systems, one is often interested in testing the equivalence of the proportions, p_j, irrespective of the equality of the joint proportions, p_{ij}. When these joint proportions are not equal, the limiting distribution of Q is no longer $\chi^2(\nu)$. Consequently, to test the more general hypothesis H_0': $p_1 = p_2 = \ldots = p_M$, percentiles (critical values) of the limiting distribution for Q under this hypothesis are required. According to Wallenstein and Berger (1981), these percentiles are given approximately by

$$Q_\alpha(\varepsilon, \nu) = \frac{\chi_\alpha^2(\varepsilon\nu)}{\varepsilon} \qquad \text{(A3.1-4)}$$

where $\chi_\alpha^2(.)$ is the percentile of $\chi^2(.)$ corresponding to a *significance level* of α (see Daniel 1990), and ε is a test dependent constant that may be estimated using

$$\hat{\varepsilon} = \frac{\left(tr[\hat{V}B]\right)^2}{(M-1)\,tr[(\hat{V}B)^2]} \qquad \text{(A3.1-5)}$$

In this expression, *tr[]* signifies the *trace* operator (Kreyszig 1988) and B and $\hat{V}$ are square matrices whose elements are given by

$$B(i,j)=\begin{cases}\dfrac{(M-1)}{M} & i=j\\[2ex] -\dfrac{1}{M} & i\neq j\end{cases} \qquad \text{(A3.1-6)}$$

and

$$\hat{V}(i,j)=\begin{cases}\hat{p}_i+\hat{p}_M-2\hat{p}_{iM}-\hat{p}_i^{\,2}-\hat{p}_M^2+2\hat{p}_i\hat{p}_M, & i=j\\ \hat{p}_M+\hat{p}_{ij}-\hat{p}_{iM}-\hat{p}_{jM}+\hat{p}_M\left(\hat{p}_j+\hat{p}_i-\hat{p}_M\right)-\hat{p}_i\hat{p}_j, & i\neq j\end{cases} \qquad \text{(A3.1-7)}$$

respectively, where $i=j=1,2,...,M-1$ in both cases. When Cochran's original hypothesis H_0 is true, $\varepsilon=1$, otherwise it takes a value between $1/(M-1)$ and 1.

Wallenstein and Berger (1981) demonstrate that their approximate percentiles are reasonably accurate (on average) for sample sizes of $N=50$ and $N=100$ blocks and $\alpha=0.05$. Consequently, recognition performances ascertained with at least 50 utterances are likely to be tested satisfactorily using Cochran's Q-test in conjunction with the percentiles given by equation (A3.1-4). This test method is referred to as *Cochran's generalized Q-test* in this thesis, to indicate the assumptions concerning the joint proportions made by the original Q-test do not apply.

A3.2 The Games-Howell Test

Regrettably, Cochran's generalized Q-test may not be used to compare false-positive error performances, like those presented in chapter 5, since these do not result from *dichotomous* expert module responses. In particular, an expert module may produce two or more false-positive errors in response to syllable utterances like those used for testing in this work, making it impractical to represent the module's responses using dichotomous random variables. In this circumstance, one might be tempted to select shorter speech portions over which to test false-positive error performance so that only *one* false-positive error is possible per portion. However, such portions are difficult to select *a priori* (as is necessary to avoid biasing) unless they correspond to those used to generated individual input tokens. Individual input tokens only yield single responses when processed so that only one false-positive error is possible in response to each. However, neighbouring input tokens are generated from overlapping speech portions, implying the responses made by an expert module to such tokens

are *not independent*. This dependence invalidates the principal assumption of Cochran's Q-test that the blocks (responses to utterances) are independent. Consequently an alternative statistical test is required.

Using traditional statistical techniques, the false-positive error performances of a pair of expert modules might be analyzed by comparing their mean performances, denoted $\bar{U}_i$ and $\bar{U}_j$. Such analysis usually compares the *difference* between these means $(\bar{U}_j-\bar{U}_i)$ to zero and relies on having an estimate of $\sigma^2\{\bar{U}_j-\bar{U}_i\}$, the variance of $(\bar{U}_j-\bar{U}_i)$. $\sigma^2\{\bar{U}_j-\bar{U}_i\}$ is given generally by

$$\sigma^2\left\{\overline{U}_j-\overline{U}_i\right\}=\sigma^2\left\{\overline{U}_j\right\}+\sigma^2\left\{\overline{U}_i\right\}-2\sigma\left\{\overline{U}_i,\overline{U}_j\right\} \tag{A3.2-1}$$

where $\sigma^2\{\bar{U}_i\}$ and $\sigma^2\{\bar{U}_j\}$ are the variances of $\bar{U}_i$ and $\bar{U}_j$, respectively, and $\sigma\{\bar{U}_i,\bar{U}_j\}$ is the covariance between these means (Neter *et al* 1988). The variance $\sigma^2\{\bar{U}_j-\bar{U}_i\}$ may be estimated using

$$s^2\left\{\overline{U}_j-\overline{U}_i\right\}_C=s^2\left\{\overline{U}_j\right\}+s^2\left\{\overline{U}_i\right\}-2s\left\{\overline{U}_i,\overline{U}_j\right\} \tag{A3.2-2}$$

provided the estimates $s^2\{\bar{U}_i\}$, $s^2\{\bar{U}_j\}$ and $s^2\{\bar{U}_i,\bar{U}_j\}$ are also available (the subscript C implies *correlated* means are assumed). Using the square-root of $s^2\{\bar{U}_j-\bar{U}_i\}_C$, test statistics of the form

$$t=\frac{\left(\overline{U}_j-\overline{U}_i\right)-\left(\mu_j-\mu_i\right)}{s\left\{\overline{U}_j-\overline{U}_i\right\}_C} \tag{A3.2-3}$$

are common (see Neter *et al* 1988), where the population means μ_i and μ_j are assumed to be equal, implying the null hypothesis H_0: $\mu_j=\mu_i$ (or H_0: $\mu_j-\mu_i=0$ equally). Assuming the random variables U_j and U_j to which $\bar{U}_i$ and $\bar{U}_j$ correspond are approximately normally distributed, the test statistic t follows a *Student's t-distribution* when n_i and n_j, the number of samples used to estimate $\bar{U}_i$ and $\bar{U}_j$ respectively, are small.

In the circumstance where $\bar{U}_i$ and $\bar{U}_j$ are *positively correlated* (implying $\sigma\{\bar{U}_i,\bar{U}_j\}>0$) and a good estimate of $\sigma\{\bar{U}_i,\bar{U}_j\}$ is unavailable, the variance $\sigma^2\{\bar{U}_j-\bar{U}_i\}$ may be estimated (crudely) using

$$s^2\left\{\overline{U}_j-\overline{U}_i\right\}_I=s^2\left\{\overline{U}_j\right\}+s^2\left\{\overline{U}_i\right\} \tag{A3.2-4}$$

(see Neter *et al* 1988) which is *likely* to result in an *overestimate* (the subscript I implies independent means are assumed). If $(s^2\{\bar{U}_j-\bar{U}_i\}_I)^{1/2}$ is then used to evaluate a test statistic like that in equation (A3.2-3), the value of this statistic will be *smaller* than if a better estimate were used, resulting in a *conservative statistical test* (the probability of *type I error*, α, is actually lower than specified by the experimenter). A conservative test may have insufficient

power to find small (but real) differences significant, however, this is inconsequential if the differences $(\bar{U}_j\text{-}\bar{U}_i)$ to be analyzed are large. Fortunately, the main mean false-positive performances compared in this work exhibit large differences, permitting a conservative test to be used successfully.

(a) Speaker JK

	BT_1	ET_1	ST_1	BT_{10}	ET_{10}
BT_1	-				
ET_1	0.99	-			
ST_1	0.95	0.94	-		
BT_{10}	0.87	0.98	0.91	-	
ET_{10}	0.87	0.90	0.83	0.93	-
ST_{10}	0.48	0.51	0.34	0.57	0.52

(b) Speaker HD

	BT_1	ET_1	ST_1	BT_{10}	ET_{10}
BT_1	-				
ET_1	0.99	-			
ST_1	0.88	0.83	-		
BT_{10}	0.97	0.98	0.86	-	
ET_{10}	0.95	0.94	0.80	0.93	-
ST_{10}	0.28	0.27	0.42	0.25	0.26

Table A3.2-1. Correlations observed between the mean false-positive error performances of the various expert module types listed when processing 10 independent sets of each speaker's 240 closing diphthong syllables not used for TDNN training.

In this thesis, it is assumed that the *mean* false-positive error performances of each speaker's expert modules discussed in §5.1.3 and §5.1.4 are *positively correlated.* Evidence for this assumption was obtained using the following experiments. For each speaker, 10 independent groups of 24 utterances were formed (randomly) from their 240 closing diphthong syllables *not* used for TDNN training. These 10 groups were then processed using their associated speaker's BT_1s, ET_1s, ST_1s, BT_{10}s, ET_{10}s and ST_{10}s (see §5.1.3.1 and §5.1.4.1) to give 10 estimates of mean false-positive error performance for each module type. The correlations between the six sets of means were then estimated for each speaker and found to be *positive* in all cases as indicated by Table A3.2-1. This experiment was then repeated for each speaker's 160 monophthong syllables by dividing these into 10 groups of 16 (randomly chosen) utterances and processing them using the appropriate speaker's BT_{10}s,

ET_{10}s and ST_{10}s (only the false-positive error performances of these expert modules are compared, see §5.1.4.2). Once again, only positive correlations between the sets of means for each speaker were observed (these correlations were similar in magnitude to those listed in Table A3.2-1 for the BT_{10}s, ET_{10}s and ST_{10}s).

Since up to six means must be compared for each speaker at a time, the statistical test attributed to Games and Howell (1976) by Sokal and Rohlf (1981) is used to compare mean false-positive error performances in this thesis. For convenience, this test is referred to as the *Games-Howell test* henceforth. The Games-Howell test permits multiple pair-wise comparisons between an arbitrary number of *independent* means. These means may be estimated from different sample sizes and their populations need not have the same variances (the populations are assumed to be *heteroscedastic*) (Sokal and Rohlf 1981). Considering two of the k means to be compared, $\bar{U}_i$ and $\bar{U}_j$, the Games-Howell test indicates that the minimum significant difference (*MSD*) between these means is given by

$$MSD_{ij} = R_{\alpha[k,\nu^*]} \sqrt{s^2\left\{\bar{U}_j - \bar{U}_i\right\}_I} \qquad \text{(A3.2-5)}$$

(Sokal and Rohlf 1981), where $R_{\alpha[k,\nu^*]}$ is an element of the *studentized range* tabulated by Rohlf and Sokal (1981) (Table 18)[2] and ν^*, a weighted average degrees of freedom, is given by

$$\nu^* = \frac{(n_i - 1)(n_j - 1)\left(n_j s^2\{U_i\} + n_i s^2\{U_j\}\right)^2}{\left(n_j^3 - n_j^2\right)\left(s^2\{U_i\}\right)^2 + \left(n_i^3 - n_i^2\right)\left(s^2\{U_j\}\right)^2} \qquad \text{(A3.2-6)}$$

If $(\bar{U}_j - \bar{U}_i) \geq MSD_{ij}$, then the null hypothesis that the population means estimated by $\bar{U}_i$ and $\bar{U}_j$ are equal (H_0: $\mu_i = \mu_j$) is rejected. Since MSD_{ij} relies on the square-root of the variance estimate given by equation (A3.2-4), it is overly large when the means compared are positively correlated, making the Games-Howell test *conservative* in this circumstance.

[2]The symbol R is used to represent an element of the studentized range instead of Q (as in Sokal and Rolhf 1981) to avoid confusion with Cochran's Q statistic.

References

Abercrombie D. (1991), *Fifty Years in Phonetics*, Edinburgh University Press.

Ainsworth W.A. (1988), *Speech Recognition by Machine*, Peter Peregrinus Ltd on behave of the IEE.

Bauer L. (1986), "Notes on New Zealand English Phonetics and Phonology", *English World-wide*, Volume 7, Number 2, pp 225-258.

Bauer L. (in print), "The History of English in New Zealand", *Cambridge History of the English Language*, Volume 5.

Benediktsson J.A. and Swain P.H. (1992), "Consensus Theoretic Classification Models", *IEEE Transactions on Systems, Man and Cybernetics*, Volume 22, Number 4, July/August, pp 688-704.

Benediktsson J.A., Svensson J.R., Ersoy O.K. and Swain P.H. (1993), "Parallel Consensual Neural Networks", *International Conference on Acoustics, Speech and Signal Processing 1993*, Volume 1, pp 27-32.

Berger and Gold (1973), "Note on Cochran's Q-Test for the Comparison of Correlated Proportions", *Journal of the American Statistical Association*, Volume 68, Number 344, December, pp 989-993

Bernard J. (1970), "Towards the Acoustic Specification of Australian English", *Zeitschrift für Phonetik, Sprachwissenschaft und Kommunikationsforschung*, Volume 23, pp 113-128

Bhapkar V.P. and Somes G.W. (1977), "Distribution of Q When Testing Equality of Matched Proportions", *Journal of the American Statistical Association*, Volume 72, Number 359, pp 658-661.

Bladon A. (1985), "Diphthongs: A Case Study of Dynamic Auditory Processing", *Speech Communication*, Volume 4, pp 145-154.

Bolinger D. (1975), *Aspects of Language* (2nd Edition), Harcourt Brace Jovanovich Inc.

Bond Z.(1978), "The Effects of Varying Glide Durations on Diphthong Identification", *Language and Speech*, Volume 21, Part 3, pp 253-263.

Bond Z. (1982), "Experiments With Synthetic Diphthongs", *Journal of Phonetics*, Volume 10, pp 259-264

Bridle J.S. (1992), "Neural Networks or Hidden Markov Models for Automatic Speech Recognition: Is there a Choice?", *NATO ASI Series, Vol. F75: Speech Recognition and Understanding, Recent Advances*, Edited by P. Laface and R. De Mori, pp 225-236.

Broad D.J. and Clermont F. (1987), "A Methodology for Modelling Vowel Formant Contours in CVC Context", *Journal of the Acoustical Society of America*, Volume 81, Number 1, January, pp 155-165.

Carstairs-McCarthy A. (1989), *Consolidated Phonology Handout*, course notes for English 223: Theory of Linguistics I, University of Canterbury, Christchurch, New Zealand.

Catford J.C. (1988), *A Practical Introduction to Phonetics*, Clarendon Press

Chandra and Lin (1974), "Experimental Comparison Between Stationary and Nonstationary Formulations of Linear Prediction Applied to Voice Speech Analysis", *IEEE Transactions on Acoustics, Speech and Signal Processing*, Volume ASSP-22, Number 6, pp 403-415.

Church K.W. (1987), *Phonological Parsing in Speech Recognition*, Kluwer Academic Publications.

Clark J. and Yallop C. (1990), *An Introduction to Phonetics and Phonology*, Basil Blackwell.

Clark T. M. (1993), unpublished doctoral thesis entitled *A Study of Features and Processes Towards Real-Time Word Recognition*, University of Canterbury.

Clermont F. (1988), "A Dual Exponential Model for Formant Trajectories of Diphthongs", *SST88: The Second International Conference of Speech Science and Technology*, Sydney, Australia, pp 146-151.

Clermont F. (1991), *Formant-contour Models of Diphthongs: A Study in Acoustic Phonetics and Computer Modelling of Speech*, Ph.D. Thesis, The Australian National University.

Clermont F. (1992), "Characteristics of the Diphthongal Sound Beyond the F_1-F_2 Plane", *SST92: The Forth International Conference of Speech Science and Technology*, Australia, pp 298-303.

Cowie A.P. (1989), *Oxford Advanced Learners Dictionary of Current English,* Fourth Edition, Oxford University Press.

Crystal D. (1980), *A First Dictionary of Linguistics and Phonetics*, Andre Deutsh.

Daniel W.W. (1990), *Applied Nonparametric Statistics* (2nd Edition), PWS-Kent Publishing Company.

Davis R. and Davis M. (1987), *Sound System Engineering*, Sams.

Dawson M.I. and Sridharan S. (1992), "Speech Enhancement Using Time Delay Neural Networks", SST92

Denes P.B. and Pinson E.N. (1973), *The Speech Chain: The Physics and Biology of Spoken Language*, Anchor Books.

Devillers L. and Dugast C. (1994), "Hybrid System Combining Expert-TDNNs and HMMs for Continuous Speech Recognition", *International Conference on Acoustics, Speech and Signal Processing 1994*, Volume 2, pp 165-168.

Dixon N.R. and Martin T.B. (1979), *Automated Speech and Speaker Recognition*, IEEE Press.

Dolan W.B. and Mimori Y. (1986), "Rate-Dependent Variability in English and Japanese complex vowel F2 Transitions", *UCLA Working Papers in Phonetics*, Number 63 October.

Ederveen D. and Boves L. (1991), "Knowledge-based Phoneme Recognition", *Eurospeech 1991*, Volume 2, pp 421-424.

Elder A.G. (1992), *Evaluation of Glottal Characteristics for Speaker Identification*, Unpublished PhD Thesis, University of Canterbury.

Fallside F. (1992), "Neural Networks for Continuous Speech Recognition", *Speech Recognition and Understanding, Recent Advances; Nato ASI Series*, Volume F75.

Fant G. (1973), *Speech Sounds and Features*, MIT Press.

Fleiss J.L. (1973), *Statistical Methods for Rates and Proportions*, John Wiley and Sons.

Fry D.B. (1979), *The Physics of Speech*, Cambridge University Press.

Fu K.S. (1980), *Digital Pattern Recognition*, 2nd Edition, Springer-Verlag.

Fukushima K. (1987), "A Neural Network Model for the Mechanism of Selective Attention in Visual Pattern Recognition", *Systems and Computers in Japan*, Volume 18, Number 1, pp 102-113.

Fukushima K. and Imagawa T. (1993), "Recognition and Segmentation of Connected Characters With Selective Attention", *Neural Networks*, Volume 6, Number 1, pp 33-41.2

Games P.A. and Howell J.F. (1976), "Pairwise Multiple Comparison Procedures with Unequal N's and/or/ Variances: A Monte Carlo Study", *Journal of Educational Statistics*, Volume 1, Number 2, pp 113-125.

Gao Y., Huang T., Chen D. (1990), "HMM-based Warping in Neural Networks", *International Conference on Acoustics, Speech and Signal Processing 1990*, S10.4, Volume 1, pp 501-504.

Gillick and Cox (1989), "Some Statistical Issues in the Comparison of Speech Recognition Algorithms", *International Conference on Acoustics, Speech and Signal Processing 1989*, S10b.5, Volume 1, pp 532-535.

Gimson A.C. (1989), *An Introduction to the Pronunciation of English* (4th Edition), Edward Arnold.

Grayden D.B. and Scordilis M.S. (1992), "TDNN VS. Fully Interconnected Multilayer Perceptron: A Comparative Study on Phoneme Recognition", *SST-92*, pp 214-219.

Haffner P., Waibel A., Sawai H. and Shikano K. (1989), "Fast Back-Propagation Learning Methods for Large Phonemic Neural Networks", *Proceedings of Eurospeech*, September, pp 553-556.

Hanes M.D., Ahalt S.C., Krishnamurthy A.K., "Acoustic-to-Phonetic Mapping Using Recurrent Neural Networks", *IEEE Transactions on Neural Networks*, Volume 4, Number 5, July, pp 659-662.

Hansen L. K. and Salamon P. (1990), "Neural Network Ensembles", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Volume 12, Number 10, pp 993-1001,

Hataoka N. and Waibel A. (1990), "Speaker-Independent Phoneme Recognition on TIMIT Database Using Integrated Time-Delay Neural Networks (TDNNs)", *International Joint Conference on Neural Networks (IJCNN)*, San Diego, Volume 1, pp 57-62.

HardCastle W.J. (1976), *Physiology of Speech Production - An Introduction for Speech Scientists*, Academic Press.

Harris F.J. (1978), "On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform", *Proceedings of the IEEE*, Volume 66, Number 1, January, pp 51-83.

Hawkins P. (1973), "A Phonemic Transcription System for New Zealand English", *Te Reo*, Volume 16, pp 15-21

Haykin S. (1994), *Neural Networks - A Comprehensive Foundation*, MacMillan College Publishing Company Inc.

Hemranski H. (1990), "Perceptual Linear Predictive (PLP) Analysis of Speech", *Journal of acoustic Society of America*, Volume 87, Number 4, April.

Hinton G.E., McClelland J.L. and Rumelhart D.E. (1986), "Distributed Representation", chapter 3 in *Parallel Distributed Processing, Volume 1: Foundations*, MIT Press, pp 77-109

Hinton G.E. and Sejnowski T.J. (1986), "Learning and Relearning in Boltzmann Machines", chapter 7 in *Parallel Distributed Processing, Volume 1: Foundations*, MIT Press, pp 318-364

Holmes J. and Bell A. (1992), "On shear markets and sharing sheep: The merger of EAR and AIR diphthongs in New Zealand English", *Language Variation and Change*, Volume 4, pp 251-273.

Holmes J. (1994), "NZ Setting Trend in Speech", *New Zealand Herald*, Monday 24 October.

Holmes W.J. and Pearce D.J.B (1993), "Sub-word Units for Automatic Speech Recognition of Any Vocabulary", *GEC Journal of Research*, Volume 11, Number 1, pp 49-59.

Jacobs R.A. (1988), "Increased Rates of Convergence Through Learning Rate Adaption", *Neural Networks*, Volume 1, pp 295-307

Jones D. (1967), *The Pronunciation of English*, 4th Edition (reprint), Cambridge University Press.

Jordan M.I. and Jacobs R.A. (1992), "Hierarchies of Adaptive Experts", in *Advances in Neural Information Processing Systems 4*, Morgan and Kaufmann, pp 985-992.

Kasabov N.K, Nikovski D., Peev E. (1993), "Speech Recognition Based on Kohonen Self Organizing Feature Maps and Hybrid Connectionist Systems, *ANNES; The First New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, Dunedin, New Zealand, November 24-26, pp 113-117.

Kasabov N.K, Shiskov S.I. (1993), "A Connectionist Prediction System with Partial Match and its Use for Approximate Reasoning", *Connection Science*, Volume 5, Number 3&4, pp 275-305.

Komori Y. (1991), "Time-State Neural Networks (TSNN) for Phoneme Identification by Considering Temporal Structure of Phonemic Features", *International Conference on Acoustics, Speech and Signal Processing 1991*, S2.22, pp 125-128.

Korff S.A. (1955), *Electron and Nuclear Counters, Theory and Use*", Van Nostrand Company.

Kreyzig E. (1988), *Advanced Engineering Mathematics*, Sixth Edition, John Wiley and Sons.

Ladefoged P (1982), *A Course in Phonetics*, Second Edition, Harcourt Brace Jovanovich Publishers.

Lang K. J. and Hinton G.E. (1988), "A Time-Delay Neural Network Architecture for Speech Recognition", *CMU Technical Report CMU-CS-88-152*, December.

Le Cerf P. and Van Compernolla D (1993), "Using Parallel MLPs as Labellers for Multiple Codebook HMMs", *International Conference on Acoustics, Speech and Signal Processing 1993*, Volume 1, pp 561-564.

Lee K.F. (1990), "Context-Dependent Phonetic Hidden Markov Models for Speaker-Independent Continuous Speech Recognition", *IEEE Transactions on Acoustics Speech and Signal Processing*, Volume 38, Number 4, April, pp 599-609.

Lee K.F. and Alleva F. (1992), "Continuous Speech Recognition", in *Advances in Speech Signal Processing*, edited by S. Furui and M.M. Sondhi, Marcel Dekker Inc, pp 623-650.

Lehiste I. and Peterson G. (1961), "Transitions, Glides, and Diphthongs", *Journal of the Acoustical Society of America*, Volume 33, Number 3, March, pp 268-277.

Lieberman P. and Blumstein S.(1988), *Speech Physiology, Speech Perception, and Acoustic Phonetics*, Cambridge University Press.

Lippman R.P, Martin E.A. and Paul D.P. (1987), "Multi-style Training for Robust Isolated-word Speech Recognition", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, April, pp 705-708.

Lippmann R.P. (1987), "An Introduction to Computing with Neural Networks", *IEEE ASSP Magazine*, April, pp 4-22.

Lippmann R.P. (1989), "Review of Neural Networks for Speech Recognition", *Neural Communication*, Volume 1, pp 1-38.

Lippmann R.P. and Singer E. (1993), "Hybrid Neural-Network/HMM Approaches to Wordspotting", *International Conference on Acoustics, Speech and Signal Processing 1993*, Volume 1, pp 565-572.

Maclagan M. (1975), "Thoughts on New Zealand English", *New Zealand Speech Therapists' Journal*, Volume 30, Number 1, May, pp 6-10.

Maclagan M. (1982), "An Acoustic Study of New Zealand Vowels", *New Zealand Speech Therapists' Journal*, Volume 37, Number 1, May, pp 20-26.

McCandless S. (1974), "An algorithms for Automated Formant Extraction Using Linear Prediction Spectra", *IEEE Transactions on Acoustics, Speech and Signal Processing*, Volume ASSP-22, April, pp 135-141.

Makhoul J. (1975), "Linear Prediction: A Tutorial Review", *Proceeding of the IEEE*, Volume 64, Number 4, April, pp 561-580.

Matthei E. and Roeper T. (1983), *Understanding and Producing Speech*, Fontana Paperbacks.

Miller G.A. (1981), *Language and Speech*, W.H. Freeman and Company San Francisco.

Minami Y., Sawai H. and Miyatake M. (1991), "Large-Vocabulary Spoken Word Recognition Using Time-Delay Neural Network Phoneme Spotting and Predictive LR-Parsing", *Systems and Computers in Japan*, Volume 22, Number 1, pp 99-107.

Mitchell R. and Shaw A. (1990), "Vowel Recognition with a Time-Delay Neural Network", *IEEE International Conference on Systems Engineering*, Pittsburgh, pp 637-640.

Miyatake M., Sawai H., Minami Y. and Shikano K. (1990), "Integrated Training for Spotting Japanese Phonemes Using Large Phonemic Time-Delay Neural Networks, *ICASSP 1990*, S8.10, pp 449-452.

Morgan D.P. and Scofield C.L. (1991), *Neural Networks and Speech Processing*, Kluwer Academic Publishers.

Morgan N. and Bourlard H (1990), "Continuous Speech Recognition Using Multilayer Perceptrons with Hidden Markov Models", *ICASSP-90*, Volume 1, pp 413-416.

Moulton W.G. (1969), "The Nature and History of Linguistics", in *Linguistics* edited by A.A. Hill, Voice of America Forum Lectures.

Neter J., Wasserman W. and Whitmore G.A. (1988), *Applied Statistics* (3rd edition), Allyn and Bacon Inc.

Okada H. (1991), "Illustrations of the IPA: Japanese", *Journal of the International Phonetic Association*, Volume 21, Number 2, pp 94-96.

Owens F.J. (1993), *Signal Processing of Speech*, McGraw-Hill Inc.

Pao Y. (1989), *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley Publishing Company.

Penny L. (1992), "Acoustic Measurements of the Diphthongs of Woman speakers of General Australian English", *SST92: The Forth International Conference of Speech Science and Technology*, Australia, pp 489-494.

Picone J.W. (1993), "Signal Modelling Techniques in Speech Recognition", *Proceedings of the IEEE*, Volume 81, Number 9, September.

Rabiner L.R. and Levinson S.E. (1981), "Isolated and Connected Word Recognition - Theory and Selected Applications", *IEEE Transactions on Communications*, Volume COM-29, Number 5, pp 621-659.

Rabiner L.R., Wilpon J.G., Soong F.K. (1988), "High Performance Digit Recognition Using Hidden Markov Models", presented at *IEEE International Conference on Acoustics, Speech and Signal Processing*, April.

Rabiner L.R. (1989), "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", *Proceedings of the IEEE,* Volume 77, Number 2, February, pp 257-285.

Reddy D.R. (1976), "Speech Recognition by Machine: A Review", *Proceedings of the IEEE*, Volume 64, pp 501-531, April.

Renals S. Morgan N. Cohen M. and Franco H. (1992), "Connectionist Probability Estimation in the Decipher Speech Recognition System", *ICASSP-92*, Volume 1, pp 601-?

Richards J., Platt J., Weber H.(1987), *Longman Dictionary of Applied Linguistics*, Longman, (Second Impression).

Robinson T., Hochberg M. and Renals S. (1994), "IPA: Improved Phone Modelling with Recurrent Neural Networks", *International Conference on Acoustics, Speech and Signal Processing 1994*, Volume 1, pp 37-40.

Rohlf F.J. and Sokal R.R. (1981), *Statistical Tables*, W.H. Freeman and Company.

Roe D.B. and Wilpon J.G. (1993), "Whither Speech Recognition: The Next 25 Years", *IEEE Communications Magazine*, November, pp 54-62.

Rumelhart D.E., Hinton G.E. and Williams R.J. (1986), "Learning Internal Representations by Error Propagation", chapter 8 in *Parallel Distributed Processing, Volume 1: Foundations*, MIT Press, pp 318-364

Sagayama S., Sugiyama M., Ohkura K., Takami J., Nagai A., Singer H., Hattori H., Fukuzawa K., Kato Y., Yamaguchi K., Kosaka T., and Kurematsu A. (1992), "ATREUS: Continuous Speech Recognition Systems at ATR Interpreting Telephony Research Laboratories", *SST 92*, pp 324-329.

Sawai H., Waibel A., Miyatake M. and Shikano K. (1989), "Spotting Japanese CV-Syllables and Phonemes Using Time-Delay Neural Networks", *ICASSP 1989*, S1.7, pp 25-28.

Sawai H. (1991), "TDNN-LR Continuous Speech Recognition System Using Adaptive Incremental TDNN Training", *ICASSP 1991*, S2.4, pp 53-56.

Seneff S. (1976), "Modifications to Formant Tracking Algorithm of April 1974", *IEEE Transactions on Acoustics, Speech and Signal Processing*, Volume ASSP-24, April, pp 192-193.

Singh S. and Singh K.S. (1976), *Phonetics Principles and Practices*, University Park Press.

Sokal R.R. and Rohlf F.J. (1981), *Biometry*, Second Edition, W.H. Freeman and Company.

Song J.M. (1992), "A Study on the Combinations of Hidden Markov Models and Multilayer Perceptrons for Speech Recognition", *SST92: The Forth International Conference of Speech Science and Technology*, Australia, pp 448-453.

Stremler G.F. (1982), *Introduction to Communication Systems*, Second Edition, Addison-Wesley Publishing Company.

Waibel A., and Yegnanarayana B. (1981), "Comparative Study of Nonlinear Time Warping Techniques in Isolated Word Speech Recognition Systems", *Carnegie-Mellon University Technical Report* CMU-CS-81-125.

Waibel A., Hanazawa T., Hinton G., Shikano K. and Lang K. (1987), "Phoneme Recognition Using Time-Delay Neural Networks", Technical Report TR-1-0006, *ATR Interpreting Telephony Research Laboratories*, Japan.

Waibel A., Hanazawa T., Hinton G., Shikano K., Lang K. (1989a), "Phoneme Recognition Using Time-Delay Neural Networks", *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. 37, Number 3, March, pp 328-339.

Waibel A., Sawai H. and Shikano K. (1989b), "Modularity and Scaling in Large Phonemic Neural Networks", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Volume 37, Number 12, December, pp 1888-1897.

A. Waibel, H. Sawai and K.Shikano (1989c), "Consonant Recognition by Modular Construction of Large Phonemic Time-Delay Neural Networks", *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, S3.9, pp 112-115.

Waibel A. and Hampshire J. (1989), "Building Blocks for Speech", *Byte*, August, pp 235-242.

Waibel A. (1992a), "Connectionist Large Vocabulary Speech Recognition", *Proceedings of the NATO Advanced Science Institute on Speech Recognition and Understanding - Recent Advances, Trends and Applications*, held July 1-13 1990 (Italy), pp 259-273.

Waibel A. (1992b), "Neural Network Approaches for Speech Recognition", in *Advances in Speech Signal Processing*, Edited by S. Furui and M. Sondhi, Dekker, chapter 18, pp 555-595.

Wallenstein S. and Berger A. (1981), "On the Asymptotic Power of Tests for Comparing K Correlated Proportions", *Journal of the American Statistical Association*, Volume 76, Number 373, pp 114-118.

Watrous R.L. (1990), "Phoneme Discrimination Using Connectionist Networks", *Journal of the Acoustics Society of America*, Volume 87, Number 4 April, pp 1753-1772.

Witten I.H. (1982), *Principles of Computer Speech*, Academic Press.

Zue V.W. and Seneff S. (1988), "Transcription and Alignment of the Timit Database", *Proceedings of the Second Meeting on Advanced Man-Machine Interface Through Spoken Language*, pp 464-473.

Zee E. (1991), "Illustrations of the IPA: Chinese (Hong Kong Cantonese)", *Journal of the International Phonetic Association*, Volume 21, Number 1, pp 46-48.