Self-Organising Maps and Embodied Conversational Agents for Computer Aided Language Learning

by

Tom Adam Frederic Anderson

Thesis
Submitted to Flinders University
for the degree of

Doctor of Philosophy

College of Science and Engineering

January 31, 2019

Contents

List of Figures

List of Tables

Abstract

Learning a language with a computer can be more affordable, personalisable, and repeatable than with a human teacher, but it is a challenge to deliver maximum results on those capabilities. Towards this goal, two problems for computer-aided language learning were addressed with an emphasis on pronunciation: (1) the evaluation and visualisation of pronunciation using Self-Organising Map (SOM), and (2) the use of Embodied Conversational Agents (ECAs) for language learning.

SOMs are a type of neural network that use a competitive algorithm for classification, clustering, and visualisation known for their unsupervised learning capabilities. The Kohonen Neural Phonetic Typewriter (KNPT) is a niche application of the SOM algorithm with the phonemes of speech. A review of the KNPT literature from the 1980s to the present reveals many of the common practices and identifies gaps.

Towards a practical evaluation and visualisation of pronunciation, parameters including feature representations, normalisation and map sizes were explored via grid search and cross fold validation. The use of the KNPT for the speech of single speaker was explored. MFCCs were found to enable significantly better classification of frames than spectrograms. It was found that the KNPT benefits from maps (e.g. 50x50) that are larger than those reported in the literature (seldom exceeding 25x25).

A multi-layer architecture was implemented, and two methods of subsetting data in the first layer were compared— auxiliary maps for disambiguating easily confused phonemes were compared with maps for silence, vowels, and consonants—with the finding that the latter had better performance. The multi-layer KNPT was used to investigate the phonemes of Australian English in the speech of different demographic groups in Austalk, a large corpus. An experiment with the KNPT for voice activity detection (VAD) demonstrated that even small maps (6x6) are able to perform this task.

Language learning with a computer can be enhanced by creating personalised and grounded experiences through the use of ECAs and virtual worlds. A system was implemented that used an ECA for video dubbing language learning task, with the finding that learners expect such a system to provide an evaluation of pronunciation. A novel framework for using ECAs for storytelling was presented that enables multimodal language learning. Finally, a system based on the KNPT for pronunciation learning through video games with ECAs was introduced. The experiments and frameworks presented in this thesis demonstrate the future of Computer Assisted Pronunciation Teaching (CAPT).

Certification

I certify that this thesis does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief it does not contain any material previously published or written by another person except where due reference is made in the text.

Signed  Dated 12 November, 2017

Tom Adam Frederic Anderson

Acknowledgements

Although this thesis has only a single name on its cover, it would not have been accomplished without the support, academic and otherwise, of other people. I therefore wish to extend my sincerest thanks to the following individuals who have facilitated this work.

- I would like to express great appreciation to my supervisors, Prof. David M. W. Powers, Dr Trent Lewis, and Dr Richard Leibbrandt, for helping to guide me towards conducting the research in this thesis.
- I wish to acknowledge the help of Prof. Chiu Yi-Hui, for collaboration with the ECA dubbing project, which further inspired the work into evaluation of pronunciation.
- I would like to offer special thanks to Prof. Barry Lee Reynolds, for motivation and collaboration.
- I would also like to thank Prof. David M. W. and Sue Powers, for proofreading my thesis. I also acknowledge that a 1463 word section of Chapter 1 was proofread by Studiosity, which provided minor feedback on grammar, punctuation and spelling.
- Special thanks to my family, especially Kate Huang and Snow Anderson, for their constant support through the long journey.
- I also thank the many others who collaborated, cowrote, or assisted along the way.
- The AusTalk corpus was collected as part of the Big ASC project (Burnham et al. (2009,1); Wagner et al. (2010), funded by the Australian Research Council (LE100100211). See: https://austalk.edu.au/ for details.
- The contribution of the Flinders University Research Scholarship and the Australian Postgraduate Award, an Australian Government Scholarship.
- The Microsoft Student Partners Program in Australia, which allowed me access to technical training and technical events, the first of several Microsoft Kinect sensors that were used towards the work presented in this thesis, and the opportunity to help my peers advance their own technical skills.

Publications

During the course of this study, the following refereed conference and journal papers were published.

Anderson, T. A. F., Reynolds, B. L., & Powers, D. M. W. (2017). An Approach to Accent Visualisation for the Reduction of Vowel Pronunciation Errors. In Chen, W., Yang, J.-C., & Ayub, A. F. M. (Eds), *Proceedings of the 25th International Conference on Computers in Education. New Zealand: Asia-Pacific Society for Computers in Education.*[1]

Anderson, T. A. F., & Powers, D. M. W. (2016). Characterisation of speech diversity using self-organising maps. *16th Australasian International Conference on Speech Science and Technology (SST2016).* Sydney, Australia: Australasian Speech Science and Technology Association.[1]

Chiu, Y. H., Anderson, T. A. F., & Powers, D. M. W. (2012). Dubbing of virtual embodied conversation agents for improving pronunciation. In J. Colpaert, A. Aerts, W. Wu & Y. C. Chan (Eds.), *Proceedings of the Fifteenth International CALL Conference* (p. 188-192). Providence University, Taichung, Taiwan.[1]

Anderson, T. A. F., Chen, Z.H., Wen, Y.-F., Milne, M., Atyabi, A., Treharne, K., Matsumoto, T., Jia, X.-B., Luerssen, M., Lewis, T., Leibbrandt, R., & Powers, D. M. W. (2012). Thinking head MulSeMedia. In Ghinea, G., Andres, F., & Gulliver, S. R. (Eds.), *Multiple Sensorial Media Advances and Applications: New Developments in MulSeMedia* (pp. 182-203). Hershey, PA: IGI Global.[1]

[1]Publication arising out of work conducted during candidature and included in the body of the thesis in accordance with Flinders HDR Thesis Rules.

Reynolds, B. L., & Anderson, T. A. F. (2015). Extra-dimensional in-class communications: action research exploring text chat support of face-to-face writing. Computers and Composition. 35, pp. 52-64.†

Wen, Y.-F., Anderson, T. A. F., & Powers, D. M. W. (2014). On energy-efficient aggregation routing and scheduling in IEEE 802.15.4-based wireless sensor networks. *Wireless communications and mobile computing*, 14(2). pp. 232-253.†

Powers, D. M. W., Atyabi, A., & Anderson, T. A. F. (2012). MAGIC robots-resilient communications: A guaranteed redundant expansible mesh built on unreliable media. In *Engineering and Technology (S-CET)*. IEEE.†

Anderson, T. A. F., Jia, X.-B., Lewis, T., Luerssen, M., Leibbrandt, R., & Powers, D. M. W. (2010). Visual speech for a Mandarin thinking head. *HCSNet Workshop on Natural User Interfaces: Multitouch and Gestural Interactions.* Melbourne: RMIT University. June 28-29.†

Anderson, T. A. F., Lewis, T., Luerssen, M., Jia, X.-B., Leibbrandt, R., & Powers, D. M. W. (2010). An application of mouth tracking for language learning. *HCSNet Workshop on Advances in Speech Production: Tools, Techniques and Recent Research.* Sydney: MARCS Auditory Laboratories, University of Western Sydney. 23-24 April.†

Anderson, T. A. F., Leibbrandt, R., Lewis, T., Luerssen, M., & Powers, D. M. W. (2009) Poverty of stimulus in language education. *HCSNet SummerFest Speed Papers.* Sydney: MARCS Auditory Laboratories, University of Western Sydney. 30 November - 4 December.†

Atyabi, A., Anderson, T. A. F., Treharne, K., & Powers, D. M. W. (2010). Magician simulator-A realistic simulator for heterogenous teams of autonomous robots. Control Automation Robotics & Vision (ICARCV). Singapore: IEEE.†

†Publication not included in the thesis.

Terminology

ASR automatic speech recognition. 3, 16, 17, 19, 30, 82, 99, 117, 138, 140, 156, 191, 195

BMU best matching unit. 37, 38, 61–63, 79, 89, 95, 103, 112, 126, 144, 196

CALL Computer Aided Language Learning. 3, 5, 6, 51, 152, 156, 162, 190–192

CAPT Computer Assisted Pronunciation Teaching. iv, 6, 155, 156, 193

DCT Discrete Cosine Transform. 16, 19, 42

ECA Embodied Conversational Agent. iv, vi, 3, 4, 6, 152, 154–161, 190, 192, 193, 199

FFT Fast Fourier Transform. 15–17, 42–44, 71, 83, 101, 102

HMM Hidden Markov Model. 17, 19, 30, 32, 42, 48, 55, 56, 60, 63, 65–67, 78

IPA International Phonetic Alphabet. 11, 12, 75, 80, 86, 105, 113, 120–122, 202

KNPT Kohonen Neural Phonetic Typewriter. iv, 3–6, 8, 22, 23, 28–34, 36, 37, 40–76, 79, 82, 87, 89, 92–95, 97–99, 101–107, 109, 111, 114–117, 124–130, 135–139, 144, 148, 150, 156, 183, 190–199, 202, 203

LPC Linear Prediction Coding. 43, 116

LPCC Linear Prediction Coding Coefficient. 14, 40, 43, 44, 75, 99

LVQ Learning Vector Quantisation. 33, 43, 45, 46, 50, 54, 59, 60, 62, 67, 72, 75, 109, 193, 196, 197

MFCC mel frequency cepstral coefficient. iv, 5, 14, 16, 17, 19–21, 23, 34, 40, 42–45, 52, 55, 63, 64, 66, 68, 72, 74, 75, 79, 81, 99, 101–104, 116, 120, 126, 131, 133, 136, 137, 139–142, 144–148, 191, 195–197

MLP multi-layer perceptron. 44, 49–51, 53, 56, 65, 67, 75, 194

mulsemedia multiple sensorial media. 6, 162, 164–166, 177, 178, 181, 182, 192, 193

PCA Principal Components Analysis. 36, 85

PER phoneme error rate. 78, 79, 95, 98, 114, 130

SAMPA Speech Assessment Methods Phonetic Alphabet. 11, 12, 201, 202

SOM Self-Organising Map. iv, 3–5, 8, 14, 15, 28–59, 61–75, 77, 79, 80, 82, 84–86, 90–92, 97–99, 104, 107, 111, 113–115, 117, 120, 123–129, 131, 132, 136, 137, 139, 141, 143, 144, 148, 150, 161, 190–198, 200

VAD voice activity detection. iv, 5, 53, 101, 102, 129, 138, 139, 143, 144, 147, 148, 150, 190–192

VER vowel error rate. 130, 137

Chapter 1

Introduction

1.1 Background and Motivation

As a teacher of English in Taiwan, I taught English at various levels, including to graduate students at National Taiwan University of Science and Technology and Fu Jen Catholic University. I also edited and proofread dozens of journal articles. I was also a guest on an English language learning TV show host on my colleague Professor Chiu Yi-Hui's show on the Chinese Television System (CTS). From 2007 to 2009, I studied for a Master's degree at the Graduate Institute of Network Learning Technology at National Central University in Taiwan.

During my Master's degree, I was involved a number of research studies that resulted in publications. In Anderson et al. (2008b), I collaborated with three classmates to create and test a language learning lesson based on a video game, America's Army. We tested the effects on language learning of instruction on how to play the game compared to vocabulary training to better understand the in-game speech. I cowrote the paper and presented it. During this time, I also collaborated with Dr. Wen, a Taiwanese professor with research interests in RFID and wireless sensor networks. I assisted in the writing of a conference paper on optimising algorithms for distributed resource and routing assignments in wireless mesh networks, which was published as Wen & Anderson (2008). Further to this collaboration, I wrote Anderson & Wen (2008), which incorporated ideas about how to apply wireless sensor networks for language learning in an outdoor setting and was presented by my colleague at the conference. I also assisted a classmate with a paper based on her Master's research towards learning Chinese with mobile devices, which was presented by the advisor as Anderson et al. (2008a).

My own Master's degree research focussed on learning English vocabulary with video games. Towards this end, I created five networked video games that ran in Adobe Flash (I programmed the full stack from scratch in ActionScript and PHP). I tested them in a classroom of elementary school students in Taiwan to play on personal laptops, once a week for five sessions. These

games involved reading, listening, and writing (using the laptop keyboard), as well as an element of fun. I varied the sequence of the presentation of vocabulary to enhance retention, with the basic idea to predict when a new word was almost forgotten, then to reteach it. I investigated how the order of presentation could improve the retention of vocabulary learning. I published the results on gaining English vocabulary through minigames as my Master's thesis (Anderson, 2009) and a conference paper (Anderson et al., 2009). Even though my games lacked any form of speaking, I saw that the students often spontaneously spoke aloud the vocabulary words as they played my games. However, the system lacked any capabilities of speech recording or automatic speech recognition (ASR), and as I found out, there may not yet be a really good and fast way to evaluate the speech of second language learners, or first language learners for that matter.

My PhD research was conducted and written in its entirety at Flinders University. The PhD candidature was full time from September 2009 to Mar 2014 and part-time from Mar 2014 to Dec 2017. I had approved periods of intermissions from January 2012 to February 2012, February 2013 to October 2013, and August 2014 to August 2015.

Initially at Flinders University, I gained experience with Embodied Conversational Agents (ECAs) and the Microsoft Kinect. I also participated in the MAGIC robot competition, with my main contributions being the implementation and testing of computer networking for our team of robots. I shifted the scope of my research plan from studying language learning in an intelligent room to working on language learning of listening and speaking with ECAs at a computer. I found that there was a need for the evaluation of pronunciation to support the language learning applications that I had in mind. On the advice of Professor Powers, my primary PhD supervisor, I investigated a type of artificial neural network known as the Kohonen Neural Phonetic Typewriter (KNPT), a Self-Organising Map (SOM) used for the classification of speech at the phoneme level. A main emphasis of this work regards those aspects of the KNPT that would enable it to be a useful tool for language learning applications.

Although my Master's research was quite different than the PhD research that appears in this thesis, and there is no overlap of the research or publications, my observations of what my systems lacked were a main inspiration for the current research on speech evaluation and Computer Aided Language Learning (CALL) with an ECA. To this end, this thesis is divided into two parts: the first covers SOM for English vowels; the second, computer-based systems for CALL. The former emphasises pattern recognition by a computer; the latter, using a computer to teach humans.

1.2 Part One

Chapter 2 – A main thrust of the current research to be presented in this thesis was towards a better understanding of speech and its evaluation and visualisation for CALL. The sounds of speech are created by pushing air through the vocal chords with change to the sound coming from articulators of lips, tongue and jaw. A phoneme is the smallest part of speech that distinguishes meanings. A computer system can accept speech as an input via microphones, but the goal for is towards a system that can evaluate phonemes and show a map that shows how phonemes are related to one another. Thus, a stream of sound can be converted into a vector of features, allowing the training of a model that can be used to describe the various sounds of speech. In this chapter, I introduce the fundamentals of speech research. I cover speech representations, including spectrograms and MFCCs. I additionally touch on the testing methods and statistics used in the thesis.

Chapter 3 – The SOM algorithm is a "clustering, visualization, and abstraction method" (Kohonen, 1995, p. VII). Rather than explore many different approaches to clustering phonemes, it was decided to restrict study to the SOM for several reasons: it has good accuracy for the task of phoneme classification reported in the literature; unlike many other approaches to classification and clustering, it clusters the data topologically; although training can take a long time, once trained, classifications are quickly determined using the maps; and it permits training on data that is not labelled (unsupervised learning). In this thesis, I explore the task of pronunciation evaluation through the use of SOMs with a goal of using its unsupervised mapping and clustering for language learning with an ECA. Although previous literature reviews have covered the idea of SOMs, there have not been any literature surveys devoted to SOM for phonemes. In literature review, I reveal a number of aspects of KNPT that many researchers do not address, such as comparisons between feature representations, suitable map sizes or methods of normalisation. These aspects gain attention in the following chapters.

Chapter 4 – Most commonly represented as a two-dimensional grid, a SOM learns to represent distributions of the data as prototypes located in a regular fashion. The organisation of the prototypes can reveal topographic aspects of the data, where the distances between classes associated with models may reveal information about the relationships between the classes. Additionally, by comparing a new data point with the different models, those models that most closely match with the data point can be used to classify the data point. A two-dimensional SOM provides a visual way to inspect data on paper or on a computer screen in order to reveal its patterns. The way that a SOM is trained means that, in general, feature vectors that are similar

to each other according to the distance metric will be located near each other on the map. As a result, the choice of feature representation can have a significant impact on results. I created a number of speech datasets with the speech of a single speaker with attention to feature representations and normalisation.

Chapter 5 – To personalise application of the KNPT requires training. The problem is that training time can grow large. For example, Kohonen (2013, p. 59) wrote, "With very large maps, both winner-search and updating are time-consuming operations." In this chapter, I explore the use of a hierarchy of maps.

Chapter 6 – Voice activity detection (VAD) is an important consideration for speech processing because it enables the separation of speech from silence. My work in the earlier chapters revealed that VAD could be performed with SOM, yet only a single study has reported using SOM for VAD (Anderson, 1991). In this chapter, I address the gap in the research regarding the detection of silence for KNPT. Using the mel frequency cepstral coefficient (MFCC) feature representation, I tested map sizes and normalisation techniques, finding that even small 6x6 SOMs can perform VAD adequately.

1.3 Part Two

The focus of the earlier chapters was on machine learning. The other part of my research is CALL. The former emphasises pattern recognition by a computer, whilst the latter is how to use a computer to teach humans. This final portion of this thesis outlines current efforts to move from the earlier portions of the research that investigated unsupervised KNPT language learning towards a practical system for language learners.

Chapter 7 — During my PhD candidature at Flinders University, I remained in contact with Professor Chiu, Director of the Centre for Teaching and Learning in the Department of Applied Foreign Languages at National Taipei University of Business. One of Chiu's research interests is film dubbing, which is an investigation of how the activity of dubbing can facilitate EFL learners' acquisition of English pronunciation. The dubbing task usually uses a video to go along with the audio, but I proposed to use an ECA instead, which can convert any text into synthetic speech. Unlike lessons that rely on videos, which must be prerecorded, ECA dubbing may be scripted more flexibly as it is produced on the fly. In this chapter, I report on this research performed in collaboration with Chiu, for which I created a virtual avatar dubbing task for one-to-one language learning. Chiu tested my software with students. On the basis of Chiu's phenomenological testing of the system, I wrote and Chiu presented a conference paper for the CALL 2012 conference (Chiu et al.,

2012), which appears in this chapter. There were two main findings of this work. One drawback with this method is that synthetic speech and video may not be as good quality as those generated by human actors. Another drawback, which the this thesis seeks to address, is a lack of automatic feedback on pronunciation.

Chapter 8 — In this chapter, I present Anderson et al. (2011), which I wrote for publication in Ghinea et al. (2012), a book that covers the potentials of multiple sensorial media (mulsemedia). My chapter explores how the capabilities of multimedia can be extended with multisensory media can be harnessed for CALL. The integration of physical interactions and speech with a computer virtual world allow for much greater immersion than a keyboard and mouse. The main contribution is a framework for mulsemedia and ECAs that allows language learners to tell stories with toys in the real world. This chapter presents a broad context for the other work on CALL and Computer Assisted Pronunciation Teaching (CAPT) presented in this thesis.

Chapter 9 — Finally, I report on a system that brings together work on KNPT with CALL. This chapter includes work that was presented as a conference poster (Anderson & Powers, 2016) that demonstrates the visualisation capabilities in the context of different Australian accents. This chapter brings together the KNPT work from Part 1 and addresses how they are applicable for CAPT.

Appendices — There are two appendices for supplemental information. Experiments using KNPT were reported in Chapters 4, 5, 6, and 9. Appendix A supplements the thesis with a brief overview the methods for data management for these experiments, including the structure of the Austalk corpora including metadata, and my methods for accessing the speech data and its annotations. Appendix B contains MATLAB code that I developed for these research efforts.

PART ONE

Chapter 2

Speech

This chapter introduces the topic of speech and how it may be analysed using a computer. The International Phonetic Alphabet is described along with the use of technology for speech sciences. Finally, I address the topics of feature representations for speech and unbiased statistical methods to be used for comparing the results of different experiments.

2.1 What is speech?

The core aspect of communication under consideration in this thesis is speech. Consider my operational definition of the word "speech":

> Speech is the production of sound for communication that is made when air is moved by the lungs past vocal organs including the tongue, lips, teeth, and palate.

This definition of speech contrasts it with other forms of communication, including sign language (movements of the hands and arms) and writing (words on paper). Humans also communicate using illustrations (pictures). Ideas can even be conveyed by the arrangement of objects in the world; for example, my wife leaves the garbage bin by the front door to communicate to me that she wants me to take out the trash.

Although these other forms of communication contribute to our understanding of language, in this thesis I am mostly restricting the attention to speech produced by humans in English. However, the definition of speech I gave above includes not only human spoken language, but also infant vocalisations, which may communicate hunger, for example, as well as the vocalisations of certain animals, such as primates and birds. Such non-verbal communications have been addressed with SOMs, and in section 3.4.3 I address the KNPT literature on speech-like communications of infants and animals. Although I do not investigate non-verbal communications in experiments presented in this thesis, I propose that the findings for KNPT, such as

map size, normalisation, and inclusion of silence in the corpus, should also be applicable to the wider domain of non-verbal speech.

To understand the aspects of language, we can look to those aspects that distinguish the speech of one language from another. There are four cues that distinguish between languages (Zissman & Berkling, 2001):

1. Phonology - different languages have different phonemes and sequences of phonemes.
2. Morphology - Different languages have different vocabularies. Some languages share base roots, so that means that they have more similar morphologies as a result.
3. Syntax - Style, sequence and choice of framing words (e.g. prepositions).
4. Prosody - Suprasegmentals such as stress, tone, rate and duration of language features.

In this thesis, the emphasis is on the phonology of English.

2.2 Accent

Accent is a distinctive pattern of human speech that is spoken by a group of people. Accents are regionally distributed, yet with the advent of broadcast media such as radio and television, accents have become less localised because some accent traits are acquired from the speakers on radio and television outside the region. Crystal & Crystal (2014) argues that certain trends strengthen accent shifts: for favouring newscasters with certain accents, the BBC has long been accused of influencing its listeners to speak English in a certain way (i.e. with a standard accent); and certain aspects of accents are learnt and adopted from television shows, such as a rising intonation pattern on the internationally popular Australian television show Neighbours. People gradually change their accents as they imitate the voices they hear, not only from friends, family and the community where they live, but also from media such as radio, television, and the Internet. Attitudes towards standard accents have relaxed a great deal, and regional accents (and foreign accents) are often appreciated and given time on the airwaves.

On the other hand, accents can sometimes get in the way. Accents may reveal aspects about a person, such as class, to the detriment of the speaker, or they may interfere with communication. When a person feels that their accent is undesirable, they may seek to change the way they speak. An ideolect is the accent and speech habits of a specific person. If the ideolect does not allow the person to communicate effectively, accent modification may be desirable.

"Sociolinguists tell us that language has two functions: to communicate and to mark the speaker as a member of a social group" (Krashen, 1997). In particular, we often do not speak

to our level of competence, and this applies to accent as well. Krashen argues that permanent improvement of accent has never been achieved through direct instruction because accent is affected by how a person speaks to the situation, with the conclusion that accent should only be measured in "optimal conditions".

Though Krashen argues against the deliberate study for improvement of accent for the second function of speech (since putting on an accent would be to pretend group membership, something that is not natural), accent can interfere with the ability to communicate effectively. Additionally, many people do want to be able to speak with a particular accent. Consequently, despite pronouncements to the contrary, then, in this thesis, I will explore representations of accent with the implied application of enable a learner to modify their accent.

Some people have learnt to pronounce with a new accent. In particular, some non-native speakers have learnt to pronounce a new language without an accent. Sometimes, this is part of the study of an actor for a role in a movie. Other times, language learners are trying to improve their ability to communicate in a new language. For an actor, the pronunciation of the language in a native way is necessary for portraying their role effectively. For language learners, it is less clear that their pronunciation must match identically the way that native speakers pronounce their language. Indeed, it is possible to communicate intelligibly and comprehensibly with an accent, and a great deal of being understood has to do with word choice. Listeners are able to adapt quickly to accented speech. However, not all accented speech is intelligible. This particularly applies for strong accents.

Current pronunciation pedagogy often deemphasises a goal of native-like pronunciation, because it is difficult to learn it even with direct study. As the aim of many teachers is simply to help language learners to be intelligible such that most of their utterances should be comprehended by others, the aim is not towards speaking like a native speaker (Munro & Derwing, 2015). The research presented in this thesis does not attempt to guide pedagogy, and although I have a particular view of how pronunciation can be taught in the learning of a language, not only for those such as actors, who must learn the precise ways to speak like a native speaker, but also for language learners, who are simply aiming to speak more understandably, the research here is primarily focused on enabling the computer to learn to evaluate pronunciation.

2.3 Speech corpora

A linguistic corpus is a collection of samples used for research into language. Often linguistic corpora consist of text but speech corpora are used for investigations into spoken language. A speech corpora is a type of language corpus that includes speech audio.

Annotating a language corpus is costly and time-consuming, but an annotated corpus has

many advantages, including easier data extraction, reusability, usability for different purposes, and legacy (adding to the permanent record) (McTenery et al., 2006). The annotation of speech may include details such as what was said (transcriptions) and how it was said (prosodic annotations). Another important annotation is the alignment of the audio to the annotations, such as to syllables or phoneme boundaries.

The transcription of a speech corpus is significantly cheaper than phoneme-level annotation. The difference is that the precise determination of when segments start and end is time consuming and open to interpretation. Many popular speech corpora, like TIMIT, were manually annotated by multiple annotators, while annotations for other corpora were prepared semi-automatically. Semi-automatic annotation involves a mix of machine annotation with human interaction to resolve uncertainty.

Canonical forms of language means dictionary forms. In corpora such as Austalk, sections of the corpus are generated by means of single word prompts. In these sections, speakers are instructed to say single words. The basic assumption for words produced in these sections is that the words will be representative pronunciations. In connected speech, the pronunciation of individual words may be more varied, and a listener is able to understand variations of pronunciation through context. When words are pronounced singly, on the other hand, the words are spoken more clearly with emphasis on clear pronunciation. In this thesis, the vowels of canonical form speech are the primary focus of investigation.

2.4 The International Phonetic Alphabet

For almost a decade, I taught pronunciation to English language learners in Taiwan using the Kenyon and Knott system (Vij, 2014). To achieve the work presented in this thesis, I needed to learn a new way to represent vowels. Similar to the Kenyon and Knott system, the International Phonetic Alphabet (IPA) was developed by phoneticians as a textual way to represent the sounds of the languages of the world. Kenyon and Knott and the IPA use mostly the same symbols; for vowels, the main difference is that Kenyon and Knott does not use length marking.

Even though the IPA is standardised, there are multiple ways to write any utterance. Firstly, there are broad transcriptions and narrow transcriptions. Broad transcription is used on phonemes that distinguish one word from another. On the other hand, a more narrow transcription might capture the way that certain sounds of a spoken language can be expressed in terms of specific positions of the articulators (lips and tongue) or the manner of articulation.

The IPA uses characters that are not in the English alphabet, but early computers used ASCII character set, which contains only 26 lowercase and uppercase letters as well as numbers

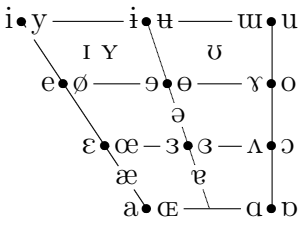


Figure 2.1: IPA vowel quadrilateral as shown in the LaTeX Vowel Package Manual (Rei, 2001).

and some symbols such as %&$#?@! (plus control characters). Alphabets like Speech Assessment Methods Phonetic Alphabet (SAMPA) were developed to allow for transcription in such ASCII-based systems. IPA symbols can be encoded in UTF-8, which allows for for characters outside the English alphabet and has become the de facto standard for text on the Internet at the time of the writing of this thesis.

In this thesis, which was typeset using pdfLaTeX, IPA symbols are represented using the TIPA package(Rei, 1996). I also prepared the vowel diagrams in LaTeX using the vowel package described in Rei (2001) from Comprehensive TEX Archive Network (CTAN). Where manual annotations were performed in transcription systems other than IPA, I implemented mappings that allowed me to convert the annotations to IPA. The Austalk corpus, for example, uses SAMPA-AUS, a system similar to SAMPA (Cassidy et al., 2014). The MATLAB code for SAMPA to IPA mapping is shown in Code B.2 in Appendix B. To display IPA using TIPA, I implemented mappings to convert the IPA to TIPA that could display in LaTeX typesetting. Note that to display the MATLAB code, the IPA was changed into TIPA and the TIPA was escaped to display the TIPA commands. An IPA vowel quadrilateral containing the vowels of many languages is presented in Figure 2.1.

"The sounds of a language are automatically and unconsciously organized by the native into structural units, which we call PHONEMES" (Pike, 1947, p. 57). The phoneme may be considered as a psychological construct or as a class of sounds, but the overall goal of using phonemes is to represent the units of sound as symbols. Phonemes serve as distinguishing units, but the specific expression of each phoneme may vary considerably in context. However, by using representing speech using a phonemic transcription system such as the IPA, a single symbol for each sound unit. For the purposes of understanding the sounds of speech, a phoneme is "one of the significant units of sound arrived at for a particular language by the analytical procedures developed from the basic premises" (Pike, 1947, p. 57). The phoneme is the unit of sound to be used for the analysis of speech.

2.5 The origins of speech technology

Originally, the study of phonetics taught methods whereby people could learn the sounds of languages without any technology required by noting the positions of the articulators. In classical phonetics, the student would learn how to write down the speech sounds in terms of articulators, thus identifying the sounds and also recording how they could be produced. The representation of vowels as a triangle was developed in classical phonetics (Ohman, 2001). The vowel triangle has now generalised to vowel quadrangle or quadrilateral, and it is used by the International Phonetics Association (IPA) for identifying vowels. But with the introduction of the spectrogram, classical phonetics began to be replaced by technological speech research.

The use of the spectrogram for speech was a spinoff of military technology. Martin Joos was a submariner in World War II, responsible for detecting other submarines through the sound they emitted using special equipment (Ohman, 2001). Joos then applied this equipment to speech (marketed as the SonaGraph) with a main argument that the classic phonetics concept of the vowel triangle was visible on a two dimensional plot of formants (F1 and F2). The IPA was influenced by this work and began to characterise vowels on the vowel quadrilateral diagram in terms of their formants. However, as Ladefoged & Broadbent (1957) pointed out, although the formants of the vowels do convey information in the context of other vowels produced by a speaker, formants have only limited applicability to the comparisons of the sounds of speech produced by different speakers. Trained phoneticians using classical methods may not be able to mark phonemes on the vowel quadrilateral without visual information about lip posture because there are multiple ways to produce similar sounds of speech.

Through the work of Joos and others, phonetics became more of an exact science. Researchers at Bell Labs published a book of speech that was represented using the Sonagraph. However, the acoustic nature of the formants F1 and F2, which were argued to relate to the big cavity and small cavity of the vocal apparatus, have been shown to be instead a continuous space of formant frequencies, "none of which was more 'affiliated' with any one cavity than with any of the other cavities!" (Ohman, 2001)

In technological speech research, therefore, the sounds become abstracted, such that we don't know how to produce them. The computer should help people do language study, but computers don't do the study—people do. Therefore, people need to have technology that is adapted to the purpose. In this thesis, I will explore the use of self-organising maps for evaluating vowels, but an important consideration to bear in mind is that of creating a tool that is not so complex or so technical that it does not adequately represent the sounds of speech.

2.6 Feature representations

One difficulty with investigating the sounds of speech is that they are abstracted from the production. Thus the sound of a given vowel is made by moving the vocal chords and articulators, such as the mouth, in a certain way through certain positions, but none of these are recorded. Only the sound wave remains. As (Ladefoged & Johnson, 2010, p. 213) put it: "Phoneticians do not really know how to compare acoustic data on the sounds of one individual with those of another. We cannot write a computer program that will accept any individual's vowels as input and then output a narrow phonetic transcription."

It is the goal in this thesis to work towards enabling the evaluation of speech. One main aspect of speech analysis is the set of features to be used. Consider the possible features to be used, specifically with an intention of realtime or near-realtime processing. I tested a number of different possible feature spaces, including spectrograms and MFCCs. Some feature spaces such as Linear Prediction Coding Coefficients (LPCCs) were not investigated in depth because such algorithms have less favourable characteristics that make it more difficult for them to be performed as online algorithms, i.e. they may be more computationally expensive or incur a delay.

A significant question is how to determine whether the feature set is working for our SOM maps. This question does not have a clear answer, but it is related to how well we are able to use the information that the map depicts. When using supervised data, it is possible to use the labels to provide a sort of an understanding of how well the training has created a map that fits unseen data to the categories of interest. Under the paradigm of unsupervised or semi-supervised training, it can be much less easy to interpret the results. In all cases, the results are subject to interpretation. In the case of vowel categories, we expect there to be overlaps and ambiguity, because the nature of spoken language is not always precise. People speed up and slow down their speech, omit and add extra sounds, speak one way at one time and then another way another time. Sometimes speech is used to convey ideas of great importance, while other times people partake in idle conversation of little consequence. In the analysis of speech, then, whether using a clearly defined or a less well annotated speech dataset, the results are what we make of them.

2.6.1 Window size

In speech processing, it is common to divide the speech stream into frames, which are small durations of information, apply a window function, and then to perform further operations (Jurafsky & Martin, 2009). The window function reduces distortion. For example, a speech signal recorded at 16 kHz may be divided into overlapping frames that are around 25 ms in duration, with 10 ms offset. This amounts to 100 frames of information per second. An

algorithm for analysis operations may be applied to each frame independently to determine a set of features for that frame. In this example, the speech signal would be characterised in terms of features with a resolution of 10 ms.

2.6.2 Spectrograms

Spectrograms are a representation of speech that visualise sound activity at different frequencies. In a spectrogram, frequency is plotted over time. Spectrograms carry much of the information of speech, and have been one of the most frequently used representations in the speech sciences. Humans have been trained to read spectrograms with 80% accuracy at the word level (Zue & Cole, 1979), although many experts are unable to do this type of spectrogram reading with repeatability (Mcloughlin, 2009, page 10).

As spectrograms can be interpreted by humans by visual inspection and there is some evidence that there are tonotopic maps in the auditory area of a brain carry equivalent information to spectrograms (Chan et al., 2014), one inference we might make is that algorithms can be designed that can recognise patterns present in spectrograms. Indeed, in his work with the neural phonetic typewriter, Kohonen (1988, 2013) used a hardware implementation of a spectrogram to transform the raw speech signal into a representation to be mapped using the SOM algorithm.

Kohonen used a hardware spectrogram, but a digital spectrogram transforms the audio signal in a similar manner to the analog spectrogram implemented in hardware. A software spectrogram typically uses Fast Fourier Transforms (FFTs) to determine the energy of the audio signal at different frequencies. When the number of FFT bins is high in comparison to the number of data points in the window, frequency information is revealed. On the other hand, when the number of FFT bins is low, time information is revealed. To determine harmonic information, 25 ms frames with overlapping so that frames are offset by 10 ms is a very common parameterisation in the speech recognition literature.

After preprocessing the raw audio, the signal is then processed by triangular (or similarly shaped) filterbanks to determine energies near specific frequency bands. Adapting the methods of Kohonen to digital spectrogram, the spectrogram resulting from processing of a speech utterance from our dataset can be mapped using a SOM. Figures displaying spectrograms with analysis are presented in section 4.3.

2.6.3 Formants

One way to represent the features of speech is as formants, which are harmonic resonances that occur in narrow frequency bands. For spectrograms at 25 ms frames, harmonic information will be observed at high frequency resolution.

The vowels of human speech may be characterised in terms of their formants. In particular, the IPA table shown in Figure 2.1 can be correlated with the first two formants (known as F_1 and F_2). In other words, the horizontal axis of the IPA table is correlated with F_2, and the vertical axis is correlated with F_1, as discussed in section 2.5.

However, formants are not clear units of comparison for all speakers. For example, due to aspects such as growing vocal tract and speech motor control, the formants of children have much more variability than adults, and the trend is not linear for all speakers (Lee et al., 1997). Although vowels produced by different speakers will generally be placed at recognisably relative positions on a formant chart, the absolute values of the formants will differ (Ladefoged & Johnson, 2010). Furthermore, if for a certain person, a specific vowel phoneme appears further to the right on the IPA table, it will generally also appear further right for all speakers of that dialect; however, the specific placement of each of the vowels cannot be determined.

2.6.4 Mel-frequency cepstral coefficients

The MFCC algorithm is an automatic method for transforming speech audio. It is a valuable technique because the transformation retains sufficient information about the speech signal for many methods to recognise the vibrational energies of speech phonemes. When comparing the speech of different people, MFCCs allow for an investigation of differences that reduces the impact of relatively minor differences in frequency. For example, people with longer necks will generally produce speech that is deeper in tone. These tone differences mean that the same vowel spoken by two people will have absolute pitch differences. When viewed in the spectral domain, as in spectrograms, the pitch differences appear as shifts. By performing log normalisation and mel scaling, many inter-speaker variational differences are reduced. Although this means that some information is lost by using the MFCC algorithm, it is useful for speech recognition, because an ASR system can function well between speakers. This is also useful for the speech analysis that we are doing here, because MFCCs allow us to observe differences between different pronunciations while abstracting away some of the aspects that different speakers have when saying the same thing.

Although MFCC is a lossy algorithm, meaning that some information is lost in the process of converting an audio file into MFCC vectors, it reduces complexity, which allows for faster processing. MFCC features have been found to be a good representation for speech recognition, improving capabilities for between speaker comparisons (Jurafsky & Martin, 2009). The MFCC algorithm can be reversed, but because it is a lossy process, the resulting audio does not sound like human speech. Nonetheless, MFCCs are widely used in ASR partly because some of the differences between humans (such as the fundamental frequency) are not essential to phoneme recognition).

Reducing the number of features, even ones that carry informative data, can often be

beneficial for learning algorithms due to the curse of dimensionality. Higher dimensionality can create complications because when using similarity metrics such as Euclidean distance, an increase in the dimensionality may increase the effective sparsity of the data.

In the MFCC algorithm, the FFT is compressed using the Discrete Cosine Transform (DCT). The premise of the DCT algorithm is to encode a signal using a codebook of cosines to weights. A two-dimensional DCT is also used for JPEG image compression. In terms of the MFCC, the higher coefficients represent finer gradations in the signal and do not contribute to the recognition task so they can be discarded.

Based on the sampling rate, there is a practical upper limit to the number of MFCCs. The maximum frequency resolution of a discrete time audio file is half the sampling rate, known as the Nyquist frequency. In the audio files used in this thesis, the sampling rate of 16,000 Hz means a Nyquist frequency of 8,000 Hz. Signals over the Nyquist frequency will not be used, as they will not be error free. In terms of a spectrogram, a 25 ms window of a 16,000 Hz signal comprises $0.025 * 16,000 = 400$ data points, so the nearest square of two is used to determine the FFT. In this case, a 512-point window is used, and the difference between the signal and the FFT window is padded with zeros.

Closely aligned with the interests of this thesis is the field of ASR. One of the main approaches to ASR is Hidden Markov Models (HMMs). The HMM Toolkit (usually referred to by its acronym HTK or as HTK Toolkit), is a widely-used software for speech recognition using HMMs. Due to its frequent use by researchers in the speech field, it can provide a standard of reference for comparison. The usage of HTK and its default values are outlined in Young et al. (2013). In this thesis, although HMMs were not used, where there exist multiple possible approaches for techniques such as delta-differencing, and I did not find improvement via experimentation, I have used HTK defaults where possible because they are reasonable starting points for ASR that facilitate comparison of my results with those of others.

MFCCs is an alternate formulation of the cepstrum. Although there are numerous variations of MFCCs, in the work presented in this thesis, following Wojcicki (2011), MFCCs were calculated using MATLAB-based MFCC procedures that are demonstrably similar to the HTK implementation of the HPARM module that can be used to transform speech audio into MFCCs[1]. I made a small modification to Wojcicki's MATLAB code that allows for an efficient computation of the sum of log energies, as is common (Vasquez et al., 2013), because both the zeroth coefficient and the sum of log energies provide information about the power of the current frame, but the former representation is more stable.

The MFCC algorithm is computed as follows. The signal is divided into frames. In this thesis, frames are 25 ms in length. Each these frames is made by creating 25 ms snippets of the

[1]The MATLAB Code for MFCCs and making comparison with HTK implementation is available from https://au.mathworks.com/matlabcentral/fileexchange/32849-htk-mfcc-matlab.

sound. These snippets, more formally called frames, overlap every 10 ms. The FFT is applied to each window, which converts the time-domain signal into a frequency domain signal.

$$\text{magnitude-spectrum} = |FFT(frames)| \tag{2.1}$$

A mel-scaled filterbank is used to bucket the FFT. The mel-scaled filterbank is a triangular filter bank on the mel scale, which is:

$$\text{mel-scale} = 1127 * log(1 + hz/700) \tag{2.2}$$

The variable hz is a frequency measured in hertz (times per second) in the range from the minimum frequency to the maximum frequency with K intervals. For example, computed in the range 300 Hz to 5300 Hz with 22 filterbanks:

$$hz = [300, 538, 776, ..., 4824, 5062, 5300] \tag{2.3}$$

The triangular filterbank is mel-scale warped and collects the log-energies as filterbank energies (FBE). The filterbank channels may be visualised as follows.

Listing 2.1: Plotting the triangular filterbank in MFCC

```
hz2mel = @( hz )( 1127*log(1+hz/700) ); % Hz to mel
mel2hz = @( mel )( 700*exp(mel/1127)-700 ); % mel to Hz
Fs = 16000; % sampling rate in Hz
K = 512/2 + 1; % Half the number of FFT points plus one
R = [300, 5300]; % frequency range in Hertz
[trif, freq] = trifbank(22, K, R, Fs, hz2mel, mel2hz);
plot(freq, trif)
xlabel('Frequency (Hz)'), ylabel('Weight'), xlim(R)
```

The magnitude spectrum is subsequently separated into filterbank channels. Cepstral coefficients are then produced using the DCT.

$$FBE = \text{triangular-filterbank} * |\text{magnitude-spectrum}| \tag{2.4}$$

The final step of the MFCC algorithm is the DCT. There are multiple types of DCT. The DCT that is most common for the computation of MFCCs is known as Type III DCT, which is the inverse of Type II. The DCT-III matrix is orthogonal, which is a format that makes it more suitable for certain mathematical operations than other formats.

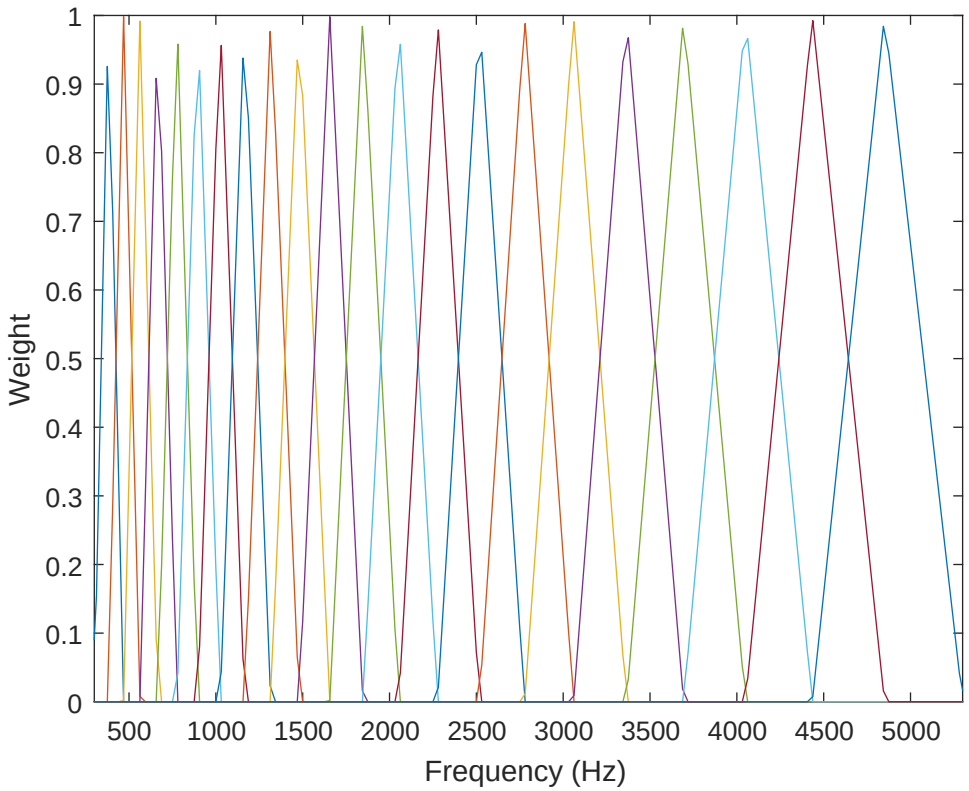


Figure 2.2: Triangular filterbanks used for MFCC processing.

$$CC = DCT * log(FBE) \tag{2.5}$$

DCT is a form of data compression as it compresses the signal into orthogonal channels. The final DCT operation in the MFCC algorithm is an artifact of ASR with HMM, which performs better when features are orthogonal, and thus linearly independent. The DCT operation is used to transform the spectral bands. The vector produced by DCT is ordered in terms of most larger to smaller features; this means the higher coefficients that contribute to the fine-grained resolution do not provide as much information that is relevant for speech characteristics. One way to conceptualise DCT is via another place that DCT is used: in the JPEG image compression algorithm. A two-dimensional DCT is practical for image compression since although a great deal of data corresponding to finer-grained details may be discarded, sufficient information about the overall image may be retained to understand it. Some of the fine details may simply be sensor noise or unnecessary artifacts. Additionally, the smaller file size for the image may be significantly faster to process. The DCT in the MFCC algorithm is one-dimensional.

Finally, a cepstral lifter operation smooths the results.

$$MFCCs = ceplifter * CC \tag{2.6}$$

All speech sounds are vibrational energies, so MFCCs are a way to understand the vibrations of the voice. As Wakita (1976) described, cepstrum methods allow for the separation of the periodic from the nonperiodic, so they are useful for the study of speech because recurring peaks arise from both the voice source (the larynx) and the vocal tract (the supraglottal spaces). This means that the MFCCs reveal vocal tract characteristics, which are not easily determined from spectral information. Furthermore, taking the log allows products to be reduced to sums, because of the following property.

$$log(x * y) = log(x) + log(y) \tag{2.7}$$

As a result of this property of the logarithm, cepstral domain expresses the convolution of signals in the time domain as addition in the cepstral domain. One reason this could be important is for finding combinations, using for example the Gaussian mixture models or non-negative least squares. If we think of the vocal tract as a processing chain, this means that if there are a sequence of things that change the sound, in the cepstral domain they are linearly separable.

2.6.4.1 Barely distinguishable difference in pitch

Humans with normal hearing are able to discriminate two tone pulses with different frequencies even when they are quite similar to each other, but there is a limit to how well people can perform this type of discrimination. Sek & Moore (1995) studied this by presenting to listeners two successive tones that had either the same frequency or different frequencies, centred around frequencies 0.25, 0.5, 1.0, 2.0, 4.0, 6.0 and 8.0 kHz. Although individuals are each different, and firm conclusions cannot be made from the study, the discriminative ability to distinguish tone difference below 2 kHz was on average "less than 0.6% of the center frequency". For this type of difference detection, known as difference limens frequencies (DLFs), the participants needed to indicate the correct direction of the change.

To tune the MFCCs according to these DLFs, then, let us use an oscillator, similar to that described in the experiment above, to produce tones at 70 dB. The values of minimum discernible difference are shown in Table 2.1.

The average human thresholds of barely detectable differences centred on different tones are listed in Table 2.1. Using sine wave generator, I visually compared the effects of difference in pitch, finding that the MFCCs of two pitches of barely detectable differences. will visually have significant differences in two or three coefficients. This provides confirmation that MFCCs

Table 2.1: Tone, threshold, and barely detectable differences in human hearing. Values from Sek & Moore (1995).

Tone (Hz)	Threshold	Barely detectable difference
250 Hz	0.6%	3 Hz
500 Hz	0.2%	1 Hz
1000 Hz	0.1%	1 Hz
2000 Hz	0.3%	6 Hz
4000 Hz	0.8%	30 Hz
8000 Hz	3%	240 Hz

are suitable for detecting subtle differences of speech. Such observations show that although MFSCs are sensitive to pitch, the variations are small, but the resulting changes in MFCCs are not so subtle.

2.6.4.2 Delta differences

A feature that is commonly added to MFCCs is deltas and delta deltas. These differences in coefficients at time increment provide information about how they change. They may be thought of as velocity and acceleration of the representation of the signal. For a given coefficient c_i at time i, the simplest way to estimate the delta, or velocity, is as an average of the values at the previous and next time steps (Jurafsky & Martin, 2009):

$$\Delta_i = \frac{c_{i+1} + c_{i-1}}{2} \tag{2.8}$$

This is equivalent to the regression formula for delta coefficients used in HTK with a deltawindow parameter of 2 (Young et al., 2013).

Similarly, the delta-delta (or acceleration) can be estimated as a function of the difference in the deltas,

$$\Delta_i = \frac{\Delta_{i+1} + \Delta_{i-1}}{2} \tag{2.9}$$

This difference is equivalently:

$$\begin{aligned} \Delta_i &= \Delta_{i-1} - 2\Delta_i + \Delta_{i+1} \\ &= (x(3:n,:) + x(1:n-2,:) - 2 * x(2:n-1,:))/4 \end{aligned} \tag{2.10}$$

Note that to calculate the delta or delta delta at the start or end of an audio file requires a different calculation because there are no cepstral coefficients from before or after the period during which a speech recording was made. It is possible to use a one-sided approximation to determine these values, following Fornberg (1988).

$$\Delta_1 = -c_1 + c_2 \tag{2.11}$$

And for a signal of length n:

$$\Delta_n = -c_n + c_{n-1} \tag{2.12}$$

The delta MFCCs at the endpoints are calculated using the one-sided approximation, so that the number of cepstral coefficients is the same as the number of deltas. The acceleration, or delta-delta, is the change in the velocity over time. So, similarly, for delta-delta:

$$\Delta_1 = c_1 - 2c_2 + c_3 \tag{2.13}$$

and

$$\Delta_n = c_1 - 2c_2 + c_3 \tag{2.14}$$

Note that if we have confidence that the important portion of the signal (i.e. the speech sounds under consideration) is not near the ends of the recording, these data points can simply be omitted from consideration.

2.6.5 Distance metrics

A method of determining distance between vectors is necessary to make comparisons. Feature space representations such as self-organising maps use prototypes and operate by means of a distance metric. Euclidean distance is a way to measure the distance between two points. Although self-organising maps can take other distance measures for determining similarity, Euclidean distance is almost invariably applicable to the KNPT for all practical purposes (Kohonen, 2013).

The Euclidean distance between two points can be computed as follows:

$$d(x, y) = \sqrt{x^2 + y^2} \tag{2.15}$$

The function d(x, y) means the distance from point x to y, which is the same value as the distance starting from point y and moving to x. For Euclidean distance, d(x, y) is always positive.

The Euclidean distance can also be formulated in terms of a subtraction of vectors, which can be a fast way to compute the distance:

$$d(x, y) = \sqrt{(x - y)^T (x - y)} \tag{2.16}$$

The normalised distance between two vectors is:

$$d(x, y) = x^T y \tag{2.17}$$

If we normalise each window to ignore the magnitude, the result is:

$$\frac{\sqrt{x^2 + y^2}}{x^2 * y^2} \tag{2.18}$$

For two vectors that are identical, such that $x_i - y_i = 0$ for all i, the distance between the two vectors is zero: $d(x, y) = 0$. In terms of matching inputs with a set of prototypes, this makes sense because if there is a prototype that exactly matches an input, then the metric should contain no difference between the prototype and the input.

Euclidean distance can be calculated between any two speech vectors using speech features such as those mentioned—spectrograms, formants, or MFCCs. Because the Euclidean distance metric can provide similarity and difference of speech frames, this enables frame-by-frame comparisons that can be calculated quickly.

2.7 Statistical measures

In the experiments conducted in this thesis, statistics are reported to convey an understanding of how well KNPT systems learn. Statistical results can supplement the KNPT visualisation of speech to enable us to get a better feeling for the language.

Recall and precision are commonly used statistical measures for evaluating the performance of a classifier. Recall is the measure of how often a classifier correctly identifies the target class in proportion to how often that class actually appeared. Precision is the measure of how often it correctly identifies the target class in proportion to how often that class was predicted. Recall and precision give an indication of how well a classifier performs based on the positive examples and predictions. Statistical bias can result when there are unbalanced classes. To reduce bias, two main methods will be used: using bookmaker informedness, and cross-fold validation. These methods are as follows.

2.7.1 Bookmaker algorithm

In this thesis, labelled maps are used to classify speech inputs; thus, the maps are used as predictors of phonemes based on speech frames. The performance of the maps is an important

consideration. A classifier that accurately predicts only the most prevalent class but guesses the others can nonetheless have high recall and precision. Bookmaker informedness (Powers, 2011) is an unbiased estimator of performance, a measure of how informed a predictor may be. Similarly, markedness measures how consistently a class is associated with its prediction. Informedness and markedness avoid population and label biases of recall and precision, which are unreliable for evaluation of multiple classes, especially when some appear more than others.

Other measures, such as ROC, Accuracy, F-Measure, Correlation and Kappa, despite taking bias into consideration, are still biased; for example, for a given recall level, their values decrease as biased towards positive predictions. Using informedness and markedness will allow for fairer comparisons between classifications made by different self-organising maps.

A two class confusion matrix is often shown as a contingency table, as shown in Table 2.2. Horizontally, the rows sum to determine predicted positives ($tp + fp = pp$) and predicted negatives ($fp + tn = pn$); vertically, as real positives ($tp + fn = rp$) and real negatives ($fp + fn = rn$). Finally, these totals sum to 1, as $pp + pn = 1$ and $rp + rn = 1$.

A confusion matrix can provide an understanding of the performance of a system. fn is often referred to as the rate of Type I errors; fp, the rate of Type II errors. Depending on the problem, these errors will have different costs (Witten et al., 2011).

Table 2.2: Contingency table used for calculating bookmaker informedness and markedness as well as other metrics such as recall and precision.

	+R	**-R**	
+P	tp	fp	pp
-P	fn	tn	pn
	rp	rn	1

Bookmaker informedness is defined for a binary case as "how informed a predictor is for the specified condition, and specifies the probability that a prediction is informed in relation to the condition." (Powers, 2011, p. 41) For the binary case, informedness is calculated as:

$$\text{Informedness} = \text{Recall} + \text{Inverse Recall} - 1 \tag{2.19}$$

Recall (sensitivity) is determined as follows.

$$\text{Recall} = \text{True Positive Rate} = \frac{tp}{rp} \tag{2.20}$$

Here, tp is the number of true positives (the number of times that true results are correctly identified by the system as true), and rp is the total number of true results. Inverse recall is determined as the prediction of negative cases.

$$\begin{aligned} \text{Inverse Recall} &= \text{True Negative Rate} \\ &= \frac{tn}{rn} \end{aligned} \tag{2.21}$$

Since $tp + tn = rn$, we can determine the second part of Equation 2.19 as follows.

$$\begin{aligned} \text{Inverse Recall} - 1 &= \frac{tn}{rn} - 1 \\ &= \frac{tn}{rn} - (\frac{tn + fp}{tn + fp}) \\ &= \frac{tn}{rn} - (\frac{tn}{rn} + \frac{fp}{rn})^2 \\ &= -\frac{fp}{rn} \end{aligned} \tag{2.22}$$

Finally, $\frac{fp}{rn}$ is the false positive rate. Thus the value of Informedness is:

$$\begin{aligned} \text{Informedness} &= \text{True Positive Rate} - \text{False Positive Rate} \\ &= \frac{tp}{rp} - \frac{fp}{rn} \end{aligned} \tag{2.23}$$

Similarly, bookmaker markedness is defined for a binary case as "how marked a condition is for the specified predictor, and specifies the probability that a condition is marked by the predictor." (Powers, 2011, p. 41) It is also known as DeltaP. For the binary case, markedness is determined as:

$$\text{Markedness} = \text{Precision} + \text{Inverse Precision} - 1 \tag{2.24}$$

Precision (confidence) is determined as follows.

$$\text{Precision} = \text{True Positive Accuracy} = \frac{tp}{pp} \tag{2.25}$$

Here, tp is the number of true positives, and pp is the total number of times that the system predicted true. Inverse precision is the prediction of negative predictions (the proportion of cases that are predicted to be negative that are indeed negative).

$$\text{Inverse Precision} = \text{True Negative Accuracy} = \frac{tn}{pn} \tag{2.26}$$

Since $fn+tn = pn$, the second part of Equation 2.24 is determined similarly to Equation 2.22 as follows.

$$\begin{aligned} \text{Inverse Precision} - 1 &= \frac{tn}{pn} - 1 \\ &= \frac{tn}{pn} - (\frac{fn}{pn} + \frac{tn}{pn}) \\ &= -\frac{fn}{pn} \end{aligned} \tag{2.27}$$

Finally, $\frac{fp}{pn}$ is the false positive accuracy. Thus the value of Markedness is:

$$\text{Informedness} = \text{True Positive Accuracy} - \text{False Negative Accuracy} \tag{2.28}$$

With a multi-class predictor for tasks such as vowel classification, where there are on the order of 15 vowels, informedness and markedness are calculated as for the binary case, using a contingency matrix, with pointwise averages across the contingency cells.

2.7.2 Cross validation

As stated, the goal is to create a model that describes a phenomenon such as phonemes. A set of speech data is used to investigate that phenomenon. Maps can be created that model speech. However, if the maps are created based on the entire set of data, there's no way to verify that they will also work for data outside the dataset. Cross validation is used to reduce this problem (Witten et al., 2011).

In cross validation, the data is divided into two sets. The first set, the training set, is used to train the algorithm, and the second set, the testing set, is used to verify the effectiveness of the algorithm. The training set is used to estimate the parameters of the model, and the testing set is used to verify that the parameter estimations work. The experiment is run a number of times. The more times the experiment is run, the more sure we can be of the consistency of the results of the experiment. However, simply running the experiment won't just result in perfect results, because the performance of the system is scaled by the number of times that cross validation was performed.

It is important to keep the sets separate to avoid a bias. Normalisation or other preconditioning operations should not be used across the entire data set, because they may cause data leakage, where information from one part of the set may influence another part that is to be held apart.

Cross validation is a statistical procedure that reduces the variance of the results. Cross validation is a common technique of running the training and testing procedures multiple times to gain a more informed assessment of the experiments. Commonly 10-fold validation is conducted, meaning that the data is split into training and testing sets 10 different ways.

Where cross validation is performed, results may be given as the mean and standard error. If n-fold validation is conducted, the mean is simply the average of the n trials.

$$\mu = \frac{\sum \text{n trials}}{n} \tag{2.29}$$

The standard deviation (σ) shows how much variation there is between the population. The standard error (SE) is an estimate of the standard deviation (σ) of the population based on the random samples that were selected for cross validation. The standard error provides information about the quality of the estimate of the mean.

$$SE_{\mu} = \frac{\sigma_{\text{n trials}}}{\sqrt{n}} \tag{2.30}$$

2.8 Grid search

The goal of the experiments with different machine learning models is to identify the best model. When there are different parameters of the model that can be varied, grid search can be used to find the best model. Consider the map size, a frequent consideration in this thesis. A larger map has more nodes so can provide a higher resolution view of the data, but larger maps are computationally more expensive to train. Also, when the quantity of labelled data is limited, the map size cannot be expanded arbitrarily without losing statistically significant observations per node. Therefore, grid search in this context explores the performance of the maps using a series of different map sizes.

The combination of grid search with cross validation is computationally expensive. Cross validation is useful because it allows for estimating confidence intervals of statistical measures (such as bookmaker informedness scores described in section 2.7.1). As grid search provides multiple results, the cross validation results could be estimated based on grid search. However, for reasons similar to those given in Hsu et al. (2003) for support vector machines, it is generally better to simply combine cross validation and grid search in practice. Firstly, cross validation arrives at statistical results that are more sound than estimations along grids. Also, assuming that not too many different parameters are being varied, grid search may not require much extra computational time. A similar approach to grid search can also be applied to the selection of other aspects that affect the results. For example, different preprocessing methods can be tested against each other to discover the best approaches for the problem at hand.

Chapter 3

The Kohonen Neural Phonetic Typewriter: A Literature Survey

The KNPT is a form of artificial neural network that is the central mechanism underlying the phoneme classification and evaluation that is presented in this thesis. The term "phonetic typewriter" came from Sakai & Doshita (1962) who described a system that should transcribe speech, essentially converting "the human voice into typed letters" (Sakai & Doshita, 1962, p. 140). The defining properties of a phonetic typewriter is that, given a stream consisting of the sounds of speech, the system should produce, as output, text corresponding to that stream of speech. In straightforward terms, the machine types out the speech that it hears. The technology of the early 1960s was too slow for the purpose, but such a machine was able to operate for "some specific words".

In the 1980s, Kohonen (1988) published a well received paper called the "Neural Phonetic Typewriter", which was a new procedure for mapping the sounds of speech to produce a phonetic transcription.[1] A KNPT produces text corresponding to a stream of speech using the SOM algorithm. Coupling the phonetic typewriter together with the inspiration from the biological structure of the brain, along with refinements, Kohonen's reported phoneme rates were as high as 95% for Finnish and Japanese. At the time, KNPT was a significant advancement of the field of speech recognition (Kohonen, 1995, p. VI). Although Finnish and Japanese are phonetic languages, such that the morphology of the script very closely matches the phonology, the use of SOMs would also subsequently influence research work into the transcription of languages with less direct phoneme-to-grapheme mappings, such as German and English, which I will cover in this literature review.

[1]The Kohonen Neural Phonetic Typewriter inspired a significant body of subsequent research that uses SOMs for phoneme classification. The current efforts are towards a new understanding of the principles of the niche application of SOM for speech. In this thesis, I refer to the Kohonen Neural Phonetic Typewriter by the acronym KNPT to differentiate it from other related systems that apply SOM for speech clustering and classification.

The neurons in an artificial neural network are adapted iteratively according to the training data via Hebbian learning (Kohonen, 1988). A fundamental mechanism that defines KNPT is its 'neural' aspect, which is brought about by the SOM algorithm, in which neurons compete for the input and the winner shares it with its neighbourhood, a kernel smoothing operation. This concept of neighbourhood parallels the structure of neurons in the brain. In particular, the neurons in the auditory cortex have spatial relationships of activation that may be described as tonotopic and periodotopic maps, which means that neurons in the brain that are near one another are more likely to correspond to similar stimuli (see, for example, Langner et al. (1997)). The SOM principle of preserving the topology of the input space is similar to that found in the brain (which has been found to have similar structures in many different regions), but it is not always closely followed by other artificial neural network architectures. Indeed, much research in the artificial neural network field focuses on suitability for particular applications rather than sharing mechanisms with the neurons of the brain.

The SOM is a neural model with correlates in the structure of feature maps in the brain. In particular, in the brain, similar sensory stimuli, such as speech, activate neurons that are near each other; conversely, more different sensory information results in activity in neurons that are further apart (Marsland, 2015). This fundamental property inherent to the brain structures for sensory information motivated the development of SOM (Kohonen, 1988). Due to this underlying mechanism, patterns in data revealed as organisation in SOMs may mirror the dynamical aspects of neurological structures in the brain (Kohonen & Hari, 1999). Although phoneme maps had not been located in the brain in 1999, Kohonen's hypothesis was that ordered representations of phonemes exist in the brain, and indeed, advances in neuroscience that have followed have pointed towards spatially organised encodings of features of speech, e.g. Humphries et al. (2010), Chan et al. (2014). It has been argued that SOM in the KNPT context may yet yield some insights into how intelligence can arise in the brain as a function of the inputs (and outputs). Although other methods may enable better results when measured statistically, it is partly for this reason that KNPT plays a central role in the current investigations.

Although interest in this application of an algorithm was greatest in the early years near when it was proposed in 1984, it has been continually revisited by Kohonen and other researchers since that time. This ongoing interest is for a number of reasons, primary among which is that the SOM is a neurally inspired approach to unsupervised clustering. Furthermore, this simple algorithm has power in the classification of speech at the phoneme level, with a capacity to produce visualisations of phonemes that group by similarity. The KNPT allows visualisation of speech that goes beyond the representation of a spectrogram or vowel formant chart to allow an understanding of the similarities and differences of phonemes and how they may be related. Although the training of a SOM can take a significant number of epochs, based on the size of the maps, the form of the input data, and the learning rates, comparisons of unseen speech with the resulting codebook can quickly allow quantisation of data that allows

classification to be made in a single step, simply by comparing the input with the codebook to find the most similar codebook vector.

Since 1986, a great deal of interest in KNPT has persisted. A number of papers have been published in areas related to this topic, as well as scholarly output such as PhD theses, with work being published virtually every year since. However, to the best of my knowledge, there has not been a literature review that describes these works in the context of the other publications in the area. In this chapter, I provide an overview of what is (and what is not) a KNPT system, revealing in a review of the relevant literature those aspects of speech or pronunciation that have been addressed using KNPT, how the area has grown, and gaps in the research. This literature review is directly applicable to the portion of my thesis work that deals with using SOMs for phoneme recognition, with applications to single speaker analysis and also for between group phonemic visualisation in a large corpora of Australian English.

3.1 Naming conventions

Many researchers in the area of KNPT identify that they have used SOMs, also variously known as self-organising feature maps (SOFMs), self-organising neural maps, self-organising networks, Kohonen maps, Kohonen feature maps, or Kohonen networks. Also, some simply refer to them in more general terms, as artificial neural networks (ANNs), but describe their mechanism of learning as that of the SOM algorithm.

To locate the scientific literature related to KNPT, articles were located using combinations of a number of search terms, including “Kohonen”, “self-organising maps”, “phoneme”, and “speech”. Papers that cited Kohonen’s initial neural phonetic typewriter papers (Kohonen, 1988; Kohonen et al., 1988)), as well as related papers that were cited by reviewed papers, were selected if they provided descriptions of KNPT systems that were implemented or tested.

An item of note to the current review, which was introduced in introductions to the bibliographical collections of SOM research (Kaskiy et al. (1998), Oja et al. (2002) and Polla et al. (2009)), is a conundrum related to searching for literature related to speech and SOMs, namely that those papers that cover speech do not always have the term “speech” in the title or abstract. The absence of a rigid mechanism or naming scheme for describing KNPT studies makes it difficult to locate them, and in the search, many unrelated research studies are also revealed. In particular, because Kohonen’s early work with SOMs applied the algorithm to speech transcription, many researchers mention speech in passing without seeking to make a contribution to speech science. Accordingly, those papers that used the related search terms, such as “Kohonen”, “speech” or “phoneme” in the title, abstract or body, but did not cover the KNPT in the experiments that were reported, were eliminated from consideration. For example, for many ASR studies, or in the HMM literature on speech analysis, the KNPT may

be mentioned in the literature review, but the research does not actually involve SOMs. For the purposes of this review, only research that reports on the application of the KNPT are addressed. This also includes variants of the SOM, such as growing maps, that seek to extend or enhance the basic competitive learning procedures of the algorithm.

Also, it is an assumption that all interesting work on phoneme-level usage of SOMs will cite Kohonen as a primary source. Even though I have made some attempt in this effort, I have not located any research articles that cover phoneme recognition using SOMs but do not cite any works by Kohonen, though they might conceivably exist. The most similar literature review to this one that I was able to find was in Kohonen (2001), which touched on a number of KNPT studies; however, that work was of a more general nature. In this review, I have tried to tie together how the different KNPT studies have explored the topographic learning of understanding speech as phonemes.

Every few years, Kohonen and his fellow researchers promulgated a list of research works that use SOMs called the *Bibliography of self-organizing map (SOM) papers* (Kaskiy et al., 1998; Oja et al., 2002; Polla et al., 2009), which collectively list 7717 papers. However, although the works are collated and in the bibliography many works are assigned keywords, the works that were identified by the list curators with keywords such as 'speech' or 'phoneme' are merely listed and not described. Furthermore, as I mentioned, the curators clearly state (with the term "speech" being their key example), the keywords are not comprehensive, nor is there any keyword for KNPT. Also, though such lists are very useful for exploring a particular area of SOM research, they simply present a number of relevant citations for general keywords, and as such, do not provide an understanding of the actual contents of the papers.

The goal of this literature review is thus not to comprehensively list every single paper or book that describes research that has similarities with the KNPT, but rather to report the main contributions that have been made over the years to using SOMs at the phoneme level and to provide a better understanding the work in this field. Due to my approach to literature search, it is unlikely that relevant work over the three decades has been missed, but the goal of this review is nonetheless to give a general understanding of the use of the topological approach of the KNPT for phoneme recognition with emphasis on the most important studies. A brief overview of KNPT circa 2001 was presented in Kohonen (2001, pp. 360-361), covering the topics of speech recognition and speech analysis, so it is therefore an aim of this literature review to emphasise works since that time. The main topics under consideration are speech recognition, isolated word recognition, continuous speech recognition, speaker identification and phoneme classification.

3.2 Scope of this literature review

A number of streams of speech analysis have been conducted with SOMs. Here, I have divided these streams into four, describing them and identifying which avenues for research were taken in the work conducted for this thesis. I consider the first of these streams to be KNPT and discuss the place of the others in that context.

The first stream uses SOMs as a system of phoneme transcription using windowed, processed speech signals as input, and it is this stream of research that is the main focus of this literature review. A number of different goals and applications have been reported for the application of SOMs to speech. KNPT was initially used for speech recognition, particularly in languages such as Japanese and Finnish with a high degree of correspondence between phonemes and writing system, but an emphasis was soon given to phoneme recognition independent of the production of spellings of words in the target language. It has been applied to aligning phoneme labels with speech. It has also shown useful in the study of speech qualities that are more removed from the phonetic in nature, for example using the KNPT to determine signs of speech (e.g. hoarseness).

Kohonen SOMs share some of those aspects of pattern recognition in spatial arrangement that are present in cortical maps in the brain. Such an arrangement can reduce the search space, but so far it is not yet fully understood how to implement this mechanism efficiently in artificial neural networks modelled on anatomical properties, i.e. in hierarchical temporal memory (George & Hawkins, 2009). Note that Hawkins argues that many ANNs, including HMMs, do not make any attempt to model biology, so they "provide little insight into neuronal or neocortical functions" (Hawkins & Ahmad, 2016). Following such logic, a more thorough investigation of systems that do, such as SOMs, would be advantageous. That being said, bioplausibility is not a primary motivation for much of KNPT research.

The second stream uses SOMs for textual data mining. The scope of such research typically has a focus other than the phoneme as a unit of analysis, as many languages, such as English, do not have strong bindings between orthography and phonology. However, this stream has at times sought to understand relationships between aspects of speech such as dialect, but use other information than speech itself as the input. Annotations, manually or automatically generated by other methods, may subsequently be analysed using SOMs. Or text in languages such as Finnish, with strong connections between orthography and phonology may be considered. For example, relationships between textual speech annotations were visualised using SOMs to observe similarities and differences between various northern-European languages and dialects (Honkela, 1996).

Although this second stream of research shares in much common with KNPT, I consider text-based studies to be distinct from the Kohonen neural phonetic typewriter. Text processing has many applications, most of which are not phonemic. Even when the focus of such research

may be aimed towards revealing relationships between phonemes, the real goal is not to map a continuous speech signal to phonemes or similar units, but rather to understand the relationship of the text based strings. Although there is significant potential for the analysis of the outputs of KNPT systems, I have not gone into this in depth in my research and so will leave such broadening of scope by investigations into textual analysis with regards to the phoneme for future research.

Finally, a third stream seeks to evaluate the quality of speech signal. For example, Barbedo et al. (2002) used various objective quality measures, which were subsequently mapped using either a polynomial mapping technique or a SOM. This type of research differs from the neural phonetic typewriter in that phonemes or similar structures are not the unit of analysis, and the goal is rather an evaluation of the quality of speech independent of the content of the speech itself. This, too, exceeds the scope of the current work so shall not be investigated in full. There is a fine distinction between those investigations that evaluate speech quality with KNPT and those that use SOMs to merely evaluate speech quality. My approach to this is to include as relevant only those studies that use the phoneme as the unit of evaluation.

A fourth stream tackles vocal productions that are not phonemic; for example, the babble of babies, the screeches of monkeys, or the productions of a parrot in the wild. These are arguably not applications of a phonetic typewriter, because they do not deal with anything phonetic. However, due to the similarity of such analysis with KNPT, I have included a section on some such applications of SOMs in this literature review.

In this review, I will present the research literature from these streams, with primary focus on the first stream (which I call KNPT research). In conducting this literature review, it was found that a number of papers included a citation of the seminal 1988 "neural phonetic typewriter" paper by Kohonen (1988), or the related Kohonen et al. (1988). Others did not, but referenced other works on SOMs (e.g. Kohonen (1995)) and conducted research mirroring the direction of the KNPT. Both types are included in this review.

In gathering references for this literature review, emphasis has been placed on locating recent papers in the field to get a better understanding of KNPT. A supervised learning algorithm similar to the SOM known as Learning Vector Quantisation (LVQ) has also played significant roles alongside the development of the KNPT (Kohonen, 2014). Classification performance of supervised tuning subsequent to SOM learning using LVQ is superior to that of a supervised SOM approach. Although the LVQ is closely related to SOM, and follows a parallel development history, I have not for the most part discussed LVQ-based speech typewriter systems where they do not involve SOMs. One of the first such LVQ systems was reported in Mäntysalo et al. (1993). I feel that a literature review of the LVQ-based speech typewriter merits study, but due to the focus of my research lab on unsupervised learning, it will be left for the future. On the other hand, I do report on research into phoneme-based classification using LVQ-tuning to improve

the results of a SOM-based approach in order to provide a context for understanding how it differs from unsupervised training of SOMs.

3.3 The SOM algorithm

Under the KNPT paradigm, a SOM learns patterns from speech. Accordingly, in this section, I will cover the procedures of the SOM, describe the common speech representations that are used as input to a KNPT system, and provide an overview of the approaches to using SOMs on the input that were reported in the literature.

The SOM algorithm is a procedure for learning codebook values for a set of nodes within a matrix (often two-dimensional) that are near the input distribution. A SOM can accept as its input any data that can take a distance measure between two subsets, from a multi-dimensional vector to a string of characters. In this thesis, the primary data under inspection is phonemes, so here, I will outline the steps of the algorithm and describe them in that context. The procedures of the SOM are well documented. Good introductions to this include Samarasinghe (2006) and Kohonen (2014).

Speech, as it is produced, is a stream of noise that varies over time, with amplitudes and frequency changes. The speech signal, which is in the time domain, may be transformed to the frequency domain (generally using the Fourier transform), producing spectrograms and formant charts that reveal regular oscillations over time. Similarly, an additional transformation of the signal into the cepstral domain captures patterns over frequency. The subsets of speech data under inspection are thus the data frames, formed from a sliding window. This short snippet of the raw audio signal is the data frame, and may undergo procedures that lead to a feature representation that enables an informative distance measure between two frames, such as the MFCC algorithm, rescaling to unit variance, or concatenation with sequential frames.

In the research literature, some researchers report different variations of the SOM algorithm. To facilitate work with SOM, the Kohonen research team produced SOM PAK Kohonen et al. (1995) and SOM Toolbox for MATLAB Vesanto et al. (2000), and many researchers have used these or built modifications on them. Due to the popularity of the SOM algorithm, there are also many other implementations available for a great deal of programming languages. The specific implementations that each research group used are not always reported or publicly available. Where multiple techniques were compared, this review will report the techniques that used SOMs with speech, but not the other techniques or applications that they were compared to, because they are beyond the scope of this review.

3.3.1 Size and shape of the map

Once the input data is prepared, the data is presented to the SOM for training. The initial size and shape of the map is determined by the experimenter. The aspects of size and shape may be selected in consideration of the input. For example, the reason that we are using a map is to learn a representation that is close to the input data. Therefore, the size of the map can be selected to ensure not only that each node will contain sufficient data to be a statistically significant representation of the local context, but also that the resolution of the map is high enough (more nodes means a finer resolution). There is debate over how to determine the size and shape of a SOM (or increase or decrease it), with different heuristics for selecting a number of nodes N for a map such as $5\sqrt{N}$ (Vesanto & Alhoniemi, 2000, p. 588) or $\frac{N}{50}$ (Kohonen, 2013).

Regular SOMs may be rectangular for simplicity, but Kohonen (2013, p. 57) suggests that a hexagonal SOM is superior to a rectangular one for data visualisation because it is easier to interpret and more exact. Hexagonal or rectangular refers to the interconnections of the nodes, not to the overall shape of the map. Each node in a rectangular map has four directly adjoining neighbours; each node in a hexagonal map has six. As a hexagonal node has more neighbours, it has more neighbour interactions, which means more information about the neighbours may be revealed.

The width and height of a map can have an effect on the mapping quality. In my research, I investigated hexagonal array networks with equal width and height (an equal number of nodes in each direction) to reduce the number of parameters to be tuned, only maps that have the same number of rows as columns. There can be advantages to having non-square maps, with either width or height to be narrower, because the visualisation can more faithfully represent the global distribution. However, an equal number of nodes in each direction does not mean that maps are square; due to the hexagonal structure, the vertical distance between rows is less than the horizontal distance between nodes. Nodes are equally spaced, such that three neighbouring nodes that are not collinear form an equilateral triangle. A "square" map in terms of an equal number of nodes in height and width fills a rectangular array with the ratio as follows.

$$height = \frac{\sqrt{3}}{2} width \tag{3.1}$$

Although the number of nodes is identical to the width of a map, as the nodes in the maps are configured in a hexagonal lattice, the height is the ratio of the height of an equilateral triangle with sides of one unit.

SOMs can be constructed in any dimensionality, but they are most commonly built as two-dimensional. In the literature, I have found that if researchers do not mention the dimensionality

of the map, we can often reason by the way they discuss the maps that they are referring to two-dimensional maps, the de facto default dimensionality. A small number of KNPT studies have used one-dimensional maps. For a literature review of dimensionality in KNPT, see section 3.6.6. Although convergence for a one-dimensional SOM can be proved analytically, a two-dimensional map allows for more complex visualisation on paper and computer monitors.

A SOM learns a manifold, which is a topological space (Kohonen, 2001). The learning process of the SOM seeks to locally fit the input manifold by iteratively reducing the local distances between the units and the input manifold (Varsta et al., 1998). Due to lateral feedback between the nodes, a single layer SOM is able to learn the structure of input data that, in backpropogation networks, requires hidden layers Kohonen et al. (1988). Fundamentally, the implication is a single-layer SOM can learn to distinguish some patterns that a single-layer backpropogation neural network cannot learn. For example, consider the classic example that a perceptron cannot distinguish, where $f(0,0) = 1$ and $f(1,1) = 1$ but $f(0,1) = 0$ and $f(1,0) = 0$. These points are not linearly separable. A 2x2 SOM could learn a codebook that consists of all these four points, and because each node in a SOM can correspond with a different label, the correct output can be made.

In contrast with Principal Components Analysis (PCA), which fits the data globally, the SOM finds local gradients. Although an entire map will take into account the global distribution of the data, at any specific location in the map, the local statistics were learnt. However, because "the reference vectors resulting in the smoothing process carried out by the SOM may not accurately represent the original statistics of the input samples, we may only qualitatively regard the two principal components of the neighbourhood set as the 'local principal components' " (Kohonen, 2001, p. 169). In other words, due to the nature of the algorithm, and to the lack of guarantees on how the map converges, quantitative claims about the manifold based on the local features in the resultant maps may not be supported. However, the qualitative relationships between nodes or classes may be quite informative.

3.3.2 Initialisation

A technique for initialising the KNPT is to use PCA to determine a starting point for the codebook values in the map. This can speed up the calculations quite considerably, because the PCA reveals the global variation in the data. A SOM starting from a random configuration can determine the global information, but it takes longer than starting from PCA. The first principal component is in the direction of maximum variance; each subsequent component is orthogonal to the previous components, and can be determined by singular value decomposition (Flach, 2012). The codebook values are determined as the range of weighted multiples of the first two principal components (for a two-dimensional map) in the range -1 to 1.

An analogy that I may make is that codebook values are like landmarks; although, unlike the

k-means algorithm, these landmarks are not necessarily equal to any of the points in the input data. Those landmarks that are close together on the map are more likely to be more similar due to the training procedures. Through the batch training process, the codebook vectors learn to represent the distribution as the the network structure approaches an equilibrium state. In terms of output, a KNPT produces a phonotopic map, which relates the speech input with phonemes. Other visualisations of phonemes in speech include such formant maps. However, the two are not equivalent.

> "It may be necessary to point out that the phonotopic map is not the same thing as the so-called formant maps used in phonetics. The latter display the speech signal in coordinates that correspond to the two lowest formants, or resonant frequencies of the vocal tract. Neither is this map any kind of principal component graph for phonemes. The phonotopic map displays the images of the complete spectra as points on a plane, the distances of which approximately correspond to the *vectorial differences* between the original spectra; so this map should rather be regarded as a *similarity graph*, the coordinates of which have no explicit interpretation" (Kohonen, 1988, p. 19).

In other words, the codebook defines a set of representative data points in the feature space that are also mapped in a two-dimensional sheet. Unlike formant maps, which place vowels according to the frequencies of formants, the actual locations of the nodes of a phonotopic map, which correspond to phonemes, do not have explicit interpretations that could show how one phoneme relates to another. This is because the SOM learns local arrangements.

3.3.3 Learning procedures

3.3.3.1 Batch SOM

A map is comprised of nodes, each of which has a value that is in the feature space. Adjustments to the codebook values are made using batch updates. First, for each of the observations, the node that has the codebook value that is nearest is determined. Then, once all the observations have been associated with the best matching units (BMUs), all the codebook values in the neighbourhood of the BMUs are updated simultaneously (the codebook value of each node and its neighbours is moved towards the average of the associated observations).

Each node c in a map has a codebook value m_c. Given a batch of data, x, which is a collection of all the observations, the distance metric is used to locate the best matching codebook value for each observation.

$$c = \arg\min_i \left(||x - m_i||\right) \tag{3.2}$$

Listing 3.1: Batch learning procedures for training

```
% the dimensionality of the SOM
dim = 2;
% use linear initialisation
sMap = som_lininit(input_data, dim, 'msize', [3 4], 'hexa', 'sheet');
% train the map using default parameters
trained_sMap = som_batchtrain(sMap, input_data);
% label the map using default labelling procedure
labelled_sMap = som_autolabel(trained_sMap, labelled_data);
```

Note that no updates are performed at this stage of the algorithm. This means that memory requirements may be large for large maps.

A Gaussian neighbourhood smoothing function $h_c i$ is used for determining the value of the distance between a node and its neighbours. The value of $sigma(t)$ is the neighbourhood training distance, and would typically initially be started at half the width of the grid and gradually be decreased to around 0.5. If the neighbourhood distance was decreased to zero during the course of training, the map would lose the property that nodes in a j neighbourhood share local similarity.

$$h_{ci} = \alpha(t) e^{\frac{\text{dist}(c,i)}{2\sigma^2(t)}} \tag{3.3}$$

Using the MATLAB SOM Toolbox (Kohonen, 2014), the training of a map can be conducted in a few steps. The dimensionality of the SOM (the number of dimensions of the nodes) and the size of the SOM (the number of nodes in each dimension) are specified. The map is initialised based on the first two principal components of the input data. The map is then trained. For this example, a single training procedure using the batch training method is followed. Finally, the trained map is labelled using a set of labelled data. The steps for accomplishing this are presented in MATLAB Code 3.1.

3.3.3.2 Sequential SOM

The original SOM algorithm used a sequential algorithm to perform updates. In contrast with the batch method, the sequential SOM performs the corresponding updates one at a time, given a single observation from the training set, the BMU in the map is determined. The codebook value of the BMU is updated towards the observation. In addition, all the neighbouring nodes are also updated (in proportion to their distance from the matching node in the map space).

In contrast with the sequential algorithm, which updates the weights after each observation, for each iteration of batch learning, the entire dataset is presented to the network before any

codebook values are updated. Batch learning is preferable to the sequential algorithm as it converges faster, does not require a time-dependent learning rate, and is "safer". Indeed, in pointing out these three reasons, Kohonen (2013, p. 53) stated, "It should be emphasized that only the batch-learning version of the SOM is recommendable for practical application..."

3.3.4 Related algorithms

The SOM algorithm is related to instance-based methods. Instance-based methods use samples from the training dataset for classifying. The nearest neighbour (NN) algorithm was first described in Fix & Hodges (1951), and it has been a tool in much subsequent research. Rather than generalise the training set into a set of codebook vectors, as in SOMs, instance based methods may also create a codebook, but the values of the vectors in the codebook match the instances from the training set. The NN algorithm is as an analogy-based method (Domingos, 2015). To predict the class of an item, the nearest item of a known is used. This simple procedure has been used earlier than the Fix and Hodges publication. For example, Domingos describes the procedure used by Jon Snow, a physician in the mid 19th century, who used a map of London and cholera cases to determine the water well that was the source of the infection.

At test time, the class of an item is predicted as the same class as the nearest item in the training set. Indeed, training is not necessary in such an approach, since the entire training set is used for the classification method. The entire training set is used for "lazy classification" (doesn't require training) in NN. Borders between items of different classes are implicitly formed. However, the drawback of the NN method is that of overfitting; a single incorrectly labelled data point affects the class prediction of all those it is nearest.

Another analogy-based method, which draws on the nearest neighbour algorithm, is the K-nearest neighbours (KNN), an instance based method. Rather than using the entire training set, as in nearest neighbour, the KNN algorithm learns clusters of the training set into K groups, each a class. For each of the K clusters, a single instance is determined.

The curse of dimensionality is a problem for nearest neighbour (and other instance based methods, and indeed all learners) (Domingos, 2015). As the number of features increases, the difficulty of learning increases exponentially. The distance metric uses each of the features for determining similarity. But this means that it is susceptible to misclassification. This is particularly so for high-dimensional data, where the number of features for each observation may be in the thousands or millions. Determining which of the features are essential may allow many low value features to be discarded, improving the abilities of the learner.

3.3.4.1 Artificial neural networks

There are many variants of artificial neural networks (ANNs). It may be noted that most types of ANNs follow a cooperative approach. The basic architecture of most ANNs is for each node to accept as input a weighted value of each of the features. The activation of each node is then determined by a combination of the weighted features processed by an activation function. The network is trained using a cost function. This differs from the SOM, which is a competitive neural network. In a SOM, each node takes as input an unweighted value from each of the features, and nodes compete to represent each observation.

3.3.4.2 Competitive learning

Rumelhard & Zipser (1985) defined a neural network that uses competitive learning as having three main properties: (1) it is comprised of nodes, each the same except for a parameter that makes each node respond to different input pattern; (2) the output of each node is limited; and (3) the nodes compete "for the right to respond to a given subset of inputs".

This three part definition describes KNPT, the earlier work of Kohonen (1982), as well as the work of others. For example, Malsburg (1973) used competitive learning for the simulation of the visual cortex, showing that lateral connections between nodes would result in topological ordering of nodes. Given an input and the output of neighbouring nodes, each node in the network would compete for the input pattern. The strength of the winner would be increased for that pattern. Malsburg showed that organisation emerged in the network, so that nodes that responded to similar angles would be located near one another in the map. Thus, it was possible to demonstrate that patterns in a network could emerge through the process of self-organisation, giving support to the idea that the functional organisation of the brain does not need to be entirely determined by genetics. More specifically, that the cells in the visual cortex that detect lines of similar angles are located near one another may be a process that arises through competitive learning.

Simplifying and generalising this, Kohonen (1982) and Kohonen (1988) extended the winners through competitive learning to also modify the neighbouring nodes. This change meant that lateral connections were unnecessary, reducing the computational overhead.

3.3.5 Input representations

The form of the various inputs, as they may be used with the KNPT, has an important relationship with the distance metric. The number of neurons in the network must be sufficient to quantise the input feature space at the desired resolution. A suitable input representation is one that provides similarity metric. The input to the map is the data frames that are used as

training data is a set of vectors. These data frames were were produced by transformations of the input data.

When humans listen to speech, the ears and the brain act on the speech signal, making it possible for speech perception to take place in the brain. For speech to be used by a computer, sound waves must be captured and converted into a format that enables information processing. The representations of speech that were reported in the KNPT literature were: spectral representations, the bark scale, MFCCs, LPCCs, and formants. These different methods of processing speech data that were used in the literature will be introduced.

3.3.5.1 Formants

A formant is an "acoustical resonance in the vocal tract" (Kohonen, 2001). The formants change over time, and are most evident for vowels. In reporting qualities of vowels, the frequencies of the first two formants are often reported as formant maps. Formant maps display vowels according to their two lowest formants (Kohonen, 1988). These formants roughly correspond to tongue height and tongue position.

Formant analysis is typically not a completely automatic process, because automated approaches lack reliability. Furthermore, due to the nature of phonemes, different speakers produce different phonemes with similar formant patterns Tiwari & Tiwari (2012), which means that formants must be considered in the context of a speaker's pitch range. Perhaps owing to the difficulties in obtaining formants by automatic methods, in my review of the literature, no studies used, as inputs to KNPT, formants obtained from human speech. Spectral and cepstral representations of speech are much more common inputs. However, a few related studies can provide some insights.

Primate vocalisations can be characterised using formants. Using SOMs for the unsupervised learning of primate vocalisations, Pozzi et al. (2012) used as inputs to a SOM the pitch and first two formants at the initial, nucleus, and coda. These were sufficient to distinguish five of the seven classes of vocal types.

In an exploration of unsupervised learning of vowel categories in infant directed speech, speech produced by mothers when speaking to their babies, Vallabha et al. (2007) used formants F1 and F2 and vowel duration. In infant directed speech, the formants are exaggerated. Vallabha did not report using SOM but a self-organising topographic map approach called TOME.

Tone languages such as Mandarin use differences in pitch to differentiate phonemes. Using SOMs, Gauthier et al. (2007) explored mappings of the pitch (F_0) and changes in the pitch (velocity profile), finding that the latter input shows clear differentiation of four tone categories on the four corners of the maps, while the former did not. However, both forms of input were

sufficient to attain a reported 80% and 97% correct categorisation, respectively. Thus, SOMs provide both clustering and categorisation capabilities; even where the global clustering does not reveal clear differentiation of the categories, the maps may still produce somewhat useful results for categorisation (80%).

Formant detection is "difficult to achieve... robustly over a wide range of speakers and speech sounds" and therefore "other approaches to speech quantization... have become firmly established" (Rabiner & Schafer, 2011, p. 642). Although formant estimation has some potential as a signal processing technique for use in the context of SOMs, because formants carry some of the essential features of vowels, the actual information rate carried in the formants can be quite low and a number of linear predictive methods may be necessary, including assuming that formants are continuous, and spectral preemphasis to reduce formants that are not widely separated over frequencies from merging. Manual correction of formant extraction through viewing of FFT plots enables linguists to correct problems in formant estimation, and reporting vowels as F1-F2 plots can enhance understanding of how people speak. Due to the difficulties in automatic formant extraction, or to the lack of information provided by such approaches, formants were seldom used as inputs in the SOM literature.

3.3.5.2 Spectral representations

Spectral representations differ from formant maps. The two-dimensional formant maps, which are typically used by linguists, map speech using only two resonant frequencies. Spectral representations, on the other hand, define the mapping by the complete spectra (Kohonen, 1990). Kohonen's original description of the KNPT used a hardware-based input that may be interpreted as a form of log scaled-frequency spectrogram with 15 filterbanks (Kohonen, 1988). Similarly, Rihkanen et al. (1994) used FFT-based spectral features. Additionally, Tashan & Allen (2011) used spectral components of the discrete Fourier transform to train SOMs to distinguish frames corresponding to specific vowels.

3.3.5.3 Bark scale

There are many features that can be used to describe the sounds that babies make. These sounds that infants produce in their first year are prelinguistic categories (vocant, squeal, and growl). Warlaumont et al. (2010) studied these productions using a SOM. Audio was processed using the FFT, then binned to a 15-bin Bark scale to create power spectral density values, which were then normalised to the maximum power for the individual utterance. Frames were concatenated so that one second of data was converted into 225 features. A 4x4 SOM was used. This feature representation is similar to the FFT-based features, but emphasises the lower frequencies.

3.3.5.4 Mel-frequency cepstral coefficients

As introduced in section 2.6.4, the MFCC algorithm is a procedure for converting digital audio into cepstral features. MFCCs are particularly well suited for HMMs, a leading approach to speech recognition. Much of the later work with KNPT has emphasised MFCCs as the feature representation. The MFCC has a number of variations, but in general it is the log of mel-scaled filterbank channels of the Fourier transform of the speech, with a final step in the MFCC algorithm of DCT, encoding the channels as orthogonal coefficients.

In the KNPT literature, 12-dimensional MFCCs was a common feature representation. 12-dimensional MFCCs were used in Lundberg (2005) for phoneme recognition. Behi et al. (2012) used 12-dimensional MFCCs for phonemic class recognition with spiking SOMs. Tashan et al. (2014) used 12-dimensional MFCCs to train KNPT that was subsequently used for speaker verification. Grashey (2003) also used 12-dimensional MFCCs for a KNPT trained for voice activity detection. Chihi & Arous (2012); Salhi et al. (2013) used MFCCs with spiking SOMs but did not report their dimensionality, though we assume it to be 12-dimensional MFCCs, as in their previous work.

Koreman et al. (1998a, 1998b, 1999) used 12-dimensional MFCCs to train one map and their derivatives to train another map. They found that the intermediate step of using KNPT to identify 14 different phonetic features, such as uvulars or plosives, improved classification rates for consonants. Kasabov et al. (1998) concatenated three vectors of 26-dimensional MFCCs, one each from the beginning, middle and end of phonemes (found through manual segmentation). Higher dimensionality was explored in Kangas (1991), who used 22-dimensional MFCCs for phoneme recognition.

In a study related to KNPT, Ranjard (2009) used MFCCs as a feature representation for bird song. 12-dimensional MFCCs plus energy were used for some experiments, while others used these 13 features plus their velocities (to make 26 features) and acceleration (39 features). The inclusion of the delta coefficients did appear to improve classification rates and aid convergence. Note that although birds do not produce speech, such an investigation into animal communication may have some relevance.

As can be seen from these studies, 12-dimensional MFCCs were used in virtually all of the literature, but it is possible to use additional delta features (velocity and acceleration) or more coefficients. I also found no comparison between the MFCCs with and without delta features.

The cepstral coefficients are sequenced in a way that the first conveys more general information and subsequent coefficients convey increasingly finer details. This means in practice that the higher number coefficients, corresponding to the finer details, may be discarded without reducing the ability to perform classification accurately. Indeed, this property of the coefficients might give us some ideas for feature scaling, because instead of discarding the higher

coefficients, the weight of the features that convey more information could be increased (feature scaling). As far as we are aware, this aspect of feature weighting for SOM remains unexplored.

3.3.5.5 Linear perceptual coding

Linear Prediction Coding (LPC) is another cepstrum-based feature representation for audio or speech that finds common use with KNPT. It combines previous speech samples to predict the next sample. A textbook explanation of LPC for SOM for speech recognition (which bypasses the phoneme classification stage) is presented in Kohonen (2014). Kohonen (2001) describes 16th-order LPC as suitable for KNPT because it retains the information of FFT in less features.

LPCCs were used in the following studies. Knagenhjelm & Brauer (1990) used 12 LPCCs for vowel classification. Dalsgaard (1992) used 12 LPCCs with KNPT to visualise and perform automatic alignment of phonemes. 10 cepstrum features were found to be suitable for a novel KNPT variant called DTW-SOM (and DTW-LVQ) that was based on dynamic time warping (DTW) as the similarity measure (Somervuo & Kohonen, 1999). De Luna-Ortega et al. (2009) found LPCCs were suitable for dimension reduction, allowing unsupervised KNPT to recognise a small number of words with accuracy above 90%. Finally, Eng (2006) used 24th-order cepstral coefficients with SOM, which performed dimensional reduction of the input to a multi-layer perceptron (MLP) for phoneme recognition. The retention of SOM activations over time is similar to the concept of leaky integration in Behi et al. (2012).

3.3.5.6 Comparisons of different feature representations

SOMs are reliant on the distance metric for classification and clustering. In virtually all KNPT studies, the Euclidean distance metric was used, a measure of similarity based on the L2 norm, i.e. the square root of the sum of the squares. Initially, KNPT research used spectral representations (e.g. FFT), but more recently, MFCCs and LPCCs have been the predominant input feature representations. I did not find formants to be used in any KNPT studies. The most frequent approach to input representations is that authors select a single method for speech representation that they feel is appropriate to their research based on previous work with the same input representations. Speech representation has evidently not been a primary area of interest and empirical comparisons of feature representations are seldom reported; however, a handful of KNPT papers presented experimental results that compared different feature representations, to be described in this section.

3.3.5.7 MFCC versus LPCC

Comparing KNPT maps trained on 13-dimensional MFCCs or 18-dimensional LPCCs to identify Korean consonants, Lee & Chang-Young (2011) found that MFCCs had better classification

accuracy than LPCCs, but on the whole, an unsupervised KNPT approach was not found to have good results for either input representation. They concluded that other methods than Kohonen SOMs would be necessary for these purposes. However, Venkateswarlu et al. (2011) used SOMs to compare features of LPCCs, MFCCs, pitch and intensity, with the interesting result that intensity was best for classification, though the maps were large in proportion to the very small data set.

3.3.5.8 PLP versus RATE

Cooke et al. (1994) investigated phoneme recognition in speech with deleted data. They compared perceptual linear predictive (PLP) coding with RATE, an inner ear hair cell model, with RATE found to be slightly superior to PLP in terms of accuracy. The frame-by-frame accuracy was around 20% for 39 phonemes (including silence) in TIMIT was considered to be too low for practical purposes, but little degradation of the performance was observed despite the deletion of data.

3.3.5.9 MFCC versus RASTA-PLP versus PLP (with LVQ)

Laleye et al. (2014) compared MFCCs, RASTA-PLPs, and PLPs for the African language Fongbe, finding that for the LVQ classifier, MFCCs were marginally better. In this work, they compared two classifiers, LVQ and Naive Bayes, for phoneme recognition and found LVQ to be superior, but that a hybrid LVQ-Naive Bayes system could outperform either singly. The authors attributed the reason for this to Naive Bayes performing better for Fongbe vowels, while LVQ had better performance on the consonants.

3.3.6 Supervised versus unsupervised training

A great deal of KNPT research has been towards using the SOM, which is an unsupervised clustering algorithm, in a supervised way. One major difference between the approach to KNPT in the literature is the type of training, whether unsupervised or supervised. Both types have their advantages and disadvantages. Unsupervised training allows the use of large unlabelled datasets, but supervised training allows labelled data to enhance or improve classification. The type of training has a significant effect on the results.

Unsupervised training of a SOM means that during the training of the map, the classes of the data points are not used. For SOM training, this procedure may be followed by labelling each neuron, for example, with the most common label of the best matching data points. The drawback to unsupervised SOM training is that the class information (which phoneme is associated with each input feature vector) is not used to create better class distinctions.

However, it might be argued that in human learning of speech as an infant, the class information is not known as phoneme recognition is learnt, so the unsupervised method may be more appropriate to some uses.

The KNPT was briefly revisited by (Kohonen, 2014, pp. 160-163) in one chapter of a book about applying SOM techniques. On the topic of unsupervised SOM, Kohonen stated, "... it is not expected that [the self-organising map would simultaneously cluster input data items at an optimal accuracy, when referred to the probabilistic methods of classification. Nonetheless its computation is very easy, and with some extra means it can be made to classify objects at a near-optimal accuracy." In other words, supervised training of the SOM, possibly using supervised SOM or LVQ-tuning, can lead to good classification.

Kohonen also states that the original work was supervised training and clarified that the phoneme labels were supplied as part of the input. That seems to vary from the original description of KNPT in Kohonen (1988), which did not explicitly describe class labels as being part of the input. This approach that uses classes that guide networks to better configurations can be described as supervised SOM, in which a one-hot vector of the class labels is supplied along with the spectral features for training. Indeed, this form of supervised SOM, which provided class labels with the training data, was not reported in the KNPT research that I was able to locate, so perhaps it is simply an omission. Kohonen (2014) ceased work on the supervised SOM in favour of LVQ-tuning, a supervised approach that follows unsupervised training.

In fact, it is not always straightforward to determine whether the work in the literature involves unsupervised KNPT. A supervised approach uses class labels to improve the classification. Thus, for approaches such as supervised SOM and LVQ-tuning, we will consider them to be supervised techniques, because the learning part of the procedure includes label information. A main focus of the research that I conduct in this thesis is unsupervised KNPT, so although I will describe supervised techniques, I will try to clearly emphasise which are the unsupervised approaches. Unsupervised training means to change the codebook values without regard to the labels. On the other hand, adding labels to the nodes that have been trained is clearly a supervised procedure in that the network is provided information. As this feedback during the labelling procedure does not change the values that were obtained during the learning procedure, the unsupervised learning is followed by labelling drawn from annotations.

One particular issue is whether the speech data that is used for training is then reused for testing and visualisation. Maintaining the separation between the training and test data is important for the situations where the maps are to be used for the analysis of speech that has not been seen before. A commonly argued benefit of SOMs in the KNPT context is that a smaller quantity of annotated data can be used for labelling maps that are created through unsupervised learning. However, I have found that implementation of this aspect is seldom reported in the literature.

For example, the work of Kangas (1994) used unsupervised training for the characterising of speech dysphonias. The corpus consisted of a set of seven different words containing the /s/ sound. Maps were trained on words using speech from speakers without dysphonia. However, Kangas does not mention how the maps were labelled. It appears that the entire set of training files was manually annotated at the frame level so that while the maps were created using unsupervised learning, the labelling procedure used the same data that was used for training. Thus, it may be that the entire dataset was annotated and the annotated labels were projected onto the map created through unsupervised training. As a result, it is to be expected that the projection of the manually annotated data will have a close fit with the dataset that is used for training the maps (in an unsupervised manner). This is because the same set of data is used for training and labelling. I would argue that such a procedure does not clearly show that a small quantity of annotated data can be used to label an unsupervised map.

Much of the prior research that uses SOMs in speaker identification is closed-set, which means that data used for testing was from speakers that were used for training of the networks. Tashan et al. (2014) argues that this approach has resulted in difficulties understanding the results that may occur in real-world applications that require classifying data from unseen speakers.

3.3.6.1 Unsupervised KNPT with limited labelling data

The use of KNPT for classification, using an unsupervised learning mechanism that relies on a large set of unlabelled speech and a smaller set for labelling is often addressed minimally, if at all, in the literature. One main advantage of such an approach is that it allows the learning of datasets for which there is only a small quantity of labelled data. However, in terms of evaluation, we are to use half of the data for map labelling and half for evaluation, this reduces the small amount of data available at each step considerably. Additionally, it may be difficult to understand the extent to which the labelled data is representative of the training data, since the phonemes of speech are known to vary between speakers.

A primary difficulty with such an endeavour is that annotated data is required for the labelling of the maps and for evaluation of the classification quality. When evaluating a classifier, it is essential to evaluate it on new data that was not used in its training (Witten et al., 2011). Later in this thesis, I will discuss approaches to work through these issues. Although the general idea that a small quantity of manually annotated data may be used with unsupervised KNPT to enable the classification of phonemes (or similar communication unit), it does not appear to be well supported in an extensive review of the KNPT literature. Most KNPT research appears to sidestep this issue.

Firstly, we may consider forms of evaluation that do not require data annotations at the frame level. For example, word recognition may be accomplished by concatenating the stream

of classifications. Thus, we might use a single word corpora, consisting of audio files that are around a second in length, for each of which there was a single word recorded. This corpora is only roughly annotated (we know what word was said, but not where the phonemic features started or stopped, or even whether all the expected features were present). Also, it may be possible to improve a roughly annotated dataset by segmentation based on the small quantity of labelled data, thus gaining more precise annotations which can then be used for labelling.

As for the question of how large the map should be, in the case of the unsupervised learning KNPT with labelling, it appears to be unresolved for KNPT. If the use of the map is for accurate phoneme classification, then labelled data could assist in the determining of large enough resolution with sufficient statistical significance for the labels, with consideration of how the labelling technique may be used for empty cells.

3.4 Applications of KNPT implementations

By the definition of KNPT that I'm using here, the systems are comprised of SOMs that are trained for phonetic purposes. One closely related area is in the study of non-human vocalisation. In this section I describe the applications where KNPT has been applied to speech data that is phonetic (or phonetic-like, in the case of animal or baby vocalisations).

3.4.1 Speech recognition

The first research into KNPT was conducted in order to work towards speech recognition (Kohonen (1988) and Kohonen et al. (1988)), "intended to transcribe orthographically edited text from an unlimited vocabulary". Kohonen acknowledges that there is significant overlap between phonemic classes, and effects such as coarticulation to complicate matters, so the motivation behind the use of KNPT for speech recognition is towards revealing features in the data that are useful for statistical analysis. The speech in the original work was processed using a hardware based system to mel-scaled frequency banks. Finnish and Japanese[2] were reported as well-suited languages for such systems, because the written language has direct phonemic orthography, meaning that there is a direct correspondence between how the language is written and the phonemes. This differs from a language like English, which does not have a highly consistent connection between spelling and pronunciation. English has a number of ways that the spellings of words can vary from the way they are spoken. For example, English has many words that have silent letters ("Wednesday"). Combinations of different letters can make the same sound ("weigh" and "way"). Words with the same spelling may

[2]Note that Japanese has two sets of symbols, hiragana and katakana, which are syllable based rather than the direct one-to-one mapping of phoneme to morpheme of Finnish.

have different pronunciations ("desert [verb]" and "desert [noun]") due to stress and voicing. Different meanings and parts of speech may be conveyed through pronunciations that are not reflected in the spelling.

Torkkola (1990) used the outputs of SOM to determine a tree decision structure of rules that were used to improve the results of the KNPT. Specifically, the rules were generated to compensate for coarticulation effects in Finnish. Despite initial work in KNPT speech recognition for Finnish and Japanese (Kohonen et al., 1988), it has also been developed for a number of languages, including Danish (Danielson, 1990) and Malay (Eng, 2006). It has also found use in the related topic of phoneme recognition for Danish and British English (Dalsgaard, 1992).

Rigoll (1990a, 1990b, 1991) investigated SOM as an input for HMM speech recognition. A variation of the learning procedures was found to be better for speech recognition using KNPT as an input to an HMM. Over the learning iterations, a constant learning factor and neighbourhoods decreasing to zero was found superior to the standard SOM, which has a decreasing learning factor and decreases the neighbourhoods, but not to zero (Rigoll, 1990a).

KNPT has significant potential for speech recognition, with much research conducted using it in the 80s and 90s, though more modern techniques such as GMM-HMM and deep learning are more popular today. Two aspects of speech recognition are the topics of ongoing research into KNPT, phoneme recognition and phoneme alignment, which I will now discuss.

3.4.1.1 Phoneme classification

Starting with Knagenhjelm & Brauer (1990), the use of the KNPT was soon reported for another application: the classification of vowels in continuous speech. Although phoneme recognition is similar in its goals to the work of the original research group, a significant difference exists in the emphasis on post-processing to improve the system's results. For speech recognition, the goal is to create text that matches the intended meaning of the speaker; for phoneme recognition, the goal is to match the phonemes that are uttered. The distinction between the two is not always clear, but the main difference is whether the system is designed to classify a complete set of phonemes or a subset.

In a comparison between MLP and Kohonen networks (and a hybrid of the two), Knagenhjelm & Brauer (1990) investigated phoneme classification for vowels. The centres of vowels in TIMIT database were extracted and the onset and codas were discarded. Using the metric of phoneme error rate, all the types of networks were found to produce similar rates of correct classifications, although different vowel confusions were observed for each.

Sequences of outputs from a KNPT trained on Finnish were used as inputs to a second SOM

for phoneme recognition (Kangas, 1991). Kangas found that classifying plosives[3] separately, in a different map rather than in the same one used for the other phonemes, could improve results. This outcome provides some indication that a focus on classifying phonemes could assist in speech recognition, though the evaluation was on the phoneme level.

Classification results are not always straightforward to interpret. Albeverio et al. (1997) introduced the concept of whether a label is sharp. Classes were considered as 2-grams, or pairs of phonemes. If the match was incomplete (a given transition matched either the first phoneme label or the last, but not both), then it was considered to be unsharp. This approach to mapping the sounds of phonemes is one that many researchers do not address, particularly when small windows (e.g. 25 ms) of an entire phoneme are considered, from onset to coda, because two different phonemes may contain identical segments but be distinguished by having different combinations of segments.

Let me give an example from Australian English. The words "hade" (/hæɪd/) and "howd" (/hæɔd/) both contain the /æ/ sound, but one transitions to /ɪ/ and the other to /ɔ/. Owing to the way that sounds are produced and distinguished, there may not always be sharp distinguishing aspects that allow us to determine whether a transition is correct or wrong. The units of analysis for the type of phonetic analysis used by KNPT may not always lead to clear categories. Albeverio et al. suggested that one way to evaluate phoneme recognition is to give partial credit for results that are not completely correct but not completely wrong.

An online version of the Kohonen SOM called ESOM that adds and deletes nodes during training was proposed in Deng & Kasabov (2003). As a result of the addition and removal of nodes, ESOM can arrive at a better fit to the data within one epoch, but the results are not two-dimensional. ESOM is a special method of constructing maps that are somewhat like SOMs. Actually, they are constructed serially instead of by a batch method, and they do not require multiple passes for training. Deng & Kasabov's research into ESOM showed it to be suitable for phonetic classification. The phonetic classification task was used to show to benchmark the abilities of ESOM against other methods of phoneme recognition; however, the researchers did not provide a direct comparison with the original KNPT algorithm. KNPT was used as just one benchmark, among many, and the particular variant ESOM was not evaluated for particular strengths or weaknesses beyond its high accuracy and rapid training.

Lee & Chang-Young (2011) trained a KNPT system using the speech of 100 male speakers who each uttered 14 consonants in the context of Korean /Ca/ utterances (syllables comprised of a consonant followed by the vowel /a/). The SOM was able to distinguish between some, but not all, pairs of consonants (e.g. distinguishing between /t/ and /d/ proved to be problematic).

[3]Plosives are phonemes for which the airstream is stopped and then released with a burst of air. In English, the plosives are /b/, /d/, /g/, /k/, /p/, and /t/.

3.4.1.2 Phoneme alignment

In contrast with the task of speech recognition, with the goal of producing a transcript of utterances, phoneme alignment is intended to produce an indication of where the phoneme boundaries occur in time. The evaluation of a phoneme alignment system looks at how well the label boundaries are determined. Early work towards phonemic speech recognition in Kohonen's group was continued in Torkkola (1988), who investigated the phonetic alignment component necessary for such a speech recognition system. This work involved the use of heuristics to determine, from the sequential output of the KNPT, the transcription of speech that best matched the audio input within frame boundaries.

Dalsgaard (1992) used KNPT for phoneme alignment in Danish and English with LPC-based cepstral coefficients and LVQ for fine-tuning. Phoneme alignment was found to have higher overall accuracy for Danish. This seems consistent with the general idea that KNPT will have better performance for languages with strong phoneme-grapheme correspondences.

As Togneri et al. (1990) describes, SOMs may be used for segmenting phonemes without making assumptions based on the labelling of the data. Supervised MLPs were trained for the desired classifications. By observing the properties of the resulting maps, Togneri et al. concluded that the KNPT is able to represent the approximate speech space. The implication for my research is that KNPT can represent speech in a way that can allow visualisation of phonemes, particularly vowels, with details of where they start and stop.

3.4.2 Dysphonia recognition

Another area of application for KNPT is the identification of non-standard speech. Leinonen et al. (1991) presented an analysis of the long vowel /aː/ as spoken by hoarse and healthy people that relies on the trajectories on a SOM trained on this data. The authors found that hoarse speech had longer trajectories, meaning that more units of the map were visited and units were further away. Healthy voices were characterised by trajectories between two to three neighbouring cells. In other words, the speech signal of a hoarse voice may include different spectral patterns that are discernible when characterised by a SOM trained on both normal and hoarse voices. I would attribute this attribute of a voice that is hoarse to less consistency. Due to the learning procedure, the neighbouring nodes in a SOM are more similar than those that are more distant, so less consistency means the trajectory must jump to a more distant node.

Rihkanen et al. (1994) explored the use of KNPT in the classification of dysphonia, roughness and breathiness in the voice. 19-dimensional spectral features were used as input. This research revealed an important question: how can the results of KNPT analysis be evaluated when human evaluations of dysphonic voices can be quite variable.

A two-layer network, consisting of a KNPT for dimensional reduction and a second layer consisting of a MLP was designed by Szczurowska (2006) to classify stuttering. The SOM portion of the study was described positively in terms of describing syllables, facilitating identification of non-fluent speech. The SOM consisted of only 25 neurons, a relatively small size network in contrast with a relatively large quantity of rich speech data. The 25-cell network size was described by the authors as sufficient for the first layer for stuttering detection.

Prolongations and repetitions are commonly present in the speech of stuttering people. The recognitions of the prolongations using a KNPT was investigated using SOMs for discrimination of non-fluency by Świetlicka et al. (2009), who again used 25-neuron networks. The smaller number of neurons in these networks is commensurate with the smaller data sets to be analysed. Venkateswarlu et al. (2011) investigated a number of different ways to pronounce (and mispronounce) a single word to identify pronunciation variation between speakers. It was found that it was possible to make comparisons between speakers, at least for small data sets evaluated particularly at the syllable level.

Dysphonia is a physical condition that prevents the speaker from producing clear speech. The focus of the research conducted for this thesis is CALL, with the assumption that discrepancies in speech are attributable to not having learnt to produce a vowel with a target accent. The above work on dysphonia reveals that some variations may be due to physical deficiencies. It may also provide some more general insights into how KNPT may be used to evaluate speech.

3.4.3 Non-phoneme vocalisations

There are many features that can be used to describe the sounds that babies make. Though babies have not yet acquired the ability to speak, the sounds that infants produce in their first year may be considered as "prelinguistic phonation categories (squeal, vocant, and growl)". Warlaumont et al. (2010) investigated the utterances of infants, finding that phonation categories could be classified using 16 neuron SOMs for feature reduction with a perceptron upper layer. Furthermore, Warlaumont et al. suggest that SOMs have significant potential for application to better understand infant vocalisations, both prelinguistic and as linguistic categories are acquired (as these vocalisations become more speech-like).

The use of SOMs is not limited to speech (or speech-like sounds) produced by human. Related work into classifying animal vocalisations both between and within primate species was carried out by Pozzi et al. (2012). The researchers used a single layer KNPT with 90 neurons, training with labelled data, reporting error rates under 20% for four of the five species. The researchers used the values of three formants (F0, F1, and F2) at the initial, nucleus, and coda of vocalisations as input to cluster vocalisations. A SOM was also used in an unsupervised way to identify that the input representation was sufficient to distinguish between five different types of vocalisations (of a set of what are traditionally considered to be seven different

vocal types). Although such work deviates from phonemes, which are solely the production of humans, the researchers expressed the potentials of the KNPT algorithm for non-human vocalisations, particularly for those animal communications for which there is limited human understanding.

Birds are well known for their communication through song. The classification of the syllables of bird songs were addressed using KNPT in Ranjard & Ross (2008) and Ranjard (2009). The specific variant of the self-organizing maps algorithm that was used was growing SOM. Ranjard explored the relationship between the number of nodes in the network and the mapping precision, and found that an initial size 50, with a small number of added nodes, was sufficient.

Similarly, the speech of wild parrots was investigated in Skrípal (2006) using MFCCs and SOMs. Classification of birdsong using SOMs in Stastny et al. (2013) enabled classification of species by the sequential arrangement of their songs.

To my mind, the successes of KNPT research with non-phoneme vocalisations with provides some confirmation that even though the SOM is not strictly neuroanatomical, perhaps the self-organisation can reveal some underlying patterns similar to how brains represent vocalisations. After all, both SOMs and the brain exhibit topological structures that emerge through exposure to vocalisations.

In the case where non-phoneme vocalisations are classified, the acronym KNPT does not really fit, because phonemes are not the output classes. Truly, the moniker Kohonen Neural Vocalisation Typewriter (KNVT) could be used for such work. In this thesis, with its goal of understanding KNPT in terms of the unsupervised learning of human phonemes, I will not address baby or animal noises, but I have mentioned them here because they have strong parallel with the current work.

Also, though Kohonen (1995) recognised that complete modelling of all the parts and processes of the brain could not occur with the technology available in the 80s or 90s (or for that matter, even today in 2017), work can still be done towards creating a model that implements in a computationally efficient way some of the interactions between feature sensitive cells. Though the cells of the brain do not compete for each input in exactly the same way that the SOM algorithm performs this procedure, Kohonen was able to build on the work of Malsburg (1973) and others to develop the principles of self-organisation. Kohonen & Hari (1999) presented an analysis of potential mechanisms for self-organisation to be performed in the brain that was mirrored by neuroscience studies of the hearing of owls and echolocation of bats that investigate how these abilities are arranged in the brain.

3.5 KNPT Voice Activity Detection

In the analysis of speech, the actual utterances are only part of the story. Before, after, and between utterances, and even within them, are pauses. Using SOMs for the analysis of speech most frequently deals with silences using a separate voice activity detection method. Here, the KNPT literature relating to the problem of VAD will be introduced. These are revisited later in this thesis in section 6, which deals with voice activity detection in the context of KNPT systems.

As mentioned, many researchers use pre-segmented data or a heuristic for eliminating silence from the training data. On close reading of the initial Kohonen work, it apparently used presegmented speech for training (and evaluation). On the one hand, there are known heuristics for distinguishing between silence and speech. But on the other, there does not seem to have been much analysis of KNPT for continuous recordings that include speech but also significant portions of silence. In other words, there seems to be a gap in the research in addressing the applicability of KNPT to audio signals that contain both speech and silence. Is the detection of voice activity an essential preprocessing step? Could it be performed by SOMs?

It has been my experience that including silence in the input can create issues for SOMs. Additionally, this issue has been noted in the literature. For example, Togneri et al. (1990) observed that the single-layer phonotopic map was unable to resolve silence, and the resulting trajectory for such portions of audio signals was erratic. This issue was resolved by an initial stage (using MLP) that would indicate whether speech was the target /i/ or not.

A common approach for KNPT is to segment the portions of speech under consideration prior to processing. For example, in an investigation of consonants, Morris1997 used, “35 ms voiced speech segments taken from immediately before consonantal closure and after consonantal release, i.e. from the vocalic portions of the transitions only.” By such an approach, which segments the speech before analysis, the phenomenon under consideration can be investigated. This is often accomplished, with datasets such as EUROM-0 or TIMIT, by simply using manual annotations to determine the start and end times. But when considering larger datasets that do not contain annotations, is it still possible to perform analysis? As Godino-Llorente & Gomez-Vilda (2004) described (for LVQ), “It is important to remove silence/noise frames to avoid modeling undesirable ambience signals.” This approach requires using annotations for segmentations, so precludes unsupervised learning.

Another approach, less frequently taken, is to train networks on audio that contains silence as well as speech. For example, Anderson (1991) used silence as a broad phonemic category, and found high accuracy ($> 90\%$) for silence portions using either audio representations of power spectrum or mean-rate. However, it is interesting to observe that the results for other phoneme classes was much lower, around 30%, although vowel accuracy was around 65%. Because input frames were normalised to unit length, my hypothesis to explain these results is that silence

caused issues for the training of the maps. This would be particularly evident for some of the speech sounds (e.g. stops) that contain portions that are very similar to, or indistinguishable from, silence.

This classification problem of distinguishing speech from silence thus seems particularly difficult for some portions of unvoiced consonants (such as /t/ or /d/), which have certain portions that are virtually indistinguishable from silence because the speaker ceases to emit air in the production of a stop consonant, for example, which means that for short periods of time, no sound is produced. To demonstrate this, I recorded the word "stop" and annotated the amplitude waveform in PRAAT, as shown in Figure 3.1. Silence before and after the word is indicated with /(...)/. Portions of the phonemes /t/ and /d/ are also silent due to this closure. As a result, simply determining silence using an amplitude threshold would result in portions of some phonemes being classified as silence, assuming that there are some frames that are sufficiently brief to capture only the silence portion of these phonemes. In this example, the /p/ took approximately 125 ms, so a sequence of about five of the 25 ms frames with 10 ms overlaps would capture only the silent portion of the /p/, in which case the signal would be virtually indistinguishable from silence.

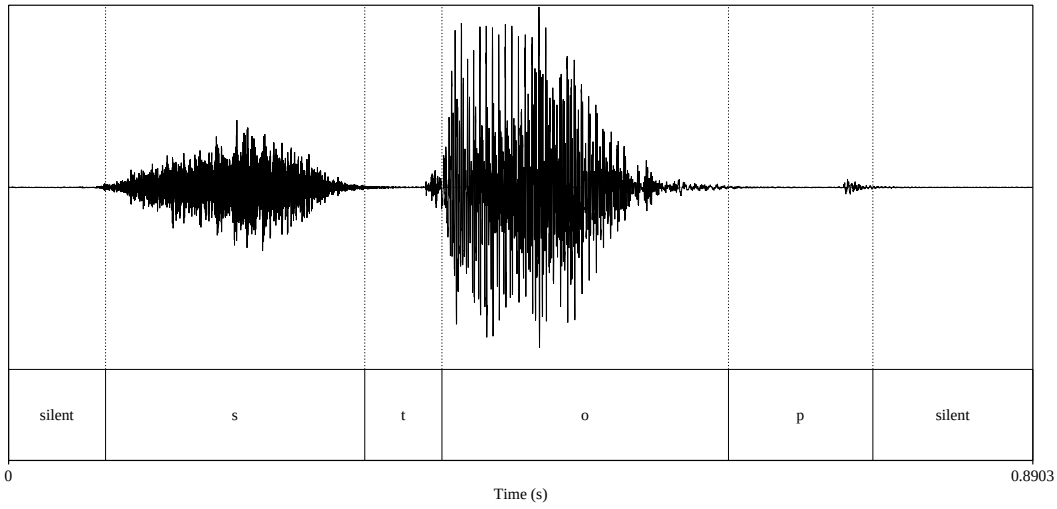


Figure 3.1: Speech spectrogram for the word "stop". The consonant sounds /t/ and /p/ are associated with moments of silence due to the breath being stopped by the closing of the lips.

In reporting the development of a a hybrid KNPT system that uses speech audio that included silence, Eng (2006) reported that an algorithm for determining endpoints (the start or end of silence, and voiced and unvoiced speech) was developed that used heuristics to fine-tune endpoint detection that was determined first by background noise, then with energy-power rms and zero crossings. The maps were then trained with Euclidean distance of cepstral coefficients. It was found found that boundaries between silence and speech were resolved by a method of Euclidean distance. What I find interesting is that the most common form

of SOM also uses Euclidean distance, but whether SOM would be an alternative method for detecting speech endpoints was not considered. This theme is common to KNPT research: when deleterious effects of silence are noted, the solution is almost invariably the introduction of another algorithm for voice detection.

In an investigation of KNPT for Spanish phonemes, Gardón et al. (1999) used silence as well as speech as inputs to the maps. In addition to training a map on all the frames, five additional maps were trained, which identified: vowels, fricatives, plosives, liquids and nasals. Using a hierarchical approach to classify the sounds resulted in better vowel recognition (reported recognition rates were 39.53% for vowels in the base map but 80.86% in the vowel map). Gardon et al. did not investigate how the inclusion silence degraded the capabilities of their classifiers, though recognition rates for silence were reported (27.51% in the base map and 67.02% in the group maps, although how the group maps, none of which was specialised for silence, enabled better classification of silence was not described).

The handful of KNPT researchers who have included silence notwithstanding, and despite the apparent need for segmentation, and the possibility that SOMs have potential for the task, I have found only one study that specifically investigates speech detection using SOMs. For voice activity detection, Grashey (2003) reported frame-level accuracy for a two-class KNPT classifier, which classified speech versus silence. Similar to the approach used in this thesis, Grashey used 12 MFCC coefficients (though he did not use information from neighbouring frames, such as are used to compute delta coefficients). A comparison was made with the classification made by an HMM, but it was reportedly difficult to determine which classifier, the SOM or the HMM, was more correct. However, it was found that by smoothing the output by taking the average output over a larger window than the frame, the difference between HMM and SOM could get smaller. Prior to this moving mean smoothing, the SOM was classifying small pauses between syllables as silence, but the HMM was classifying them as speech.

To address this gap in the literature, I will report more aspects of KNPT for voice activity detection, as well as for vowel segmentation, as revealed by my investigations. The main research hypothesis in terms of voice activity detection is that SOM can segment speech from silence, or vowels from non-vowels (comprised of both silence and consonants as included in the audio input). This will address the question of whether another algorithm than SOM, such as a voice activity detection heuristic, is really necessary for KNPT research. If the goal is an unsupervised learning method for classifying vowels in unlabelled speech files, we should understand how silence affects the results of KNPT. If the elimination of silence prior to classification is advantageous, perhaps it would be preferable to use KNPT for this purpose rather than a different algorithm.

3.6 KNPT Architectures: Ensembles, hybrids, and variations

KNPT architectures refers to how the inputs, maps, and outputs are interconnected to work together. Even in the original work, Kohonen describes difficulties with certain phoneme confusions being registered with the same neuron in the map. One of his proposals was to have another layer of maps, which were tuned to better distinguish these confusions (Kohonen, 1988). To advance the basic algorithm, multiple maps may be combined in various ways. Additionally, other machine learning and statistical tools may be used to enhance results. A significant interest in KNPT is thus how to improve results using ensembles, hybrids and hierarchies of maps.

It seems that researchers have coined almost as many different terms as there are different network architectures. Rather than explore this diverse terminology, I will describe the two major types: ensembles and hybrids. KNPT ensembles use multiple maps to improve results. This arrangement may be hierarchical, such that data is processed by one map then sent to one or another map. Or it may be an ensemble of maps, such that classification results from multiple maps, trained or labelled differently, that are combined using data fusion techniques. Hybrid SOM techniques use SOM alongside other classification techniques like HMM or MLP, generally in a hierarchical manner, with the results from one technique feeding into the other.

I prepared Figure 3.2 to illustrate the three fundamental architectures used in KNPT research. Firstly, the SOM map may be used by itself or as an ensemble of SOMs. Secondly, some recurrence may feed the output of a SOM back into itself, which is called a recurrent SOM (RSOM). Finally, the results of the SOM algorithm may be used in a hybrid architecture, perhaps feeding into a MLP or HMM.

3.6.1 Single-layer SOM

The single-layer architecture is the most basic structure for a SOM. The details of the initial KNPT algorithm, as proposed in Kohonen (1988); Kohonen et al. (1988) and described in Kohonen (2014), were elaborated in section 3.3. Kohonen et al. reported both word recognition accuracy and phoneme labelling accuracy, as well as segmentation accuracy. I will describe my own explorations of the single layer architecture in Chapter 4. Here, I will describe how other researchers used and evaluated the single layer SOM architecture for phonemes.

KNPT is both a clustering method and a classification algorithm. Evaluation of the classification results of KNPT are reported here. The methods of evaluation are reported, but not the results, because there are too many variabilities to make fair comparisons between the experiments by different researchers. Accordingly, rather than focus on the specific statistical

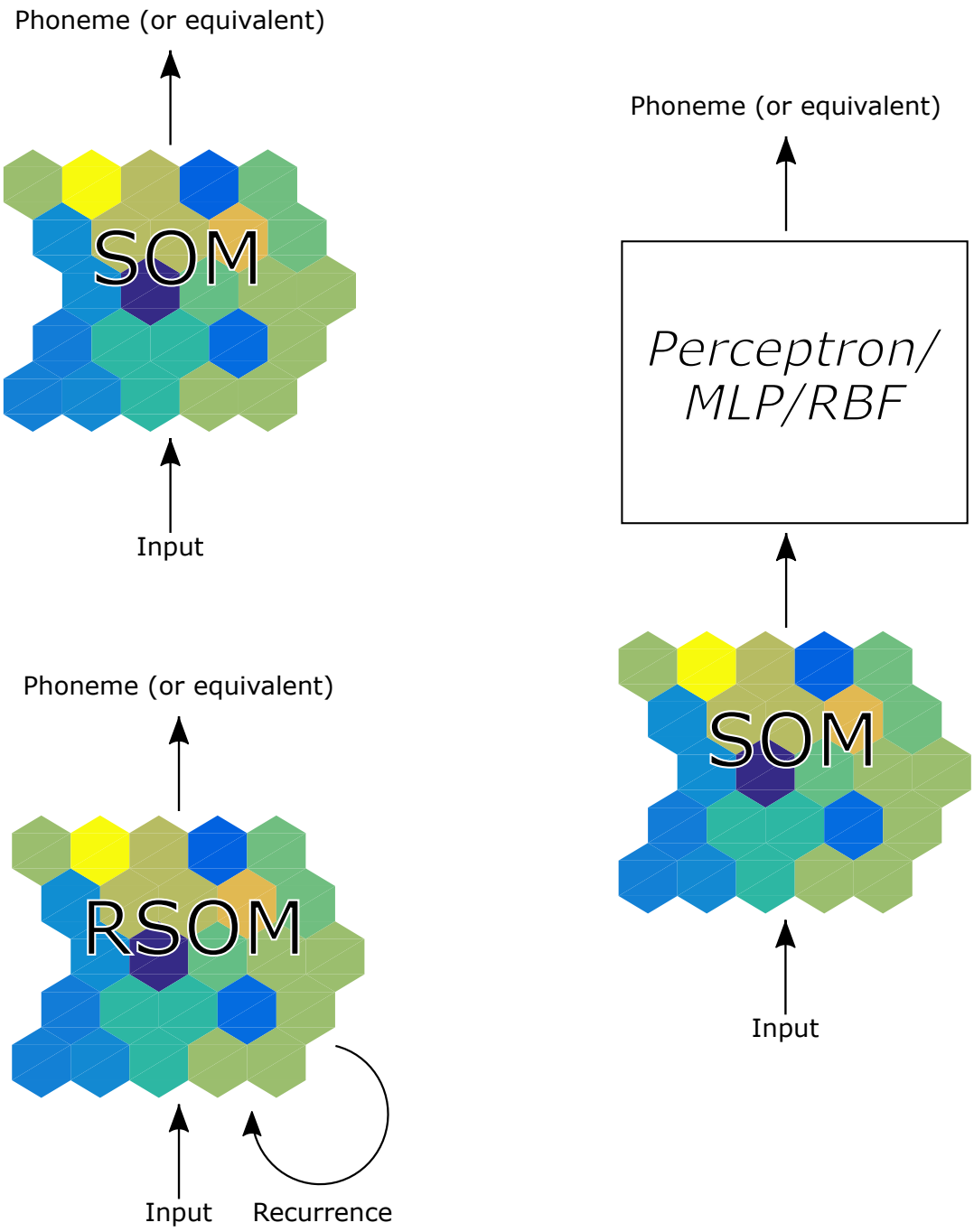


Figure 3.2: Three architectures typical to Kohonen Neural Phonetic Typewriter research: SOM, RSOM, and hybrid.

values that are reported, I will instead describe the different ways that researchers report their results so as to inform how KNPT systems may be evaluated. For classification, this ranges from word recognition accuracy to frame accuracy for specific frames in the speech stream.

As described by Kohonen (1990), "To someone familiar with practical speech recognisers it will be clear that it would be meaningless to evaluate and compare different test runs statisti-

cally; the results obtained in each isolated test depend so much on the experimental situation and the content of text, the status and tuning of the equipment, as well as on the physical condition of the speaker." Thus, while it is possible to report a general accuracy for phonemes between 80% and 90%, and with the addition of compensatory methods (such as rule-based smoothing) to around 92% and 97%, making comparisons between studies, even from the same research team, is not really possible.

The research literature on KNPT measures success in a number of different ways. The application, or purpose, of a KNPT system that a research study intends affects the particular metrics that are presented. Two main tasks, word recognition and phoneme recognition, are common in the literature. Word recognition uses words as the unit of analysis, seeking to resolve how well the system can be used to automatically transcribe the words of continuous speech. Phoneme recognition, on the other hand, uses transcription of phonemes as the goal, but analysis may be undertaken at the phoneme level, the frame level, or the window level. Finally, the endpoints of the units may also be of interest to the researcher, with such measures as vowel onset time.

Anderson (1991) used more traditional speech recognition metrics to evaluate the performance of a supervised KNPT system for phoneme recognition. Results included average percentage correct, along with percentages for deletions, insertions, substitutions. Other researchers used the sum of insertions, deletions, and substitutions (misidentification) to calculate the phoneme error rate, including Torkkola et al. (1991) and Kurimo (1997).

For the single-layer map, the size of a map is a hyperparameter that may be selected by a heuristic or varied to determine ideal values. For fixed valued maps, Dalsgaard (1992) reported on experiments with a single fixed size: 20x20 SOMs. Similarly, Rihkanen et al. (1994) used 17x13 maps.

Dalsgaard (1992) used expert knowledge of the features of phonemes for both British English and Danish. For vowels, the vocalic features were sonorant, closed, mid, open, front, centre, back, and round. For consonants, the consonantal features were labial, alveolar, velar, glottal, fricative, plosive, approximant, liquid, and nasal. Silence was also mapped. A small corpus of two minutes was used for the unsupervised training of a SOM, with some tuning with a method similar to LVQ. The neurons were then associated with probabilities for each of the vocalic and consonantal features (and silence).

Other researchers used grid search, meaning that they varied the size of maps to find how map size affected the results. Rigoll (1990b) used varied rectangular maps between 10x10 to 20x20 and found that 17x17 maps gave the best recognition results. Conversely, Torkkola & Kokkonen (1991) used similar dimensions (using a range of size from 100 to 450 cells), finding that there was no effect of size on final recognition accuracy.

Not all KNPT maps are large. Some can be quite small. For the study of lemur com-

munications, Pozzi et al. (2012) used unsupervised SOMs to explore classification of unknown vocalisations. Three map sizes were reported: 10, 20 and 40 units. These are among the smallest maps in the literature. Similarly, Lundberg (2005) used 2x2 and 4x4 maps, i.e. 4 and 16 units, respectively. If the size of the input data is very small, consisting of only a few hundred observations, it makes sense that the maps are small.

A variation of the single-layer SOM is to use an autoencoder to perform the preprocessing of the audio data. Mitra et al. (2016) used 80 bottleneck neurons from the third layer of a five layer autoencoder as input to a SOM. After the autoencoder was trained on other language data (650 hours of speech in languages other than the language used for testing), it was used to encode the test data that was then presented to the SOM. This approach demonstrates how autoencoders can be used as a preprocessing step for the single layer architecture.

The map sizes used in single layer KNPT systems have almost always been low, between 100 to 1000 nodes. Apparently, since the original KNPT research was conducted with small maps, and computation time increases with the size of a SOM. Although work with larger maps has been conducted in other areas, such as a SOM for classifying a patent database that was over a million nodes (Kohonen, 2013), I have not found that larger map sizes have been explored in the KNPT literature. Since few researchers describe map size as an issue for KNPT research, and when they do, they do not elaborate on what they are looking for when they do, it took me a great deal of time to uncover that this is a potential gap in the KNPT research, particularly now that we have recourse to more powerful CPUs and memory capabilities than in the 80s and 90s. I will address map size in the following chapters.

3.6.1.1 Phoneme error rate

Although KNPT systems often operate on data frame by frame, phonemes have durations over time. The characterisation of phonemes by a KNPT system may be achieved by identifying phoneme boundaries or by chaining the sequence of classification. evaluating the resulting trajectory over time.

Phoneme labels produced by a KNPT system were compared with those produced by an alternate method (and with hand annotated utterances) in Torkkola (1988). Torkkola also checked accuracy of boundaries within one frame and within three frames of the boundaries. Torkkola (1988), who noted that he was not a professional phonetician, identified that phoneme boundaries may be identified as where the classification changes from one phoneme to another. Using a similar technique, Kangas (1991) reported phoneme error rates for different segmentation algorithms to demonstrate the effectiveness of the new methods that used contextual information.

Rigoll (1990a) used entropy to show that a constant learning rate of 0.1 was optimal, with neighbourhood decreasing to zero, reporting results in terms of phoneme recognition both with

and without the entropy maximization. It was found that vector quantisation (the codebook entries) was better for HMM than using the labels (Rigoll, 1990b).

3.6.1.2 Frame error rate

A window consists of multiple frames that are near each other. A window may also be called a segment. The results for windows are determined by associating the results from a number of different sequential frames. Windows are different from phonemes because they do not extend to the ends of the phonemes.

Togneri et al. (1990) reported classification accuracy for the /i/ sound, with a finding that most misclassification occurred near manual annotation borders, which could indicate suboptimal annotations.

Knagenhjelm & Brauer (1990) extracted the central portion of vowels from a hand-labelled speech dataset to report classification rates at the frame level. This enabled comparison between different types of networks. It was also observed that inaccuracies were not serious, which meant that confusion was most commonly associated with similar phonemes like /e/ and /i/. The confusion matrices were shown for some networks, so that readers could observe that different confusions were found for different networks.

Both frame level and segment level accuracy was reported in Danielson (1990). Segments were determined from the manual annotations. The results were used to show that LVQ could improve recognition accuracy.

A measure dubbed the "phonemic transcription error rate" was reported in Torkkola (1990); Torkkola et al. (1991), with smoothing based on majority voting in dynamically determined windows to improve results (other more complicated rules were considered, but did not perform as well as simple majority voting). Some problematic phonemes (plosives) that are classified separately on an auxiliary map were considered the same for error reporting purposes in the base map. The errors and correct classifications were reported, with comparisons between uncorrected and corrected presented to show that the smoothing was beneficial. The goal of the dynamic focus was to successively narrow windows to enable phoneme-level reporting.

3.6.1.3 Clustering

The KNPT algorithm is not only a classification method, but it also has clustering capabilities. Evaluation of a clustering method may be considered under internal, relative or external criteria (Halkidi et al., 2001). One way that many KNPT researchers addressed clustering was to provide visual displays of labelled maps. Such a representation allows for inspection of the maps, which reveals that neurons that are near each other typically have the same labels.

Kohonen et al. (1988) showed a labelled map. Those neurons that did not have a clear majority winner were indicated with a question mark ("?").

Trajectories are also displayed in many articles. Trajectories indicate the series of nodes that are visited over time as they best match the input. The size of jumps in trajectories were used to identify hoarse voices from healthy ones (Leinonen et al., 1991). Trajectories were also depicted in Torkkola & Kokkonen (1991). It was identified that the occurrence of multiple sequential hits would not be visible on trajectories. These happen when successive frames of audio result in the same matching unit. This was resolved for visual inspection by performing some filtering in time, such that the hits would be displaced from the BMU for increased visibility.

Anderson (1991) used metrics of cluster distortion and codebook entropy to evaluate the maps. The cluster distortion was defined as the average squared error between the codebook value (the centre of a cluster).

$$D = \frac{\sum_{i=1}^{N} \sum_{\epsilon_{c_i}} |\bar{x} - \bar{m}_i|^2}{N} \tag{3.4}$$

Here, N is the number of neurons in the map. The error is determined between the codebook value and all the data points for which a neuron is a BMU.

Another measure that was introduced was entropy. The entropy was computed for each codebook value as follows.

$$E = -\sum_{i=1}^{N} p_i \log_2(p_i) \tag{3.5}$$

The relative frequency p_i is determined as the number of hits for each neuron in proportion to the total number of hits for the entire map. A map that spreads the data evenly will have a lower value, and higher entropy will occur when the data is spread out less evenly. When entropy is higher but distortion is lower, the map may be considered better than other maps for the same data that have lower entropy but higher distortion.

3.6.2 Improving results

The results of KNPT can be improved in a number of ways. The input to the network can be preprocessed, for example by normalisation of the inputs. For a SOM, it is common to scale the inputs using the variance or to a 0-1 scale so that each feature is similarly weighted. Parameters that affect learning such as neighbourhood radius can be explored, as can those that affect the resolution such as the size of the maps.

3.6.2.1 Learning vector quantisation

In addition to his work with SOMs, Kohonen developed a similar algorithm to SOM called LVQ. LVQ is a supervised approach counterpart to SOM, but "unlike in SOM, no neighbourhoods around the 'winner' are defined during learning in the basic LVQ, whereby also no spatial order of the codebook vectors is expected to ensue" (Kohonen, 2001, p. 245). For LVQ to be used in KNPT as a final tuning stage, it must be modified to include the SOM algorithm neighbourhood function to reduce the loss of topological structure (Kohonen, 1995). Indeed many researchers in the KNPT area use supervised techniques including LVQ to improve the mapping of speech to phonemes.

Kohonen developed a number of different variations to the LVQ algorithm (LVQ1, LVQ2, LVQ2.1, LVQ3, OLVQ1, and LVQ4). Essential to all forms of LVQ is a competitive learning method that is similar to SOM in that it is a map with associated codebook vectors, but each neuron in the LVQ map is also associated with an output class. The LVQ algorithm matches an input data point to find the best match. During learning, the BMU is updated according to reward/punishment rules that are defined by the LVQ algorithm variant. For example, in LVQ1, if the BMU comes from the same class as the neuron, it is moved toward the data point (reward). If it comes from a different class, it is moved away (punishment). Another difference is that in LVQ, only the winner is updated, i.e. no neighbourhoods are used (Kohonen, 1995). In this treatment of KNPT, I restrict consideration to those maps that use the SOM concept of neighbourhood as fundamental.

It is common to use SOM as an initialisation procedure for LVQ. The results of the SOM algorithm have been found to be improved by the use of LVQ fine-tuning subsequent to SOM training. Knagenhjelm & Brauer (1990) found that the results of KNPT for phoneme recognition could be improved slightly by LVQ2, however it was not mentioned whether the topographic information was adversely affected. If the topological nature of the maps is to be preserved, the LVQ algorithm must be modified to include the SOM concept of neighbourhood (Kohonen, 2014).

A number of KNPT studies that I surveyed used LVQ in a fine-tuning role, c.f. Danielson (1990), Kangas (1991), Torkkola & Kokkonen (1991), Dalsgaard (1992) and Somervuo & Kohonen (1999). The main drawback to LVQ is that it's reliant on class labels. An area for future study could explore the use of techniques such as weak supervision to LVQ tuning by tuning the positions of nodes to enhance the discriminative capabilities.

3.6.3 SOM algorithm with modifications

In addition to the basic SOM algorithm, a number of modified versions have been used to improve performance of a KNPT. These include leaky integrators and adaptive SOM, which I

will now describe.

3.6.3.1 Leaky integrators

Leaky integrators were introduced in Kangas (1994). Leaky integrators is a method that holds activation of neurons over time. The subsequent output of a time sequence can be used as the input of another map.

More recently we have seen leaky integration incorporated in recurrent SOMs, which adds to the temporal encoding by incorporating the previous activations as additional features in the training of the SOM (Chiraz et al., 2015). They additionally incorporate a growing aspect, such that maps are increased in size when it is determined that certain neurons are associated with a more than sufficient quantity of input data, based on quantisation error. Their work focused on vowels in the TIMIT corpus.

3.6.3.2 Adaptive SOM

Arous & Ellouze (2003) created a variant of the SOM algorithm called Adaptive SOM that uses neighbourhood radius, which is selected based on the quantisation error, a measure of the similarity in the distance of the codebook vector to the input vectors. Adaptive SOM differs from the standard algorithm in that the BMU is selected with consideration of the quantisation error. Arous also fused the outputs of adaptive SOM and four other forms of the SOM algorithm (standard, optimised, and supervised SOM as described in Kangas et al. (1990) and conscience SOM as in DeSieno (1988). Experiments were conducted using the TIMIT speech corpus, using three frames from the centre of each phoneme. Three different map sizes were 10x20, 20x20, and 40x40. It was remarked that some vowels are quite different, and thus easy to separate using any SOM variant and map size, but that similar phonemes were found to be difficult to separate, so future work was suggested. Indeed, these results are similar to the initial difficulties I report in section 4.7 and eventually found were resolved with larger maps.

3.6.4 Ensembles of SOMs

An ensemble of SOMs combines the output of multiple SOMs to gain better results. The outputs can be combined by voting, averaging, or data fusion. One example of a hybrid ensemble of SOMs was presented by Koreman et al. (1998a), who used two SOMs, one trained using MFCCs as features and the other using delta MFCCs (the result of subtracting the previous frame of MFCCs from the current frame). The outputs of both networks were then combined using HMMs.

Kohonen introduced a similarity measure for strings for SOM that allows maps to be constructed of symbol strings (Kohonen & Somervuo, 1998). I do not consider these networks as KNPT because they do not take speech audio as their inputs, but they are of interest in the context of KNPT because they use SOMs and can be used as a subsequent step to evaluating speech for phonemes. The output of KNPT, a string of phonemes, can be processed by a SOM of symbol strings in a hierarchical manner.

That there are some classes of phonemes without clear separability is an important consideration for KNPT. To tackle this problem, Arous & Ellouze (2003) compared several variants of the SOM algorithm, combining them into a system of SOMs, and found that using only 12 MFCCs from three central frames, no KNPT alone was able to separate vowels that are phonetically similar. However, by hybridising SOM and genetic algorithms, the SOM parameters could better disambiguate similar phonemes.

Chiraz et al. (2015) described a hierarchical system of SOMs using the principle of a growing hierarchical self-organising model (GHSOM) that allows maps to grow horizontally and hierarchically. To grow horizontally, a map unit can be divided into multiple units to narrow the input space of each unit. To grow hierarchically, a single map unit can be represented by a new SOM. Vowel recognition rates were compared with the best results achieved by a cooperative system that combined the results of different variants.

3.6.4.1 Temporal KNPT

Speech is a stream of information carried in an acoustic signal. The analysis of speech must somehow represent its temporal nature. In presenting a taxonomy of temporal sequence processing with SOMs, Guimarães et al. (2003) presented three main ways that temporal data is modelled with SOMs as follows: 1) implicitly considering temporal aspects only in the processing of the data, either before or after the SOM; 2) incorporating temporal dependencies through a modified SOM, in the learning or classification rules; or 3) modifying the SOM topology, either hierarchically, from one map to another, or incorporating feedback, from one node to another.

In this thesis, I explore the first representation through SOM preprocessing of the data and training SOMs on the windows. However, an analysis of the third (feedback) reveals that using concatenated frames as inputs to the maps is akin to incorporating feedback from a previous frame with the current.

3.6.4.2 Auxiliary maps

A hierarchy of SOMs means to have classification performed by multiple maps. The classification made by the first layer can be used to select portions of data that are used for training

other SOMs. This idea of using "auxiliary maps" was mentioned in the original work on the neural phonetic typewriter (Kohonen, 1988) Investigating the capabilities of auxiliary maps for speech recognition, Kangas (1994) found that their use significantly improved the classification of unvoiced plosives[4].

Gardón et al. (1999) used a hierarchy of maps with a 10x20 map and five smaller 5x10 maps that grouped by phoneme types (vowel, plosive, nasal, fricative, or liquid). Training of the base map was unsupervised but the phoneme labels of the training data were used to label the map. The best results were found for detecting vowels and silence.

And in my own work, published as Anderson & Powers (2016) and reported in Chapter 5 (section 5.7), ensembles of SOMs were used such that the first layer to distinguish between silence, consonants, and vowels, and the second layer to classify vowels was explored.

3.6.5 Hybrid KNPT

Despite the capabilities of SOMs for machine learning of phonemes, or perhaps because of these capabilities, a number of researchers have joined KNPT with other machine learning techniques. In addition to the ensemble techniques previously described, I found that there were three main areas of hybrid KNPT research: HMMs, MLPs, and fuzzy neural.

3.6.5.1 Multi-layer perceptron

SOM may be used as a first layer in a multilayer network. For example, Torkkola & Kokkonen (1991) used the topological properties of the SOM algorithm to generate trajectories that could be classified by a pattern recognition algorithm such as a feed-forward ANN.

Eng (2006) used a SOM for feature extraction with a second step of a three-layer MLP with 30 output nodes. Performance was evaluated on the recognition of 30 two-syllable words at the syllable level and at the word level. It was found that a 15x15 SOM would provide better recognition accuracy than smaller maps. Additionally, using more cepstral coefficients was advantageous (using 20 or 24 LPC coefficients outperformed 12 or 16).

Tashan et al. (2014) trained a single dimensional (64 units x 1 unit) SOM with MLP for speaker verification. An individual map was trained for each speaker, using an energy measure to detect the vowel portions from three prompts: "two", "five", and "eight". It appears that supervised tuning may have allowed the network to better characterise the vowel codebooks.

[4]For Finnish, the plosives are /k/, /p/, and /t/.

3.6.5.2 Hidden Markov Models (HMMs)

HMMs are one of the most popular approaches used in speech processing. HMMs are a sequential Bayesian network that perform inference (Marsland, 2015). Some KNPT researchers have explored the use of HMMs with SOM.

Rigoll (1990a, 1990b, 1991) used SOMs trained on speech and labelled with phonemes as input to HMMs. Mutual information in the label stream, which is related positively with the entropy of the data, was associated with better distinction of sounds, but may lead to more "wrong" labels. Zhao Zhao & Rowden (1992) used KNPT as an input to HMMs, reporting speech recognition accuracy for digits.

3.6.5.3 Fuzzy neural

Kasabov et al. (1998) used a multi-layer network comprised of different types of networks. Firstly, mel-scale coefficients (MSC) were used to train fuzzy neural networks (the error reduced using back propogation), one network for each phoneme. Each phoneme was thereby transformed into phoneme features by the fuzzy neural network. Subsequently, word recognition was achieved using a fuzzy SOM, which takes as input those phoneme features that were produced by the fuzzy neural networks. In this architecture, the phonemes are determined by the fuzzy neural networks (that did not use the Kohonen SOM algorithm) but the words are determined by sequential information as processed by the SOMs.

3.6.6 Different dimensionalities of SOMs

SOMs can be more than two-dimensional, but usage of other dimensionality maps were rarely reported by researchers in the KNPT literature. Virtually all KNPT research has been conducted with two-dimensional maps. The main reason for this dimensionality is that it is the most suitable for visualisation. When a goal of the KNPT is visualisation, then maps may be best represented as two-dimensional hexagonal sheets (Kohonen, 2013). It is not as straightforward to display higher dimensionalities in a static way on paper or a computer monitor (though they may be explored interactively).

Togneri et al. (1990) investigated different dimensionalities of SOM, from two-dimensional to five-dimensional. The quality of the maps was also indicated by the amount of "crinkle", meaning how much differed from planar. They reported that four-dimensional maps minimised crinkle, indicating that the signal representation that was used may be best mapped in four dimensions. This idea of crinkle is similar to what Kohonen (2014) calls "folds" in a map. It has been found that the width of the neighbourhood function has significant effect on the

smoothing of folds. Accordingly, perhaps higher dimensional maps reduce the crinkle factor by controlling the neighbourhood width more appropriately to the data distribution.

Although the four-dimensional torus was observed to have a lower crinkle factor than the two-dimensional plane for 12-coefficient MFCCs (0.09 vs. 0.04), the significance of that crinkle factor was not really addressed beyond the observation that lower crinkle would seem to better suit the input distribution. Given that selecting the suitable neighbourhood width was not directly addressed in Togneri et al. (1990), this could be an potential avenue of research for those interested in higher-dimensional spaces.

A novel form of one-dimensional SOMs was explored by Tashan et al. (2014) for the task of speaker verification. A single dimensional SOM trained on three vowels (/æ/, /u/, and /i/) was used as an input to a MLP, which was able to improve performance classification over that of the SOM. Although the approach was successful for a three vowel classification task, the restriction of SOMs to a single dimension significantly reduces the interconnectedness, which means that less information is revealed about how each phoneme is related to each other.

Another exception is Lee & Chang-Young (2011), who used one-dimensional torus maps for consonants, reporting that satisfactory results were not found for KNPT.

As for the research presented in this thesis, I will restrict attention to two-dimensional maps, given that the projection of multidimensional space by the SOM algorithm learns the local topology. The above studies notwithstanding, a great deal of KNPT research demonstrates that static two-dimensional representations have practicality for the visualisation of speech in the context of phonemes.

3.7 Different Languages

Initial work in KNPT was towards developing speech recognition for Finnish and Japanese (Kohonen et al., 1988). Further work from the research group using Finnish was reported in Kangas (1991); Torkkola (1990). Owing to those early efforts, KNPT has also been developed for a number of other languages. As I discuss in section 3.4.1, both Finnish and Japanese are languages with a strong grapheme-phoneme relationship, a factor that adds to the suitability of KNPT for speech recognition.

The attention of many researchers in the KNPT area was clearly towards the development of the algorithm and enhancing its understanding in general ways, towards learning phonemes, not specific languages. For example, Rigoll (1990a) investigated SOM as an input for HMM speech recognition, but I was unable to find the language in reading these papers. It was only through personal correspondence with him that Rigoll was able to let me know that he recalls that German was most likely the language of the speech used for applying information theory principles to the KNPT problem.

In the literature, KNPT researchers reported languages including Danish (Danielson, 1990), and Malay (Eng, 2006). KNPT has also found use in the related topic of phoneme recognition for Danish and British English (Dalsgaard, 1992). Dalsgaard used 80 seconds of speech from a single speaker from SAM EUROM0 speech corpus as well as a supplementary corpus recorded for Danish. More recently, Lee & Chang-Young (2011) applied KNPT to Korean consonants, though results were reported as not satisfactory so investigations turned away from KNPT and LVQ was conducted instead.

Finally, the English language is widely considered to be the lingua franca of the scientific community. As such, most published articles on KNPT are written in English, and it follows that a number of researchers have applied KNPT principles to the analysis of English speech. One of the main corpora used in the KNPT study of English has been TIMIT. TIMIT is a speech database collected from several hundred speakers (Zue et al., 1990). TIMIT has been widely used in the speech sciences. The TIMIT database is a well known speech corpus for speech recognition that has been manually annotated. Manually labelled datasets such as TIMIT lend themselves to supervised training, but some KNPT researchers have used the TIMIT database for unsupervised learning of SOMs.

Anderson (1991) used the TIMIT dataset to first perform unsupervised training of maps, followed by labelling that used the labels from the training data. Similarly, numerous researchers on SOM variants have used the TIMIT corpus, which has been manually annotated at the phoneme level. For example, Cooke et al. (1994) described unsupervised learning, but labelled using a portion of the TIMIT database. Arous & Ellouze (2003) used TIMIT for training, comparing several variations of SOM, some of which used unsupervised training and others, supervised (labelled data was used during training).

In the research of Arous & Ellouze and others, it seems that all maps were labelled using the labelled training data. SOM is often cited as unsupervised technique that requires only a small portion of the dataset to be labelled, but in practice, KNPT research seldom if ever uses this feature. In Chihi & Arous (2012), training was described as unsupervised they did not explicitly describe how labelling is performed. In Behi et al. (2012), the researchers describe their systems in terms of modules. This is an interesting conceptualisation; the association of labels with the unsupervised map may be considered as a module that follows the learning module that accepts the input from the pre-processor module (e.g. as MFCCs). In Salhi et al. (2013), despite recognising that the unsupervised classification SOM training allows for a wider range of corpora, the procedure used for labelling was again not explicitly addressed.

3.8 Normalisation of KNPT inputs

Normalisation is an important consideration for KNPT. In my review of the literature, I found that many researchers simply do not describe any normalisation, if any, that was performed. This is despite Kohonen's (1988) original KNPT specification, in which spectral features were demeaned and normalised to constant length, and his research that followed strongly suggests normalisation. Indeed, with sufficient dimensionality and large enough maps, an appropriate normalisation procedure would improve the convergence of a network. As described in (Kohonen, 1990, p. 1470), "Normalisation is not necessary in principle, but it may improve numerical accuracy because the resulting reference vectors then tend to have the same dynamic range." Additionally, some audio representations, including MFCCs, have different features with variances that are in roughly the same range. Thus, normalisation may not always be strictly necessary, but it merits consideration.

My own findings for utterance normalisation are that normalisation at the word level (normalising each feature to unit variance) is a good strategy, particularly for small maps. Particularly for the task of separating speech from silence, I found that different types of normalisation had a significant effect, particularly for small maps. These findings are presented in section 6 on voice activity detection.

3.8.1 Mean and variance normalisation (normalisation to unit variance)

A recent article by Kohonen (2013) gives us some insights into the normalisation procedures for Kohonen SOM. When we compare the feature vectors by distance measures, features with larger scalings will have more distance than those with smaller scalings, so normalisation so that they have either the same variances or they all have the same minimum and maximum. Normalisation to unit variance is used to improve results when using Euclidean distances between observations.

Standardisation is a type of normalisation that is often helpful to use in machine learning (Witten et al., 2011). To standardise an input, the mean and standard deviation are determined for each attribute. Thus, a vector of means and standard deviations are determined, one for each feature. This allows the vector of means to be subtracted from each input vector, then be scaled by the corresponding standard deviation. This procedure results in all the features having zero mean and unit variance.

Consider a set of attributes, each of which has a value determined by the set of observations. Here, a subscript indicates an observation, and a superscript indicates an attribute. If there are k attributes, then observation x_i is made up of these attributes, as follows.

$$x_i = a_i^1, ..., a_i^k \tag{3.6}$$

The entire set over which we are

$$a_i = \frac{v_j - mean(v_j)}{std(v_j)} \tag{3.7}$$

In a SOM, it may not be necessary to subtract out the averages of each feature if the subtraction is to be performed across the entire set if Euclidean distance is the similarity metric. This is because the distance between two points is not changed by first subtracting the same value from both points. Such a subtraction would simply shift the value for the feature across the entire map. In some KNPT systems, each vector of features for a frame may be normalised Kohonen (1990). When a multiplication or division is performed subsequent to mean subtraction, a shift cannot reverse the mean subtraction. Thus, it is important to consider where in the normalisation process that mean normalisation may occur.

Anderson (1991) performed mean subtraction followed by normalising vectors to unit length. It was argued that information may have been lost in the normalisation process, but that the effects of noise would be reduced in the process. The reason that information is lost is that the feature length is based on some aspect of the features. By setting all features to unit length, at the frame level, that information is not used in the argmax comparison.

There is another more practical reason to perform mean normalisation. When there are many attributes under consideration, it may simply be good practice to perform mean normalisation so that the values are easier for humans to make comparisons, even if it is not numerically necessary (Kohonen, 1995). In other words, mean normalisation requires extra computation steps, but if that makes it easier for a human to understand the compare the values visually, it may be warranted. However, for many audio representations, the features are already relatively comparable by eye. In such a case, it may be superfluous to perform mean normalisation. My observation is that normalisation to unit variance is the most standard type of normalisation in the KNPT literature.

3.8.2 Dot product normalisation

The Euclidean distance SOM is the most commonly used. Dot product SOM differs from Euclidean distance SOM in the distance measure that is used. Nearly all KNPT implementations used Euclidean distance. However, normalisation can make Euclidean distance SOM equivalent to dot product SOM. Dot product normalisation is also performed at the frame level.

$$c = argmax\{x \cdot m\} \tag{3.8}$$

Each codebook vector c_i in the codebook is calculated as follows.

$$c_i = argmax\{x_i m_i\} \tag{3.9}$$

As described in Kohonen (2014), normalisation to constant length makes Euclidean distance metric equivalent to the dot product, with the proviso that the codebook values must also be renormalised to constant length after every update. (Kohonen, 2014, pp. 41-42) stated, "If the vector dimensionality is high, and also the input vectors are normalized to constant length, the difference between SOMs based on the Euclidean distances and the dot products is insignificant." (Kohonen, 2014, p.40) also notes that, "the SOM Toolbox does not have provisions for computing the dot-product maps".

3.8.3 Range normalisation

Range normalisation means to change the ranges of features, or attributes, so that they are all on similar scales. A common style of range normalisation is to scale the features between 0 and 1. This can be accomplished by determining the maximum and minimum of each feature as follows.

First, for inputs $x_i \in X$, each x_i is an observation of each feature. If the X is represented as a matrix, where the observations are the rows, then the columns are the features. To determine the range of each feature, the minimum and maximum value for each feature is thus calculated as the minimum and maximum of each column.

$$\tilde{X} = \frac{X - min}{max - min} \tag{3.10}$$

$\tilde{x_i}$ is the normalised input vector. When all the input is scaled, and the codebook is learnt from the input, the result is that each codebook vector will be range normalised scaled as well.

Dalsgaard (1992) performed normalisation at the frame level to ensure that the values of cepstral features were from -1 to 1. Similarly, Eng (2006) performed range normalisation of cepstral coefficients. This procedure uses the minimum and maximum for each coefficient to scale the values into the range -1 and 1. Guenther & Gjaja (1996) applied normalisation to KNPT in a novel way using formants as inputs. Normalisation was used to create agonist-antogonist pairs. Using the minimum and maximum values of the formants created a positive (x^+) and a negative (x^-) value, so that a large value of the formant would result in a high x^+ and low x^-, whilst a small value for the formant would result in the opposite, a low x^+ and high x^-.

The above research notwithstanding, most KNPT researchers do not use range normalisation. In my opinion, this is due to the nature of speech data, which is generally already within

a fairly narrow range.

3.8.4 Leaky integrator normalisation

In a series of several experiments, Kangas (1991); Kangas et al. (1990) reported on the "leaky integrator", which considers sequences of neuron activations in a SOM over time. Leaky integration, without normalisation, was also explored by Chappell & Taylor (1993) and Reiss & Taylor (1991). Kangas introduced a method of normalisation for the leaky integrator. This approach to normalisation differs from feature scaling because it scales the output, or activation, of a node rather than the input. Although I did not test such an approach, it is a novel approach that could be an area of interest for future research.

3.8.5 Utterance level normalisation

In Warlaumont et al. (2010) infant vocalisations were classified. FFTs were binned to a 15 bin Bark scale to create power spectral density values, which were then normalised to the maximum power for the individual utterance. One second of data was thus converted into 225 values. A 4x4 SOM was trained on each of 225 value observations. As only one second of each utterance was used, this study was the only one out of all the KNPT studies that were reviewed to explicitly state that normalisation was being done at the utterance level. Although infant vocalisations are not clearly phonemes, I explore the normalisation at the utterance level for voice activity detection in section 6 with some positive results.

3.8.6 Avoiding normalisation

As I mentioned, there were quite a lot of studies that did not report normalisation. More studies omitted mention of anything about normalisation procedures than did mention it. One interpretation is that if normalisation is not mentioned, no normalisation was carried out. On the other hand, the researchers might have considered it to be such a basic step that it was not necessary to describe it, or the software they used may have automatically performed normalisation. In my experience, certain types of normalisation may be well suited to certain feature representations, so I feel that an investigation of normalisation procedures in the context of KNPT is merited.

There were a few isolated studies that specifically mentioned why normalisation was avoided. Lundberg (2005) claimed that it was possible to avoid pitch normalisation by restricting the corpus to only include male speakers with fundamental frequencies around 110 Hz. Additionally, data was selected from the centre of vowels (the median), which was the portion of the phoneme thought to suffer the least effects of coarticulation that occur from other phonemes

produced nearby in the speech stream. Thus, it may be possible to avoid a need to normalise by careful selection of the training data, though Lundberg's maps were the smallest in the KNPT literature.

3.9 Conclusion and Future Directions

Attention to the KNPT has persisted since its origins in the 80s. SOMs are still widely used in linguistics (Kohonen, 2013). The KNPT has been used in research for a number of speech representations, including filterbank channel energies, MFCCs, and formants. The information present in the speech signal at the phoneme level allows the KNPT to reveal aspects of the nature of speech through classification and clustering. Annotated speech is difficult (and expensive) to obtain, so unsupervised training allows using KNPT to classify larger speech datasets. Furthermore, with supervised labelling, the SOM can determine nearly optimally decision boundaries, as was shown in the early KNPT experiments using supervised SOM and LVQ-tuning.(Kohonen, 2014, p.160)

The use of KNPT has numerous advantages for visualisation and classification of speech and speech-like signals. The naive user does not always gain the advantages as might be expected, as using KNPT effectively might require consideration of parameterisation, map sizes, learning rates (for sequential SOM) or decreasing neighbourhood sizes, as well as aspects such as normalisation or selection of feature representation.

It is beyond the scope of this chapter to explore every development in SOM in other areas that could affect the KNPT systems. For example, Kohonen (2013) reported the non-negative least squares method for interpreting the output of SOMs, but we have not seen any KNPT research that uses this technique. Additionally, deep learning and neural modelling are hot topics in the research literature. Future research into KNPT will surely use such developments and more. The research presented in the literature seldom compares more than one feature representation. Work could be conducted to better understand how different parameterisation of feature representation algorithms such as SOMs could affect topological distribution of the SOM with respect to phonological classification.

Chapter 4

Implementation of Single Layer KNPT Systems

In this chapter, I report on aspects of automated speech analysis using KNPT using the speech from a single speaker. In particular, the overall goal is to explore what can be accomplished in terms of online speech analysis, with the proviso that although training may be undertaken offline as it requires a significant amount of time, the performance of the system for analysis (at test time) should be rapid and real-time. The presentation of the research I conducted and report on in this chapter moves towards an understanding of automatic speech analysis. The results of these investigations provide insights into factors of evaluation that contribute to a successful system.

I present an exploration of speech representations and evaluate their suitability for SOMs. In particular, I address two representations of speech: spectrograms and MFCCs. I transformed raw speech audio into these two speech representations and performed clustering to understand how the SOM will be affected by the different feature representations. I was able to explain different aspects of my voice with them, and understanding the applicability of these representations that transform the speech signal, to also develop an understanding of how the KNPT can be evaluated.

The goal here is to demonstrate with the speech of a single speaker some aspects of KNPT for English. The speech of a single speaker is called an ideolect. The speech of a single speaker is less varied than for comparing speech between speakers, because a speaker has a fairly constant physiology, such as bone structure and muscle size, and certain habits of speaking. These experiments provide context for the abilities of the system for one speaker. Extending it to a different system architecture will be the subject of subsequent chapters.

The basic information processing steps used in a KNPT system are depicted as a flowchart in Figure 4.1. Speech input is recorded using a microphone. This information is preprocessed using procedures including steps such as filtering out excess noise or framing the signal into

windows. Next, feature extraction is accomplished by an algorithm such as MFCC or LPCC. These features are presented to the classifier, which in a KNPT system includes some variant of the SOM algorithm, but may also include other steps such as LVQ tuning, or mix other machine learning approaches, such as a MLP, for improving classification. Finally, the system outputs a stream of phonemes. The alignment of the phonemes may correspond to the frames in the preprocessing stage.

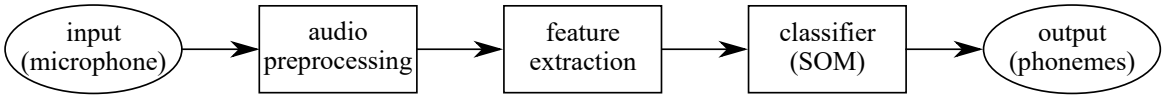


Figure 4.1: KNPT flowchart

4.1 Single speaker datasets

For the experiments presented in the present chapter, with the purpose of exploring the capabilities of KNPT for automatic speech analysis with attention to those aspects that were not reported in the literature, I created several collections of different types of speech data. Specific details about each of the datasets that were created will be presented in the corresponding sections of this chapter (sections 4.4, 4.6 and 4.7). Here I will describe the processes that went into producing or selecting each speech dataset. Speech was produced by a single male speaker (the author) and was recorded using a Microsoft Kinect microphone array. This goal for using this type of dataset was to improve understanding of the factors that contribute to gaining an understanding of within-speaker differences. A main goal of this work is to identify methods of determining useful representations, and the speech data would allow this.

With an end goal towards the creation of a tonotopic map of speech, single speaker recordings were made over the vowel inventory of English as determined by the IPA inventory for English. The vowel and dipthongs were spoken for three seconds each, with an emphasis on making smooth transitions between the different sounds or phones. This intentionally slow manner of speech was designed to allow closer inspection of the steady states of vowels. Recordings were made in a quiet room, although it was not soundproofed. Extraneous noise from the speaker were avoided, and particularly, mouse clicks were not used (the clicks would be captured by the microphone). Interactions were made by using a touch screen during non-speech segments.

Using the microphone array in the Microsoft Kinect, I first recorded /hVt/ words at the four corners of the vowel quadrilateral. This dataset is discussed in greater detail in section 4.4, along with an analysis of these vowels through spectrograms and a single-layer KNPT system.

The next data set that was recorded was a wider inventory of vowel sounds in the /hVd/ context. The idea of vowels in this context may include the “r” liquid, which is considered by

many to differ from a vowel. Linguist Pike (1947) introduced the term *vocoids*, which includes these sounds of speech as well as vowels that do not have friction on the airflow in the mouth. Although I did perform experiments with r-coloured vowels, they are not reported in this thesis because Australian English is one of the main foci. Australian English is non-rhotic and therefore does not really have r-coloured vowels. Therefore, I will not include those experiments that involve the r-coloured vowels of California English.

It was then determined that to control for coarticulation effects, I would then record more three-second vowels. These recordings were towards generating a complete (for the accent of the single speaker) inventory of vowel sounds with no consonants. Four tokens of each of the 21 vowels in the vowel inventory was recorded in a session. For this corpus, four sessions were recorded at standard pitch, and then four further sessions were recorded with the speaker using a slightly higher pitched voice. Another condition with the voice at a lower pitch was attempted, but it was too uncomfortable for the speaker so was not completed.

Each session was recorded on a different day. This dataset therefore consists of 2 conditions (pitch) x 4 sessions x 21 vowels x 4 tokens x 3 seconds, for a total of 2016 seconds of vowels. These vowels were then annotated for phoneme boundaries, which indicate the start and stop of each vowel to aid subsequent analysis. I performed annotation in PRAAT. To capture the wider variations in speech, recordings were conducted over several sessions. The speech of a single speaker was recorded within the span of a few weeks. The complete vowel inventory was recorded at a time in each session, with a randomised sequence each time to control for order effects.

4.2 Evaluation metrics

In conducting the research review in Chapter 3, I identified a number of measures that have been used for the evaluation of KNPT systems. Different studies with their disparate goals require ways to determine the success (or failure) of the system to perform its task adequately given the training data. I also touched on measures to reduce the effects of bias when reporting statistics in section 2.7.

In these efforts presented in this work towards a system for the evaluation of pronunciation, consideration is given to the methods for measuring how well our system is performing. To perform this effectively requires a consideration of the nature of the task. The ability to perform classification or evaluation is significant in understanding the utility of the system for the task. In this case, I am seeking to develop a system that allows for and supports an identification of unexpected speech productions. To do this effectively requires the classification of phonemes, so that the phonemes themselves are recognised in the speech stream. To recognise a phoneme that is not pronounced as expected may necessitate an ability to detect an anomaly, a pronunciation

heretofore unobserved. Alternately, a target phoneme may be pronounced more like another phoneme that has already been observed.

At this point I may clarify that an eventual goal of this research is towards measuring deviation from the target and displaying phonemes in the context of a fixed phoneme inventory. Thus, rather than to seek a characterisation in terms of an anomaly, I aim to show how an anomaly matches what is known to be true of phonemes in our target language. This is similar to human learning of language: prior to the age of six months, an infant will recognise the sounds of all languages; however, after that age, their perceptions are coloured by their experience of language. Thus, I shall operate according to a principle of following the examples of human perception: our maps have learnt the prototypes of the target language. This is the basis on which the classification and evaluation of phonemes is to be conducted.

4.2.1 Gold standard comparison

Phonemes have certain characteristics that may lead us to conclude that they are inherently overlapping categories. Nonetheless, by definition, the phoneme is the unit that serves to differentiate one word from another. In a supervised paradigm, in which features such as phoneme labels and start points are known (through mechanisms such as human annotation), we may compare the classifications produced by the system to the known features. In other words, as the targets for identification of phonemes become known, we can compare them to the phonemes that are generated by our mechanism of classification.

Given a file that has phoneme annotations, including the target phoneme as well as the start and end time, we can compare the output of our system to the annotations. The frame error rate (FER) compares the output of the classifier with the manual annotations at corresponding time intervals. For example, let us consider ten frames from the word “had” (pronounced /hæd/). I trained a SOM on the four vowel inventory of Dataset 1. The speech was manually segmented to include only the vowel portion. Consider a 150 ms time interval with a frame offset of 10 ms; in this interval, there are 15 overlapping frames. Thus, the target vowel æis repeated 15 times to match the output of the classifier. As an illustrative example, the results I received from the SOM for one section of the phoneme are as follows:

- Target: æææææææææææææææ
- Output: æææææææææɛɛæææææɛæ

By comparing the target and the output, it is evident that of the fifteen frames, there are three times that /ɛ/ appears. The number of errors can be determined between target and output. To enable comparison with other time intervals, the number of errors is divided by the number of frames. For this example, the frame error rate (FER) was as follows.

$$FER = \frac{3 \text{ errors}}{15 \text{ frames}} = 0.2 \tag{4.1}$$

Another metric that can be used is the phoneme error rate (PER). For the example given previously, it is known that the target phoneme is /æ/. To determine a prediction for the phoneme, we can simply select the most common output as the most representative phoneme. More sophisticated methods can take into account the connectedness of the output, the interdependence of one frame on the neighbouring frames (e.g. considering each frame output as a state transition in a HMM), or the predicted effects of coarticulation. Assuming that the 15 frames was the entire phoneme, using PER, the most common output method, is determined as follows:

- Target: æ
- Output: æ

$$PER = \frac{0 \text{ errors}}{1 \text{ phoneme}} = 0.0 \tag{4.2}$$

In contrast, if more frames were classified:

- Target: æ
- Output: ɛ

In which case:

$$PER = \frac{1 \text{ error}}{1 \text{ phoneme}} = 1.0 \tag{4.3}$$

As the system processes more phoneme identifications, the average PER is adjusted accordingly.

4.2.2 Partially labelled data

The metrics of FER and PER as described can also be used to evaluate if only some of the data is labelled. For instance, the system can be trained on unlabelled data, labelled with a portion of the labelled data, then evaluated with unseen labelled data. Since not all the data is labelled, determining error rates on speech files must be adjusted accordingly.

The labels we have described to this point include the target phoneme as well as the start and end time. But what can we do for audio files that contain known utterances, but (as they are not

manually annotated) the start and end of phonemes are not known? For example, a recording may have been made of a speaker who was prompted to say the word "had" and nothing else. For example, if there is a recording that was prompted by the phonemes /(...)hæd(...)/[1], the file should contain that word, but it is still not known where those phonemes are to have started and ended.

In such a case, since only one phoneme in the file is relevant (the vowel), the PER for the vowel can be determined by simply taking as the prediction that vowel phoneme that was produced most frequently for the file. If the files contain a great deal of non-vowel information, and the classifier incorrectly labels a significant portion of the non-vowel as a vowel, this will bias the results. However, I found that the SOM as trained was fairly accurate in identifying periods of silence and non-vowel uniquely from the vowels. For this dataset, the errors were generally produced as a substitution of other vowels.

4.2.3 Unsupervised training

One aspect of SOMs is that a category can be associated with multiple neurons. The common approach to labelling categories simply selects the BMUs, which means that if data from a number of different phonemes best match a certain neuron, selecting only one category means the neuron is associated with the most common matching phoneme at the expense of other matching phonemes. Although the common approach is the simplest, in that it provides an unambiguous category, it is also possible to retain information about the multiple categories that are associated with a neuron. An approach I have considered is to ignore as insignificant the data from phonemes that represent less than 5% of the input to a neuron. Adopting a minimum threshold such as 34% will ensure that a maximum of two categories will be associated with each neuron. Adding multiple categories complicates the inter-category and intra-category measures, but may provide additional valuable information about the nature of the observations.

In order to compare the different feature representations of the speech signal, namely to compare spectrogram with MFCC, for the task of phoneme analysis with KNPT means that the aim is to understand the quality of clustering for each feature space. In constructing the SOMs, there is also an interest in finding out how the cluster quality may vary as a function of the size of the SOM, so each experiment can be run multiple times, each with a different size of SOM.

4.2.4 Manual inspection

The final metric that I will introduce for evaluation of the output of a KNPT system is of a manual nature. It is envisioned that a KNPT system will provide a characterisation of phonemes

[1]The silence around the word is indicated as a pause in speech, /(...)/.

that is useful for a task such as phonemic analysis. Rather than evaluating the performance of the system, the output could be evaluated for how clearly it shows the results. In other words, although quantitative metrics are useful, qualitative metrics that show the utility of the outputs may also be desirable.

The general form of this qualitative metric is how well the output can fulfil the envisioned concept of displaying a visual form of phonemes in a way that can be understood by the user. This could take a form similar to the vowel quadrangle (shown in Figure 2.1), but the way that the classification scheme has been implemented using SOM is not specifically constrained by the formants, and as such, may be subject to a different interpretation.

4.3 Initial exploration with corner vowels

Due to a number of factors, including accent variability, it is difficult to define a standard transcription system (Ladefoged & Johnson, 2010, p. 69), "there is no one right way of transcribing even a single accent of English." In an attempt to enable increased consistency of transcription, the cardinal vowels are a reference system used by phonologists. They come, not from any specific language, but instead as specific sounds that serve as landmarks for those interested in phonology, such as linguists. The cardinal vowels are produced with the tongue at its extreme positions (as used in speaking).

In this section, I will introduce an analysis of four vowels produced by a single speaker that have maximal separation on the cardinal vowel chart. Three of these cardinal vowels (/iː/, /ʌ/, and /ɑ/) are referred to as the corner vowels. Along with the fourth vowel, /æ/, these four vowels provide a reference that helps to show some of the properties of vowels (Ladefoged & Johnson, 2010).

With reference to IPA charts for American vowels I prepared Figure 4.2 for these four vowels. An IPA chart like this is common in linguistics and phonology, and teachers of English as a foreign language show them to students to help when explaining how the different vowels are produced. Vowels on the left and right sides of the quadrilateral are considered to be front and back, respectively. Similarly, those vowels that appear at the top and bottom of the quadrilateral are high and low vowels, respectively.

The terms front and back, and high and low, refer to the auditory qualities of a vowel. Generally, this quality is affected by tongue position. When a vowel is generally produced when the tongue is at a certain position in the mouth, it is considered to be that type of vowel. For example, a vowel such as /iː/ as in "heed" is generally produced with the tongue in *high front* position. A front vowel is an indication that the top of the tongue is closer to the front of the mouth than the back. These tongue positions are not precise locations but rather are relative places. The terms are indicators of the sounds of speech, not the positions

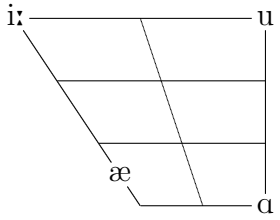


Figure 4.2: Four American vowels placed on the vowel quadrilateral

of the tongue. Traditionally, phoneticians used these terms (and others, such as rounded or unrounded lip shape) to describe the mouth positions, but in actuality, the positions only very roughly indicate the positions of the articulators in the production of vowels (Ladefoged & Johnson, 2010).

The sounds of speech are produced with rapid positions and movements that affect the airstream. One of the reasons that many students of new languages (or accents) find it difficult to pronounce words in a new way is that there are different ways to distinguish the sounds. For example, Japanese only has five vowels. When Japanese speakers are learning to speak English, they are learning many more distinctions than they have needed in their mother tongue. The idea behind using a limited vowel inventory such as the corner vowels shown above is that some insight will be gained, though it is not necessarily true that a method that is able to distinguish easy to separate categories will also be useful for larger vowel inventories.

4.3.1 Four vowel dataset

I created a dataset of four vowels. The four vowels used in this dataset are representative of the four corners of the vowel quadrilateral shown in Figure 4.2. This dataset was created to allow an exploration of a smaller number of vowels that we expect have maximum separability because of the way they are distributed in the vowel formant quadrilateral.

The four vowels were produced in an /hVt/ context. The words were “heat”, “hat”, “hot”, and “hoot”. Each vowel was produced for approximately three seconds in duration. There were eight recordings of each word for a total of 32 audio recordings, each containing four repetitions of the vowel. For the four vowels data set, there are thus 32 repetitions of each vowel and there are 4 vowels, for a total of 128 three second long vowel utterances separated by about a second each.

The vowel audio files were resampled to 16 kHz and divided into 10 ms overlapping windows (100 per second), so there were 256 vowels for three seconds each plus approximately four seconds of silence per file. This means about 76,800 windows of vowels and 25,600 windows of silence. Each window was converted to 39-feature MFCC format, with 12 cepstral coefficients, sum of the log energies, as well as delta and delta-delta features. The MFCC parameters are

consistent with those used by many researchers (for example, they are equivalent to those that would be produced with the default values produced by HTK Toolkit (Young et al., 2013)).

During the recording process, noises such as mouse clicks were kept minimal; the only noise from interacting with the recording interface was made by a finger on a touch sensitive screen at the start and end of a sequence of producing four /hVt/ context vowel tokens of three seconds each. The noise of a finger on the touch screen was very quiet (much quieter than a mouse click), and this interaction did not occur at the same time as the vowels were being produced. The data set was created with an intention to increase the differentiability between speech and silence using regular starting and stopping points. However, the time at which the vowel starts and stops is still only known with a certain degree of imprecision, because inherent to the nature of the task of saying a vowel, the exact starting point depends on human factors that determine the exact instant that the utterance will begin to be produced. What this means is that although each vowel started close to a certain moment, it is expected that each start and stop point of a vowel is slightly different. To mitigate this effect, the method we have used to capture the essence of the vowel without silence is to trim off the ends.

Stratified analysis was used, with $\frac{7}{8}$ of the files used for training; $\frac{1}{8}$, for testing. Stratification ensured that the same proportion of each vowel was used for training and testing. A number of different combinations were tried to explore the capabilities and parameterisations of the technique. For the results presented here, four SOMs were trained. The size of the map was varied at four steps from 3x3 to 9x9. These four sizes will be used to explain the effects of different levels of resolution for unsupervised SOM. The trial-and-error method for visualisation using SOM is a general technique, as described in Kohonen (2014). Thus, I will present an analysis based on the trial-error method.

A primary consideration that will be revealed through these four maps is how unsupervised KNPT with supervised labelling has two different aspects of resolution on the same map: (1) the density of the training data; and (2) how well the data used for labelling aligns with the trained map. As far as I am aware, this specific concept has not been discussed in the KNPT literature.

4.4 Spectrogram Analysis of Corner Vowels

Developed in the 40s, spectrograms are one of the oldest automatically generated speech representations used for linguistic analysis. I reviewed speech spectrograms in section 2.6.2. As Zue & Cole (1979) demonstrated, it is possible for humans to read speech spectrograms. Arising out of that work, efforts were made towards ASR using spectrograms and associated representations of speech (Zue, 1985). Indeed, the initial feature representation in Kohonen (1988) was basically spectrogram with logarithmic binning reducing the number of features to 15.

Through spectrograms, I will demonstrate that there are clear differences between each vowel, but that there are also differences between each different utterance. For my initial tests with SOMs, I used the spectrogram as the input representation. A spectrogram is a common tool used for speech analysis, produced by windowing the data and applying a FFT to each frame. Spectrograms for the corner vowels are shown in Figures 4.3 to 4.6. 32 tokens of four different vowels were created. Each word was recorded with at least three seconds per vowel. As the emphasis of the research is vowels, the speech file was trimmed so that only the vowel remains. In Figure 4.3, the vowel /æ/ in the word "hat" is represented. There are clear similarities for

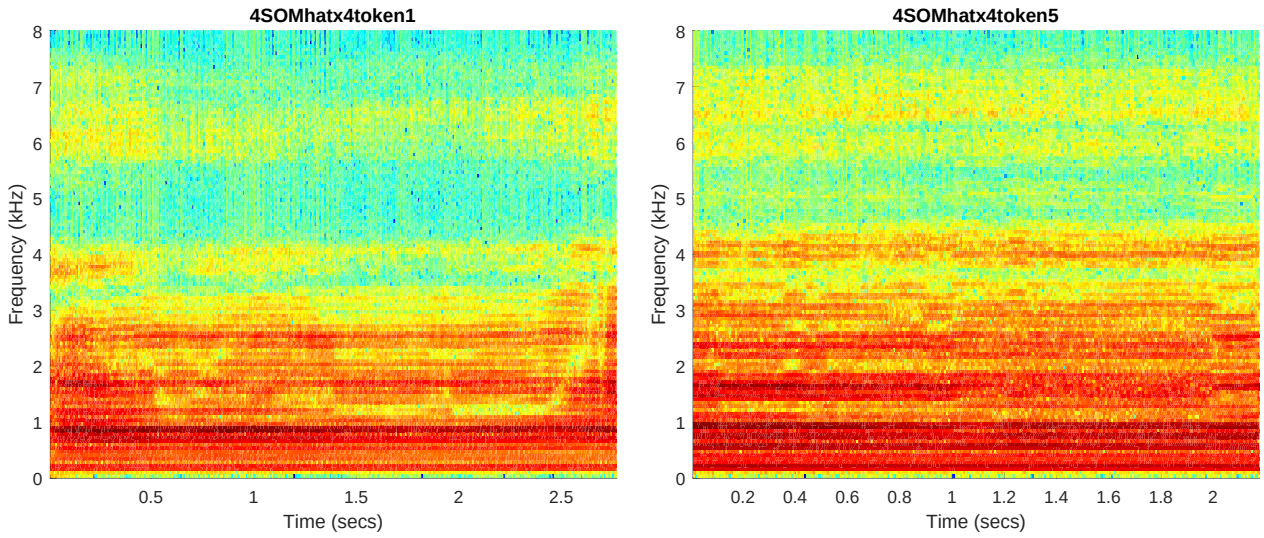


Figure 4.3: Spectrograms for two utterances of vowel /æ/ in the word "hat".

the two spectrograms. In particular, the energy levels are high between 500 up to 3000 Hz in both. However, although the same person spoke both tokens, it can be seen on the spectrogram on the left that there is not much energy between 3000 and 4000 Hz, but in the spectrogram on the right this band has quite a bit of energy. Next, in Figure 4.4, in which the vowel /iː/ in the word "heat" is represented, there is quite a bit of similarity between the two utterances. Energy is high between 2000 to around 3700 Hz and low elsewhere. In Figure 4.5, energy levels are high between 500 and 1500 Hz. There are additionally some higher formants beginning around 2500 Hz. Finally, in Figure 4.6, the vowel /ɑ/ in the word "hot" is represented. Energy levels are high between 500 to 1300 Hz, as well as some higher formants between 2500 to 3000 Hz (and above). The spectrogram on the right shows more variability in the sound. Increased variability in the vowels may indicate that the speaker was more tired or needed to drink more water. In all these figures, there is a bar of high energy between around 50 to 500 Hz. This voice bar is an indication that a voiced phoneme was produced. In this case, all the phonemes were vowels, so they all have this voice bar throughout. Using the spectrograms of eight tokens each of the four vowels from the words "hat", "heat", "hot" and "hoot" as input (similar to those shown in the spectrograms), I trained a SOM. As I will show, the resulting map had learnt to distinguish these four vowels.

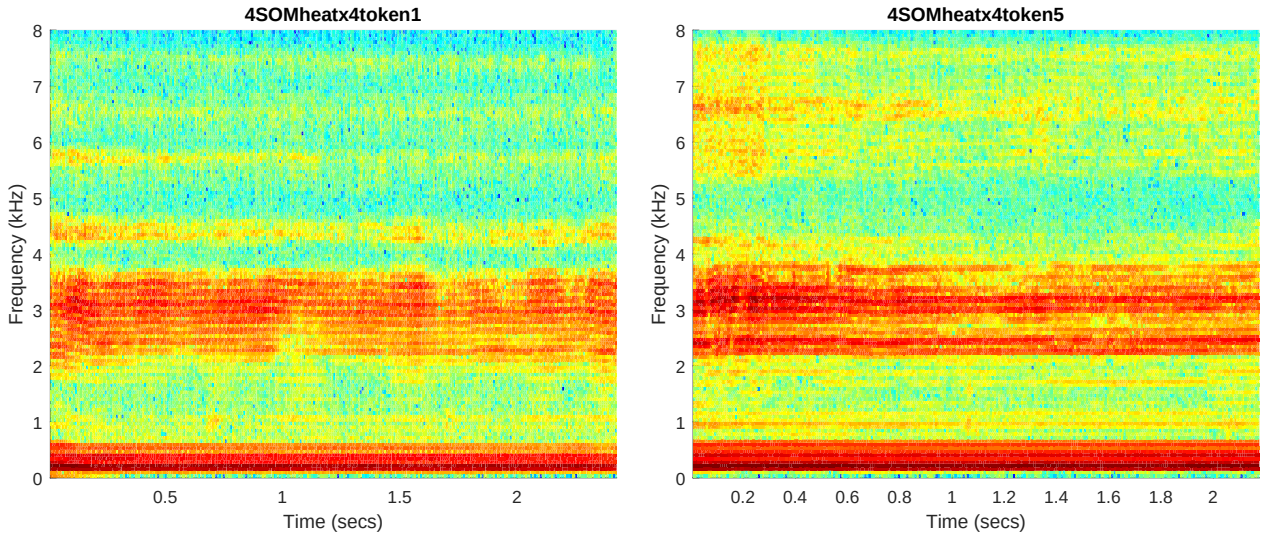


Figure 4.4: Spectrograms for two utterances of vowel /iː/ in the word “heat”.

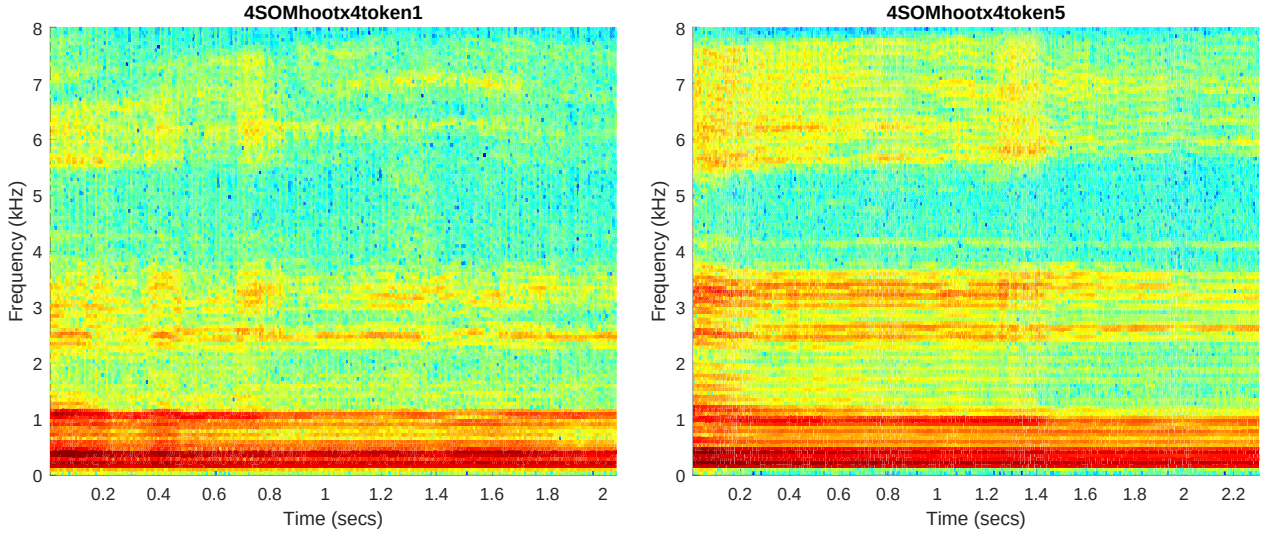


Figure 4.5: Spectrograms for two utterances of vowel /ʉ/ in the word “hoot”.

In Figure 4.7, the activations of the trained SOM are shown for the phoneme /æ/. The 10x10 unlabelled SOM shows the visited nodes. The size of the blue circles is in proportion to the number of visits, with larger circles indicating more hits. Also, it was possible to see that not just one cell responded to the sequence of frames that made up a vowel, but a trajectory.

4.4.1 Methods

MATLAB Listing 4.1 shows how the data is loaded and stored for training and visualisation. First, the data is loaded using my custom data class, which reads the file names from the data folder, loads the wav files from disk, then converts each to spectrogram format using procedures as in MATLAB Listing B.3. The data is split into training and visualisation files. Only the first of each set of eight files was manually annotated. These labels are loaded from PRAAT

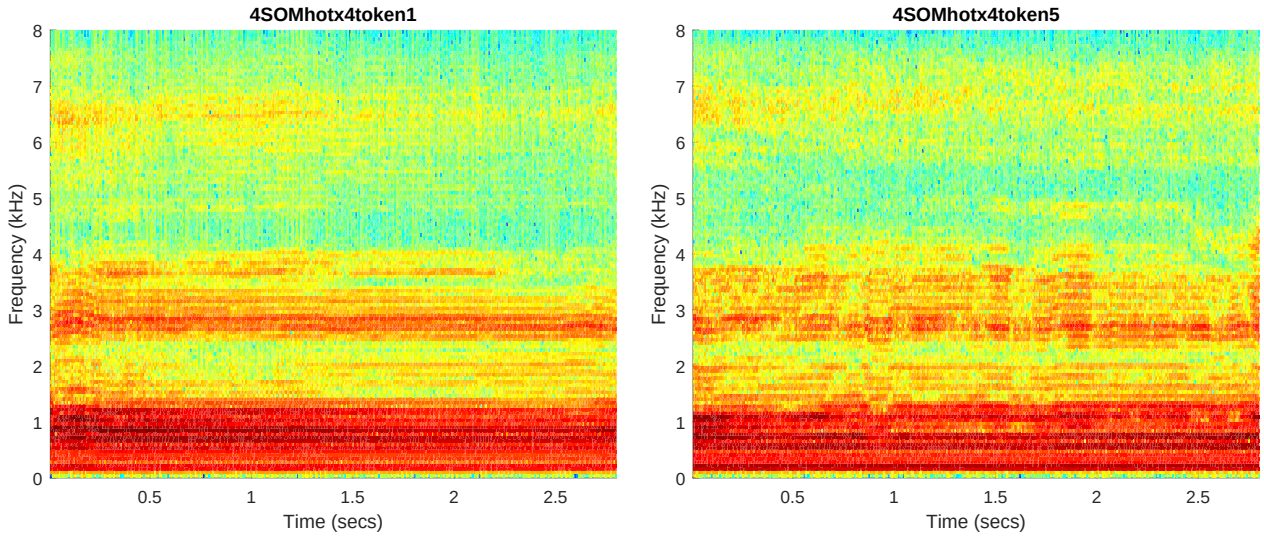


Figure 4.6: Spectrograms for two utterances of vowel /ɑ/ in the word "hot".

TextGrids and stored in a ta_praat_textgrid object (Details shown in MATLAB Listing B.4).

Once the data has been loaded, it is used to initialise a map. PCA The som_lininit routine uses MATLAB's PCA implementation to determine the two principal components that account for the most variance in the data. The resulting map is named init. Rough training is commenced starting with a neighbourhood radius of a quarter of the width of the map and ending at two. Fine training was then conducted with neighbourhood radius starting at two and ending at 0.5. The train length was 20 for both rough and fine training. After Kohonen (2014) published his book on using the SOM Toolkit, I revisited these parameterisations. The values were found to be suitable for the purposes of the analysis I have presented. They are shown in MATLAB Listing 4.3. Next, the visualisation of unseen data was produced using the resulting maps. The maps were rotated 180 degrees so that they would more closely align

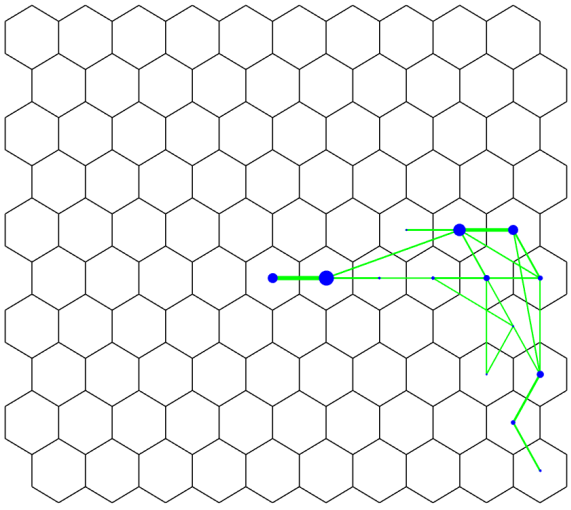


Figure 4.7: Trajectory of the vowel /æ/ in the word "hat". The SOM was trained on spectrograms of the corner vowels.

Listing 4.1: Loading data

```
%% load data
% load the data from file, including mfcc39 format data
complete_ta_data = ta_data_class('dirpath', data_folder, 'mfcc39', true);

% set aside 1/8 of the files for visualisation
train_indices = true(size(complete_ta_data.json_metadata{:}.prompt));
train_indices([1, 9, 17, 25]) = 0;
visualise_indices = false(size(complete_ta_data.json_metadata{:}.prompt));
visualise_indices([1, 9, 17, 25]) = 1;
```

Listing 4.2: Labelling maps

```
%% divide and label data
train_ta_data = complete_ta_data.ta_clone('mask', train_indices, '
    single_speaker', true);
visualise_ta_data = complete_ta_data.ta_clone('mask', visualise_indices, '
    single_speaker', true);

% load praat textgrids that were manually labelled
visualise_ta_data.ta_get_TextGrids()
visualise_ta_data.ta_praat_textgrid_to_mfcc39_label('sampa_to_ipa', false)

% get training and test data into som data structures
train_sData = train_ta_data.ta_get_som_data_struct();
visualise_sData = visualise_ta_data.ta_get_som_data_struct('label', true);
```

Listing 4.3: Training

```
%% train
init_sMap = som_lininit(train_sData, 'msize', [somWidth, 18]);
init_sMap.name = 'init';

% rough training
rough_sMap = som_batchtrain(init_sMap, train_sData, 'radius', [5, 2], '
    trainlen', 20, 'tracking', 1);

% fine training
fine_sMap = som_batchtrain(rough_sMap, train_sData, 'radius', [2, 0.5], '
    trainlen', 20, 'tracking', 0);
fine_sMap.name = 'fine';
```

with the IPA cardinal vowel map above. The rotation does not change the codebook data, only the cells in which it is stored. The distances between the old cells and the new cells are precisely the same. Note that 90 degree rotations or flips would not be possible with this map because of the offset. For flips, it could be suggested to use a symmetric map size with rows of different lengths. As it was not necessary to perform such a procedure to reach a satisfactory visualisation, mechanisms for improving the comparison of dissimilar maps are left for future research.

The map is visualised with the background colour of each cell a shade of grey in proportion to the number of hits received by that cell. Each individual phoneme is drawn on the map in a hexagon of proportional size to the number of hits for each. Procedures for the visualisation of the map are shown in MATLAB Listing 4.4.

Listing 4.4: Visualisation of the map

```
hits = som_hits(sMap, train_sData);
hits_temp = hits + 0.5 * max(hits);
hits_temp = hits_temp/max(hits_temp); % scale between 0.5 and 1.0 so not too dark
grey = repmat(hits_temp, 1, 3);
colormap_temp = sort(grey);
sMap = ta_rotate180(fine_sMap);
h = som_show(sMap, 'color', grey, 'bar', 'vert', 'footnote', '', 'colormap', colormap_temp);
h.label.String = '';
tick_labels = sort(hits);
labelled_hits = cell2mat(arrayfun(@(word) som_hits(sMap, ta_data_struct.(word{:}).ta_get_som_data_struct('label', true))', words, 'uni', false))';
som_show_add('hit', labelled_hits, 'MarkerColor', [[1 0 0]; [0 0 1]; [0 1 1]; [1 0 1]], 'MarkerSize', 0.5);
labelled_sMap = som_autolabel(sMap, visualise_sData, 'ta_vote');
```

The manual annotations provide a more precise value for start and stop times of the various phonemes. To show only the vowel portion, it is sufficient to perform a string comparison to select only the frames that were for a given phoneme, as follows in MATLAB Listing 4.5.

Listing 4.5: Subselect for vowels.

```
vowel_portion = temp_sData.data(strcmp(temp_sData.labels, phoneme),:);
```

4.5 Spectrogram-based KNPT

One way to analyse a KNPT map is to plot the speech on it. Visualisation of the trajectories allows us to get an intuition for the vowels. In later sections of this thesis, larger vowel inventories are used, which precludes an in-depth analysis of each of the maps. Here, in contrast, only four vowels were used (/æ/, /iː/, /ɑ/, and /u/). This enables an inspection of the trajectories of a representative portion of the vowel tokens. Two exemplars for each vowel will be shown to get an understanding of a single-layer KNPT system. For these purposes the vowels were truncated so that the beginning and end were within the voiced section of the utterance, thus the /h/ and the /t/ portions were removed from this analysis.

In Figure 4.8, two trajectories are depicted. The utterances selected for visualisation were the first and the fifth, and they were produced on different days. The same map, which was

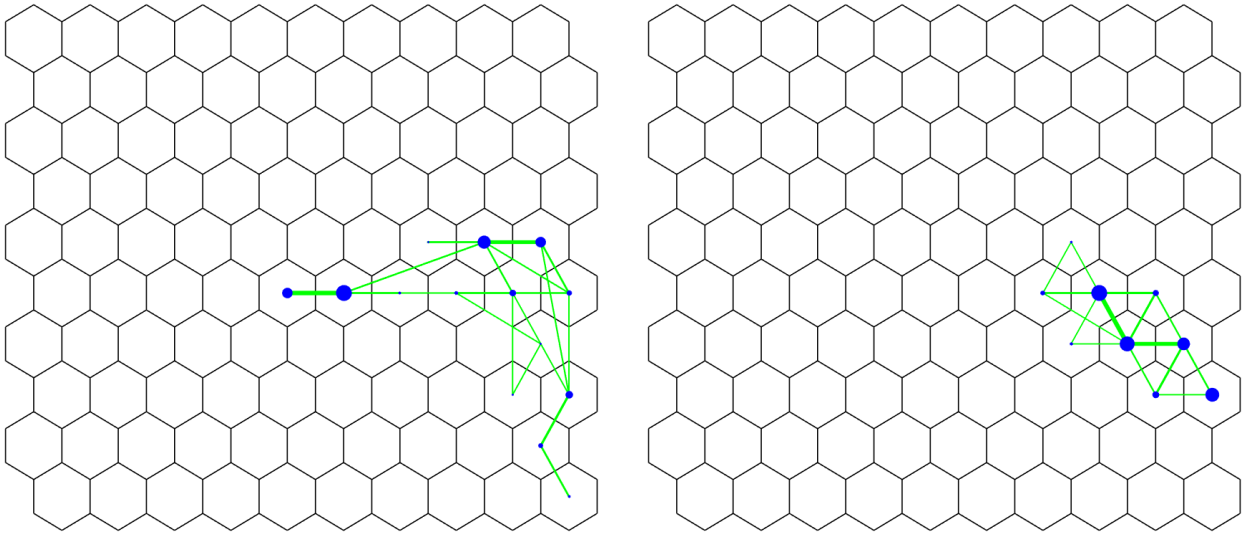


Figure 4.8: Trajectories for two utterances of vowel /æ/ in “hat”.

trained on the four vowels, is shown for each. An extended vowel phonation of about three seconds is transformed into a sequence of approximately 300 frames. Lines are drawn on the map to indicate where transitions occur. A transition occurs when the best matches for subsequent frames are in different cells. The size of the blue circles is in proportion to the number of hits for corresponding cells. On inspection, it can be observed that different cells are favoured for each of the two utterances. However, the trajectory is restricted to the same area of the map (the right middle).

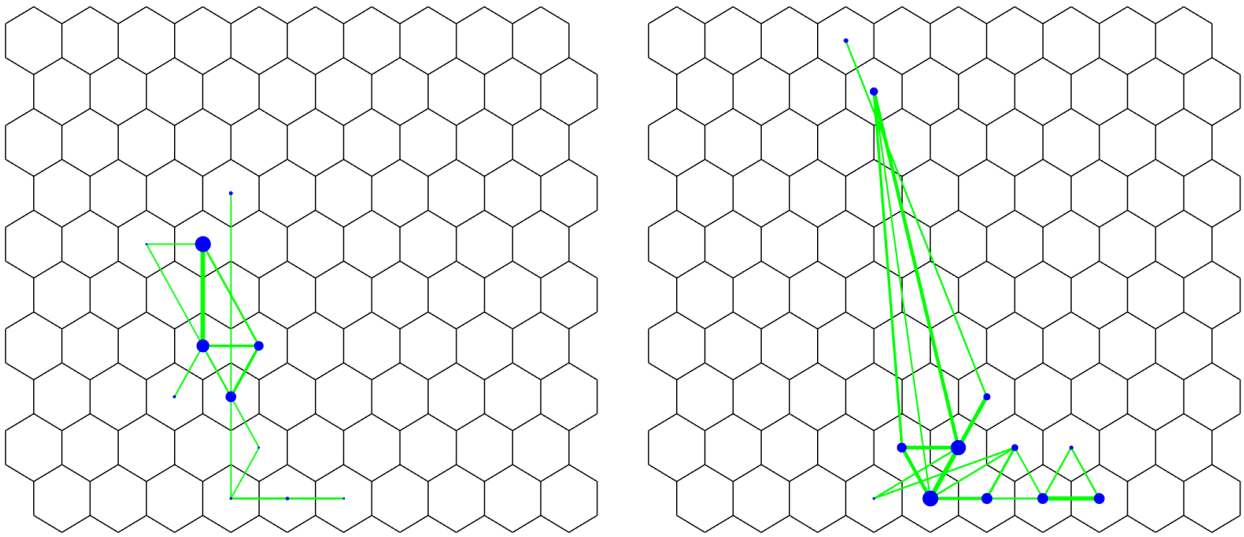


Figure 4.9: Trajectories for two utterances of vowel /iː/ as in “heat”.

Next, in Figure 4.9, trajectories for the vowel /iː/ as in the word “heat” are shown. As with Figure 4.4, the first and the fifth utterances were visualised. The map shown here is the same map for all these figures. Again, it can be observed that different cells are favoured for each of the two utterances. However, unlike the trajectories for /æ/, which centred around the right

middle, the BMUs for /iː/ appear in the bottom middle. A few frames in the figure on the right matched cells near the top middle. As the corresponding blue circles are small, we can tell that this classification was not made for a significant portion of the vowel.

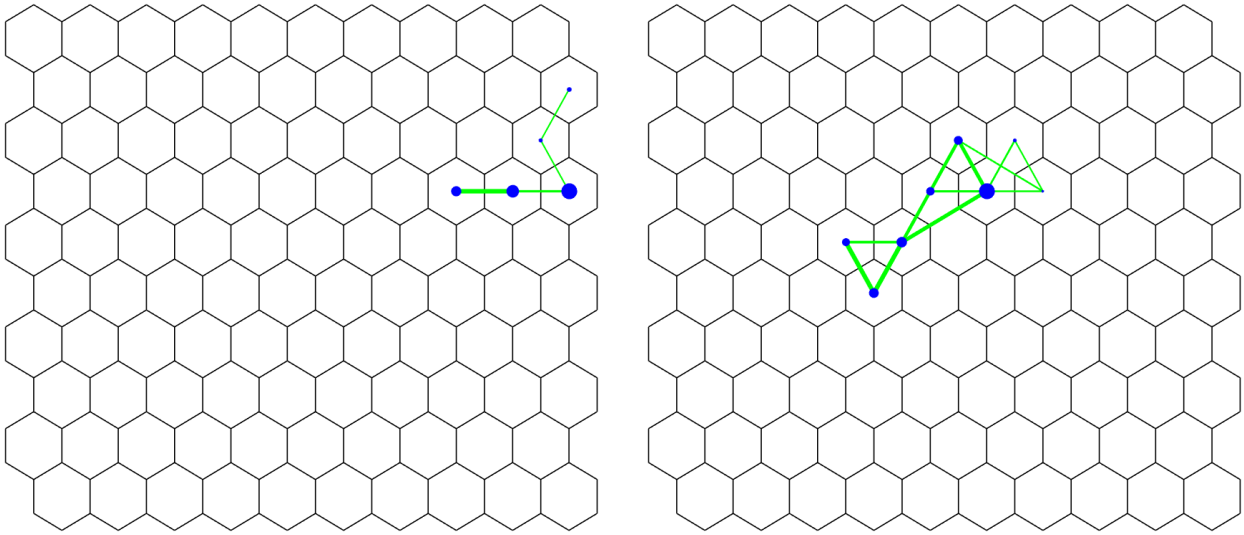


Figure 4.10: Trajectories for two utterances of vowel /ɑ/ as in "hot".

Next, in Figure 4.10, trajectories for the vowel /ɑ/ as in the word "hot" are shown. The same speech files were used to generate the spectrogram in Figure 4.6. The same map was used to generate Figures 4.8 to 4.6. In this case, the resulting visualisations for the two utterances seem to be more different than in the previous visualisations. The /ɑ/ shown on the left resulted in a trajectory that was near the top right. The trajectory of the /ɑ/ shown on the right resulted in the middle of the map. Without the assistance of labels, it might be difficult to know that the /ɑ/ shown on the right is different than the the /iː/ shown on the right of Figure 4.4. One reason for this is that the distances between cells on the map are not an indication of the distances in the original space.

Finally, in Figure 4.11, trajectories for the vowel /u/ as in the word "hoot" are depicted. The same procedures and map as in Figures 4.8 to 4.10 were used. For these two utterances, the visualisations show more similarity than the previous vowels showed. The trajectories of the /u/ both appeared in the top left corner of the map. As I discussed in 3.4.2, Leinonen et al. (1991) used the size of the jumps in KNPT maps to identify certain characteristics of the voice, such as hoarseness. I would argue that for the comparison of different vowels, some indication of the variability of the vowels may be revealed by a similar idea. For example, it seems that the sound /u/ is less variable than /iː/. I hypothesise that this might be due to different mouth characteristics of different vowels, with certain mouth positions causing increases in the turbulence of the air.

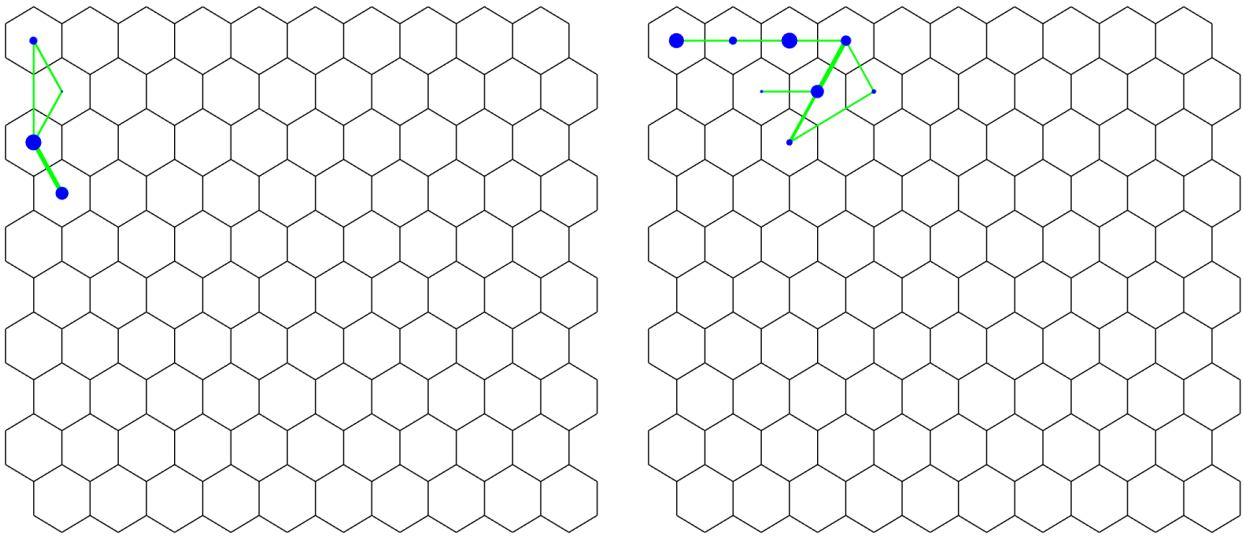


Figure 4.11: Trajectories for two utterances of vowel /u/ as in “hoot”.

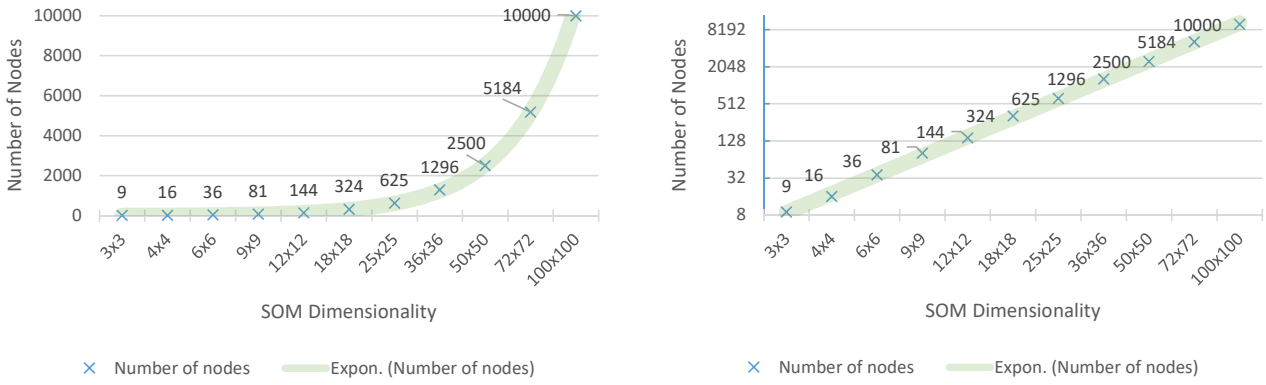
4.5.1 Map sizes

As noted in Kohonen (2014), there are two main methods for selecting the size of the map. Firstly, it may be set dependent on the data, e.g. a rule-of-thumb estimate of 50 data points per map cell may be used, which relates to finding a statistically significant number of datapoints per cell. Secondly, the size of the map may be determined by trial-and-error. Although a larger map gives more resolution, the larger map has less data points per cell, which means it has less statistical significance. For the experiments to follow, where the size of the map is to be found through trial and error, I have developed a sequence of map widths.

$$6, 9, 12, 18, 25, 36, 50, 72, 100, 144, 200, \ldots \tag{4.4}$$

The reasoning behind using these widths is as follows. To test the effects of different map size, a series of map sizes were determined with the goal that each step in the series would approximately double the size of the map. The number of nodes in each dimension must be an integer, but the increases in width for a doubling of area are related by an irrational number, $\sqrt{2}$. This means that there is no series of integers that can exactly match this criterion of area doubling. Furthermore, the goal is to perform a grid search to identify approximate sizes, not to fit the exact doubling. I additionally aimed to use a sequence of numbers that were aesthetic, i.e. the values were pleasant to look at with few primes and many perfect squares or near squares. Although my initial explorations used a width of 20, a much better series (by the criteria above) was determined as follows. Anecdotally, these values are also much easier to remember than odd numbers such as might be selected with a starting point of 20 or more strictly adhere to the doubling.

A single cell, 1x1 SOM, is too small for classification. Its value approaches the average of



(a) Linear scale and an exponential curve.

(b) Logarithmic base 2 scale.

Figure 4.12: Relationship between SOM dimensionality and its area in the doubling sequence.

the dataset, since the SOM algorithm adjusts the codebook values towards the average of the observations that best match each node in each epoch of training. With only a single node, every observation would match the single node, so the map would therefore converge towards the average of the dataset. Accordingly, the smallest map with equal sides to consider is the 2x2 map. Assuming there is no a priori reason to insist on one particular size over another, then we may take into account certain intrinsic factors to SOMs, such as that maps of larger sizes have higher resolution but less generality (as fewer observations are associated with each unit).

Starting with a 25x25 matrix, an 18x18 matrix is composed of approximately half the number of cells. And a 36x36 cell is composed of approximately twice the number of cells. To get to the next doubling, the size is increased again by doubling the initial 25 to create a 50x50 matrix. Using this procedure, up and down, we arrive at sizes: 3x3, 4x4, 6x6, 9x9, 12x12, 18x18, 25x25, 36x36, 50x50, 72x72, 100x100, 144x144. The number of cells in each of these matrices is 9, 16, 36, 81, 144, 324, 625, 1296, 2500, 5184, 10000, and 20736, respectively.

The exponential curve of closest fit to this line is $y = b * m^x$ for $x \in 1, 2, \ldots, 11$ for some constants b and m. The values of approximately $m = 2.03$, $b = 4.34$, with a standard error for $m = 0.0056$ and a standard error for $b = 0.038$. Although, of course, the sizes of the matrices were determined using a heuristic to find an integer sequence that doubles in area and is comprised of many squares and double squares. The standard error for this estimate of y was 0.059. Figures 4.12a and 4.12b depict the relationship between the SOM dimensionality and area, and show that the fit of the exponential curve is quite close despite adopting regular values. The area of each SOM dimensionality is shown as a label to the left of the data marker. The two figures are identical except that the first shows the number of nodes with a linear y-axis, but the second shows it with a logarithmic scale. Although the values are not at the centre of the exponential line of best fit, it should be clear that this series is approximately

equivalent to the doubling of map size yet still retains the quality of an aesthetically acceptable integer sequence.

Using this sequence of map sizes, the performance of different sizes of SOMs can be evaluated, with an approximate doubling of area at each successively larger map size. If different sized maps exhibit similar performance, smaller map sizes are preferable to larger maps, so this grid search enables increased granularity at the smaller sizes and allows for an exploration of the effect of increasing map size on the performance of our system for phoneme recognition and analysis.

4.5.2 Results for the four vowel dataset

In the case of the unsupervised KNPT with a small quantity of labelled data, the training set is larger than the labelling set. The training data is used to train the map, while the labelling set is used to label it. This makes it possible to leverage larger data sets that are only partially labelled. However, this raises a question: which set of data should be used, training or labelling, when setting the map size based on the number of data points per node? What other factors should we consider?

The map size that we choose depends on our purpose for training and labelling the maps. As there are two different sets, one for training and one for labelling, the distribution of each set will be different. As a result, the labelling set of vowels may not map to all the cells that learnt from the training set. To reduce this effect, a smaller size map will increase the probability that the labelling set covers the similar area in the training set.

In Figures 4.13, 4.14, 4.15, and 4.16, the background grey shade of each cell represents the number of hits, which is the number of data points in the training data that were mapped to that cell by the end of training the map. The number of hits indicated to the right of the colourbar provides an indication of how many data points there are for each shade of gray. Note that for larger maps, the number of different data points is too high to show the count for each shade, so intermediate counts are omitted to save space.

The labelled data was used to colour the map. Each phoneme is represented with a different colour. The size of the circle for each phoneme in each cell is scaled to the number of observation that was best matched with its codebook value. The colours for each phoneme follow MATLAB colour specification[2], with the mapping as shown in code listing 4.6.

The labelling of phoneme text was accomplished by TA_VOTE, which is a custom labelling method that includes all labels that comprise over 34% of the data in that cell. This ensures

[2]The MATLAB colour specification provides short names for six colours plus black and white and may be consulted at https://au.mathworks.com/help/matlab/ref/colorspec.html.

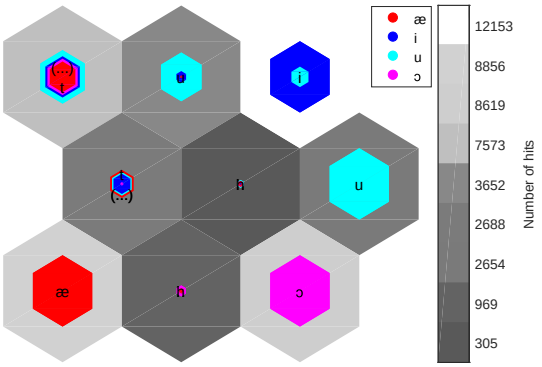


Figure 4.13: Four vowels on a 3x3 map

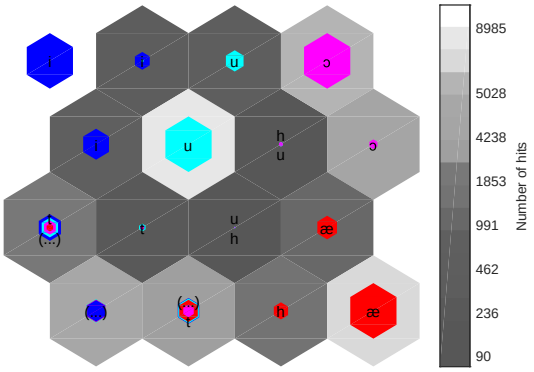


Figure 4.14: Four vowels on a 4x4 map

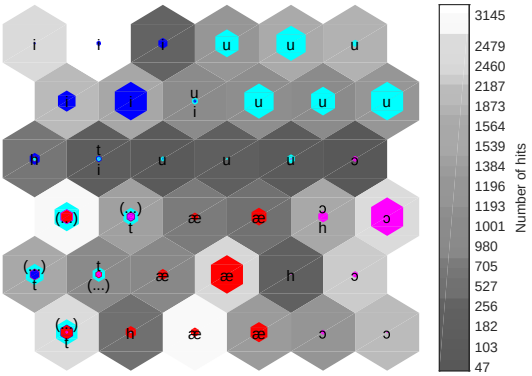


Figure 4.15: Four vowels on a 6x6 map

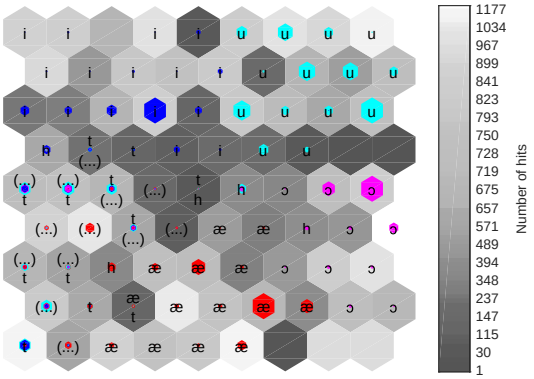


Figure 4.16: Four vowels on a 9x9 map

that no more than two labels appear in any cell, excluding those minority labels that had lower likelihoods of appearing.

We can see in Figure 4.16 that the larger map has increased resolution. As a result, we can see some differences between the training data and the labelling data. In the /i/ portion of the map (at the top left), many of the cells are lighter grey, which indicates that they had a significant number of hits from the training data. However, the small size of the blue hexagon in the centre of some of the cells indicates that the labelling data did not fit the training data. Although it was sufficient for providing labels to most of the cells in the top right, we can observe that there is an unlabelled cell in the middle of cells labelled by /i/. The likely explanation for this cell is that it had learnt to recognise /i/ from the training data, but that no significant labelling data was similar enough. As the size of the map increases, this phenomenon of unlabelled cells also increases. Due to space considerations, I will not provide further maps to illustrate this here, but it is an important consideration for any unsupervised KNPT approach that has only been mentioned.

Listing 4.6: Phonemes and colour coding

```
phonemes = {'æ', 'i', 'u', 'ɔ'};
colours = {'r', 'b', 'c', 'm'};
```

The phenomenon of unlabelled cells occurs even where the entire training set is labelled and the labels come from the training data. As a result, a number of methods for resolving unlabelled cells have been developed. For example, the k-nearest neighbours approach in (Kohonen, 2014, p.101) takes the labels from a number of neighbours (this may be a dozen or a hundred in practice) and determines the unlabelled cell by a vote.

At the other extreme, Figure 4.13 depicts a small map. It can be observed that the resolution is so low that there is considerable overlap between /i/ and /ʌ/ cells (blue and cyan). For the purposes of phoneme classification, it would appear that this map would provide poor results.

Figure 4.14 shows complete coverage of the labelling data over the training data map, with no gaps. There is a direct correlation between those cells with higher number of hits from the training data (shown in lightest shades of grey) and the highest number of hits from the labelling data (shown with the largest coloured hexagons). For example, the cell labelled /i/ at the top left is light grey and has a large number of labelling data hits as well.

It can be observed that a number of cells include labels for both "silence" /(...)/ and "t" /t/. This occurs for part of the "t" phoneme because it is a plosive (an oral stop), in which the airflow is stopped and then released. During the portion of the "t" where the air is stopped, the speech stream is basically silent. As a result, some windows of both the "silence" and the "t" are mapped to the same cell.

On the other hand, if the purpose is the classification of phonemes, and there is sufficient labelled data for both labelling and testing, then the results may be evaluated quantitatively. One method is to use a number of different sized maps to find the size that achieve the purposes for using the KNPT.

Code listing 4.7 shows the method of labelling.

Listing 4.7: Four vowel density

```
% training data hits with labels
figure
som_show(sMap, 'empty', '', 'footnote', '');
som_show_add('hit', som_hits(sMap,train_sData), 'MarkerColor', 'y');
arrayfun(@(index) som_show_add('traj', vowel_file_bmus{index}, 'MarkerColor',
    colours{index}, 'TrajWidth', 0), 1:size(words, 1));
labelled_sMap = som_autolabel(sMap, visualise_sData, 'ta_vote1_34');
som_show_add('label', labelled_sMap);
```

4.5.3 Phoneme error rate

Most of the utterances used for this analysis were not annotated (only 1/8 of the files were manually annotated). However, they were produced according to prompts, so although the precise locations of the phonemes is not known, the content of each file is known because of how it was produced. As a result, without further annotation, it is not possible to determine the frame error rate, which is frequently reported in the KNPT literature. On the other hand, PER at a file level can be addressed as follows.

Each file is named with a description that allows us to know the phoneme that was produced. Under the training paradigm, a map is trained using the unlabelled data, without any information about each of the phonemes. Using the small set of annotated data, the labels from the annotated data are projected on the map to the BMU, and the label that appears the most frequently for each cell is indicated as its representative label. Then, a test utterance, which was not used for either training or labelling, is labelled using the map. A 16 second audio file has approximately 1600 frames that get labelled. The most simplistic method for determining the classification of the file is to accept the majority of labels. Although each file includes silence and consonants as well as vowels, the goal is here is to identify the vowels among these other sounds. One assumption is that the vowel frames are classified as vowels and the non-vowels are classified as non-vowels. For this four vowel dataset, it was observed that this approach, though simplistic, had a certain degree of success.

This dataset was produced in such a way that a significant portion of the map can be expected to be dedicated to each of the vowels. Specific to the dataset is that the vowels were produced with extended phonation, so that means that the vowels represent much more time in each file than the consonants /h/ and /t/. Figure 4.16 shows the proportions that are similar to all maps: approximately 20% of the map is filled with silence, /t/ and /h/, with the remainder split fairly equally between each of the four vowels, each being mapped to a corner.

Each file was produced in essentially the same way. Using the annotated files, we can determine an approximate proportion of each segment to the length of each file. Importing the PRAAT TextGrid in MATLAB means that I can access the information detailed in the annotation. It is now possible to determine the total duration of all segments of a particular phoneme (or silence) for a manually annotated file, as follows.

Based on the timings derived from my annotations, Table 4.1 was prepared. It may be observed that the vowels take up the majority of each file. It is assumed that the extended vowels in these files are easier to classify than more diverse phonemes. However, a 2x2 map contains only four cells. In such a small map, it was found that the four cells do not get dedicated to each of the four vowel phonemes that are the targets of the investigation. Two cells are assigned to vowels, and the other two are assigned to consonants and silence. Since there are more than four classes, as we are also considering consonants and silence, the maps

Listing 4.8: Durations

```
interval_heat = visualise_ta_data.textgrid{:}.praat_textgrid(2).items{:}.
    interval
interval_heat_table_temp = cellfun(@(c) interval_heat(strcmp(interval_heat.
    text, c), 1:2), unique(interval_heat.text), 'uni', false);
duration_temp = cellfun(@(itt) varfun(@sum, rowfun(@(xmin, xmax) xmax - xmin,
     itt)), interval_heat_table_temp, 'uni', false);
interval_heat_duration = vertcat(duration_temp{:});
cellfun(@(s, d) sprintf('%s %.2f', s, d), unique(interval_heat.text),
    table2cell(interval_heat_duration), 'uni', false)
```

Table 4.1: Four vowel corpus total phoneme durations

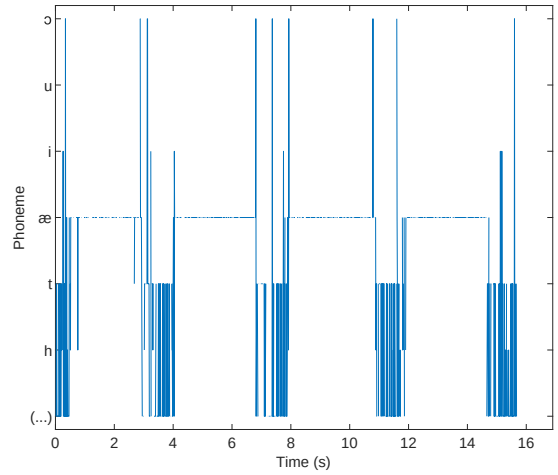
hat		heat		hoot		hot	
phone	time	phone	time	phone	time	phone	time
/æ/	11.57	/i/	11.09	/ʌ/	11.31	/ɔ/	12.18
/h/	0.64	/h/	0.68	/h/	0.38	/h/	0.57
/t/	1.11	/t/	2.03	/t/	1.90	/t/	0.93
/(...)/	2.29	/(...)/	1.80	/(...)/	2.01	/(...)/	1.92

included four vowels as well as silence, /h/ and /t/. A 2x2 map must therefore contain multiple classes in a single cell and is not sufficient for identifying the different classes.

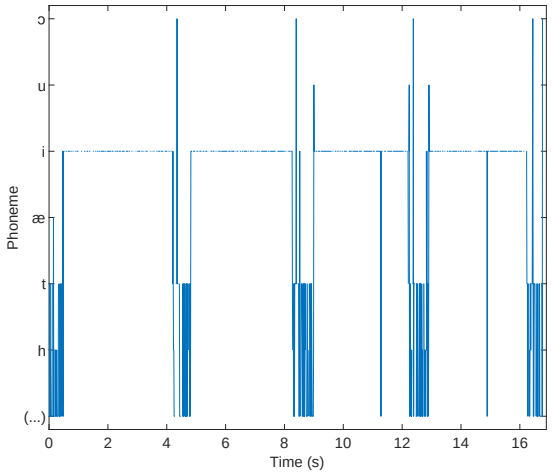
In Figures 4.17a, 4.17b, 4.17c, and 4.17d, the results of classification for one fold of cross validation (with a 50x50 map) are depicted. Note that because the output from all map sizes is relatively similar, only a single fold at a single map size is presented. In these figures, it can be seen that for periods that were not classified as vowels. As a result, the successful classification of the phoneme can be carried out in all cases. Smoothing, such as using a mean, would improve the stability of classification. However, if the vowels were shorter in proportion to the length of the file, more difficulty would be expected.

It should be noted that the consonants /h/ and /t/ are not clearly separated from silence. The phoneme /t/ contains significant portions of time that the voice is not active, which may significantly increase confusion. Also, the /h/ occurs for only around 5 frames per utterance, which is much shorter than the vowels, /t/ or silence. As a result, that phoneme may not be well represented on the maps. As attention in this thesis is restricted to vowels, the question of how to improve recognition results for these consonants was not pursued, but it would be a good topic for future investigations.

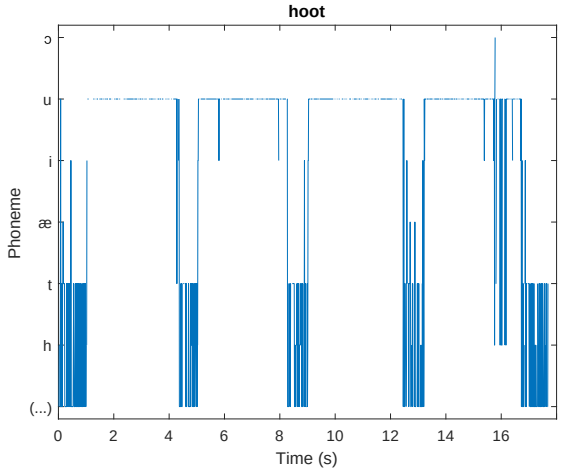
In the final repetition of the word “hot”, shown in Figure 4.17d, it may be observed that near the end of the utterance (near the 16 second mark), increased confusion with /i/, /u/ and /h/ was observed. I listened to that audio file but did not notice any significant difference occurring. At this point in the investigation, it was not clear why the classification was degraded at that point in time (i.e. there was no evidence of incorrect clipping), but it can be noted that smoothing will decrease such discrepancies if classification is the goal.



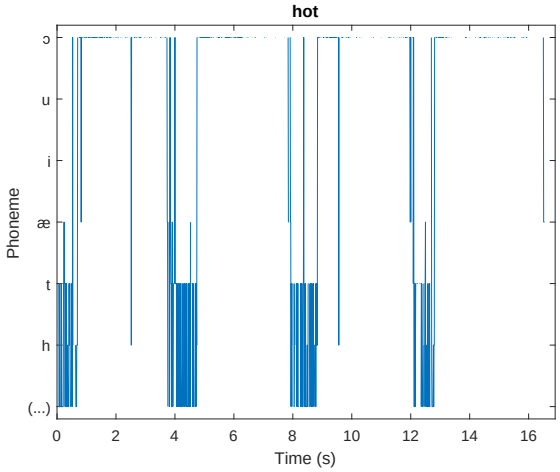
(a) Four repetitions of 'hat'.



(b) Four repetitions of 'heat'.



(c) Four repetitions of 'hoot'.



(d) Four repetitions of 'hot'.

Figure 4.17: Labelling of cVt utterances using unsupervised KNPT.

4.5.4 Discussion

These results reveal that for unlabelled data, the KNPT is able to present a visualisation that clearly indicates a separation between the four vowels. For this speech data, a small proportion was manually labelled. This manually labelled data is then used for labelling the map. This allows the SOM, which was trained in an unsupervised manner, to be used for classification.

In this case, all the annotated data was used for labelling the map. This means that there was no annotated data remaining for statistical evaluation of the classifier at the frame level. Nonetheless, observation of the maps reveals the relationship of the labelled data to the trained map. The range of the dataset used here is limited, consisting of only four vowels that appear to have very minimal overlap on the map. Rather than annotating more of this limited dataset, it was decided to create a new dataset that consists of all the vowels. That work will follow in

the next section.

The work that I have presented so far, with a four vowel single speaker corpus, reveals that the unsupervised SOM can clearly separate some vowels, but that for statistical evaluation, annotated data must be held out, for labelling and for verification. This gives rise to a question: Should the data that is used for labelling also be part of the data that is used for training the map? On the one hand, using the labelled data in the training set, the labels correspond to labelled data points that will more closely align with the trained map, if included in data that was used to train the map. On the other hand, by holding out labelled data during training, it may be expected that the classifier may generalise better. Additionally, some approaches may rely on using the same map with different labelling schemes or different labelled data. Leaving the labelled data out from the training data may thus improve the flexibility of the models.

Issues with classification were found at small and large map sizes. The small maps have too little resolution to perform class separation. On the other hand, large maps have too many unlabelled cells to enable error-free classification of unseen data. Following the advice presented in Kohonen (2014), those cells that remain unlabelled after projecting the labels from the manually annotated data can be labelled with a vote by the neighbours. However, this procedure cannot always work properly. For example, in the 50x50 maps, some unlabelled space was situated between the silence, /æ/ and /ɔ/. Relabelling large clusters of unlabelled cells in such cases can be shown to lead to incorrect results at times. However, for maps that are not excessively large, the vast majority of unlabelled spaces occur in regions surrounded by the same label, and this procedure should work without error. In my experience, this issue affects mostly the framewise error rate, not the PER, because when determining PER, unlabelled cells are disregarded.

Judging by the results obtained to this point, it appears that the unsupervised KNPT approach can work successfully, at least for a small vowel corpus. Accordingly, the technique can be relevant for studies such as those that rely on small inventory of vowels, such as Kalashnikova et al. (2016) that investigated three corner vowels in the target words *sheep*, *shoe* and *shark*. In personal communication with Denis Burnham, a coauthor of the study, it was mentioned that there was an ongoing need for automatic methods for locating the vowels in these types of studies. However, most studies are not limited to such a small vowel space, so the next investigation I will report in the next section is into the applicability of unsupervised KNPT for larger vowel inventories.

Some weaknesses were found for the approach. Due to the low quantity of annotations, the frame error rate could not be reported. Rather than spend time manually annotating the dataset, it would be preferable to work with a more complete vowel inventory. Also, as no phoneme-level misclassifications were found for any but the smallest of maps, the suitability of the map size was not determined; therefore, for such a small dataset, it would be advisable to use a smaller map size, as smaller maps take significantly less time to train.

4.6 KNPT feature representations: Spectrogram versus MFCC39

In the previous experiment, speech was represented as spectrograms. Although the initial explorations for the four corner vowels did appear to have some promise, it was limited to a small part of the vowel inventory. The next step was to create a larger vowel inventory (the entire vowel inventory of a California speaker).

Once I introduced more vowels, degradation of the performance was observed (classifying one phoneme as another), which persisted despite my attempts to tune SOM training parameters such as map size, initial/final neighbourhood size, and normalisation. We identified the feature representation (spectrograms) as a possible area for improvement. Although spectrograms are similar to the feature representation as used by Kohonen (1988), the channels in that research were filtered, grouped into 15 bins, and normalised. Even when I tried similar procedures, the results were not better. Perhaps this is because Finnish and Japanese have fewer or less variable vowels. In this section, I will describe a comparison between feature representations of spectrograms and MFCCs for vowels of English.

Comparisons between feature representations are rarely reported in the KNPT literature (see section 3.3.5.6). The few comparisons between feature representations were: MFCC versus LPCCs, PLP versus RATE, and MFCC versus RASTA-PLP versus PLP. KNPT research since 1990 rarely uses spectrogram-based feature representations, but to the best of my knowledge, there are no published results that explicitly compare spectrograms and MFCCs that demonstrates the superiority of one representation over the other for speech.[3] If MFCCs are better than spectrograms, as will be revealed in this section, then it might be suggested that the feature representation could be an appropriate area of investigation for these types of research.

MFCCs are a feature representation that is common to ASR and speech processing in general. I introduced KNPT research literature that uses MFCCs in 3.3.5.4. Additionally, I described how the MFCC algorithm operates in section 2.6.4. Since MFCCs are much more common in modern ASR research, the hypothesis is that they would demonstrate better for KNPT systems.

4.6.1 Methods

This experiment was designed to compare spectrograms and MFCC39 feature representations for KNPT systems. Spectrograms were processed using the same methods as section 4.3 and consisted of 129 features. MFCC39 consisted of 39 feature MFCCs using 25 ms windows and 10

[3]Related work with bird song (Ranjard & Ross, 2008) and infant vocalisations (Warlaumont et al., 2010) presented comparisons between spectrograms and MFCCs.

ms offsets. Single layer 20x20 SOMs were trained in an unsupervised way and then labelled by projecting the labels from the data back onto the map. The testing was carried out with unseen utterances that were held out for this purpose under a 10-fold cross validation paradigm.

Each vowel token was produced by a single speaker (with a California English accent) with sustained phonation for three seconds and a second of silence between each. Each vowel was repeated 32 times. What I refer to as the corner vowel dataset contains four vowels (/iː/, /uː/, /æ/, and /ɑ/) that are the four corners of the vowel quadrilateral. The full vowel dataset contains the four corner vowels as well as ten other vowels (/aɪ/, /aʊ/, /eː/, /iː/, /o/, /uː/, /æ/, /ɑ/, /ɔɪ/, /ɚː/, /ɛ/, /ɪ/, /ʊ/, and /ʌ/).

The procedures for cross fold validation are shown in MATLAB Listing B.2. Training was faster for MFCC39 than spectrograms. The following times were required for loading data, training, labelling, and testing for each dataset (Corner vowels or full vowel inventory) and speech representation (MFCC39 or spectrogram).

1. Corner vowels
   (a) MFCC39 (20x20): 57 seconds
   (b) Spectrogram (20x20): 107 seconds
2. Full vowel inventory
   (a) MFCC39 (20x20): 165 seconds
   (b) Spectrogram (20x20): 326 seconds

Training was carried out in a similar way to MATLAB Listing 4.3. The training time for MFCC39 was approximately half that of spectrograms for both the corner vowels and the full vowel inventory. It was faster to train using corner vowels than the full vowel inventory. The difference between the corner vowels and the full vowel inventory is the amount of data. The full vowel inventory is approximately 3.5 times larger than the corner vowel set. For both spectrograms and MFCC39, the corner vowel set consisted of approximately 512,000 frames; the full vowel set 1,792,000 frames.

4.6.2 Results

Table 4.2 presents a comparison between recall, precision, F Score and bookmaker results for the corner vowel. The higher score between spectrogram and MFCC39 is highlighted in bold. As can be observed, for this dataset, the MFCC39 performs much better than the spectrogram feature representation. The bookmaker informedness of 0.96 to 0.97 for each of the phonemes, and the 0.85 informedness for identifying silence frames, indicates that performance is consistent

for all these phonemes and that they exhibit excellent class separability at all parts of the utterance. These differences will be explored in the following section (4.7). Additionally, it can be observed that the bookmaker informedness for silence using spectrogram is worse than chance (-0.11), which means that the system is performing worse than chance.

Table 4.2: Framewise comparison between recall, precision, F Score and bookmaker for full vowel inventory (California American accent) with a 20x20map.

	Spectrogram				MFCC39			
Phoneme	Recall	Precision	F	Bookmaker	Recall	Precision	F	Bookmaker
/(...)/	0.30	0.37	0.31	-0.11	**0.87**	**0.91**	**0.89**	**0.85**
/iː/	0.50	0.52	0.49	0.40	**0.98**	**0.94**	**0.96**	**0.97**
/uː/	0.61	0.41	0.48	0.50	**0.98**	**0.95**	**0.96**	**0.97**
/æ/	0.60	0.41	0.48	0.50	**0.98**	**0.94**	**0.96**	**0.96**
/ɑ/	0.44	0.37	0.37	0.32	**0.98**	**0.93**	**0.96**	**0.97**

In Table 4.3, a comparison between recall, precision, F Score and bookmaker results for the full vowel inventory is presented. As in the previous table, the better score of spectrogram and MFCC39 is highlighted in bold. The results for individual phonemes are worse with this larger vowel inventory for both the spectrogram and MFCC39 feature representations. Again, for this dataset, MFCC39 enables superior classification performance in comparison with spectrogram for every metric for every phoneme. It can be observed that even with the MFCC39 feature representation, the 20x20 KNPT framewise classifier for certain phonemes, such as the diphthongs /aɪ/, /aʊ/ and /ɔɪ/ is less accurate than for others.

The classification of silence using spectrogram is again near chance. The KNPT research reported in Kohonen (1988) using spectrograms (FFT) did not clearly describe whether the speech data included silence. A VAD algorithm could be employed for this purpose, which might improve results. I address this issue in section 6.

4.6.3 Conclusion

The results clearly show that for a single layer KNPT, the MFCC39 feature representation outperforms spectrograms. Zue & Cole (1979) showed that humans are able to learn to read spectrograms, but I'm not aware of any attempts to do the same with MFCCs. Since the 20x20 networks in this experiment were able to learn vowels at frame level to a high degree without any higher level techniques, I would argue based on this experiment that it would be easier for a human to learn to read MFCCs than spectrograms.

The system was observed to perform poorly for identification of silence using spectrograms. It is expected that this effect could be reduced by normalisation. As I mentioned in section 3.5, most researchers use data that has been manually segmented so that it does not contain silence (e.g. TIMIT), or a VAD algorithm is used prior to evaluation. As the main goal of my research

Table 4.3: Framewise comparison between recall, precision, F Score and bookmaker for full vowel inventory (California American accent) with a 20x20 single layer KNPT.

	Spectrogram				MFCC39			
Phoneme	Recall	Precision	F	Bookmaker	Recall	Precision	F	Bookmaker
/(...)/	0.33	0.55	0.41	-0.06	**0.87**	**0.88**	**0.88**	**0.84**
/aɪ/	0.14	0.03	0.05	0.10	**0.40**	**0.33**	**0.36**	**0.36**
/aʊ/	0.23	0.12	0.16	0.19	**0.60**	**0.57**	**0.58**	**0.58**
/eː/	0.40	0.19	0.24	0.36	**0.54**	**0.53**	**0.52**	**0.51**
/iː/	0.40	0.42	0.38	0.37	**0.76**	**0.79**	**0.76**	**0.75**
/o/	0.15	0.06	0.08	0.11	**0.72**	**0.76**	**0.74**	**0.70**
/uː/	0.34	0.22	0.25	0.31	**0.88**	**0.86**	**0.87**	**0.87**
/æ/	0.42	0.32	0.35	0.39	**0.91**	**0.90**	**0.91**	**0.91**
/ɑ/	0.34	0.25	0.28	0.30	**0.79**	**0.82**	**0.80**	**0.78**
/ɔɪ/	0.34	0.13	0.17	0.29	**0.40**	**0.32**	**0.35**	**0.36**
/ɚː/	0.31	0.19	0.23	0.28	**0.93**	**0.93**	**0.93**	**0.92**
/ɛ/	0.32	0.21	0.24	0.28	**0.80**	**0.89**	**0.84**	**0.80**
/ɪ/	0.25	0.16	0.18	0.21	**0.75**	**0.78**	**0.76**	**0.74**
/ʊ/	0.48	0.32	0.37	0.44	**0.86**	**0.83**	**0.84**	**0.85**
/ʌ/	0.34	0.25	0.27	0.30	**0.81**	**0.85**	**0.83**	**0.81**

is to classify vowels, not silence, rather than biasing the maps towards better classification of silence using spectrograms, the work will be using MFCCs. I report on VAD using MFCCs in Chapter 6.

When I started work with KNPT for vowel evaluation, I was keen to use spectrograms as the feature representation, because research has shown that neurons in regions of the brain (the primary auditory cortex) are tuned similar to spectrograms Mesgarani et al. (2007), and because Kohonen (1988) used a representation similar to mel-scaled spectrograms. As I described in section 2.6.4, MFCCs also use an FFT transformation that then undergoes further processing.

4.7 KNPT mapping of monophthongs and a full vowel inventory

In the previous section (4.3), an unsupervised KNPT was used with spectrograms for phoneme classification. Here, attention is turned towards a more complete inventory of vowels, using the MFCC feature representation. The end goal is to be able to evaluate speech, but it is important that we know when KNPT is being used effectively (and when it isn't). Since the basic building block of speech is the phoneme, and vowels are primary phonemes for accent, the idea is to use maps to build a model that will allow the evaluation of vowels. Rather than restricting the input to just the corner vowels, a complete vowel inventory of English will be used for

training, labelling and testing purposes. Speech was recorded from a single speaker with a California English accent. In addition to visualisations, we need metrics to understand how well the system can perform tasks such as phoneme classification. Although the spectrograms did result in somewhat satisfactory results for the small dataset, I found that MFCCs enabled the larger dataset. Subsequently, the remainder of experiments presented in this thesis will use MFCCs.

4.7.1 Single speaker KNPT

The work in this chapter involves speaker dependent KNPT, which trains and tests maps with the speech from a single speaker. A review of the KNPT literature reveals that much of the original KNPT research involved single speaker corpora. The seminal KNPT research trained 10x15 maps to output the phonemes of a single speaker (Kohonen et al., 1988). Other work by this research group that used speech of a single speaker for speaker-dependent KNPT for phoneme sequences includes Torkkola (1988); Torkkola & Kokkonen (1991). Subsequently, other KNPT researchers also pursued this vein of research. However, I found that there have been only around a dozen KNPT publications that involved training maps with a single speaker, and I will outline them here to give context to the current study.

Danielson (1990) trained maps on a single Danish speaker on 2 minutes of speech using cepstral feature representation. The map size was not explicitly given, but a figure showing a 15x15 map was included in the paper, so it seems likely that the maps would have been the same 15x15 as the figure. Danielson stated that it took 2 hours to train a map, and that work was made towards implementing the identification of BMUs using a "DSP32 board (250 nsec instruction time)" that could complete training in approximately 30 minutes. In comparison, I was able to train and label a 15x15 map on 30 minutes of speech in MATLAB in live script mode in less than 4 seconds. Although comparing results between papers is problematic for reasons I have already discussed, my results for vowels seem much higher, which I think is probably due to the map size and larger dataset. Another thing that differentiates my research from these earlier works is that they only reported results based on a single map, but I present 10-fold cross validations at 11 map sizes, up to 100x100.

Work around the same time included Knagenhjelm & Brauer (1990), who trained and tested a 16x16 map with the vowels of a single Swedish speaker. Kangas (1991) used a 216 unit map[4]. Rigoll (1990b) investigated the considerations of mutual information in KNPT of a single German speaker using 16x16 maps. Dalsgaard (1992) trained a 20x20 map with the speech of a male speaker.

More recently, a handful of researchers have studied speaker dependent KNPT. Yet despite the improvements in CPU and memory bandwidth, larger maps were not reported as part of

[4]Kangas didn't give the map dimensions but based on the factors of 216, it seems likely that they were 27x8

these efforts. Kaski & Lagus (1996) used 6x4 maps to classify the phonemes of a single speaker. Using 10x20 maps trained on the speech of a single Spanish speaker, Gardón et al. (1999) reported classification results for silence and vowels that seem consistent with the results that I will present here. Male Spanish speaker, only four words, and results that seem quite adequate (91% accuracy was reported) with a 25x5 map (De Luna-Ortega et al., 2009).

Finally, only one KNPT researcher reported tests for different map sizes for a single speaker, but those tests were limited in scope to a series of maps from 10x10 to 20x20 maps (Eng, 2006). In comparison, the map sizes that I have considered in this chapter range from 2x2 to 100x100, and it seems clear from the results that maximum classification performance is not approached with 20x20 maps. In other words, I believe that Eng's results showed that map size was a limiting factor but not whether an increase of map size would always obtain better results.

Eng remarked that other SOM researchers have described the importance of testing different map sizes. Similarly, "It is not possible to guess or estimate the exact size of the array beforehand. It must be determined by the trial-and-error method, after seeing the quality of the first guess" (Kohonen, 2013, p. 56). Finding how map size affects the results is an important consideration, but previous technological limits influenced researchers to use small maps. In other words, challenge is to determine whether using larger and larger maps can provide a key to better results, or, if there are limits to improvements, how to find them.

4.7.2 Methods

This experiment involved a KNPT system for the study of an ideolect, which is the accent of a single person. A single native speaker's voice was recorded over their vowel inventory. Monophthongs, diphthongs, and r-coloured vowels were recorded in isolation (without any initial or final consonants). Similar to the vowels produced for the earlier work, the vowels were sustained for approximately three seconds each, with a second of silence between each. One file consisted of four repetitions, and there were eight files created for each phoneme. I recorded each of the different vowels shown in Table 4.4. For each vowel, 32 recordings were made. Each of these vowel productions was three seconds in duration with a second of silence between each. In total, there were 1280 seconds of audio.

As with the previous experiment, a single channel (mono) of speech was converted to 39 feature MFCC using 25 ms windows and 10 ms offsets. The feature representation thus included 12 mel-frequency coefficients, the sum of log energies, and additionally, delta and delta-deltas were computed for each of these 13 features using the preceding and following frames. After MFCC processing, with frames overlapped at 10 ms intervals, the result was 1,280,000 observations with 39 features for each observation in the dataset.

The goal was to perform unsupervised training for the vowels, label them using a small

Table 4.4: Vowels of the General American accent.

monophthongs		diphthongs		r-coloured vowels	
IPA	example	IPA	example	IPA	example
/æ/	hat	/aɪ/	hide	/ɑɹ/	arm
/ɑ/	hot	/aʊ/	now	/ɛɹ/	mare
/ɛ/	het	/eɪ/	hate	/ɚ/	hurt
/ɪ/	hit	/ɔɪ/	hoit	/ɪɹ/	deer
/i/	heat	/oʊ/	hoed	/ɔr/	hort
/ʌ/	hut			/ʊɹ/	tour
/ʊ/	hood			/jʊɹ/	pure
/uː/	hoot				

labelled training set, and then to classify each of the vowels. It was expected that the successful classification rate would not be as high as the four vowel corpus, because the phoneme classes have significant overlap. However, in accordance with the high classification rates in the literature, it was expected that it would be possible to attain high levels of classification, despite English not being as easy to classify as other languages such as Finnish and Japanese that were used in the initial KNPT literature.

After consulting Ladefoged (2007) it was determined that the speaker's accent would be most similar to the Californian accent. Therefore, from the dataset that was created, the vowels of the California accent were selected. For context with Figure 4.2, I prepared a vowel quadrilateral for eleven vowels of the California English accent, as shown in Figure 4.18.

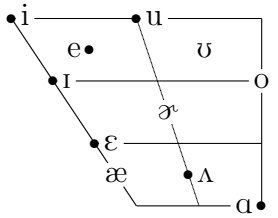


Figure 4.18: Eleven monophthong vowels of California English accent placed on the IPA vowel quadrilateral.

Monophthongs are phonemes that are regarded to consist of a single vowel sound. In contrast with diphthongs, a monophthong does not move from one position of articulators to another. The monophthong vowels for a California accent consists of eleven vowels: /eː/, /iː/, /o/, /u/, /æ/, /ɑ/, ɛ/, /ɚ/, /ɪ/, /ʊ/, /ʌ/.

The diphthongs of this accent are shown in Figure 4.19. Both these formant charts follow the example shown in (Ladefoged, 2007, p. 42). Through the course of this research, I learnt that the California accent is slightly different than the general American accent shown in Table 4.4. The diphthongs are formed of a nucleus and an offglide but following (Ladefoged, 2007, p. 30), "English long vowels and diphthongs are often analyzed as unitary phonemes such as /iː/ (as

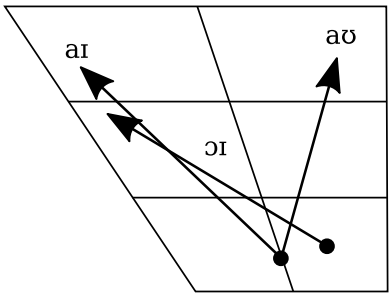


Figure 4.19: Three diphthongs of California English.

in *heed*) and /aʊ/ (as in *how*)", as is done here. One alternative to this method is to treat them as combinations of short vowels, which could be the topic of future KNPT research.

Using the diphthongs and monophthongs of California English, various sizes of maps were trained and labelled. The classification performance of the maps was evaluated at the frame-level and word-level, and for specific phonemes and for the entire system of phonemes.

4.7.3 Results

The classification performance for a single layer KNPT system trained on two sets of California vowels, the monopthongs and the set of all vowels, with varying map size are given here. The results are presented using bookmaker informedness so that unbiased comparisons can be made with the results in the previous sections, which covered the four vowel map and the comparison between spectrograms and MFCC39 feature representations. In the previous sections, results of frame-level classification were presented. Here, I introduce a simple way to evaluate classification at the word-level. An unbiased estimator allows comparisons of results between sets of different sizes, but the granularity of the evaluation is important. To make a fair comparison with the results of the previous sections, the frame-level bookmaker informedness should be consulted.

4.7.3.1 Monophthong vowels

As I have described, the vowels of a single speaker were presented for varied map sizes. Here, we consider a subset of the vowels known as the monophthong vowels. Using modern hardware (Intel i7 with 16 GB RAM), the training and testing of the entire series of maps from size 2x2 to 72x72 can be completed in around 42 minutes. It took 41 minutes to complete 10 folds of the 100x100 maps. Thus it takes approximately the same amount of time to complete 100x100 maps as ten times the number of maps of varying size. The training of maps dominates the time required to load and process data, train, label, test and store the results.

In Table 4.5, the informedness scores for classification of phonemes are presented. Each utterance was recorded separately, and training and test separation ensured that the classifica-

Table 4.5: Bookmaker informedness at the frame level for California accent monophthong vowels with varying map sizes.

phoneme	2x2	3x3	6x6	9x9	12x12	18x18	25x25	36x36	50x50	72x72	100x100
/(…)/	0.39	0.84	**0.86**	0.85	**0.86**	0.85	0.85	**0.86**	0.85	0.85	0.83
/eː/	-0.04	-0.04	0.03	0.51	0.66	0.78	0.74	0.79	0.82	0.85	**0.86**
/iː/	0.19	0.47	0.62	0.68	0.86	0.92	0.93	**0.94**	0.92	**0.94**	0.93
/o/	-0.04	-0.04	0.62	0.76	0.77	0.76	0.79	0.77	0.77	0.79	**0.83**
/uː/	-0.04	0.53	0.66	0.73	0.74	0.82	0.85	0.87	0.85	**0.90**	0.89
/æ/	-0.04	0.59	0.95	0.96	0.96	0.96	0.96	**0.97**	**0.97**	0.96	0.96
/ɑ/	0.11	0.20	0.54	0.72	0.83	0.89	0.92	0.92	**0.94**	**0.94**	**0.94**
/ɛ/	0.20	0.08	0.58	0.79	0.84	0.84	0.86	0.88	0.91	0.91	**0.92**
/ɚː/	-0.04	0.60	0.91	0.97	0.97	0.96	0.96	**0.98**	0.97	0.97	0.97
/ɪ/	-0.05	0.41	0.55	0.67	0.76	0.84	0.81	0.83	0.91	0.91	**0.92**
/ʊ/	-0.04	0.00	0.66	0.66	0.81	0.87	0.88	0.88	0.88	**0.90**	**0.90**
/ʌ/	0.19	-0.02	0.58	0.82	0.85	0.90	0.92	0.92	**0.93**	**0.93**	0.92

tion performance shown is for unseen utterances. The yellow to green colour scale ranges from minimum to maximum, respectively. The maximum value of bookmaker informedness for each phoneme is indicated in bold font. It can be observed that the classification of silence /(...)/ nears peak bookmaker informedness level even with the 3x3 maps and that performance does not continue to increase significantly beyond this level. This provides an indication that even small maps are able to classify silence successfully using the MFCC39 feature representation.

Bookmaker informedness is an unbiased estimator that improves our ability to make comparisons between tests (Powers, 2011). If we use a high enough value such as a 0.80 bookmaker informedness as the cutoff for significance to determine if classification was successful, then small maps (6x6 and above) are able to perform the classification of silence successfully. Larger map sizes (50x50 and above) allow for the best discrimination between the monophthongs. Some of the vowels appear easier to classify that others. Two monophthongs, /æ/ and /ɚ/, are classified well even with small maps (KNPT performance exceeds the 0.80 informedness level for 6x6 maps and above).

In contrast, the classification levels for /eː/ and /o/ do not achieve this value except with larger maps (50x50 and 100x100 maps, respectively). This provides an indication that these two phonemes are more difficult to classify. In Figure 4.20, a 6x6 map is shown to better understand why some classes may result in failure when others succeed. For this map size, the /eː/ phoneme has 0.03 informedness. If we inspect this map, no nodes have been assigned the label /eː/. At most, the other 6x6 maps in the 10-folds assign one node to this phoneme. As a result, most of the time that a frame from this phoneme is presented to the SOM, it best

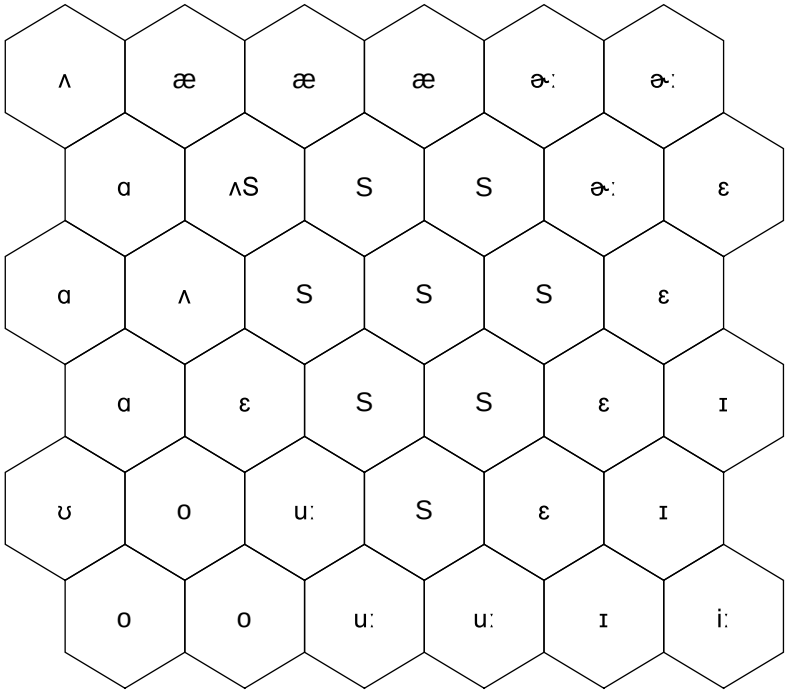


Figure 4.20: Example of a 6x6 map trained on MFCC39 frames of California accent monophthong vowels.

matches other nodes.

In Table 4.6, bookmaker measures of markedness and informedness are shown for map sizes from 2x2 to 100x100. Bookmaker informedness is highest for a 100x100 map, which means that a larger map provides more information. However, the results of the 50x50 map are within one standard error. The highest markedness is reached for the 36x36 map (and the most parsimonous model within one standard error is 12x12), which indicates that the predictions of classes are most consistent with the distribution with this size map.

When working with 20x20 maps as in the previous sections, I conducted attempts to improve the voting mechanism to allow for better recognition. Although associating a node with statistical information about the training data that best matches the node can improve the classification performance, it is still a limiting factor. To reduce this problem with KNPT, Kohonen investigated LVQ tuning, which deliberately changes the self-organising nature of the network to better recognise phonemes when these kinds of issues occur. However, when using an unsupervised approach, such as is being used here, these approaches are impractical because it is the intention to develop a system that does not need the classes for training. The classes are not known during training, so when factors such as not having sufficient nodes present occur, it is not possible to use the classes to improve classification during training.

Table 4.6: Markedness and informedness scores at the frame level across all California accent monophthong vowels with varying map sizes.

	Markedness		Bookmaker	
Size	Mean	Std. Dev.	Mean	Std. Dev.
2x2	0.58	0.032	0.11	0.014
3x3	0.75	0.031	0.37	0.014
6x6	0.76	0.054	0.66	0.039
9x9	0.79	0.028	0.77	0.027
12x12	0.83	0.013	0.83	0.012
18x18	0.85	0.019	0.86	0.017
25x25	0.84	0.017	0.87	0.015
36x36	**0.85**	0.022	0.88	0.023
50x50	0.84	0.021	0.89	0.015
72x72	0.83	0.012	0.90	0.010
100x100	0.80	0.018	**0.90**	0.006

4.7.3.2 Full vowel inventory

A complete vowel inventory includes the diphthongs and monophthongs. The full vowel inventory for a California accent contains the monophthongs used in the previous section with the addition of three vowels: /aɪ/, /aʊ/ and /ɔɪ/.

The entire series of smaller maps from 2x2 to 72x72 were completed in 47 minutes. It took 46 minutes to complete 10 folds of a 100x100 map. As with the monophthongs, it took approximately the same amount of time to complete 100x100 maps as ten times the number of maps of varying size. It took longer to learn the 14 vowels than the 11 monophthongs.

In Table 4.7, bookmaker measures of markedness and informedness at the frame level are shown for map sizes from 2x2 to 100x100. Bookmaker informedness is highest for a 100x100 map, which means that a larger map provides more information. The highest markedness is reached for the 100x100 map. Predictions for the 36x36 map and above are within 1.96 standard errors, which indicates that the predictions of classes are most consistent with the distribution with this size map and above.

The addition of the diphthongs to the classification task results in somewhat poorer results overall than were achieved for monophthong vowels. With this more comprehensive dataset, there is more variation in the speech data. Larger map sizes can reduce some of this difficulty, but the increase in training time is significant.

The classification results for both monophthongs and dipthongs are shown in Table 4.8. The classification of silence nears peak classification performance at around 6x6 maps. This is similar to the results seen for monophthongs, but shows that adding more vowels may create more difficulty with identifying portions of silence in the speech data files. One explanation for

Table 4.7: Markedness and informedness scores at the frame level across the full inventory of California accent vowels with varying map sizes.

	Informedness		Markedness	
Size	Mean	Std. Error	Mean	Std. Error
2x2	0.53	0.020	0.02	0.002
3x3	0.75	0.045	0.15	0.009
6x6	0.64	0.072	0.42	0.023
9x9	0.80	0.046	0.70	0.028
12x12	0.88	0.045	0.80	0.017
18x18	0.91	0.021	0.87	0.012
25x25	0.95	0.029	0.95	0.008
36x36	0.96	0.033	0.96	0.008
50x50	0.96	0.025	0.96	0.008
72x72	0.98	0.029	0.98	0.010
100x100	**0.98**	0.028	**0.98**	0.007

Table 4.8: Bookmaker informedness at the frame level for the full inventory of California accent vowels with varying map sizes.

Phoneme	2x2	3x3	6x6	9x9	12x12	18x18	25x25	36x36	50x50	72x72	100x100
/(...)/	0.40	0.71	0.84	0.85	0.84	0.85	0.85	**0.86**	0.85	0.85	0.85
/aɪ/	-0.03	-0.03	0.25	0.27	0.25	0.31	0.40	0.45	0.50	0.57	**0.61**
/aʊ/	-0.03	-0.03	0.19	0.41	0.45	0.50	0.57	0.62	0.74	0.75	**0.80**
/eː/	-0.03	0.06	0.00	0.24	0.34	0.55	0.60	0.67	0.69	0.75	**0.78**
/iː/	0.15	0.26	0.44	0.67	0.71	0.68	0.76	0.72	0.76	0.74	**0.80**
/o/	-0.03	-0.03	0.41	0.63	0.71	0.70	0.71	0.75	0.75	**0.77**	0.76
/uː/	-0.03	0.63	0.62	0.76	0.80	0.86	0.83	0.86	0.87	**0.89**	0.89
/æ/	0.01	0.42	0.72	0.81	0.84	0.89	0.89	0.92	0.90	0.91	**0.93**
/ɑ/	0.14	-0.03	0.34	0.63	0.60	0.74	0.78	0.79	0.84	0.86	**0.86**
/ɔɪ/	-0.03	-0.03	0.12	0.37	0.41	0.33	0.40	0.50	0.52	0.62	**0.63**
/ɚː/	-0.03	0.13	0.90	0.91	0.92	0.92	0.93	0.96	0.96	**0.98**	0.97
/ɛ/	0.32	0.47	0.59	0.69	0.72	0.78	0.83	0.82	0.84	**0.87**	0.86
/ɪ/	-0.03	-0.04	0.33	0.53	0.60	0.71	0.78	0.81	0.82	0.85	**0.87**
/ʊ/	0.03	0.33	0.11	0.51	0.70	0.84	0.84	0.87	0.85	0.87	**0.89**
/ʌ/	-0.03	0.16	0.34	0.63	0.74	0.82	0.84	0.86	**0.90**	0.90	0.90

this is that more phoneme classes increases map crowding, leaving less units available for the classification of silence.

With a double in the map size, the training size is approximately double. Training for smaller maps can be accomplished in seconds on modern hardware, but 100x100 maps require several minutes each. Under the 10-fold cross validation paradigm conducted here, this means that training takes hours if 100x100 maps are to be considered. One observation here is that since silence classification can be achieved by smaller maps, a hierarchy of maps may significantly

reduce computational costs. This is further investigated in the following chapters.

The results shown in Table 4.5 provided an indication that map sizes of 36x36 and above allow for the best discrimination between the vowels. However, the worst performance was observed for the diphthongs /aɪ/ and /ɔɪ/. The results in section 4.7.3.1 show that classification of the monophthongs neared peak performance at map sizes of 18x18. Although these map sizes may seem similar, actually a 36x36 map has four times the number of nodes as an 18x18 map. Therefore, it has a much higher resolution.

Table 4.9: Bookmaker informedness scores at the utterance level across the full inventory of California accent vowels with varying map sizes.

Phoneme	2x2	3x3	6x6	9x9	12x12	18x18	25x25	36x36	50x50	72x72	100x100
/aɪ/	0.00	0.00	0.19	0.22	0.38	0.28	0.79	0.89	0.92	0.96	**1.00**
/aʊ/	0.00	0.00	0.29	0.65	0.72	0.88	0.93	**1.00**	1.00	1.00	0.99
/eː/	0.00	-0.03	0.00	0.45	0.67	0.84	0.89	0.99	0.90	0.97	**1.00**
/iː/	0.51	0.73	0.58	0.82	0.91	**0.97**	0.95	0.82	0.86	0.86	0.90
/o/	0.00	0.00	0.39	0.72	0.82	0.88	**1.00**	0.92	0.99	0.99	0.99
/uː/	0.00	0.63	0.97	0.99	0.99	0.87	**1.00**	0.90	0.93	0.90	0.90
/æ/	0.00	0.87	0.86	1.00	**1.00**	0.93	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
/ɑ/	0.55	0.17	0.36	0.75	0.82	0.93	**0.97**	**0.97**	0.93	**0.97**	0.93
/ɔɪ/	0.00	0.00	0.12	0.28	0.26	0.54	0.73	0.78	0.86	0.93	**0.96**
/ɚː/	0.00	0.00	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
/ɛ/	0.90	0.73	0.97	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
/ɪ/	0.00	-0.01	0.45	0.69	0.86	0.99	0.90	0.93	0.97	1.00	**1.00**
/ʊ/	0.00	0.85	0.05	0.57	0.69	0.83	**1.00**	0.97	0.97	**1.00**	0.97
/ʌ/	0.05	0.06	0.46	0.82	0.99	1.00	0.96	**1.00**	**1.00**	**1.00**	**1.00**

Table 4.9 shows the bookmaker informedness scores for error rates for individual vowels at the utterance level are presented. The informedness for each phoneme indicates that with 50x50 and larger maps, the KNPT system is able to identify the phonemes (true positives) of the single speaker without an undue quantity of false positives at a rate of over 0.80. These results differ from previous frame level results because the frame level classifications for the entire utterance are collected to determine the results at the utterance level. In other words, given the classification results for each frame, they are collected to determine the most likely vowel for that utterance. With the smaller maps, diphthongs /aɪ/ and /ɔɪ. have the lowest results.

Figure 4.21 shows a labelled single-layer SOM that was trained on the full vowel inventory of a single speaker. Transition periods silence is indicated with an "S_" or "_S" where the vowel takes the position of the underscore. This map differs from the maps described in section 4.7.1 because it shows silence along with the phonemes. Unless silence is removed from the dataset, it should appear in the maps; however, as I have discussed, many KNPT researchers do not

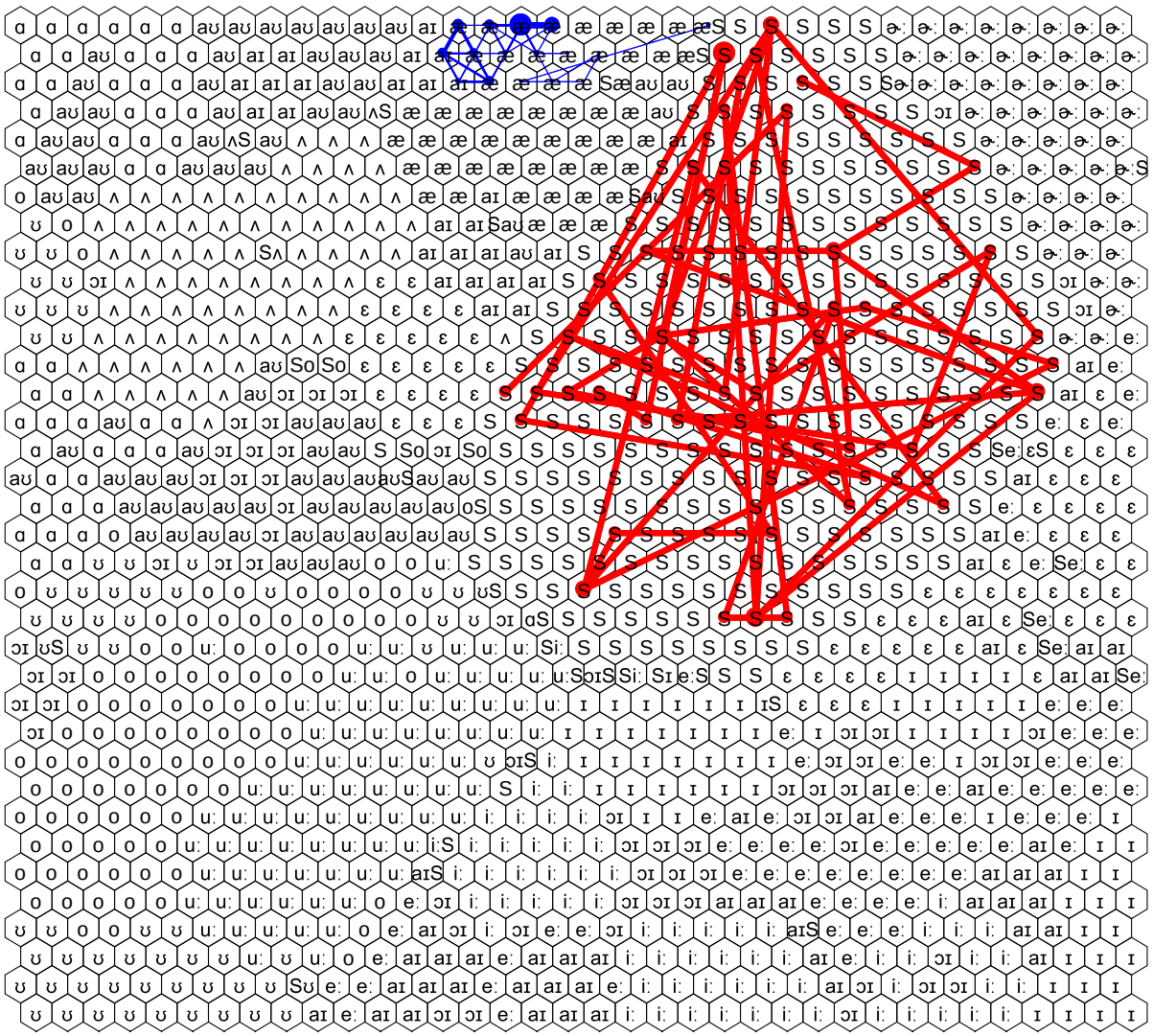


Figure 4.21: Labelled 36x36 map trained on full vowel inventory of a speaker with a California accent. Label "S" indicates silence. Trajectory for one repetition of the vowel /æ/ as in "hat" is indicated with silence in red and vowel in blue (as determined by the map classification).

mention silence (or show maps with silence). This aspect will be explored in more detail in Chapter 6.

One way to use this map is simply to inform. We can see how the different phonemes are clustered. Vowels that are produced with relatively similar positions such as /o/ and /uː/ are often located near one another. Using trajectories, we can show where the sound appears on the map. In Figure 4.21, a trajectory was drawn for one vowel utterance. Although silence was a small portion of the recording, its trajectory (shown in red) takes up a lot of the map because the sound is not stable. Subsequent frames of silence do not result in BMUs that are located nearby on the map. On the other hand, neighbouring frames from the vowel /æ/ as in "hat", which is shown in blue, have a strong tendency to BMUs that are near one another. I

have implemented this classification so that it can be shown in real-time.

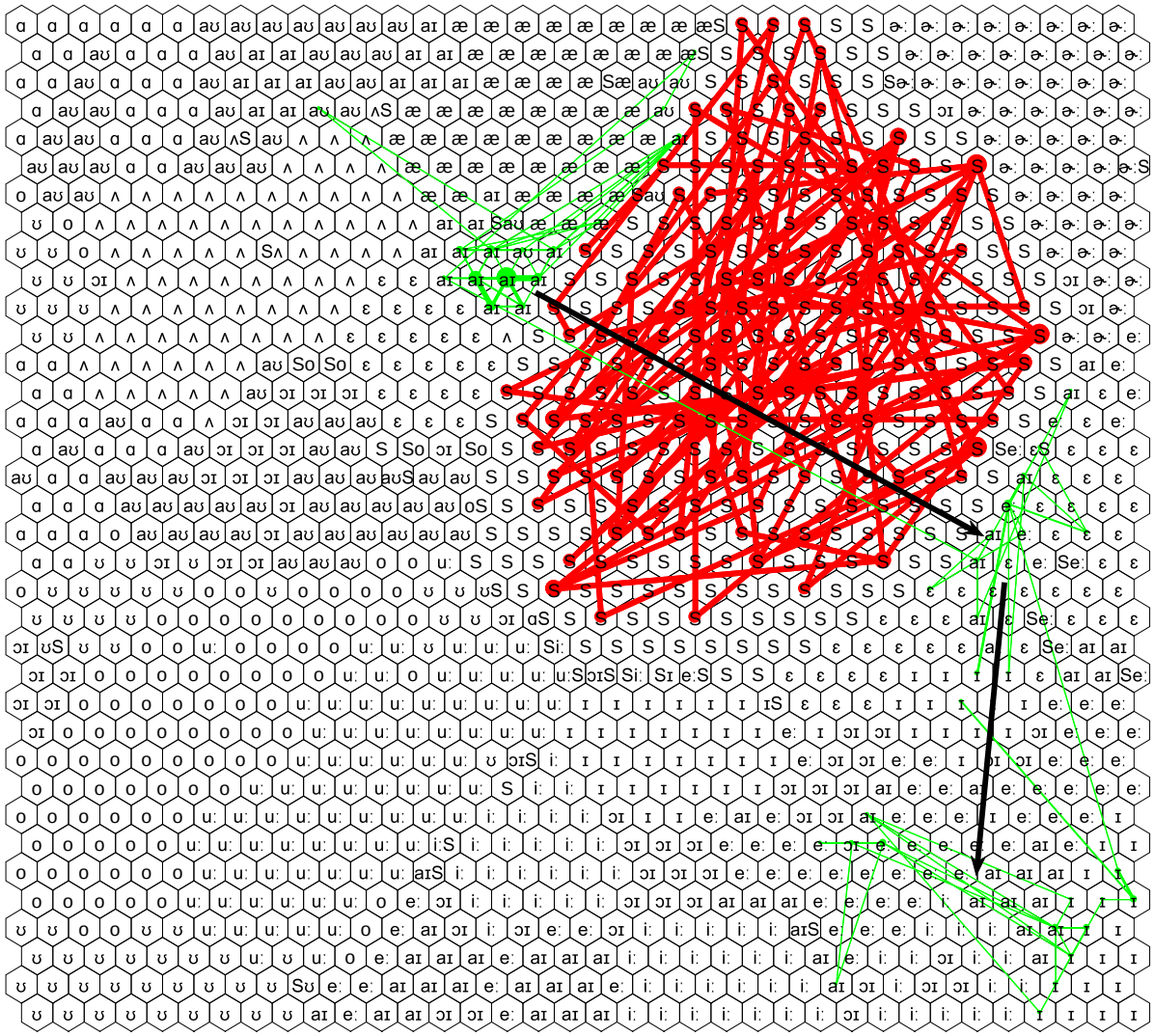


Figure 4.22: Labelled 36x36 map trained on full vowel inventory of a speaker with a California accent. Label "S" indicates silence. Trajectory for one repetition of the vowel /aɪ/ as in "hate" is indicated with silence in red and vowel in green (as determined by the map classification). Two black arrows have been added to indicate the general movement of the trajectory on the map.

In comparison with the trajectory of the vowel /æ/ as in "hat" that was shown in Figure 4.21, the trajectory of a diphthong is more complex, as shown in shown in Figure 4.22. As I showed previously, at the word level, the diphthong /aɪ/ as in "hate" was classified by a 36x36 map at 0.892 bookmaker informedness score (and a standard error of 0.066). Despite a high recognition rate, the trajectory can be seen to travel across the map. This may be expected, since a diphthong by definition has multiple vowel sounds. The vowel /aɪ/ was drawn on the IPA vowel quadrilateral shown in Figure 4.19 with a similar motion across the map. This type of map can therefore provide a visualisation of how a vowel changes over time. However, it should be viewed in conjunction with the statistical analysis to ensure that SOM procedures such as

the selection of map size are sufficient for the classification task.

4.7.4 Discussion

Extending the previous work with four corner vowels to larger vowel inventories, the effect of different map size was explored. Experiments were first conducted using monophthongs, and then adding in the diphthongs of a single speaker with a California accent. The investigations targeted two different measures of classification: frame-level and word-level, with the general finding that sufficiently large maps enable very high rates of success.

It was found that recognition improves with increased map size to a limit (around 50x50) at both the frame-level and the word-level granularity of analysis. Also, we recall from section 4.6, the comparison between spectrograms and MFCC39 showed that a set of four corner vowels that were selected to have the least overlap was also successful at the frame level with 20x20 maps for the MFCC39 feature representation (though not for spectrograms).

The experiments were conducted at the frame level and the word level with varied map sizes from 2x2 to 100x100 with monophthongs and with full vowel inventory. 50x50 maps and larger were able to classify frames at a 0.80 informedness criteria for informedness for the entire set of monophthongs, and even 18x18 maps were able to achieve nearly 0.80 level for all phonemes (/eː/ and /o/ were slightly lower at 0.78 and 0.76, respectively).

It was observed that even the largest 100x100 maps cannot attain perfect recognition for all phonemes. One explanation for this is that the phonemes have a great deal of overlap. For example, it would be expected that the two phonemes /ɔɪ/ and /ɔr/, which both contain the phone /ɔ/, should have more similar activations over the initial part of the vowel, but less similar activation for the final.

Some researchers have sought to remedy this by incorporating the time aspect into the map, such as through RSOM, as described in Chapter 3. For presentation to language learners, it might be suitable to keep the KNPT as presented in this chapter but teach them how the classification results vary over time for certain vowels. It's clear that although the frame error rate does not reach perfect recognition, because the results near 100% for PER even with a simple counting method, KNPT systems with consideration of parameterisation found through cross fold validated grid search has significant capability to learn to recognise the vowels of a single speaker.

One main explanation for this is that early work was probably constrained by memory and CPU speeds. Later researchers have followed the example of earlier work. Therefore, I examined map size for the vowels. My conclusion here is that map size is a very important consideration. If results are lacking, doubling the size of the map can improve them significantly. Doubling

the size of the map increases computational cost, but since the classification performance can be greatly improved, I argue that it is worthwhile.

The size of the map is an essential parameter of a SOM. I revealed in the single speaker literature review (4.7.1) that all but one of the KNPT researchers reported a single map size. It is quite probable that they all tried different map sizes but only reported the successful one. The earliest researchers (in the 1990s and 2000s) would be restricted for the small maps because of the computational time required, which put a very hard bound on how big the maps could be. For example, Danielson (1990) stated that a 15x15 map took 2 hours to train. In contrast, on an fourth generation i7 Intel CPU, I am able to train a single 100x100 map on ten times the amount of data in tens of minutes, not hours.

One factor to discuss is that of whether the system is overfitting. That is to say, is the SOM finding patterns in the data or is it simply storing the data. A 100x100 map consists of 10,000 codebook values. The size of the speech data set used here is 30 minutes x 60 seconds/minute x 100 windows/second = 180,000 observations. With the largest of maps, there are thus 18 data points per node on average, an 18-fold reduction in the size of the data. As I have mentioned, (Kohonen, 2013, p. 56) proposes, "A very coarse rule-of-thumb may be that about 50 items per node on the average should be sufficient, otherwise the resolution is limited by the sparsity of data." The main consideration for maps that are large is that there are sufficient observations for each node to have statistical accuracy. Thus, maps above 100x100 should be trained with more data than the corpora that I used for the work presented in this section. However, the results were almost as good (near perfect recognition at the word level) with a 36x36 map, which is 4 times smaller still (72 times compression, around 150 observations per node).

A sufficiently small SOM prevents overfitting because each node is associated with a high number of observations. One way to test for overfitting is to ensure that the test set is independent of the training set. This was the procedure that I used to generate all the results here; however, it is worth keeping in mind that in experiments that are speaker dependent, the same speaker produces both the training and test set. A more general test could train with one speaker and test with another.

4.8 Discussion

In this chapter, I explored KNPT systems for vowels. I presented evaluation metrics for KNPT systems. I also reported several experiments that used a single-map KNPT system on single speaker datasets. My review of the research literature revealed that not all research in this area reports the best way to determine the proper values of aspects including map size and feature representation. The experiments in this chapter are a summary of the efforts I made to reveal what factors enable successful classification using KNPT. Furthermore, it seems that if earlier

studies did not always use best practises, this can be a stumbling block for new researchers.

Most KNPT research is towards speaker independent phoneme recognition, meaning that the training data (and testing data) comes from multiple speakers. Of the 78 KNPT journal articles, conference papers, technical reports, and dissertations, only 11 research publications described research that involved a single speaker, and of these, only one presented anything but a single map size. Additionally, the maps were all relatively small, generally under 20 nodes in width.

The initial feature representation under consideration was the spectrogram. The spectrogram was found to be useful for corner vowels that have clear separability. However, I encountered unsatisfactory performance when larger vowel inventories were under consideration. Spectrograph-based KNPT systems suffered when using larger vowel inventories. Next, I compared spectrograms with the MFCC39 feature representation, finding that MFCCs provided much better performance for a single layer 20x20 KNPT system for every phoneme or silence, particularly for larger vowel inventories. Finally, I investigated how map size affects KNPT systems. Results indicate that some KNPT researchers may have obtained less than optimal performance from their systems due to overly small maps.

These investigations provide a strong basis for preferring MFCCs rather than spectrograms in KNPT research, as well as supporting the approach of testing varied map sizes, which previous research often fails to report. Additionally, the methods I have presented here, grid search with cross validation, can be used to evaluate different factors, such as varying training parameters, different feature representations such as LPC or lossless speech compression, or other architectures.

A single speaker dataset can show the effectiveness of KNPT as an aid to track the progress of a learner. An initial dataset recorded by the language learner could be used to train maps, and over time their speech could be compared with the initial baseline. This would make it possible to evaluate the speech of a learner in isolation. Alternately, a map trained on the speech of a historic person might serve as a useful aid for a person who seeking to emulate the speech of a specific person, such as a movie actor preparing for a role of a historic person.

Although the accent of the single speaker was under consideration here, an understanding of the speech of a single speaker in relation to the speech of other people would require a comparison with the speech of other speakers. This, together with multi-map KNPT architectures, is the topic of the next chapter.

Chapter 5

Multi-level KNPT for Australian Speech

5.1 Introduction

One main goal of the research that I am presenting in this thesis is to inform the visualisation of speech with an emphasis on vowels. In particular, for phonemes, as units of speech, to be displayed as locations over time, or trajectories, on two-dimensional maps. A specific application in mind for this form of visualisation is the identification of accents. Towards this end, the work in this chapter utilises an Australian speech corpus, Austalk (described in section 5.2), and moves towards an understanding of how speech from multiple speakers can be analysed using a hierarchy of SOMs. As I revealed in the KNPT literature review presented in Chapter 3, previous work using SOMs for speech has put emphasis on ASR, not pronunciation evaluation, the main difference being that when recognising speech between speakers, one generally seeks to eliminate individual differences to enhance classification performance, whilst when exploring pronunciation, one seeks to reveal individual differences in speech.

5.2 Austalk

Austalk was an Australia-wide research collection project tasked with creating a corpus of speech[1]. The Austalk corpus is a collection of recordings created with around 1000 speakers of Australian English. The main target of the collection was participants from Australia who were schooled in Australia. The Austalk project provides an insight into Australian English, with both audio and video being collected. During my PhD candidature, I worked as a paid research

[1]"The AusTalk corpus was collected as part of the Big ASC project ((Burnham et al., 2009,1; Wagner et al., 2010), funded by the Australian Research Council (LE100100211). See: https://austalk.edu.au/ for details.

assistant on the Austalk national research project. The main work I performed for this project was the recording of speech (with video) of speakers in Adelaide. I additionally assisted with the compression and uploading of data from our site, which even in its compressed form was in the hundreds of gigabytes due to multiple microphones and cameras. We stored our data locally in uncompressed form as well[2]. I also conducted testing of the repository software and evaluation of problematic files (sometimes two or more audio files existed for a single utterance; by listening to the files, it was sometimes possible to identify if one or more of the files was included erroneously). As a result of the efforts of myself and the many others on the project, the Austalk corpus of Australian English now includes the speech of nearly 1000 speakers. I also created and contributed software tools for the processing of Austalk data in the MATLAB environment that I have made publicly available at https://github.com/twocs/alveo-matlab.

As a prerequisite for participation in the recording process, speakers completed all of their primary and secondary schooling in Australia. This means that they studied in Australian English from kindergarten to the end of high school (or earlier if they dropped out). Tertiary studies, university, could have been completed overseas. As a result of this requirement to have been schooled in Australia, these speakers are all to be considered speakers of Australian English. Differences in accent, such as regional and ethnic variation, may be captured in such a corpus.

The Austalk corpus includes speech recorded in a number of conditions: Participants read words, sentences, and stories. They also participated in unscripted speaking as well, producing monologues and participating in a pair activity using maps. Although I did conduct some experiments that used more of the available data for training, these tests were not conclusive. In this thesis, which focusses on the unsupervised learning and visualisation of vowels, I will report on my research that used only the word list speech data. As an aside, although it may be that the inclusion of additional speech produced by the participants could improve the maps, I found that the additional quantity of data also meant increased training time, but could also produce effects that made it more difficult to determine proper parameterisations. Thus, the main reason for narrowing the focus of the research to only vowels of read words was to reveal parameters and strategies, with the broadened scope reserved for future research.

5.2.1 Participants

All participants who provided their speech to the Austalk corpus had the entirety of their primary and secondary education in Australia. This means that they attended English language schools in Australia from first grade until the time at which they finished high school. On the advice of my supervisor, three groups of speakers were selected from the Austalk corpus: general Australian, Educated Melbournians, and Chinese background speakers, and no other groups

[2]Uncompressed, a single speaker's data from a single session is often over 300 gigabytes.

were to be studied. One main reason for selecting these populations was diversity in populations but sufficient size for statistically significant results, so each of these groups has a significant number of members (Chinese (29), Melbournian Educated (41)). Other large populations of similar size were not selected for this study, such as Persian (29) or Vietnamese (13), so have potential for future research.

Note that in the previous paragraph, I have listed the total numbers of the population in parentheses. These are the number of participants that are reported by the Alveo query engine; however, at the time of my research, this number was not indeed available. For example, only 14 were obtained from the Chinese language group despite having 29. Since that time, the corpus has been expanded to include more of these records, so future research could include held-out data from more speakers. Similarly, a sufficient number of speakers from Sydney were recorded as part of the Austalk collection, but only four were actually uploaded or otherwise made available at the time of my research.

The speech of participants was considered to be in the Melbourne educated demographic group was made if participants were recorded at the Melbourne site, indicated that they had received a bachelor's degree or above, and were not a member of the Chinese group (i.e. they did not indicate speaking Chinese as a mother tongue or as a second language). The Melbournians were all aged 25-34. There were 7 males and 14 females in this group.

Inclusion in the Chinese as a language group was made if participants indicated that Chinese was their first language (three participants) or, those for whom English was a first language, listed another language either "Chinese", "Mandarin", "Mandarin Chinese", or "Chinese Mandarin". Note that the first language group (Chinese) was selected as a multiple-choice response, whilst the 'other language' indication was made as a fill-in-the-blank response. Accordingly, there were a variety of different ways that second language was indicated, so individuals were selected if Chinese or Mandarin was part of the response. All but two members of the Chinese group were born in Australia; the two born outside reported that they were born in Germany and Hong Kong. There were 7 males and 7 females in this group. None lived in Melbourne.

The General Australian group was selected to capture a broad Australian dialect. All the members of this group indicated that English was their first language, and none spoke Chinese. They did not have university education. All were born in Australia. There were 7 males and 14 females in this group. They indicated that they lived in Perth (9), Adelaide (3), Armidale (1), Castlemaine (1), Bathurst (1), Canberra, Townsville (1), Sydney (1), and Brisbane (1). None lived in Melbourne.

Although Austalk has been anonymised, and personal information, such as names and contact details, was not retained for the privacy of the participants, metadata was collected from each speaker. This information was revealed through a text-based questionnaire that was completed prior to participation in the first session. The information collected included aspects

such as gender, other languages spoken at home, and so forth. Responses to some of these questions were made from multiple choice selections, but for others they were fill-in-the-blank.

I accessed the metadata lookup through austalk-query.apps.stevecassidy.net, which is hosted by Associate Professor Steve Cassidy at Macquarie University's Centre for Language Technology. Cassidy's sample code[3] also helped me to get started with the Python interface with Alveo, which enabled me to write the Python code to download the speech files I needed for this research. Additionally, I wrote code in MATLAB to load the files into a structure that I could then use to find out the metadata for each file. The approach and associated methods evolved over time as I created and improved on strategies for various data management needs, but a representative sample of the scripts used for data management with comments to describe their function are presented in Appendix B.

Each of the speakers selected for my research participated in two to three sessions. In total, data for each person contained approximately 1000 folders. Seven microphones were used to record the speakers. For the purposes of this research, only a single stream of 16 kHz speaker data was used. A JSON file contained information about the file. The prompt, which was the word that was elicited from the speaker, was the main information of importance for this research that was contained in the file. Additionally, three speakers had manual annotations for their speech. These TextGrid format files were created in Praat, a speech analysis software. I discussed how I enabled the use of Praat TextGrids in MATLAB in section A.3. When considering only hVd words, 18 words spoken in three sessions meant 54 relevant folders in total per speaker, though if only two sessions were completed that number decreased accordingly. More than half the speakers completed three sessions. In total I used the speech of 56 speakers, of whom 33 completed three sessions (and the other 23 completed two).

5.2.2 hVd Words

The hVd speech that was used in this chapter was recorded with an AudioTechnica AT892c headset microphone during the Austalk recording process as I described earlier (see Burnham et al. (2011) for more details). The 16 kHz bitrate files from each group of participants were converted to 39-feature MFCCs and used to train a hierarchical network of SOMs. The 39-feature MFCCs are comprised of 12 MFCCs, with the first MFCC being replaced by the sum of log energy, along with 13 delta features and 13 delta-delta features. The delta and delta-delta features are calculated by the method used in the previous chapter, using first order approximations to the derivatives.

The full list of hVd words used for the research in this chapter along with their IPA transcriptions is presented in Table 5.1. Each prompt is presented 'as in "hVd"'. I created this table

[3]Examples https://github.com/Alveo/pyalveo/examples

Table 5.1: The vowels of Australian English in an hVd context.

Prompt	Prompt IPA	Vowel IPA	Type
as in "had"	/hæd/	/æ/	Monophthong
as in "hade"	/hæɪd/	/æɪ/	Diphthong
as in "haired"	/heːd/	/eː/	Monophthong
as in "hard"	/hɐːd/	/ɐː/	Monophthong
as in "head"	/hed/	/e/	Monophthong
as in "heared"	/hɪəd/	/ɪə/	Diphthong
as in "heed"	/hid/	/i/	Monophthong
as in "herd"	/hɜːd/	/ɜː/	Monophthong
as in "hid"	/hɪd/	/ɪ/	Monophthong
as in "hide"	/hɑed/	/ɑe/	Monophthong
as in "hod"	/hɔd/	/ɔ/	Monophthong
as in "hode"	/həʉd/	əʉ/	Diphthong
as in "hood"	/hʊd/	ʊ/	Monophthong
as in "horde"	/hoːd/	/oː/	Monophthong
as in "howd"	/hæɔd/	/æɔ/	Diphthong
as in "hoyd"	/hoɪ/ /hoɪd/	/oɪ/ /oɪ/	Diphthong
as in "hud"	/hɐd/	/ɐ/	Monophthong
as in "who'd"	/hʉd/	/ʉ/	Monophthong

for reference during the course of my research, so it may have some redundancy in presenting the IPA version for both the prompt and the vowel that is produced as part of the prompt, i.e. for the word "had", both /hæd/ and /æ/ are given, representing in IPA the full pronunciation and the pronunciation of the vowel, respectively.

It may be mentioned that it is possible that some of the prompts may have been read by speakers in an improper way (Cassidy et al., 2014). Although most utterances were produced properly, as a trained research assistant was guiding the recording process, the fact that a given prompt may have been misread should be a consideration when interpreting results. Personally, I found several improperly produced utterances, though none in the sample used for this research. Note that an error in pronunciation may involve different mechanisms. For example, it may occur as a duplication (a participant repeats the word correctly more than once), correction (e.g. the utterance is made incorrectly, and is then repeated correctly), or omission (no speech utterance is recorded in its corresponding audio file). Additional noise may also be present in the recordings, such as mouse clicks or speech that is unrelated to the prompt, although this is more common for longer, more difficult prompts than for the single word, canonical form productions that were my focus. Due to the brevity of the noise caused by a mouse click, it is not expected to be mistaken for a vowel, though it could cause minor difficulties for those researchers who investigate consonants such as /t/ (mouse clicks were made in periods of non-speech, so the absence of any temporally proximal vowels could reduce the likelihood of a mouse click to be identified as speech).

For the research presented in this chapter, Australian English consists of twelve monophthongs and six diphthongs, as shown in Table 5.1. Typical of Australian English, a non-rhotic accent, vowels followed by an "r" are not r-coloured but lengthened. Thus, the r-prompts here are associated with monophthong vowels, while in the previous chapter, they were associated with r-coloured vowels (because I was mapping the speech of a single speaker who has a general American accent). For example, the vowel "or" in an hVd context is "horde", which is produced in American English like /hord/ but in Australian English as /ho:d/. As a result, in the table of Australian pronunciations, there are twelve monophthongs and no r-coloured vowels, in contrast with the work of the previous chapter, which had a different IPA representation for the prompts that correspond with a California accent.

One difficulty that is encountered when using vowels in an hVd context is that some of the words on the wordlist are uncommon or not real words at all. Ten of the hVd words have real, common meanings in English so were presented simply with the single word as the prompt. On the other hand, to clarify the pronunciation of those words that are not, the prompt is given in a sentence for context with the structure "hVd sounds like word". The research assistant ensured that the participant reads only the hVd word, not the entire sentence. Eight of the hVd words were presented in this manner, as follows.

- "hade sounds like made"
- "haired sounds like scared"
- "heared sounds like beard"
- "hod sounds like pod"
- "hode sounds like mode"
- "howd sounds like crowd"
- "hoyd sounds like Boyd"
- "hud sounds like mud"

5.3 Testing paradigm (data considerations)

To increase the likelihood that the results of experiments are statistically sound, the testing paradigm of cross validation was utilised. The basic principle of cross validation is to hold out some of the data for testing so that the multiple tests can provide better statistics of the phenomena being observed. Given the three groups (General Australian, Educated Melbournian, and Speakers of Chinese), the data from three participants from each group was held out for

testing. The remainder of the data was used for training the SOMs. All maps were labelled using the small set of annotated speech (data from these speakers was not part of any of the three groups). Secondary maps were trained, as described in the following section. Finally, the classification performance of the maps was determined using the held-out data. As no data from the held-out speakers was used for training or labelling, the classification performance can be used to indicate how well the system performs between speakers or between groups.

In order to test for each speaker, I used stratified cross validation with gender balancing. This entire procedure of holding out data, training, labelling, and testing on the held-out data was repeated to ensure that the data from each speaker was used for testing exactly one time. The main disadvantage of cross validation is that the amount of time required to run the tests increases proportional to the number of folds. Also, data management becomes more cumbersome. In particular, the data used in the training and test sets cannot simply be saved for each fold for future verification purposes without incurring significant storage overhead. Instead, access to the files is maintained using logical addressing without physically duplicating the files.

5.3.1 Cross validation and random seeds

Cross validation relies on random permutations of the dataset. To enable reproducibility, a seed is used by the random number generator to regenerate the random numbers for subsequent verification as necessary. Note that the fixed seed will ensure that the sequence of random numbers will remain the same, but to extract the same data, it is also essential that the sequence of steps that uses the random number generator is the followed in the same order. In other words, to obtain the same data as was used in the experiment, the steps must be the same and their ordering must be retained. No intermediate steps that use the random number generator can be introduced. Some of these difficulties can be reduced by never setting the global seed (which is used for all random number generators), and instead by specifying a seed generator for each subroutine, but this adds the complication of requiring more seeds which must also be stored.

One way to improve the trust that we have in our code is to ensure as a programming convention that the global random seed is not modified by subroutines. Changing the global seed creates issues, for example, when external code is introduced to perform cross validation. It can waste considerable time when the cross-validation procedures are varied slightly by introducing poorly written functions that change the global random seed because the combinations of data will be changed. If the external code is not to be trusted, prior to running an external function, the internal state of the random number generator should be stored, to ensure that the random sequence can be returned to its original position when returning to the program. This applies specifically to MATLAB but is also generally true in other programming languages.

5.4 KNPT

The training and test sets for each fold of cross-validation were controlled so that no speech from a participant that was used for training was used for testing. In other words, the testing process used only speech from participants that had never before been heard by the network. This means that the evaluation is a judgement about how well the KNPT learns between-speaker.

Initially, the single layer SOM that was used in the previous chapter was used for the between speaker evaluations with the Austalk corpus. However, it was found that performance was not as stable as for a single speaker. This is a common phenomenon for ASR in general, intuitively because different speakers have unique characteristics—both physiological (personal) and cultural (accent).

The solution that I used in this research for improving the reliability of pronunciation evaluation is the hierarchical SOM. The idea is that the classifications of a first map are used to divide the data so that subsequent maps may learn more specific features. Working together as an ensemble means more maps, which means more time for processing, but because they are smaller maps, it speeds up the learning. After training the ensemble of maps, there are also more maps at test time. Again, though there may be more maps, if they are smaller, the amount of computation necessary for classification does not necessarily increase.

5.4.1 Hierarchy of Self-Organising Maps

To improve the classification of the maps, I introduce a method of data sampling that uses supervised labelling to produce submaps. The hierarchical SOM is shown in Figure 5.1. First, the base SOM is trained using unlabelled data in an unsupervised manner. Speech data for only three Austalk participants was manually annotated due to cost and time concerns. These participants with manually annotated speech data were not in any of the groups that were used for unsupervised learning in the first step. The labels from the manually annotated speech data are projected onto the base SOM. The training data is run through the base SOM to be split according to some criteria based on the labels. Finally, this split data is used to train the submaps. The submaps may also be labelled using the manually annotated speech data.

Under the unsupervised learning paradigm, speech is not assigned to the nodes based on assumptions that we may have for them (e.g., based on phonetics or other mechanism). For a collection of nodes to be considered a class, the question then is how to resolve the question of which phonemes are in similar classes. The solution here is to use a small set of labelled data to determine those phonemes that fall into the same category. There are several interpretations of how this may be accomplished, a primary one being to use knowledge of phonetics to guide the selection of phonemes for specific auxiliary maps. Indeed, auxiliary maps were commonly

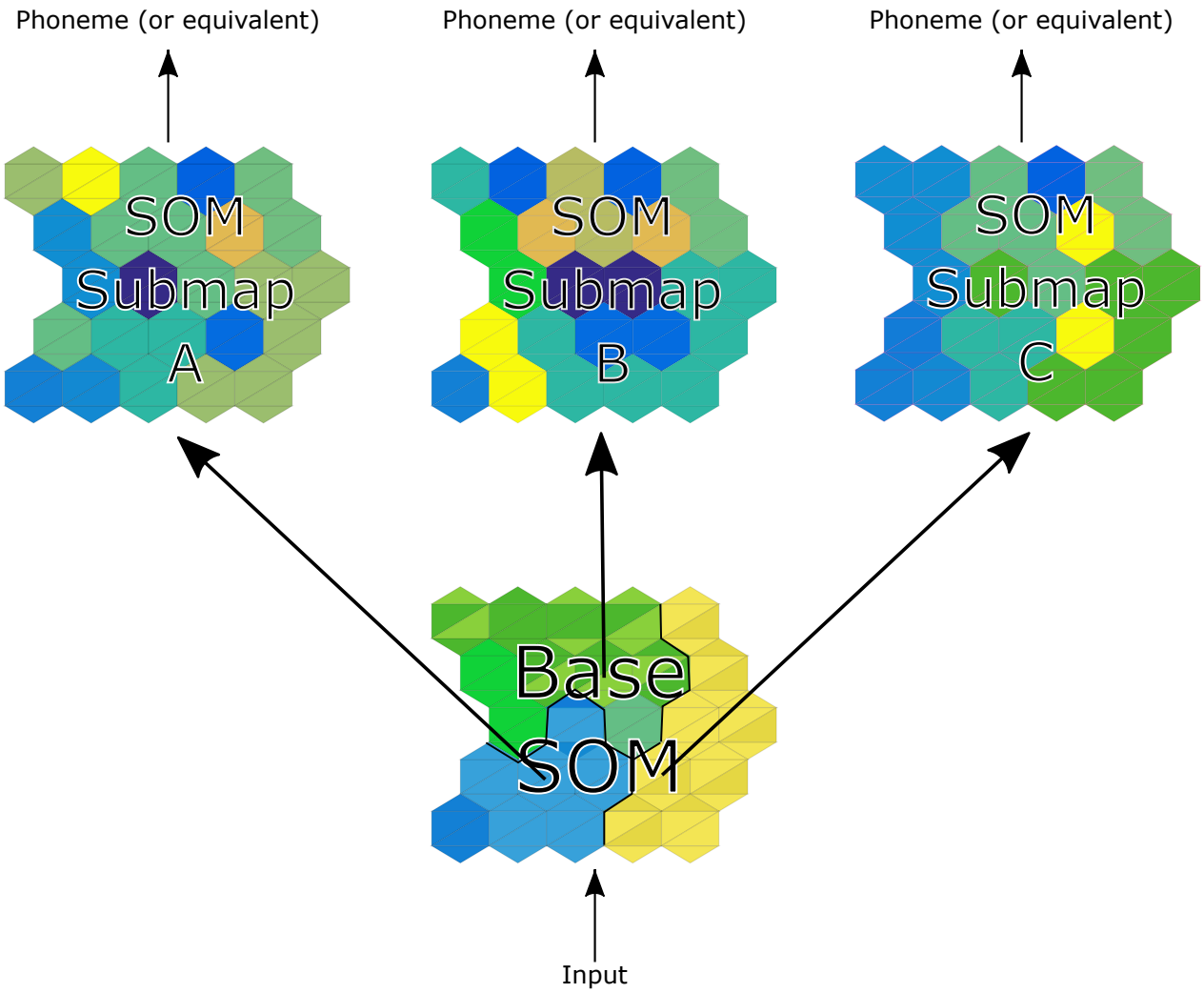


Figure 5.1: Procedures of the multi-level self-organising map.

used for distinguishing certain classes of consonants, for example for Japanese consonants in Kohonen et al. (1988). In other words, the hierarchies are based on linguistic categories.

The criteria may be membership in linguistic groups, such as vowels and consonants, or in specific groups of confused phonemes. Previous work with SOMs has indicated the idea of auxiliary maps. Kohonen (1988) indicated that such a multi-level map structure could be used to distinguish between categories of phonemes that are confused. In the seminal work on unsupervised KNPT, Kohonen (1988) mentioned that auxiliary maps could be used for better distinguishing those sounds of speech that are in the same phonemic category.

One factor to consider is how the different categories are to be determined; judging by Kohonen's description it appears that they are based on phonetic knowledge that we have beforehand. Such an approach aims to improve maps by biasing learning towards what we already know (or think we know) about vocal language, which is composed of phonemes, the target of my investigations. However, I suggest that such an approach biases learning toward

phoneme substitution, which is responsible for the serious errors of pronunciation, but does not encompass the entirety of phoneme mispronunciation. Additionally, the errors made by maps may not be the same as the errors made by humans. As a result, investigations in this chapter will be restricted to categories that are learnt by the system and not to phonetic categories already known to be confused.

In the hierarchical method to disambiguate confused phonemes, a first map is trained on the data. The phonemes that are frequently confused on the first map are then identified. A method to determine groups based on phoneme confusion is an open question. For these studies, I developed a greedy method based on high off-diagonal values in the confusion matrix from the base map. To identify the phoneme confusion groups, this implies successively putting the phonemes with highest confusion into the same group until all phonemes are in one of the groups. Subsequently, the submaps may be trained.

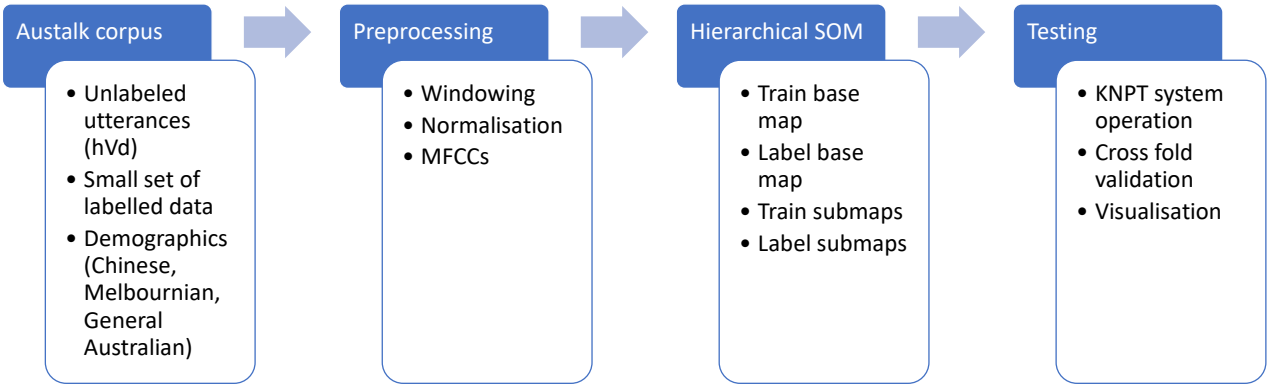


Figure 5.2: Experimental procedures for training and testing.

The process of training used in this chapter that results in a KNPT system capable of classifying the vowels is presented in Figure 5.2. The training data was taken from the Austalk corpus, with the vowels of three speakers in an hVd context that were manually annotated used for labelling. Each utterance was produced by a human participant. The sound was captured using a microphone and saved as a file in a computer. The data management of audio files was managed by my MATLAB class, with further details in Appendix A. The stream of audio data was split into a series of overlapping windows, which was then processed using the MFCC algorithm. Finally, a KNPT system, which includes at its core the SOM, classifies the stream by revealing the BMUs for each window (or combination of windows). The map is then labelled using manually annotated audio. Using these phoneme labels, the unlabelled data may be visualised. The main differences in this chapter is that the utterances were produced by multiple speakers, and the clustering procedure uses multiple maps for improved performance.

The main direction of my research aims for better visualisation of the vowels of the utterances. As each recording of speech contains not only the vowels, but also consonants and non-speech, this means that the maps learn portions of the signal that are not the vowels. In

general, more than half of each file is non-speech. I feel that there is therefore a potential to take into consideration what is important for the task of evaluating vowels, and to narrow the input accordingly. A multistage procedure would be beneficial for the training of maps, because, as shown in the previous chapter, larger maps can in general produce better classification results, but require much more time to train.

5.4.2 Unsupervised KNPT training

During unsupervised training, a SOM learns a two-dimensional mapping of the data. Labelled data is then projected into the map to reveal local relationships in the maps. Although distance and direction are not preserved, useful information is revealed in the maps, which provide a convenient simplification for visualisation on a flat computer screen. Each node learns with regard to its neighbouring nodes, thus serving as a model of its local area.

A considerable amount of work on SOMs has worked towards using labelled data for the learning procedures rather than unsupervised learning. Part of the challenge for conducting this thesis was therefore to learn what parameters are appropriate. I was able to discover a number of the parameters on my own through experimentation and in reference to the literature on SOMs, but halfway through my thesis research, Kohonen (2014) released a book on implementing SOMs in MATLAB that covered many of the significant details. For example, if a map consists of too few nodes, the resolution will be too coarse, and classification will suffer. On the other hand, if too many nodes are used, the number of observations for each node will be too low to get statistically significant results. This parameter (the number of nodes in the map) can be selected with consideration of these factors to produce the desired results. In his book, Kohonen gives considerations and heuristics for selecting the number of nodes in a map (and the width and height), but indeed these are starting points that allow us to understand and discover the parameterisations that will result in the desired visualisations. The actual numbers used will depend on the type of the data, so particular emphasis was given to KNPT research.

In conducting this research into the understanding of phonemes with SOMs under an unsupervised learning paradigm, I have learnt that there are a number of parameters that must be selected appropriately for conditions to be suitable for learning. A main contribution of my work is therefore to clarify the important considerations (or parameterisations) for this type of unsupervised learning. These will be particularly useful for the evaluation of speech, but also by outlining how selecting different values may vary the results, they may also be useful for researchers in other fields.

5.4.3 Model hyperparameters

For this research presented in this chapter, a number of hyperparameters were fixed in order to evaluate the effects of other parameters. Model hyperparameters may be referred to as tuning parameters. The values of the hyperparameters that are used to guide the learning of a SOM are dependent on the data but cannot be learnt from the data. What we can do instead is to vary the hyperparameters, commonly using a grid search algorithm, to find out more about what types of models work best.

There are not many hard-and-fast answers about which values to choose, but rather parameterisation is guided by experience. For an overview of how to select hyperparameters for the unsupervised SOM, I consulted Kohonen (2014) and Kohonen (2001) as primary sources. One goal was to fix as many hyperparameters as possible, which reduces the computational complexity. Hyperparameters here include map dimensionality, map size (e.g. number of nodes in height and width directions, and their ratio), number of iterations, initial and final neighbourhood distance. Additionally, there are various normalisation procedures, neighbourhood types (e.g. Gaussian, cut, bubble) that may be varied. By selecting reasonable or standard values for many of these, the goal was to achieve consistent results and allowing exploration of those aspects that are of interest. In the KNPT literature, I have found that not all researchers indicate what values were used for hyperparameters in training, but I will outline the values I used and how they were selected.

Maps were hexagonal lattice structure. The codebook values were initialised using linear initialisation with the first two principal components, which simply speeds up the time that is needed for the maps to become ordered. Batch learning was used, meaning that for each iteration the entire dataset is presented to the network before any codebook values are updated (this is in contrast with the sequential algorithm, which updates the weights after each observation). Batch learning is preferable to the sequential algorithm as it converges faster, does not require a time-dependent learning rate, and is "safer". Indeed, in pointing out these three reasons, (Kohonen, 2013, p. 53) stated, "It should be emphasized that only the batch-learning version of the SOM is recommendable for practical application..."

Maps were trained using a rough (or coarse) and a fine phase, such that the global topography is learnt during the rough phase, and the local dynamics are learnt during the fine phase. The Gaussian neighbourhood function in the rough phase commences with a value that was dependent on the width of the map: half the map width. The rough phase ended and the fine phase commenced with a neighbourhood radius of two, and the fine phase ended with a neighbourhood radius of 0.5. For Gaussian neighbourhoods, ".5 is a typical final value for small maps (and also the smallest value in general)" (Kohonen, 2014, p. 49). Neighbourhood effects decrease exponentially, so drop off rapidly as the neighbourhood radius approaches zero.

The number of iterations for a phase depends on the size of the map and the dataset that

is being learnt. I varied the number of iterations per phase, which affects how slowly the neighbourhood radius is decreased, finding that a value of 30 iterations for each phase was not too time consuming but yielded stable results (multiple runs would result in similar maps). One of the main investigations was how the size of the map affected the results. Indeed, for smaller maps, the number of iterations could be smaller, but since the amount of time to train smaller maps is negligible in comparison with the amount of time needed for larger maps, the number of iterations was set in regards to the largest maps. The smallest maps found in the literature are Pozzi et al. (2012) who used 10 unit maps (for simian vocalisations), while the largest is around 20x20. To test the effects of different map size, a series of map sizes were determined with the goal that each step in the series would approximately double the size of the map.

5.5 Visualisation

In Figure 4.21, the speech of the speaker was projected onto a map using a vote method that selects the most common label as the label. By observation, it is clear that a large portion of the map is devoted to silence, which is represented as "S". A similar pattern was also observed with other vote strategies. The test data is not labelled, which creates some difficulty in observing whether the map is a true representation of the labels. However, there are some clues that indicate that it may indeed be valid. The vast majority of data frames from the manually labelled speech that best match cells that have been labelled as silence are indeed silence. Additionally, when labelling unlabelled speech, by manual observation, the labelling of segments near the beginning and ends of files are consistently labelled as silence (the way that the data was collected, speech should occupy the central portion, so the beginning and end should be silent). Finally, the proportion of speech to silence in the files is very close to the proportion of cells devoted to speech and silence in the maps. As a result, I suggest that a map trained on both speech and silence must learn to classify silence. Although Grashey (2003) did find preliminary results indicating that VAD can be successfully accomplished by SOMs, I suggest that when the task is the evaluation of phonemes, dedicating large portions of the maps that used for classification or evaluation to silence may be counterproductive.

To avoid this problem, many KNPT researchers, following the example of Kohonen (1988), eliminate the issue of silence by relying on annotated speech corpora such as TIMIT, training and testing only on the speech portions. Others such as De Luna-Ortega et al. (2009) and Tashan et al. (2014) use VAD algorithms that are not SOM-based as a pre-processing step. Although this is a valid approach, I do think that an end-to-end KNPT approach that reduces reliance on such external algorithms appears to be an interesting avenue that has not been well addressed in the KNPT research. This is an important detail for researchers who may not be intimately familiar with the SOM and its application as KNPT, still an active area of research,

that may be overlooked but can have significant effects on the results. I further note that the question of whether the silence from the corpora was included in the analysis is omitted from a number of research articles in the KNPT area, making it more difficult to interpret the findings (particularly negative findings).

5.6 Analysis

The PER of a phoneme classification system may be calculated as the number of incorrectly identified phonemes divided by the total number of phonemes that were identified. For the purposes of vowel classification, the PER may be referred to as the vowel error rate (VER). The KNPT-based classifier presented in this chapter produces frame-by-frame classifications. In terms of VER for hVd words, each word is understood to contain only a single vowel (though a diphthong contains multiple phones). The performance for each utterance was analysed based on the most frequent vowel label for that file. An alternate method that some researchers used for calculating the VER was to locate the frame that is in the centre of a vowel and compare only the results of the central frame. The centre frame method was not used here because it is dependent on knowing where the vowel is in the utterance.

PER is calculated in regards to a gold standard; thus, the putative categories of the gold standard are reliant on the quality of the standard itself. The manual transcriptions were broad transcriptions, and as such, were closely associated with the prompts that were used to elicit the utterance (i.e. the goal of the annotators was the labelling of phonemes, not allophones or finer nuances of speech, so they did not transcribe or otherwise provide information on small differences in pronunciation). If the participant misinterpreted the prompt, a difference between the prompt and the speech uttered may have occurred. Although, for the Austalk corpus as a whole, I have located a small proportion of utterances that deviate from the prompts (such as an absence of speech or idle chat), it appears that this did not occur for any of the hVd utterances used here. I used a number of methods to identify possible deviations, including: each file length was consistent with a brief hVd word, and I listened to the files where any unusual output of the system were identified. Additionally, I inspected the manual annotations for the hVd utterances to determine that there were no annotations that deviated from the text prompts.

5.6.1 Statement

The following poster paper presentation is based on the hierarchical KNPT system with a focus on training maps on a group of speakers. These maps are then used to evaluate the speech of others, who were not part of the training set. As Kohonen (2001, p. 274) describes

this dilemma: "Interpretation of free speech from an arbitrary speaker, without prior speech samples given is impossible with the present techniques." In the previous chapter, speech from the same speaker was used for training and for testing. Here, then, we are seeking to reveal some of the techniques that may enable the evaluation of accent based on the speech of dialect groups. The publication that follows was prepared to share this work with the speech science community.

The system framework was developed by Tom Anderson in consultation with David Powers. This paper and an associated poster was written by Tom Anderson. Tom Anderson edited with proofreading and feedback from David Powers. The poster was presented by Tom Anderson at the 16th Speech Science and Technology Conference (SST2016) held at Western Sydney University.

5.7 Characterisation of speech diversity using self-organising maps

5.7.1 Authors

Anderson, T. A. F., and Powers, D. M. W.

5.7.2 Introduction

We report investigations into speaker classification of larger quantities of unlabelled speech data using small sets of manually phonemically annotated speech. The Kohonen speech typewriter [1] is a semi-supervised method comprised of self-organising maps (SOMs) that achieves low phoneme error rates. A SOM is a 2D array of cells that learn vector representations of the data based on neighbourhoods. In this paper, we report a method to evaluate pronunciation using multilevel SOMs with /hVd/ single syllable utterances for the study of vowels, following [2] (for Australian pronunciation).

5.7.3 Methods

We used 18 /hVd/ words from AusTalk, an Australian speech research corpus collected from speakers of Australian English [3]. Audio was converted to conventional 39-feature mel-frequency cepstral coefficients (MFCCs, log energy, and differential and acceleration calculated on 25 ms windows with 10 ms offsets). MATLAB scripts are available from the authors on request. Scripts for MFCCs were modified from mathworks.com/matlabcentral/fileexchange/32849-htk-mfcc-matlab.

To investigate the sensitivity of our phonemic model to dialect, the system was trained using unlabelled speech from one of three different groups, General Australian (21 speakers), educated Melbournian (21 aged 25-34y), or speakers of Chinese (14), and labelled using a subset of manually annotated data (phonemes and start/end points) from three further speakers, as follows: a base 25x25 unit SOMs was trained using all audio windows from the unlabelled data; three-quarters of the annotated data was assigned to the units in the base map; the resulting labelled map was used for separating the unlabelled input data for submap boosting. Two approaches to submapping, each with three 20x20 submap SOMs, were trained on data segmented by the base map by: (1) three maps for subsets of vowels with shared confusion in the base map (Kohonen's method); or (2) three maps of either /h/, vowels, or /d/ (linguistic).

5.7.4 Results

Frequently confused phonemes (on the first map) were identified using a greedy method based on high off-diagonal values in the confusion matrix from the base map:

- Vowel Group 1: /ɔɪ/ (hoyd) and /oː/ (horde)
- Vowel Group 2: /e/ (head), /eː/ (haired), /ɜː/ (herd), /ɪ/ (hid), and /ɪə/ (heared)
- Vowel Group 3: /ɐ/ (hud), /æɔ/ (howd), /æɪ/ (hade), /ɐː/ (hard), /ɑe/ (hide), /ɔ/ (hod), and /əʉ/ (hode)

The vowel error rate, shown in Table 5.2, was calculated by comparing the most frequent vowel label outputted for an audio file with the vowel noted by manual annotators. Our method of using initial, vowel, or final for submaps was observed to be slightly better than Kohonen's method of using submaps to disambiguate confused phonemes, and particularly for Chinese, marginally significant.

Table 5.2: Vowel Error Rate for single (25x25 classifier), or multi-SOMs trained on confused vowels or initial, vowel, or final (/h/, V, or /d/) (standard error in parentheses) based on 10 CV over word tokens.

Map Type	General	Melbournian	Mandarin
Single	0.487(0.062)	0.600(0.025)	0.506(0.041)
Vowels	0.059(0.021)	0.047(0.019)	0.090(0.016)
h/V/d	0.057(0.014)	0.042(0.022)	0.027(0.015)

5.7.5 Discussion and Conclusion

The goal of the experiments conducted in this work was to explore the general capacity of labelled data for revealing characteristics of dialect in the unlabelled speech of Chinese-background Australians and educated Melbournians vs Australians generally. The results are competitive with the original phonetic typewriter results, demonstrating effectiveness with 39-feature MFCC vectors. There is the possibility for improving on these using context, but for the future directions of this research, we are more interested in characterising differences between the three groups.

5.7.6 Acknowledgements

The AusTalk corpus was collected as part of the BigASC project, funded by the Australian Research Council (LE100100211). See https://austalk.edu.au/bibliography.html for details.

5.8 Discussion

Due to the brief nature of the SST paper in the previous section, a number of points of discussion were omitted.

Observe that Vowel Group 1, which had the most mutual confusion and least confusion with other vowels, contains the sounds: /oɪ/ and /oː/. In Australian English, the "r" in "horde" is not pronounced, so the two vowels appear to have very similar sounds, differing only in the lengthening, which makes the sound appear longer. Here, the information is being processed at the frame level, so if lengthening is indicated only as more frames, then it would not be detectable without an additional procedure such as counting the number of frames; however, if there are pronunciation differences at different parts of the vowel, such as a different timbre of voice at the end of a lengthened vowel, then we could expect to be able to differentiate these vowels without relying on counting the number of frames or ms used in the pronunciation of the vowel itself (noting that speech rates can vary quite considerably to reduce the reliability of such an approach).

Vowel Group 2 also shows a grouping of lengthened vowels (/e/ and /eː/), but this group is not as pure as group 1 (it also includes /ɜː/, /ɪ/, and /ɪə/).

Finally, Vowel Group 3 contains two obvious categories. The monophthongs /ɐ/ and /ɐː/ appear next to each other, differing only slightly and one is lengthened. Additionally, it includes four diphthongs, three of which have overlapping trajectories near the location of HUD/HARD.

As a result, I found that these two categories of phoneme with the addition of /ɔ/ were commonly confused, so as a result applied an auxiliary map for their disambiguation in the spirit of Kohonen's phonetic categories.

The main result here is that the single 25x25 map was much worse at classification than either of the multi-level maps. We developed a method of that did not rely on auxiliary maps for confused phonemes, but rather kept all vowels together in the same map. This method was able to perform marginally better than the auxiliary map approach followed by Kohonen and others. Additionally, it has the advantage of keeping all the vowels together on the same map for the purposes of visualisation. The use of these maps for teaching language is explored in Chapter 9.

5.8.1 Future work

Here I present proposals for several practical directions for future research. These include improving the MATLAB pipeline, incorporating larger speech datasets, and drawing comparisons with competing methods for phoneme transcription.

The systems that I implemented for the cleaning and management of speech data are described in Appendix A. In terms of enhancing the capabilities of MATLAB, Mathworks has created bindings for Python. As Alveo has Python API, one direction for the future to enhance research efforts would be to take advantage of those bindings in my data management. Currently my data flow is as follows: data and metadata are downloaded through Python, the metadata is converted into a MATLAB data structure and the data is loaded from the file structure. By calling the Alveo Python API, the procedures of downloading the data and metadata, and storing them as a data structure, could be accomplished from inside MATLAB. I believe that this would significantly improve the data management pipeline, which would benefit researchers who use MATLAB and are interested in accessing Austalk or other data repositories from Alveo, such as AVOZES or GCAusE, among others. Currently, my scripts reduce the manual workload required, but tapping into the Python scripts in MATLAB would further reduce the amount of time necessary for accessing, preparing and storing the data.

Future research may use data from more Austalk participants as it become available through Alveo. During the course of my research, the Alveo database did not have the entirety of the Austalk corpus online because it was still a work in progress. To work around this absence of data, the groups that I used were those with speech files available. However, the Chinese group in particular had a number of speakers that were not used in my study, because they were not available at the time. As speech from those speakers who were omitted becomes available on Alveo, it would be interesting to improve the quality of research by including them. Additionally, some groups initially of interest, such as the participants recorded in Sydney, were not available in sufficient numbers due to the data ingress issue, so were left for

future research once that data is made available. Finally, once the techniques are refined, the speech of participants from smaller groups, such as speakers from smaller areas or those who speak European languages, could be investigated to reveal patterns of accent.

The transcription or annotation of speech in Austalk has been performed by professionals and by automatic processes with manual verification. Updates to Alveo in May 2017 enabled attaching those types of associated data (particularly phonetic annotations) to the speech files[4]. In relation to this new data source, an area of future research would therefore be to make comparison with those annotations with those made by my KNPT system. This would enable locating potentially problematic areas in those phonetic annotations, and also improved tuning of the KNPT itself.

Another area of interest is to learn features from the video, which was recorded using a stereo camera and therefore allows for extraction of 3D information. Although the production of phonemes has a wide degree of individuality, there are also some basic principles that underlie the mechanics of how people speak. In order to provide feedback about the speaking, it is intended that the system would first learn to understand the individual characteristics, but then based on the accent maps, be able to provide information that could reveal something about basic anatomical changes such as posture or tongue movements. For the language learner, the goal would be to identify those changes that would make them more intelligible. For the learner who is seeking to develop a particular accent for the stage, the goal might be more fine tuning. In any case, by taking into account features extracted from video as well as those from audio, I believe that future research would enable better feedback. The drawback with video processing is that the files are much larger, and although image processing can provide specific information about some of the articulators (the jaw and lips are visible), other aspects, like the tongue, are not revealed by an external camera.

5.8.2 Identification of specific points of accents

The research that was presented in this chapter might be applied in a language learning context. An extension of this research to language learning is how the accent of one person can be compared with the accents of others. The goal of one-to-one pronunciation instruction is that the lessons are customised to the specific needs of the learner rather than a one-size-fits-all approach that must be usable by most learners.

Firstly, it would be of interest to see which of the phonemes are pronounced most differently so that pronunciation of those phonemes could be modified via form focused instruction. In this way, the segmental accuracy of the learners could be improved. Visual feedback might be provided so that the learner could find out what they could do to pronounce a specific utterance

[4] http://alveo.edu.au/2017/05/23/austalk-updates

more accurately in the target accent. Additionally, it would be of interest to see whether there were some more general speaking habits that can be identified through the KNPT paradigm. General differences in pronunciation between groups of phonemes can be identified, so that more basic tongue or jaw postures, mannerisms, or tendencies could be addressed. For example, the degree of lip rounding for certain vowels may affect the pronunciation. The attention of the learner could be directed to form a more characteristic shape of the lips in order to address this aspect. A video camera trained on the learner can be used to reveal certain of these aspects (mouth openness), although the interior tongue movements of frontness and backness that are tightly connected to vowel production are less evident through video because of frequent occlusion of lips and teeth. As I learnt in a workshop on speech technology, the internal activity can be revealed using a technique such as ultrasound on the neck and chin. Although these technologies are not currently available or affordable for home or school users, application of KNPT to speech could prove practical in more clinical applications.

Another area of interest is investigations into better ways to provide learners feedback about their pronunciation as it related to the target accent so that they can gradually, through self-directed learning, achieve an awareness of their accent and particularly any shortcomings that may reduce the intelligibility or comprehensibility of their speech to a listener. Similar to the learning mechanism that enables improvements in pronunciation to be achieved through listening to perceive differences in minimal pairs[5], accent may be shifted by improving the ability of learners to perceive how their utterances relate to the target accent. In doing so, it may be suitable to select for submaps those commonly confused on a linguistic basis. For instance, it has been observed that English learners from a Mandarin language background have less ability to differentiate front vowels such as /e/, /æ/ and /ei/), and may substitute one phoneme for another. In contrast, the over-nasalisation of English vowels is also a common phenomenon among speakers from China. Such a tendency might not be revealed in a system that is trained only on native speakers, so speech corpora of learners of English could be useful training material.

Indeed, as Derwing et al. (1998) found via experiment, although instruction for general speaking habits and instruction for segmental accuracy improve comprehensibility and accentedness for reading sentences, instruction for general speaking habits improves comprehensibility and fluency for spoken narratives that is superior to a focus on segmental accuracy (or no instruction at all). In a Master's thesis that outlined a prototype Vowel Pronunciation Training System, Hecker (2006) explored the concept of KNPT for foreign accented speech, revealing that the unsupervised training of a SOM would result in a topological structure that held similarities with physical characteristics of speech such as lip rounding and tongue occlusion.

[5]My opinion on this form of instruction for accent improvement is that minimal pairs can provide a tractable starting point, but any real change in accent will be brought about only through the modification of the relationships between and within vowel categories. Thus that awareness may be served by learning how each specific sound relates to all the others in the target accent.

Hecker also proposed the similarity of a vowel could be observed by comparing the MFCCs of an utterance with a prototypical one, thus providing a mechanism to measure differences between accents, but did not implement this in his prototype to show that it could work. Similarly, in a Master's thesis that advanced the concept of using SOMs for dialects, Nti (2009) presented the idea of using KNPT for visualising dialects, in particular identifying ACCDIST Huckvale (2004) as metric of interest for this purpose. The main conclusion was that it was found to perform less than adequately for both k-means and SOMs; therefore, the improvement of the visualisation was left for future research.

What these forays into the research niche of accent with SOMs establish, I would argue, is that although the potentials of SOMs are recognised for multiple tasks (such as identifying accent differences and visualisation) there are several complications that may not be immediately apprehended until one is experienced in the craft. In particular, ACCDIST is a comparative measure that makes a distance matrix that describes how all the vowels are related for each speaker. This process enables comparisons between speakers, but it also reduces the size of the dataset to a single array for each speaker. In order to create maps of significant size, the KNPT approach needs many observations, but in averaging, ACCDIST effectively reduces the number of observations to speaker (or even further, to a single observation per group).

5.9 Summary

In this chapter, I evaluated a hierarchical approach to unsupervised KNPT for the evaluation of Australian speech. This approach allows the application of patterns learnt from phonemes via unsupervised learning for classification and clustering. The hierarchical approach was compared with a single layer approach. The results demonstrated that multi-level maps were superior to single maps in terms of VER, but that the method of selecting submaps is also an important concern. I found that linguistic category was better for this purpose than disambiguating confused phonemes. The aspects of normalisation and MFCC feature selection were addressed to elucidate concerns necessary to constructing such a system. I reviewed a specific implementation of unsupervised KNPT for language identification and explained its characteristics through experiments. I furthermore touched upon several pre-processing and parameterisation concerns that should be relevant for others seeking to use similar approaches.

Chapter 6

KNPT Voice Activity Detection

The research conducted for this section was conducted to get a better idea of the ability of the networks to differentiate speech and silence. As I described in the KNPT literature review in section 3.5, the vast majority of research used pre-segmented speech data, such that no silence was presented to the network in training or testing. This is attained by manual annotation or via an algorithm for separating speech from voice. In fields such as ASR, this problem of isolating speech is referred to as VAD. Although many algorithms exist, e.g. the ITU-T G.729 VAD used in mobile telephony, VAD is still a field of ongoing research.

Many KNPT researchers thus sidestep this issue by using a heuristic or manual segmentation to determine speech from silence. VAD is an important consideration for phone networks, because it may save a great deal of bandwidth if portions of a phone call when a person is not talking can be omitted from transmission. Nonetheless, it seemed reasonable in the initial conception of the research, the philosophy was to use VAD as part of the system, thus that a separate method (using a separate VAD algorithm) would not be used for isolating speech from non-speech in our inputs.

One important question to justify the use of KNPT on audio that includes silence as well as speech is whether it is justified to use such input. In other words, can KNPT operate properly when part of the problem is to resolve the speech versus silence? From a psycholinguistic viewpoint, can produce only a limited range of vocalisations due to our speech apparatus. Similarly, we can hear and perceive only a limited range of sounds. It would follow that human perception and thus human memory is tuned to the sounds of speech. As studies such as Humphries et al. (2010) demonstrate that tonotopic maps are present in the brain, with gradients, then a possible idea is that topographic maps can probably approximate in relatively low dimensionality those sounds such that the different phonemic classes may be distinguished. Phonemes in the midst of non-speech sounds can be identified without consciously distinguishing between sounds that are like voices and sounds that are not.

VAD is a well known textbook problem. Rabiner & Schafer (2011). However, in reviewing

the KNPT literature for voice activity, it was revealed that only one author (Grashey, 2003) reported using a SOM for VAD. Both speech and silence were used as inputs in my technique for a significant amount of time; therefore, it is worthwhile to analyse whether a SOM can perform VAD.

In the KNPT literature, it is standard practice to use a VAD step (not based on SOM). Commonly, KNPT applications do not operate on speech files directly, i.e. raw audio. Audio files are necessarily preprocessed, since a Euclidean distance measure does not make much sense when comparing raw audio. An audio file may be normalised so that the volume is not too quiet and not too loud. The speech is converted to a more usable feature space, such as MFCCs and its associated derivatives. But another thing that most researchers do is to segment the files into sections of speech, and only train on the speech. This means that portions of the speech files that contain silence are ignored.

In KNPT applications, removing silence frames by another method may be performed prior to feature normalisation, such as was performed in (Alam et al., 2011). However, the ability of the SOM to perform VAD remains largely unexplored. An alternative approach, which was the first one that I used in exploring KNPT systems, would be to take an audio file, convert it into feature space, for example filterbank energies or MFCCs, then to classify the speech directly using SOMs. This differs from the approach of VAD preprocessing because the detection of speech activity is performed by SOMs rather than a separate algorithm for distinguishing between speech and silence. However, as a consequence of this decision, there is the potential that the silence, being retained in the signal, will adversely affect our results. To verify that the approach, which retains silence, works for SOMs, the explorations presented in this section were conducted.

I additionally seek to go beyond the classification problem to visualise the topological nature of speech in context, where it exists with silence, for different feature spaces. Note that here for vowel analysis, speech may be considered to consist of vowels without consonants. Therefore the work to follow may be biased towards vowel-based VAD rather than the more general voice activity VADidentify all types of speech. However, future work should explore the differences.

6.1 The significance of silence in the evaluation of speech

The sounds of speech are many and varied, but there are even more different sounds that are not speech. Learning to classify speech using a SOM, or any classification method, is most often accomplished by first identifying speech, isolating it from the non-speech portions of the audio. As discovered in the literature, there is a second approach using SOMs. Some researchers do not distinguish between the speech and non-speech portions in the training of their systems. Subsequent labelling may provide the mechanism for determining where the speech begins and

ends. In this type of approach, silence is treated as another feature to be detected by the system.

The main disadvantage to training a map with both speech and silence is that the silence portions of the audio may not be sufficiently distinguished from the speech portions on the basis of the metric that is used. As silence has different statistical properties than speech, normalisation can reduce this issue. A second disadvantage is that normalisation becomes dependent on the length of the audio. For example, if we have a corpus in which each word was approximately one second in duration but the length of each file was between two seconds and sixteen seconds, with the word appearing somewhere in the middle of the file. Indeed, many canonical word corpora, in which single words are spoken clearly, are of this form. In such a case, if the normalisation was to be conducted at the file level, it would be expected that silence will dominate some normalisation calculations and not others.

Another approach to normalisation might be to normalise across the entire training set. However, it may be common to compare multiple training sets, with different normalisation factors for each. Normalisation might normally assist in controlling for aspects such as microphone placement that increase the overall sound level of the recording without significantly affecting the evaluation of the pronunciation. Without knowledge of how long each utterance was in proportion to the amount of silence, there may arise some bias from such a normalisation procedure. Additionally, set normalisation is not well-suited to real-time applications where the entire test set has not yet been collected (although there are many).

I conducted some experiments to investigate different approaches to normalisation, normalising at the file level, the set level[1], not normalising, and significant differences were observed, particularly for smaller maps. Learning the statistical properties of the silence portions of the corpora requires devoting some of the cells of the map to silence portions. I found that for learning the phonemes of speech, there is a relationship between the size of the map and the normalisation granularity.

Thus, if patterns of non-speech and speech are both to be learnt, the non-speech portions of the audio may require significant area of the maps. The number of cells that are most frequently associated with non-speech may vary by the characteristics of the non-speech portions. This is less likely for carefully controlled corpora, but generally, this phenomenon makes it more difficult to make evaluations of speech, which is the real goal here.

6.2 Methods

Vowels were recorded by a single speaker, each containing one second of silence and three seconds of vowel, plus silence at the end. It was determined that because MFCC is well known

[1]Grashey's VAD research used set normalisation.

Listing 6.1: Parameters used

```
% 10 ms overlapping windows (100 windows per second)
parameters.windows_per_second = 100;
% files are a bit over 16 seconds in length
parameters.max_file_length = 25; % no file is over 25 seconds in length
% SOM parameters
parameters.SOM_width = 25; % 50 data points per node would be approximately
    an 36x36 map.
```

as a common method of representing audio for ASR, we would represent the data as MFCCs. Additionally, delta and delta-delta features are common. Additionally, the first coefficient is basically a measure of energy of the signal, but it has been found that the sum of log-energies is actually a more informative. This means that the total number features for our system are 39. The parameters for these experiments were set as in MATLAB Code 6.1.

The audio files were loaded and converted to 13-feature and 39-feature MFCC format. Note that in recording these files, only the vowels (and "r" closures) were produced, no consonants. Post-vocalic "r" was included because the word list is similar to Austalk, the Australian speech corpus, but this speaker used American English. Data was loaded using the procedures shown in MATLAB Code 6.2.

Files were then split into training and test sets to enable cross fold validation procedures. Splitting the data was achieved using procedures like those that are shown in Appendix B.2.

The SOMs for each form of normalisation were hexagonal sheets of the same size. The procedures for performing linear initialisation of maps using different normalisations of training data are shown in MATLAB Code 6.3. The wrapper methods ta_som_normalize and ta_som_normalize_sMap will be used later when normalising the testing set is necessary. This permits the testing of different normalisations by simply switching the method name and sData.

The values of mean and standard deviation used in the normalisations are retained in the corresponding sMap, which enables the testing set data to be normalised using the mean and standard deviation of the training set. The method ta_som_normalize_sMap is used to normalise sData using the values that were used for normalising sMap (here, to unit variance).

Listing 6.2: Loading and preprocessing speech files

```
tic
% Load speech audio files and process
variable_names = {'folder', 'fieldname', 'word', 'MFCC', 'sData_word', '
    textgrid'};

rcoloured = {'cute', 'arm', 'hort', 'hurt', 'mare', 'pure', 'tour'};
monophthongs = {'heat', 'hit', 'hate', 'het', 'hat', 'hot', 'hood', 'hoot', '
    hut', 'hurt', 'hoed'};
diphthongs = {'hide', 'now', 'hoit'};
words = horzcat(monophthongs, diphthongs, rcoloured);
data_table = cell2table(cell(0,size(variable_names, 2)), 'VariableNames',
    variable_names);

for word = words
  for number = 1:4
    for condition = {'V', 'VG'}
      folder = sprintf('vowels %s', condition{:});

      % fieldname format is '<word>_x<condition>_<number>.wav'
      fieldname = sprintf('%s %s %d', word{:}, condition{:}, number);

      % filename format is '<word> x<number_of_repetitions> <number>.wav'
      filename = sprintf('G:/Dropbox/datasets/kinect_audio/%s/%s %s %d.wav',
    folder, word{:}, condition{:}, number);

      % textgrid_filename format is '<word>_x<number_of_repetitions>_<number
    >.TextGrid'
      textgrid_filename = sprintf('G:/Dropbox/datasets/kinect_audio/%s/%s_%s_
    %d.TextGrid', folder, word{:}, condition{:}, number);
      textgrid = ta_praat_textgrid(textgrid_filename);

      % read wav file
      [wav_file, Fs] = audioread(filename);

      % use only one microphone as the input (Kinect has 4 channels)
      wav_file = wav_file(:, 1);

      % downsample to 16 kHz, consistent with other datasets
      wav_file = resample(wav_file, 16000, Fs);

      % run the MFCC algorithm using filterbank_num and retaining 13 MFCCs
      [MFCC39] = ta_mfcc(wav_file)';

      data_table = [ data_table
              table( ...
                string({folder}), string({fieldname}), ...
                string({word}), {'MFCC39'}, ...
                som_normalize(som_data_struct('MFCC39'), 'var'), ...
                textgrid, 'VariableNames', variable_names) ];
    end
  end
end, toc % Elapsed time is 12.968220 seconds.
```

Listing 6.3: Normalisation of the training data

```
% Normalise sData to unit variance
ta_som_normalize = @(sData) som_normalize(sData, 'var');

% Normalise sData using the values that were used for sMap
ta_som_normalize_sMap = @(sData, sMap) som_normalize(sData, sMap);

ta_som_linit = @(sData, parameters) som_lininit(sData, ...
  'msize', [parameters.SOM_width parameters.SOM_width], ... % size of the map
  'lattice', 'hexa', ... % hexagonal map
  'shape', 'sheet'); % sheet shape

%% Linear initialisation with on normalised data
% No normalisation
sMap_inited_no = ta_som_linit(sData_no, parameters);

% Word-level normalisation
sMap_inited_word = som_lininit(sData_word, parameters);

% Set-level normalisation
sMap_inited_set = som_lininit(sData_set, parameters);
```

6.2.1 Types of normalisation

In the experiments for VAD presented in this section, three different types of normalisation were compared, as follows.

1. No normalisation
2. Word normalisation - Normalisation is conducted such that each word has unit variance. This means that all input files in the training set and test set are normalised individually without respect to one another.
3. Set normalisation - Normalisation is conducted to create unit variance for the training set and unit variance for the test set. The variance for the entire training set is determined, and the training set is scaled to unit variance. Normalisation affects not only the mapping of the input data onto the SOM, but also the mapping of the test data onto the SOM. The same scaling used to create unit variance for the training set was used to normalise the test inputs.

The application of set normalisation to the test set has two main interpretations. The training set is first normalised using the statistics of the training set (the mean and standard deviation). One method of normalisation of the testing set is to retain the normalisation values from the training set and use them to shift and scale the testing set. Alternately, the test set can be normalised using its own mean and standard deviation, which might be considered because the training set and testing set are different. Results from both these methods are

reported. Other combinations of normalisations were tested informally, such as normalising the training set at the word level and the testing set at the set level, or one set being normalised with the other not, but these types of combinations lead to much worse results, as would be expected.

An different method of normalisation is to resize vectors into the range $[0 - 1]$. Further details on range normalisation are described in section 3.8.3. In the KNPT literature, only three authors (Dalsgaard, 1992; Eng, 2006; Guenther & Gjaja, 1996) reported using range normalisation.

I did not find good results for range normalisation on this data set, so the experiments I report for VAD are focussed on unit variance normalisation. Range normalisation might be expected to degrade performance slightly, because the log-scaled ranges of the coefficients resulting from the MFCC algorithm allow for comparison by design, but it might depend on the type of normalisation (word or set) that is used.

The method for training the SOM is shown in MATLAB Code B.3.

6.2.2 Labelling without annotations

As described in previous sections, speech was recorded in sets of four vowels, three seconds long each, with one second between each vowel. Therefore, we know the basic structure of the audio files. However, these times are approximate and subject to human variability, since it was a human producing the sounds. Using this as a guideline for rough labelling, we will observe how well the maps distinguish speech and silence. Due to the way that the speech was recorded, with one second of silence followed by three seconds of vowel, repeated three times, I have good confidence that the beginning of the speech happened within 250 ms of the intended start time and completed within 250 ms of the intended end time. Similarly, the silence endpoints should be within 250 ms of the start and end times. Although occasionally a sound might stop or start a little before or after the strong confidence window, it will almost always occur within the widened speech windows. The idea is thus to use the rough idea of start and stop times to label the maps. The labels apply to both the training and testing sets, but because we are interested in unsupervised training, only the labels from the training set are applied.

The procedures for determining labels are presented in MATLAB Code 6.4. The procedures that were used for labelling the maps are shown in MATLAB Code B.5. The labelled training data is run through the maps. Each of these labels are associated with the BMU for each data frame. The collection of labels associated with each node are then determined by a vote procedure. Maps for the various normalisations procedures are displayed using the procedures shown in Appendix B.4. All results are reported at the frame level.

Listing 6.4: Prepare rough labels

```
%% Create a labelled array of rough labels for speech/silence windows
speech = array2table(parameters.windows_per_second * [[1.25, 3.75];[5.25,
    7.75];[9.25, 11.75];[13.25, 15.75]],'VariableNames', {'start','stop'});
silence1 = array2table(parameters.windows_per_second * [1/parameters.
    windows_per_second, 0.25], 'VariableNames', {'start', 'stop'}); % first
silence = array2table(parameters.windows_per_second * [[0.25, 0.75];[4.25,
    4.75];[8.25, 8.75]; [12.25, 12.75]], 'VariableNames', {'start','stop'});
transitions1 = table(speech.stop, speech.stop + .5 * parameters.
    windows_per_second, 'VariableNames', {'start', 'stop'});
transitions2 = table(silence.stop, silence.stop + .5 * parameters.
    windows_per_second, 'VariableNames', {'start', 'stop'});
my_tables = {speech, silence1, silence, transitions1, transitions2};
labels = cell(file_length * window_length, 1); % cellstr of labels.. one hot
    vector for speech, silence, transition
label = {'V', 'S', 'S', 'T', 'T'}; % vowel, silence, transition, unlabelled
indices = cell(3,1);
labels = cell(parameters.max_file_length * parameters.windows_per_second, 1);
labels(:) = {'S'};

for index = 1:size(my_tables, 2)
  current_table = my_tables{index};
  indices{index} = rowfun(@colon, current_table);
  indices{index} = horzcat(indices{index}{:,:});
  indices{index} = reshape(indices{index}, 1, []);
  labels(indices{index}, 1) = label(index);
end
```

6.3 Results

Table 6.1 shows classification results for different map sizes. No normalisation was used in this experiment. Classification is performed at the frame level. The largest values are indicated in bold, along with the other values that are not significantly different.

The classification of silence is achieved above a 0.80 bookmaker informedness level for 3x3 maps with both the MFCC13 and the MFCC39 feature representations. The highest informedness for MFCC13 is reached with an 18x18 map. The highest informedness for MFCC39 is reached with an 9x9 map.

It can be observed that the bookmaker scores are higher for the MFCC13 feature representation than for MFCC39. The difference is statistically significant, but with sufficient map sizes, classification is performed at around 0.98 or 0.97 bookmaker informedness values.

In Table 6.1, despite recall and precision scores of 1.000 and 0.7637 for a 2x2 map with MFCC13 feature representation, the bookmaker informedness score was 0.00. The reason that this occurs can be seen in Table 6.2, which shows a contingency matrix for the results of classification of one of the 2x2 maps with no normalisation. Every node in the map has been labelled as a vowel, so it classifies every frame in the test set as a vowel. As a result, the map achieves a perfect recall score, as every vowel was correctly identified as a vowel. Furthermore, as vowels are approximately $\frac{3}{4}$ of the map, the precision scores for this size map are also near this proportion.

Table 6.2: Classification of vowel and silence by a 2x2 map with no normalisation.

		Actual	
		Vowel	Silence
Predicted	Vowel	1309	401
	Silence	0	0

6.3.1 Normalisations for VAD

The classification results shown in Table 6.3 are shown for normalisation that was performed at the utterance level. Subsequent to putting the utterance in the MFCC13 or MFCC39 feature representation, the values of each utterance were standardised to unit variance. Classification performance is above 0.95 bookmaker informedness for all maps with the MFCC13 representation. Classification performance of maps trained with MFCC13s was better than for MFCC39s except for the largest 50x50 maps, when they were slightly higher for MFCC39s.

In the classification results shown in Table 6.4, the normalisation was performed at the utterance level. Subsequent to representing the utterance as MFCC13 or MFCC39 features, the values of each utterance were standardised to unit variance. Maps 3x3 and above were

Table 6.1: Mean percentage (standard error) for varying map sizes using MFCC13 or MFCC39 features with no normalisation.

	MFCC13				MFCC39			
Size	Recall	Precision	Markedness	Bookmaker	Recall	Precision	Markedness	Bookmaker
2x2	**1.00(0.00)**	0.764(0.0044)	0.764(0.0044)	0.000(0.00)	**1.00(0.00)**	0.761(0.0058)	0.762(0.0058)	0.00(0.00)
3x3	0.999(0.00035)	0.964(0.0052)	0.960(0.0051)	0.889(0.014)	0.998(0.00034)	0.970(0.0029)	0.962(0.0033)	0.898(0.0088)
6x6	0.998(0.00048)	0.987(0.0030)	0.980(0.0033)	0.955(0.0093)	0.995(0.00062)	0.991(0.00079)	0.974(0.0022)	0.964(0.0025)
9x9	0.997(0.00091)	0.995(0.00037)	0.985(0.0028)	0.982(0.0012)	0.994(0.00063)	**0.994(0.00049)**	**0.975(0.0017)**	**0.973(0.0016)**
12x12	0.998(0.00035)	0.995(0.0011)	0.990(0.0018)	0.983(0.0043)	0.995(0.00072)	0.992(0.00074)	**0.975(0.0029)**	0.968(0.0030)
18x18	0.999(0.00055)	**0.997(0.00037)**	**0.994(0.0015)**	**0.989(0.0014)**	0.994(0.00053)	**0.993(0.00086)**	**0.974(0.0014)**	**0.972(0.0027)**
25x25	**1.00(0.00019)**	**0.997(0.00046)**	**0.995(0.00052)**	0.988(0.0017)	0.994(0.00039)	**0.993(0.00076)**	**0.975(0.0015)**	**0.972(0.0027)**
36x36	0.999(0.00036)	**0.997(0.0005)**	0.993(0.0011)	**0.989(0.0017)**	0.993(0.00076)	**0.993(0.00076)**	0.972(0.0018)	0.971(0.0023)
50x50	0.999(0.00036)	**0.998(0.00044)**	**0.994(0.00075)**	**0.990(0.0016)**	0.993(0.00068)	0.992(0.00085)	0.972(0.0021)	0.969(0.0029)

Table 6.3: Mean percentage for varying map sizes using MFCC13 or MFCC39 features with normalisation performed at the utterance level for all utterances in the training set and the testing set.

	MFCC13				MFCC39			
Size	Recall	Precision	Markedness	Bookmaker	Recall	Precision	Markedness	Bookmaker
2x2	0.9742	0.9922	0.9149	0.9502	0.9735	0.9912	0.9102	0.9462
3x3	0.9905	0.9915	0.9501	0.9516	0.9892	0.9769	0.9406	0.9147
6x6	0.9903	0.9942	0.9631	0.9712	**0.9900**	0.9728	0.9323	0.8839
9x9	**0.9928**	0.9938	0.9673	0.9697	0.9812	0.9854	0.9243	0.9343
12x12	0.9901	0.9923	0.9586	0.9634	0.9829	0.9892	0.9326	0.9468
18x18	0.9928	0.9939	**0.9688**	0.9713	0.9788	0.9903	0.9208	0.9461
25x25	0.9905	0.9941	0.9627	0.9710	0.9832	0.9908	0.9337	0.9518
36x36	0.9918	0.9942	0.9682	**0.9730**	0.9849	0.9954	0.9460	0.9695
50x50	0.9911	**0.9943**	0.9646	0.9717	0.9853	**0.9967**	**0.9472**	**0.9735**

able to achieve a high level of VAD for both feature representations with this normalisation procedure.

Table 6.4: Mean percentage for varying map sizes using MFCC13 or MFCC39 features with normalisation of the training and testing sets based on the statistics of the training set.

	MFCC13				MFCC39			
Size	Recall	Precision	Markedness	Bookmaker	Recall	Precision	Markedness	Bookmaker
2x2	**0.9990**	0.8802	0.8769	0.5313	**1.0000**	0.7611	0.7617	0.0000
3x3	0.9980	0.9744	0.9672	0.9137	0.9975	0.9659	**0.9569**	0.8798
6x6	0.9949	0.9896	0.9723	0.9597	0.9869	0.9894	0.9463	0.9524
9x9	0.9932	0.9925	0.9700	0.9683	0.9840	**0.9957**	0.9445	**0.9702**
12x12	0.9956	0.9913	0.9769	0.9662	0.9849	0.9934	0.9440	0.9629
18x18	0.9974	0.9925	**0.9840**	0.9725	0.9796	0.9914	0.9246	0.9506
25x25	0.9974	0.9918	0.9830	0.9687	0.9784	0.9938	0.9235	0.9574
36x36	0.9957	**0.9928**	0.9809	0.9733	0.9798	0.9947	0.9290	0.9619
50x50	0.9963	0.9927	0.9834	**0.9747**	0.9800	0.9941	0.9269	0.9596

Table 6.5 reports the result of a mixed normalisation procedure that I performed to explore some different options. The training set was normalised to unit variance using the mean and variance of the entire training set. However, each utterance in the test set was normalised using its statistics. It can be seen that MFCC39 outperforms MFCC13 in this scenario.

In Tables 6.3 to 6.5, compared classification performance with different normalisations. The results were that the maps trained and tested with unnormalised MFCC13 or MFCC39 performed significantly better than any form of normalisation. However, normalisation using training set statistics or normalisation at the utterance level was only a percentage point or two lower than the unnormalised classification performance, which for practical purposes may be considered as negligible.

Table 6.5: Mean percentage for varying map sizes using MFCC13 or MFCC39 features with normalisation of the training set based on the statistics of the entire set, and normalisation of the testing set based on the statistics of each utterance.

	MFCC13				MFCC39			
Size	Recall	Precision	Markedness	Bookmaker	Recall	Precision	Markedness	Bookmaker
2x2	0.9910	0.8965	**0.8682**	**0.5785**	**1.0000**	0.7611	0.7617	0.0000
3x3	0.9912	0.7866	0.6062	0.1302	0.9872	0.8444	0.7253	0.3743
6x6	0.9865	0.8849	0.8004	0.5401	0.9680	0.9037	0.7475	0.5976
9x9	0.9882	0.8199	0.6969	0.2781	0.9515	0.9452	0.7645	0.7506
12x12	0.9898	0.8845	0.6898	0.2950	0.9438	0.9497	**0.7661**	0.7764
18x18	**0.9926**	0.8894	0.7478	0.3530	0.9211	0.9329	0.6701	0.6898
25x25	0.9921	**0.9205**	0.7752	0.4410	0.9311	0.9522	0.7254	0.7637
36x36	0.9890	0.9119	0.7443	0.4111	0.9371	0.9555	0.7495	0.7846
50x50	0.9877	0.9188	0.7789	0.4837	0.9359	**0.9607**	0.7427	**0.7918**

In Figure 6.1, an example of the VAD labelling is presented. In the labelled set, transition periods were used to indicate the general start or stop of speech within 500 ms. Even without smoothing, the frame-by-frame classification performance has captured the speech portions almost entirely (hence the 0.989 bookmaker informedness for this map size). These results show that although the exact start or stop of speech was never provided to the system, it has learnt it based on the 500 ms transition window.

6.4 Discussion

The results of these experiments on the task of VAD with KNPT demonstrate that VAD can be achieved to a high level even with small maps around 3x3. The unnormalised maps achieved the best classification results, probably because normalisation boosted coefficients with less information that was relevant to the detection of speech. As the results were reported at the frame level, this means that using techniques such as smoothing can increase the performance.

The results that I have shown in this chapter are consistent with the work of Grashey (2003), which was the only KNPT research that I have seen suggesting that SOMs could be used for VAD. Every other researcher seems to have used a subset of a dataset that did not contain silence, used an external VAD algorithm, or ignored the problem of silence in the dataset. Grashey tested the MFCC12 feature representation with 25x25 maps. My results here expand on these results to demonstrate that VAD can be achieved with a variety of map sizes.

Additionally, Grashey did not compare different normalisation procedures. Although my results found that unnormalised maps performed the best, it was only marginally better than two of the other normalisation procedures. This result can allow KNPT practitioners to understand their results in the context of the more general speech versus silence question. Many

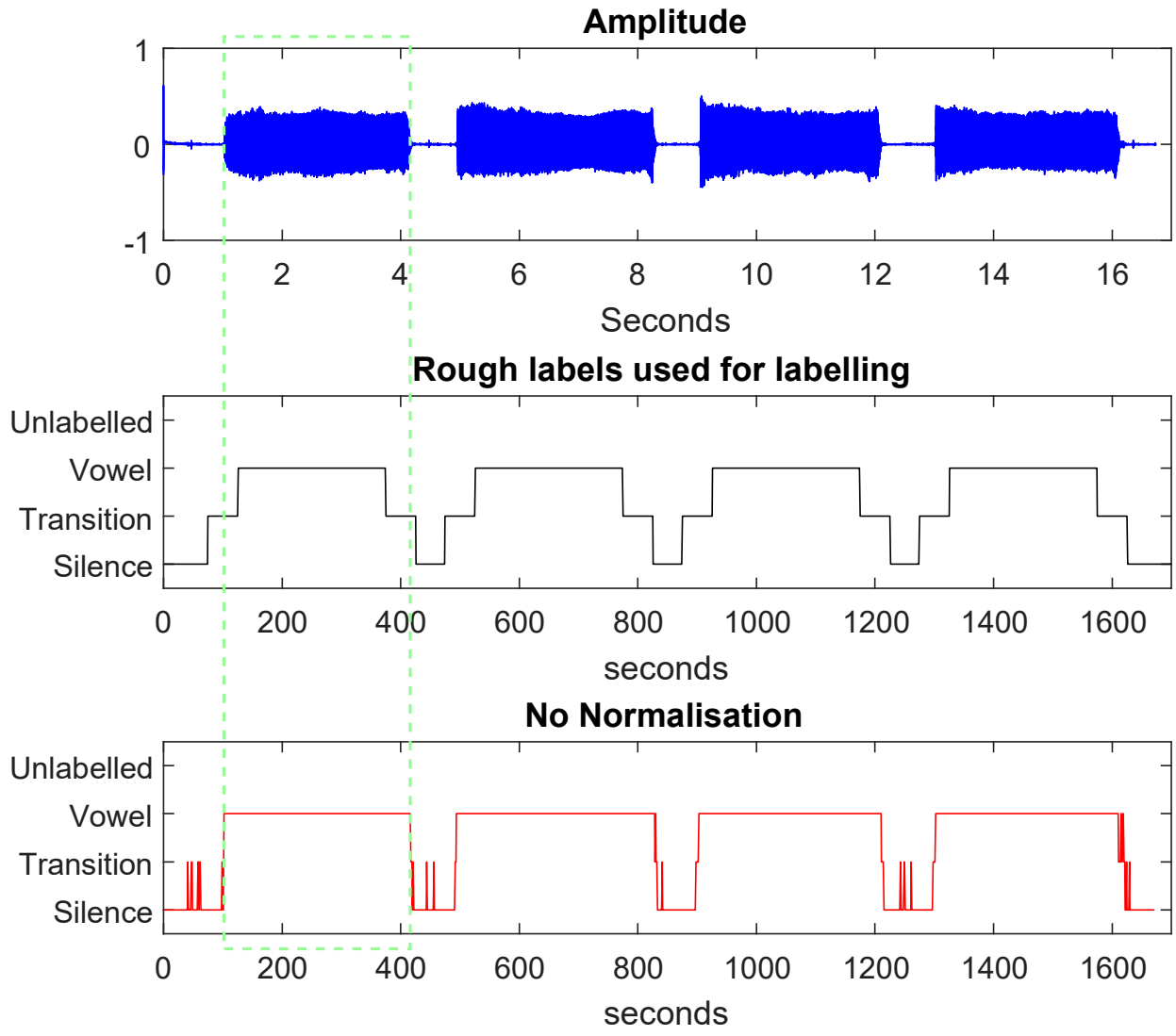


Figure 6.1: VAD for a file containing four utterances of the vowel /i/ as in *heat*. The first subplot shows the actual amplitude of the speech. The second subplot shows the rough label scheme used in training as it would be applied to this unlabelled file. The third subplot shows the results of classification using a 18x18 map with no normalisation. I drew the green dotted rectangle over the three subgraphs aligned with the results of classification to give the reader an idea of how the start and stop can be determined based on the frame-by-frame classification made by the 18x18 SOM.

researchers in this area have used datasets that are presegmented so that they have not needed to classify silence. However, my results indicate that using a hierarchical structure of SOMs will be sufficient for this purpose.

Grashey suggested that a KNPT VAD classifier must have many more than two nodes in the map. I also found that this was true for the unnormalised SOM. However, with a 2x2 map, bookmaker informedness was higher than 0.95 for MFCC13s, when normalising at the utterance level. This provides a strong indication that by controlling for variations at the utterance level, one or two nodes is sufficient to capture much of the difference between speech or silence.

The results presented clarify the results of Chapter 5, which showed that using a hierarchy of silence, vowel or consonant was superior to Kohonen's method of disambiguating confused phonemes. As we have seen here, the SOM has very high performance for distinguishing silence from speech. Using large classes at the base level as in the silence, vowel or consonant method is better than trying to disambiguate smaller vowel classes at the same time as classifying silence. This could be because silence still dominates the map at this early stage. By first performing the easier classification task, which seems to be VAD, the higher level maps are able to better distinguish between the vowel classes.

PART TWO

Chapter 7

Dubbing Embodied Conversational Agents

7.1 Background and significance of Embodied Conversational Agents

In this chapter, I introduce embodied conversational agents and my work on enabling them for a CALL activity for pronunciation that was developed. An ECA is a computer software driven animated face that can speak and move its mouth according to the words that it speaks, and with computer generated animation, allowing a computer the embodiment to converse with humans. The Head X embodied conversational agent was developed at Flinders University in my laboratory over a number of years as part of the Thinking Head project, a multi-university, multidisciplinary investigation into intelligent agents for interactions between humans and computers (Luerssen et al., 2011).

An agent such as Head X for learning can assume many roles. One common application of embodied conversational agents in a learning context is as a tutor, or teacher. Pedagogical ECAs are not necessarily tutors or teachers, however—they may also assume the role of learning companion, such as peer tutor, peer tutee, collaborator, competitor, or critic (Chou et al., 2003). To these roles, I propose that there is another, that of portraying roles in visual media presentations; in other words, the conversation agent can assume the part of an actor that can be dubbed. The advantages of using computer software for language learning are myriad, but unlike human teachers or conversation partners, computers never tire, they can be available at any time of day or night, yet do not charge an hourly rate for the one-on-one instruction they provide. As actors, computers can adapt their performances to the situation, such that it can be new every time; thus, a lesson built around a virtual conversational agent can be flexible, unlike video, which typically takes a single path and will play back exactly the same every time.

With these considerations in mind, I created software for language learning that harnesses the flexibility of a virtual conversational agent. I used the virtual avatar mentioned above that was developed to work with text to speech to enable realistic mouth movements that correspond with the speaking voice. Although it was built to work with different speech engines, the primary speech engine was SAPI 5.3, Microsoft Text to Speech (TTS). Microsoft publishes free voices, but there are also commercial voices available that work with SAPI 5.3.

7.1.1 Dubbing of Embodied Conversational Agents for language learning

In this section, my contributions to the published work (Chiu et al., 2012a) that is included later in this chapter are presented. In conversation with my colleague Yi-Hui Chiu, professor of English at the National Taipei University of Business in Taiwan, I came to better understand Chiu's research interest of video dubbing for her pronunciation classes (in which the students perform live voiceovers for movie scenes). I created a software application that enabled the repeated practice of dubbing a virtual embodied agent for arbitrary sentences. Professor Chiu used that software as the basis of exploratory research with some learners of English in Taiwan to better understand how learners would perceive an embodied conversation agent as used in a dubbing task. I wrote a basic description of the system, her findings, and our joint conclusions as a paper I titled *Dubbing of Virtual Embodied Conversational Agents* (Chiu et al., 2012a). Chui presented this paper at the Fifteenth International CALL Conference. Here I will discuss some of the motivating factors and how the future of video dubbing could be computer generated."

The basic flow of my virtual dubbing task is fairly straightforward. The software presents the target utterance to the learner, who can first see and hear the example, observing the speech production as movements of the head and lips. Subsequently, after sufficient observation, the learner has the task of speaking the words while the head moves, with the goal of dynamically matching their speech with the lip movements. Thus, they synchronise their voice with the animation. Finally, they are able to view the results of their efforts with the animation, which plays along with their own voice. Once a given sequence of utterances was recorded to the learner's satisfaction, they could advance to the next sequence.

Accordingly, I created software that would enable the student to participate in this sequence of turns. I created an interface for playing and recording that could play back any recorded voice or the TTS voice. According to a student-centred activity model, these steps can be repeated and old steps revised. The scripts for each lesson can be set by the teacher, the learner, or by an intelligent tutoring mechanism. Some screen shots of the different parts of the activity are shown along with a brief explanation of how the buttons and text work for the activity. Basically, an utterance is generated with voice and animation. The learner listens to the utterance as many times as is necessary. They can modify the viewing angle to observe.

The record button is pressed. The student records their voice while the virtual avatar moves their lips, either silently or for choral practice. Finally, the playback sequence allows the learner to listen and watch as the virtual agent moves in time with their voice.

Practice for video dubbing can be done in the student's own time with technology such as YouTube and a video recorder, enabling the student to hear themselves speaking in synchronisation with the video. The activity of practising at home for a class assignment that consists of a dubbing task allows the student to reflect on their speaking in a low risk environment, without fear of embarrassment. It was my aim to provide a virtual avatar that could be similar to the video. It has some advantages and disadvantages but I believe that the shortcomings can be overcome, and also provide a benefit of evaluation. Using video can provide a natural model, while on the other hand, a virtual embodied avatar can provide a reduced example that allows for specific attention to items of interest. The activity flow of the ECA dubbing task is as follows.

Activity Flow

1. The model text and video and audio is presented to the learner.
2. The learner practises reading aloud.
3. The learner presses record and records their voice in synchrony with the model video.
4. The system evaluates the speech.
5. Feedback is presented to the learner.
6. The learner can repeat the process flow from some step above or move onto a new utterance.

The lip dubbing pedagogy uses an audio-visual model that the learner is to voice dub. In other words, the learner speaks at the same pace and in synchrony with the model such that their voice appears to drive the lip movements of the model. The problem of alignment of speech is a feature of the system, such that if the learner is unable to match the lip movements properly, they are aware that they are unable to speak with the same accent as the model speech. Training the system on the audio can alert the learner where the differences between their speech and the model speech are located. In particular, in reviewing the speech and video they are able to attend to.

Difficulties with this approach: Inter-speaker variation can mean that varying characteristics between the model and learner are responsible for the observed differences. When these differences arise from physical traits, it may simply be not possible for the learner to match

the speech of the model. For a most basic example, if the model is a male and the learner is a female, the learner may find it too difficult to match the voice characteristics of the model. In this circumstance, we may accommodate differences by mapping the model space into a learner's space. Care must be taken to ensure that the transformation of the model space retains desirable characteristics, otherwise an unnatural pronunciation model may occur.

Alternately, with a sufficiently large corpora of models, for example, a 1000-speaker database, a suitable model would be selected that closely matches the physical characteristics of the learner. For example, it is not necessary that a male student must always learn from a male's voice, since people naturally learn pronunciation from others regardless of their gender. On the other hand, for the video dubbing task, it would be unusual to see a male's face with a female's speaking voice. Future research could investigate gender differences for pronunciation teaching with an ECA.

The general result of Chiu's use of my software for experiments was that this type of agent can be useful in the video dubbing task. Sentences can be selected on-the-fly for the students to say, and as such, lessons aren't restricted to a set script as occurs with video dubbing, which uses videos from YouTube, for example. Through interviews with the students, it was revealed that the main thing that they felt was lacking was an evaluation of their performance in the speech dubbing task. It is this finding that motivated the next direction of my research, to pronunciation evaluation. As I mentioned, the video dubbing pedagogy allows repeatable practice, but it's limited by the videos that are available, thus learners cannot construct their own dialogues. Additionally, evaluation of the speaking produced by the learner only occurs as a final performance mark or during in-class-practice. Indeed, these problems are typical in many aspects of teacher centred class-based language learning, not just pronunciation. Although it is possible to gain benefit from students communicating with one another in the unfamiliar language, there remains the difficulty that it is unlikely that novices provide high quality feedback to their peers as they do not know the language well.

Video dubbing is an activity for improving pronunciation. The activity of practising at home for a class assignment that consists of a dubbing task allows the student to reflect on their speaking in a low risk environment, without fear of embarrassment. Through repetition of utterances and mimicry of muscle movements, learners can work toward improvement in their speech intelligibility. It was my aim to provide a virtual avatar that could be similar to the video. The methodology used for the prototype study could be improved, but I believe that the shortcomings can be overcome, and the findings point toward evaluation of pronunciation as a incremental goal with the eventual aim of supporting future efforts into creating software based language tutoring systems.

Motivated by the findings of the dubbing agent research project, I pivoted my research toward automated pronunciation evaluation, with the goal of creating an evaluation mechanism for a virtual agent for the dubbing task. Despite being a topic of research for decades, CAPT is

still in its infancy (Levis, 2007), with wide gaps between pedagogy and practice. Levis reports that existing systems fail to diagnose pronunciation errors, or they provide feedback that is incorrect, due to weaknesses in using ASR for accents of language learners. However, with "intelligibility as a principle for CAPT", CAPT has been shown to work for specific areas of pronunciation, such as pedagogies that focus on segmentals with a focus on those that are targeted to enhance intelligibility.

7.1.2 Statement

The main outcome of our collaboration was learning that the learners had a strong concept that pronunciation evaluation would be a component of an exercise such as dubbing. The next focus of my research thus became the evaluation of pronunciation, culminating in the main body of KNPT pronunciation evaluation research as presented in this thesis. I prepared the following conference paper to disseminate this research to the CALL research community.

The ECA system was created by Tom Anderson, tested by Yi-Hui Chiu, and this paper was written by Tom Anderson on the basis of that testing. The presentation was made at CALL 2012 by Chiu, Y.-H.

7.2 Dubbing of Virtual Embodied Conversation Agents for Improving Pronunciation

7.2.1 Authors

Anderson, T.A.F., Chiu, Y.-H., and Powers, D.M.W.

7.2.2 Keywords

Second language learning, pronunciation, speaking, dubbing, embodied conversation agents, computer-aided language learning.

7.2.3 Abstract

In this work, we introduce work on an Embodied Conversational Agent (ECA) used by language learners to practice pronunciation. Using a realistic animated head with synchronized lip movements in conjunction with text-to-speech technology, we are able to provide a student-centred learning environment. Earlier work on the dubbing of film clips shows that dubbing can

be a powerful pedagogy for pronunciation. This work investigates the dubbing of ECAs, with a focus on the perceptions of language learners and the problems encountered when dubbing animated characters. With the eventual aim of creating a paradigm for using ECAs in language learning that reduces the burden on language teachers, we also investigate weaknesses of such a model. We present the results of an experiment on the dubbing of ECAs by second-language learners. The main contribution of this work is an investigation of the viability of dubbing ECAs to improve pronunciation. We also present a summary of possible future directions in research on ECAs.

7.2.4 Introduction

In this paper, we uncover problems that language learners confront when dubbing animated characters. Computer generated scenarios are feasible for language teaching (Anderson et al., 2008b). Film dubbing, or the synchronisation of voice over a film clip, is an effective teaching approach in an EFL context (Chiu et al., 2012b), with benefits including: reducing mispronunciation, improving fluency, raising awareness of intonation, increasing linkages with actual use, and meeting learners' perceptions. Film clips, through voice and video models of pronunciation, provide opportunities to improve pronunciation, particularly in regard to supersegmentals.

Dubbing is a productive activity that reduces demands on the teacher. Chan (2010) defined the mathematical definition of productivity as:

$$productivity = \frac{\text{student output}}{\text{teacher input}} \tag{7.1}$$

Accordingly, productivity increases with more student output and less teacher input; thus, an automatic system is advantageous if students learn more, whilst lessening teacher load. Film dubbing offers authentic, contextualised scenarios, but the content of film clips are not altered easily. To integrate dubbing into a wide variety of EFL courses, we propose: (1) a specifically designed interface; and (2) flexible scripts to cover all possible sentences suitable for personalised courses.

Video content is produced *a priori*, but an Embodied Conversational Agent (ECA) is a programmable virtual human that can speak any text. ECAs transcend text-to-speech (TTS) to provide embodiment, linking face and lip movements with speech. For an overview of ECAs, see Massaro (2004a). ECAs are suitable for activities like storytelling in language teaching (Anderson et al., 2011). HeadX, the ECAs in this study, is a realistic ECAs with synchronized lip movements (Luerssen et al., 2011), as seen in Figure 1. Visual speech can improve phonemic awareness in the acquisition of a language (Massaro et al., 2008). Research with HeadX has shown benefits for learners of lip reading (Gebert & Bothe, 2010).

Figure 7.1: The HeadX ECA, depicted reading a sentence in the pronunciation activity.

7.2.5 Methods

The purpose of this experiment was to gain an understanding of how language learners would perceive an ECAs for learning pronunciation. English language learners from Taipei were recruited (nine English college students and two with Ph.Ds).

A customised sentence selection software application was created using the default female face and voice of HeadX. To investigate user experience, ten random sentences of six to twelve words were used. The user interface, displaying sentences, can be seen in Figure 7.2. The four buttons depicted are as follows.

- Play HeadX —Activates animation and TTS audio.
- Record —Activates animation without TTS audio, and records the learner's voice.
- Play Back —Activates animation and plays the learner's voice.
- Next —Advances to the next sentence.

Participants viewed a PowerPoint presentation describing system usage. Participants were instructed to pretend dubbing was their job, to perform voice dubbing to improve the ECA. Lesson flow advanced through the sentences as follows: first the sentence was played, participants next recorded their voices, then played the recording back. They then alternated between these three actions. When they were satisfied with their pronunciation and voice synchronisation, they advanced to the next sentence. Subtitles were shown throughout the process.

After completing all ten sentence recordings, an oral interview was conducted to elicit feelings. Results from each participant were combined, revealing the most common responses.

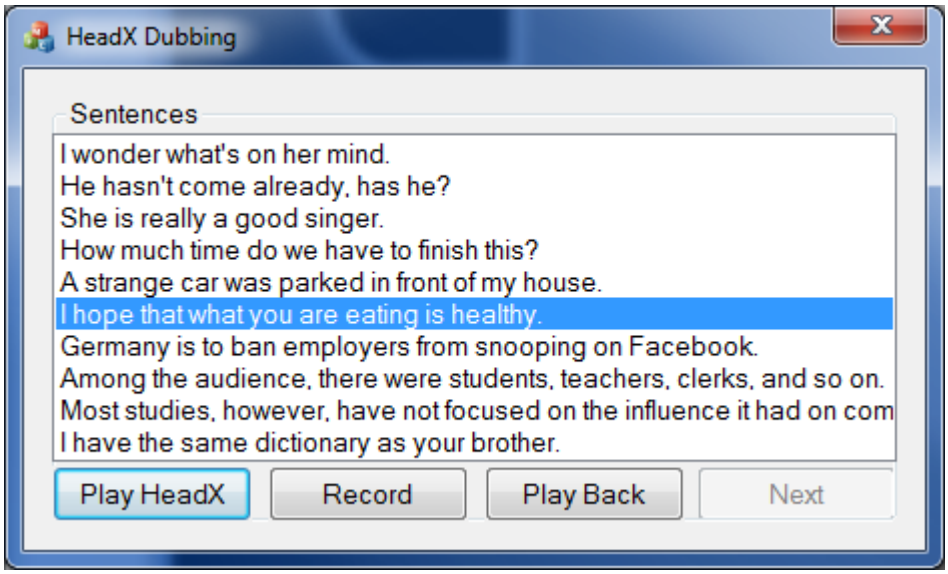


Figure 7.2: Figure 2. The dubbing system interface. The ten sentences depicted were used in this study.

7.2.6 Results

Responses gathered from the participants revealed their feelings. Overall, participants felt that the interface was easy to use for experienced computer users. Learners expressed that lip movements were helpful to learning proper pronunciation, but not as helpful as voice. Some suggested that interactive dialogues could be preferable to a dubbing activity.

Although film dubbing is a real-world activity, most participants viewed the exercise as a pronunciation drill. Most of the responses regarded the initial sentence reading phase (initiated with button "Play HeadX"). Participants felt the intonation of the system was awkward. Although they felt that HeadX performed sentence reading better than a translation dictionary, MyET provided a better intonation model. Many participants also said that unlike a human teacher, lip movements were not exaggerated, nor was attention placed on the lips (participants were not trained to use multi-angle or zoom views). Additionally, many participants expressed that the system should provide variation, such as accent, male/female, or adjustable speed.

Many participants drew comparisons with MyET (Tsai, 2006), a CALL pronunciation system well-known among language learners in Taiwan, which could indicate that the software was considered marketable. Many respondents preferred the MyET model of intonation, saying that its wave distribution and scoring mechanism assist in understanding mispronunciation, although even with MyET it was not easy to tell which words are mispronounced. Compared with MyET, the weaknesses of an ECA without such mechanisms are easily expressed. On the other hand, participants said the experiment was interesting and that they would be very happy to recruit participants for future studies.

The graphics capabilities of some computers were found to be insufficient to display HeadX

Figure 7.3: Figure 3. HeadX display as used with graphics problems.

correctly. Due to elements out of our control, the appearance was black-and-white (Figure 7.3) for some participants. Animation functioned properly, but without colour. Participants said the visual speech cues were helpful, but many reported that the head looked scary like this.

7.2.7 Discussion and Conclusion

As interviews revealed, participants felt that the experience could be improved through better graphics, intonation, contextualisation and interactivity. We are working with the creators of HeadX to provide better compatibility. In future, attention should be directed towards the mouth as pronunciation model rather than on the whole head.

Although ECAs with TTS provide lesson planning flexibility, learners feel less improvement compared with video or human teachers. For future work, we aim to conduct quantitative comparison of dubbing ECAs with pre-dubbed human voices. Alternatives to pre-dubbed voices include peer-generated voices or film clips. Additionally, longer contextualised passages or interactive dialogues would be more beneficial for learners. Despite problems uncovered in this pilot study, ECAs for pronunciation learning merit further digital classroom experiments.

7.2.8 Acknowledgements

Luerssen and Lewis from Flinders University of South Australia provided invaluable assistance with customising HeadX. The sentences used were from http://tatoeba.org/eng under CC-BY license [1] from authors CK, CM, Guybrush, sacredceltic, and Source_VOA.

[1] http://creativecommons.org/licenses/by/2.0/fr/

7.3 Enabling Head X to speak Mandarin

In addition to the work with the English pronunciation, I also worked on enabling a Mandarin version. In order to create a virtual head that can speak Mandarin, the SAPI output must be decoded and sent appropriately. In the case of English, the syllables are transmitted and they can be mapped to visemes. The difference between Mandarin and English for TTS is that Mandarin has the same syllables, as does English, but as a tone language it also has tones, which influences the voice quality. The basic assumption of the HeadX interpretation of the viseme stream is that it is linear, which is to say that there is one time frame and one viseme. However, the main difference is that Mandarin SAPI output consists of two visemes per time frame, one viseme for the syllable and the other viseme for the tone. Each of these visemes and its corresponding length are the same, so my work on HeadX enabled the avatar to properly process the stream of input from the TTS voice, since it had been misinterpreting the two streams of information.

To enable the HeadX software to work, I programmed support for the Mandarin SAPI input by modifying the source code of the HeadX software. Then I constructed an XML template that included all the various speech sounds of Mandarin. The XML file would describe how each of the various speech sounds of Mandarin should be mapped to visemes, thus enabling the HeadX avatars to work with Mandarin TTS voices. This modification that I made allowed researchers in China to use HeadX in their research.

7.4 Discussion

The main outcome of the research efforts presented in this chapter was to motivate the subsequent research into SOMs. Students who used the dubbing activity with ECAs agreed that automatic evaluation of speech was a necessary component of the system, which would allow for less reliance on a human teacher. Going along with this suggestion led to the work that appears in Chapters 3 to 6 in this thesis.

Chapter 8

Thinking Head MulSeMedia: A Storytelling Environment for Embodied Language Learning

8.1 Statement

There are a number of approaches to improve language abilities when there is a human tutor or teacher, or for self-directed learning. When considering computer-based methods, it is often the case that existing methods (used by human teachers) are considered, but the philosophy I take is that the particular pedagogy or teaching method that will be used is not the specific concern. Previous chapters worked towards a system that can evaluate an utterance in regards to some understanding of the target, such that the teaching method can provide feedback about the speech that was produced. In addition to providing for speech analysis, e-learning provides many other affordances of interaction, even beyond those of a keyboard and mouse. In this chapter of the thesis, I address how a multimedia environment for storytelling can provide a general framework for CALL.

The book chapter, as presented here, was written entirely by Tom Anderson with review and consultation with the other authors[1]. This chapter appears in the bibliography of this thesis as Anderson et al. (2011), and it was published in the book "Multiple Sensorial Multimedia Advances and Applications" (Ghinea et al., 2012).

[1]This version of the book chapter excludes the Related Work section, which was contributed by the coauthors. That section has been removed and replaced with a brief summary (written by Tom Anderson).

8.2 Authors

Tom A. F. Anderson, Zhi-Hong Chen, Yean-Fu Wen, Marissa Milne, Adham Atyabi, Kenneth Treharne, Takeshi Matsumoto, Xi-Bin Jia, Martin Luerssen, Trent Lewis, Richard Leibbrandt, David M. W. Powers

8.3 Abstract

With technological improvements and widespread computer usage, the need for user-friendly human-machine interfaces is steadily increasing. Computer graphics and audio provide a colourful world of visual and auditory experiences; nevertheless, tutoring systems have traditionally used keyboard and mouse almost exclusively, limiting significantly the input mechanisms available to content designers. Multisensory learning is beneficial for learning a new language (Birsh, 2011), and through increasing the affordances of interaction, learning scenarios can be significantly augmented.

The Thinking Head is an embodied conversational agent that be used in intelligent tutoring systems as the representation of a teacher and tutor for one-on-one computer learning and teaching. Although there are many applications that benefit from an embodied conversational agent and there has been a push in our research community towards improving embodied conversational agents, our main area of research lies in the creation of applications for second language learning (as introduced in Powers et al. (2007); Powers et al. (2008)). We have explored speech synthesis and speech recognition to permit users to interact with computer systems with their voices. To facilitate language learning through conversation, we have incorporated these auditory capabilities into the Thinking Head; additionally, we seek to create a bridge between the virtual and physical worlds through physical manipulation, achieved primarily through visual input recognition with affordable devices such as webcams.

The hybrid world provides a framework for the creation of lessons that teach and test knowledge in the second language. In one of our example grammar and vocabulary lessons, the Thinking Head instructs the student to move a tiger to a lake—a student moving the toy tiger in the real world effectively moves a virtual tiger in the virtual arena. This type of interaction is beneficial for computer-based language learning especially because we are able to see a student has successfully understood the directions if the tiger is moved to the vicinity of the virtual lake. Physical movement helps the learner to internalise the novel relationships in the second language. We also provide for additional forms of communication, including dialogue with an embodied conversational agent and writing stories using markers on a whiteboard. In summary, our system provides a natural interface with which to communicate with the hybrid-reality learning environment.

8.4 Introduction

In this chapter, we explore interactions with the Thinking Head computer interface arising from a fusion of inputs of voice, objects, and writing—without keyboard and mouse. The Thinking Head system for language learning teaches through stories; however, the Thinking Head is not just the storyteller, but rather the medium in which the storyteller and story exist, and through which a learner learns to tell the stories. A learner engages in the multisensory storytelling activity by interacting with physical props within a room to learn the stories.

A user manipulates real-world objects within the Thinking Head virtual arena. In our implementation, objects are most commonly small graspable things; however, our use of the term object applies more generally to refer to things of all sizes, persons, places, and ideas. Indeed, humans have given names to objects for thousands of years, and the meanings of objects take form as the words we use as tools in communication.

In SIMULA 67, arguably the first object-oriented computer programming language, objects were representations in a discrete event system (Dahl et al., 1968). As (Lorenz, 1993, p. 12) characterises objects in computer programming, "An object is anything that models 'things' in the real world. These 'things' may be physical entities such as cars, or events such as a concert, or abstractions such as a general-purpose account." Thus, for our purposes, an object is any natural thing or combination of things that can be quantified, described, or elaborated by man or machine.

Traditionally, computer multimedia applications consist of audio and video, while mulsemedia, or multiple sensorial media, enhances the user interaction experience. The use of realia (physical props such as apples, hats, and magazines) adds to the language learning experience because realia engages the senses. Object recognition determines the identity, location, and movements of known physical objects, and classification heuristics categorise unknown objects (Yilmaz et al., 2006). Those objects are rendered with respect to landmarks in the virtual world, creating a hybrid media experience that enhances class-based learning and provides a strong practical component that is easily accessible in the classroom or at home. Finally, writing practice reinforces the lessons and adds more latitude for expressing creativity.

The first question we need to address is how to promote modes of mulsemedia learning beyond the standard keyboard and mouse interface. Although interactive systems challenge the brain, Crawford (2003) notes that the typical scenario for play with a computer works with most of the body immobile. The advantage of learning in a multisensory computer environment is that the user can retain the benefits of computer processing while interacting with tangible objects in the environment. The goal of language learning is to become functional in the real world. As it is impractical and expensive for many language learners to travel to the part of the world where people speak the target language, it is desirable to learn by oneself in an immersive and interactive environment.

Language is produced as a reflection of multisensory perceptions of the world around. As children, we naturally acquire language of our physical, social, and cultural environment, but as we grow older, the gaining of fluency in a new language takes a different route. Although some believe that it is more difficult for teens and adults to learn a language, Krashen (1987) argues that it is not that the older learners proceed more slowly, it is rather that the interactive and comprehensible nature of the environment of the younger learners favours their learning. It is common knowledge that adults simplify the complexity of the language when they speak with children. By gradually and incrementally increasing language complexity while maintaining comprehension, learners of all ages come to perceive language not simply as a logical collection of information but also as a system of understanding the world. We believe that when a learner engages in mulsemedia experiences that are challenging yet stimulating, the language that is learned is grounded in experience, providing basic language skills that are later necessary to engage in reading.

There are substantial technical barriers to providing an interactive and adaptive language experience, difficulties which arise in the way that the computer perceives, represents, and portrays the world. For many years, the computer has been restricted to two-dimensional representations on flat screens. The computer, equipped with sensors, thereby has an agency that allows it to perceive meanings in the world, channelling the actions of the learner towards effective practices.

Furthermore, by perceiving learners and gaining a better model of the learner in three-dimensional spaces, we can gain a closer understanding of how the learner views the language as a mechanism of conveying understanding about the world. Through video, the computer can communicate with the learner, gently informing the learner that their actions have somehow inadequately achieved the goal communicated to them, or alternately, commending their satisfactory actions.

The computer has become a very large part of people's lives, and as such, it is necessary for a language learner to acquire the language corresponding to computer-based environments. Experience gained with the Thinking Head for learning natural languages extends to learning programming languages and the interfaces of computer systems. The interface that we are talking about has no keyboard or mouse, but it does include speech recognition and speech synthesis, and object and gesture recognition. Arguably, even speech recognition is multisensory, because the throat muscles move and the vocal chords produce sound, and the listener perceives feedback with multiple senses on multiple levels—directly as proprioception, and indirectly through other respondents.

It is theoretically possible to explicitly hard-wire the behaviour of the system in response to each change in relationship, but as the number of objects grows, the complexity of the system increases accordingly. Therefore, in order to reduce complexity it is desirable to pass many of the decisions about the intended result of user interactions to the system itself. In later

sections, we will discuss the creation of a system to support interactive stories for the purposes of language learning.

8.5 The State of the Art

A room-based system for second language learning is a system of interactive services that provides different outputs based on the locations and behaviours of people and objects. First, we review previous work on systems for interactive storytelling, including room-based systems, where embodied conversational agents portray the role of teacher, confidante, or storyteller. Then, we present factors involved in facilitating interaction with a computer system primarily through spoken language and movements of body and objects, including machine use and recognition of human language, speech synthesis and recognition, vision, wireless sensors, and machine learning. We provide a brief overview of second language learning for context, but then present a more in-depth research basis of our approach to language learning through stories. Subsequently, we present the literature on computer-based education with a focus on mulsemedia and interactive education with embodied conversational agents.

8.5.1 Storytelling platforms

It has traditionally been difficult to build an interactive story-based language learning computer system, which is essential to defining the space of a story environment that uses a tangible interface. More recently, however, a wide array of low-cost multiprocessors and sensors have become available, making the visions of dream-based research[2] increasingly possible. One-to-one[3] technology-enhanced learning (i.e., each student has at least one computing device to help his/her learning) brings promising potentials for students, including active, productive, creative, and collaborative learning (Chan et al., 2006). When all students have a computing device that has wireless access to Internet resources and multiple sensors to identify tangible objects in a learning context, students can acquire relevant information and can even interact with different learning objects. Such technology support is crucial for some learning activities, especially for storytelling activity, because an enriched environment could facilitate students' productive thinking, creative ideas, and collaborative behaviours. Storytelling platforms provide a composite and changing environment in which to act together with a computer Crawford (2004). In the StoryRooms of Alborzi et al. (2000), it was reported that a multisensory environment for storytelling is intrinsically motivating. Ribeiro et al. (2009) demonstrated that interaction with robots is a useful educational tool for arts and languages, and can be applied

[2]Dream-based research is the pursuit of innovation and potentialities for some distant future, as defined in Chang et al. (2007).

[3]More information can be found in the G1:1 website (http://www.g1on1.org)

to storytelling. Their learners used robots as their characters in a drama, but rather than using a natural language interface, they scripted the interactions by using a visual programming language. Likewise, participatory design researchers Druin et al. (1999) demonstrated that children could successfully build personas for robots that could be a medium for storytelling.

8.5.2 Tangible interfaces

Although computers and virtual reality offer many advantages, real-world tangible objects may be superior to them in certain ways. O'Malley & Fraser (2004) categorised tangibles for education into four main classes: (1) digitally augmented paper and books; (2) physical items used as interfaces to virtual worlds; (3) digitally enhanced tangibles (e.g. robots or other devices that interact with other tangibles); and (4) sensors, which gather information about the world. Our storytelling world incorporates all these types of tangibles.

Tangible interfaces compare favourably to virtual ones. Wooden blocks, which are moved by grasping, contrasted with virtual blocks that are moved by a computer mouse, were found to have distinct advantages for learning (Verhaegh et al., 2008). This phenomenon is attributable to the ease of manipulation of real-world objects, which can be moved into different arrangements more efficiently, with positive effects on learning. An implication of this is that the innate human ability to manipulate real-world objects can improve computer interfaces.

Yet real objects need not be without the advantage of computer processing. "A networked toy may provide aural, visual, motion, tactile and other feedback, and be able to sense speech, physical manipulation, and absolute and relative location." (Srivastava et al., 2001, p.2) Such toys equipped with wireless sensors in a smart room for learning also offer significant capabilities for the lesson designer. Tangible interfaces, in the form of physical objects that are to be touched and moved, allow for tighter representational mappings between the objects and the meanings they represent (O'Malley & Fraser, 2004).

Research into multimodality finds implications for language learning. Audio-only learning is inferior to multimodal conditions in which language is visually grounded; not only in terms of the quality of machine learning of language, but also, for a computer system learning human language, the availability of more modalities of inputs offers more understanding (Roy, 1999). Computational models of language provide insight for improving the capabilities of computer systems, and furthermore also shed light on the abilities of humans to use language.

Handheld objects and devices offer an additional level to the hybrid reality. These location-aware objects represent many different things for language learning—learners grasp and move them to reflect changes in the story, and these objects provide outputs such as haptic feedback or sound. Despite having relatively smaller screens than computers with situated LCD monitors, users perceive that handheld devices have a larger field of view and provide a natural interface that promotes proprioception and immersion (Hwang et al., 2006).

Online and offline character recognition mechanisms allow users to write on real-world surfaces with everyday marking utensils, for example, by using a marker on a standard whiteboard. Handwriting recognition forms an important component in building human-machine interfaces (Plamondon & Srihari, 2000), and the interactional aspect of real-time processing of handwriting offers a great deal to educational media designers. The accuracy of classification results increases as a result of tracking the position of the hand and movements of the pen as users write, but more significantly, this mechanism also provides the user with a natural language interface with which to communicate with the learning environment.

8.5.3 Embodied conversational agents

Embodied conversational agents are known by several names; depending on the domain, they may be referred to as intelligent, animated agents, as pedagogical agents or as educational agents (for a more in-depth discussion, see Veletsianos et al. (2010)). Embodied conversational agents are computer-generated animated characters that engage in multi-modal dialogue involving speech, intonation, and gestures. They are most commonly used in video games, but over the years, a number of such agents have been developed for a variety of applications in a broad array of disciplines. The trend in recent years has been towards increasing realism in terms of graphics and agent behaviour. Their inclusion in interactive systems makes user interactions with the computer more like interactions with a human.

Virtual agents can be used in the storytelling context in the roles such as the storyteller, characters in the story or the audience. Some embodied conversational agents are tailored for specific teaching tasks, e.g. Baldi Massaro (2004b). Others like Greta (Poggi et al., 2001) take a more holistic approach. On the other hand, a pedagogical agent with only a single identity is constrained to that identity, which might not be advantageous among some target audiences. Our previous work was to create a platform for embodied conversational agents, specifically designed to portray multiple and varied personas (Luerssen & Lewis, 2009).

Embodied educational virtual characters have the capability to drive learning in terms of two factors: learning motivation and learning experience (Chen et al., 2009). By involving the learners emotionally in the virtual characters, we maintain their attention and drive to succeed in the educational activity. And by improving the emotional ties of the learning experience, we enhance the learning opportunities based on relationships within the environment. As students build relationships with virtual characters, they increase their affinity for the domain subjects.

When humans interact, the participants modify their strategies based on the conversation. In embodied conversational agents, template matching is a common strategy, and audible or visual speech is generally processed to and from text-based expressions. Thus dialogue systems are natural language systems that represent dialogue and productions as text. The findings of Chu-Carroll (2000), who developed MIMIC, a dialogue manager with strategic generative

mechanisms that adapted to the interaction, suggest that dialogue managers should determine when to take the initiative to better match the expectations of humans who interact with embodied conversational agents.

8.5.4 Second language learning

In early childhood, we acquired language through the activities of play with objects, not in deliberate efforts to transfer knowledge from one person to another but in symbolic activities with intentions to imitate basic events (Nelson, 1998). We can draw some useful analogies from the way that a child learns their mother tongue, but in the case of learning a second language, learners may be of any age. It is thus relevant to consider that the main purpose of learning a language is to gain a tool. The production and comprehension of the foreign language vocabulary in the context of sentences is thus a tool to receive and create the expression of thoughts of others. A word or phrase is a symbolic representation, "a close amalgam of thought and language that it is hard to tell whether it is a phenomenon of speech or a phenomenon of thought" (Vygotsky, 1964). Reality, generalized in a word, takes different meanings and associations according to the circumstances and the evolutions of the language. Thus, from our perspective, the learning of language is the learning of shared cultural meanings towards participating in that culture.

There are a number of popular approaches to language teaching, including: grammar-translation, the audio-lingual method, and the communicative approach, but these, as with most language learning methods, have little to do with physical action in the real world. Total Physical Response (often known as TPR) is a method that involves physical movement in the learning of a new language (Asher, 2000). In Total Physical Response, as in more traditional classrooms, a teacher leads the second language lesson, but unique to the Total Physical Response methodology, the teacher produces utterances that demand a physical response from the learners, many times involving the manipulation of objects. The involvement of the body in such kinaesthetic learning activates portions of the brain—in particular, the right hemisphere—which is considered to be understimulated in standard language learning through lectures and repetition. Through Total Physical Response, learners acquire a new language through whole-body interactions much faster rather than the rote learning and analysis of the logistics of language that is prevalent in many language lessons. In particular, we feel that the learners gain a shared cultural understanding of the components of language grounded in physical interactions.

8.5.5 Stories for language learning

Our current work centres on the telling of stories as a form of language learning. We seek to enhance language learning by incorporating the methodology of Teaching Proficiency through Reading and Storytelling (TPR Storytelling), which extends movement-based learning to the learning of language in stories (Ray & Seely, 1997)[4]. The storytelling procedures elaborated in this chapter arise in TPR Storytelling, the cornerstone of our current efforts. A classroom teacher who adheres to the TPR Storytelling method engages learners in the stories through active responses and involves them directly—both as actors and as storytellers. Language learning through storytelling creates a social context that brings together the participants as a community that has certain stories in common (Brune, 2004). Allowing for sufficient questioning to tease out the finer details, the learners of a language taught through stories are learning as insiders. Furthermore, learners are using all their human senses and are therefore implicitly learning the language that conveys meaning about things perceived.

Stories have many of the essential elements required for successful language learning. Stories are handed down for generations because stories communicate something through language that echoes within our imaginations. It is through hearing stories that we come to understand the relationships between things in the words of another. Much of human cultural memory is stored in stories, and the language that facilitates the telling of stories is the heart of language. Knowing the basics of a language allows any learner to be able to learn more of the language through comprehensible stories (Krashen, 1987). To understand a story, it is necessary to understand how the relationships between things change over time. Stories communicate the changes in interrelationships between elements over time, and when stories are interesting, those who receive the stories are motivated to listen.

There are many different kinds of stories. We describe these with words like comedy and tragedy—terms which describe the emotions evoked. The learner attaches meaning to an emotional context that is grounded in the experience. In a story, the intertwining of language learning and the content of the story rewards the learner, who understands the language because there is a story to understand. The learner understands why things happen, gaining insight into the important objects in the story.

TPR storytelling provides for scaffolding and expansion of language. The first time that the story is told, it is told in broad strokes, in generalities that the learner gradually recognises as the basic story; successively, as the learner becomes more able to understand the story, increased levels of details emerge. For example, a love story can be expressed in general terms as between a man and a woman, but attention is piqued by insider details. Questions are bound

[4]Note that TPR Storytelling emerged as an evolution of Total Physical Response, though the acronym for TPR differs for each. Hereon in, we reference to TPR Storytelling in the sense of Teaching Proficiency through Reading and Storytelling as our primary language learning pedagogy.

to arise when a storyteller reveals that the man is a 112-year-old man named Dave Armstrong who is a new resident of Manhattan. There is always more to any story than can be received on its first telling. Additionally, as details emerge, the audience begins to relate the objects with the cultural memory. When the audience find out that Dave has been in cryogenic sleep for 65 years, they will begin to draw analogy to stories or folktales—both Rumplestiltskin and Snow White depict the events that unfold for people who sleep for long periods.

To learn a new language, it is vitally important to learn a great deal, but it is also quite important to have something new to learn about. Television and movies are the media with the most familiarity in society today, but lessons delivered through the television are not successful for language learning, as it is not enough to just see and hear language learning media: you must be able to understand it (Krashen, 1987). We extend Krashen's argument, believing that it is just not sufficient to click a mouse to communicate to the system when you do not understand—you need to be able to talk with the storyteller, to ask and answer questions. Furthermore, the storyteller should be able to ask questions, and the system receives the answer by way of natural speech or physical movements. The people who learn in this way can apply the memorable experiences that they gain with the stories to interactions in the outside world; for example, when the learner gets the chance to interact with a native speaker, they will know how to ask for clarification because the computer system frequently asked for clarification in a real and friendly way.

The greatest stories transcend language barriers because of their themes and narratives, as in the mythology of Joseph Campbell's hero journey (Inchauste, 2010). A compelling story follows a storytelling arc, meaning that the events in the story build repeatedly towards a climax. A compelling story is one that influences our emotions, which arouses the curiosities. Therefore, to allow the curiosity to drive the language learning, we must also make the story change in a way that is adaptive to the learner. Using stories in language learning, when the learner comes to understand that certain ways of speaking are natural, we have succeeded to a certain degree. When we ask questions about the occurrences in a story, we have a shared space that contains a memory of things and events that have occurred.

8.5.6 Nonverbal communication

Human communication is a means of expression and is more than just the words of the language; accordingly, interaction with an embodied conversational agent is not limited to communication with words. "Believable nonverbal behaviours for embodied conversational agents (ECA) can create a more immersive experience for users and improve the effectiveness of communication" (Lee et al., 2006). To indicate the location of an object, a turn of the head or a move of the hand suffices better than a detailed explanation. Similarly, when a speaker increases the rapidity of their eye blinks, it signals that the speaker has an increased emotional attachment to the topic.

An embodied conversational agent can also employ nonverbal behaviours to deepen the experience of interaction and to improve the quality of communication. Many elements of nonverbal communications are culture-specific, thus of interest for language learning. An embodied conversation agent has many qualities that make it an ideal teacher for teaching language, including: it never tires of teaching the same thing repeatedly with multitudes of different students, and it can collect user data for grading the learning and for system evaluation. For an overview of expression and virtual characters, see Vinayagamoorthy et al. (2006).

8.5.7 Visual speech synthesis

Many streams of information are manifest in human perception of speech. The visible movements of the face of the speaker contribute a great deal to the comprehension of speech. Modelling correct pronunciation is more than just the correct modelling of acoustic phenomena. If a student is to learn to control the position of their mouth in a suitable fashion, it is also favourable to interact with a working example, for example a human teacher or a realistic virtual head, to provide a mechanism for learning to know how to produce a particular phoneme. Vision is an important part of the listening process (Kellerman, 1990), and particularly for language learners, this means that designers should provide ways to supplement audio with mouth movements, such as with a talking head (Massaro, 2004b).

Figure 8.1: This representation of the Thinking Head, an embodied conversational agent, is used as a mediator for storytelling.

8.6 Implementation

The Thinking Head system is an extensible framework for interaction with a multimedia system. First, we present the implementation of the Thinking Head system for second language learning and the system architecture. We then describe the system's embodiment in terms of the perceptions of users of the language learning system. Subsequently, we recount the capabilities of the system to perceive and act in the world. Finally, we provide a sample language lesson to give readers a sample of the capabilities of the system.

8.6.1 Thinking Head

The Thinking Head is an animated head with several screens, cameras, and tangible objects—but conspicuously, there is no keyboard or mouse. The system perceives and interprets inputs primary from the visual and auditory domains captured by the respective sensors. It responds in accordance with the in-built dialogue manager, which defines the function of the Thinking Head as a Teaching Head that provides instruction and companionship, and which has been extended to present multimedia lessons that promote second language acquisition.

Although perception by the system is the main feature of this project, the appearance (or synthesis) of the system what users perceive most immediately. Since first impressions matter, we have implemented virtual characters that animate realistically and are visually appealing and likeable. The framework that drives the system provides both the external form (e.g. through monitors, speakers and tangible objects) and a perceptive system that allows it to produce reactions to human stimuli that appear intelligent. It can be configured to specific language learning tasks and is interoperable with other subsystems, such as with vision processing and audio/visual speech synthesis. We use both pre-recorded audio, which allows for high quality pronunciation, and a text-to-speech engine, which allows the natural language production system to produce unique utterances. Additionally, the speaking voice synchronizes with lip movements of the Thinking Head.

In terms of outputs, the Thinking Head is in many ways like standard media, which is dependent on sight and sound. On the other hand, the Thinking Head provides for multiple modes of inputs and outputs to engage the senses of the language learner, particularly to situate the learning of the new language in vivid, multisensory experiences. We have a sound system to produce audio and a colour monitor for vision; many of the monitors we use are standard LCDs, but we also have implementations that use LCD projectors or 3D monitors. On the other hand, the fixed location of a standard computer monitor restricts interaction: in our implementation, a number of user-movable LCD screens provide output devices that can provide 3D information. Users of the system can manipulate these screens, which may be

unattached or anchored, and the outputs of the system change in accordance with the changes in spatial locations and orientations.

Objects of interest, including the monitors and screens, are more than just display devices; they also provide affordances for interaction. The holding and moving of these objects is a form of input for the system, and the learner sees and touches these objects—and potentially smells or tastes them, in the case of fragrant objects or food, respectively. Additional output capabilities of objects of interest include localised sound production and haptic feedback, but the changes in outputs are not restricted. The environment—or the objects of interest themselves—responds by providing contextualised output, which provides learners with feedback naturally grounded in the language system.

As a form of hybrid reality, many tangible objects in the learning space have correspondences in the virtual world. A child's toy held in the hand, merely a simple graspable object in the real world, becomes any object or objects in the virtual world, promoting multimodal channels for messages that expose spatial and relational meanings.

8.6.2 What information does the system perceive?

The inputs to the system come from the environment of the learning space, which is comprised of objects, some movable and others fixed. In this context, the learner and the infrastructure are also elements of the environment. The system gathers information about the environment and the interactions with the environment through sensors in the room. Some of our cameras possess pan-tilt mechanisms; by way of moving the camera judiciously, the learning space system can fine-tune its survey of the visual field. The perceptive components of the system additionally observe the outputs that the system produces, allowing for improvements based on these observations.

The Thinking Head system can see and hear. Unlike traditional systems that simply forward or record video or voice data, the Thinking Head can analyse the data to perceive certain attributes and changes to the environment. The system is able to upload data from the learning space that it perceives to the Internet using web services. This data would be stored securely to maintain the confidentiality of learner data. Future use of the data allows for fine-tuning of algorithms, inter-learner comparisons, and re-evaluation of results under new theoretical frameworks.

8.6.3 How are objects located and tracked?

An important component of our language learning lessons is the use of tangible objects. It is through the interactions of learners with these objects that we are able to determine comprehension levels for the stories. We have investigated a number of different techniques to

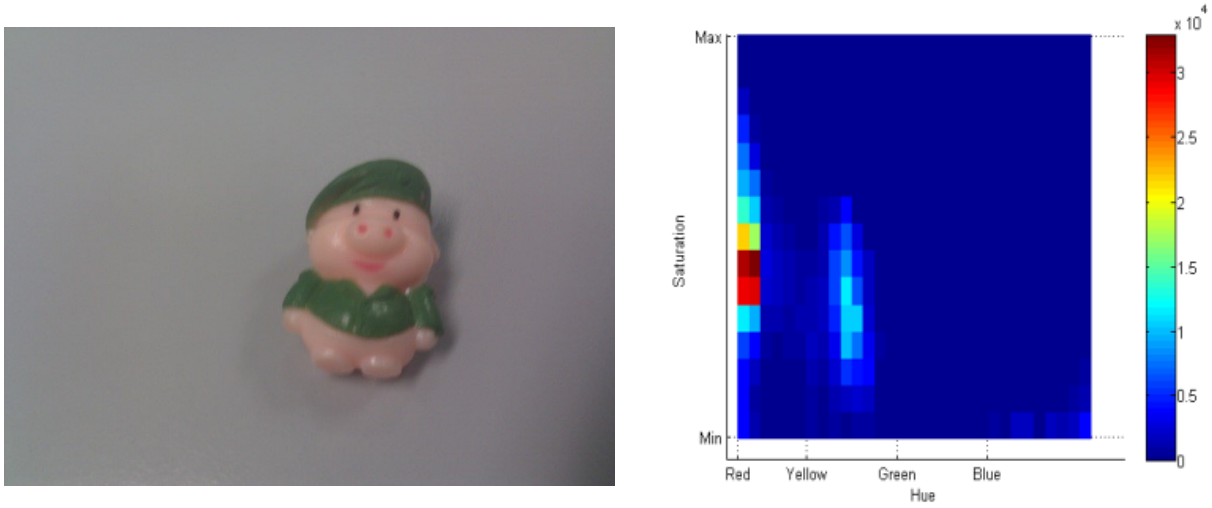


Figure 8.2: An image of a pig. Hue-saturation fingerprint of the image.

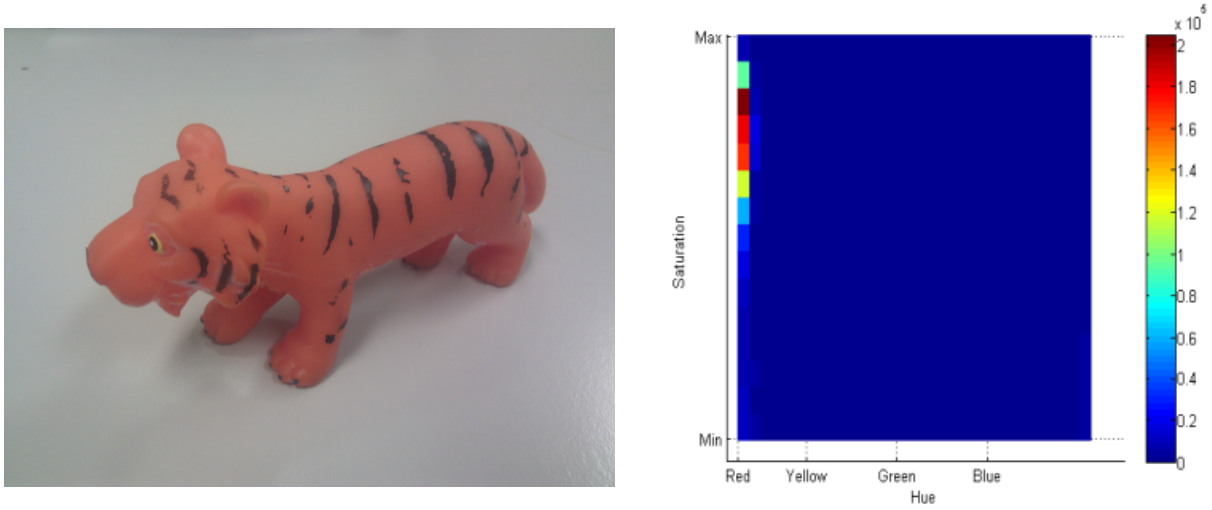


Figure 8.3: An image of a tiger. Hue-saturation fingerprint of the image.

recognise and track the movements of objects. Our focus is on the tracking of objects with low cost webcams through primary geometric representation of objects detected through colour channels.

Each of the tangible objects used in the language lessons has a unique colour signature, determined based on hue and saturation, which enables the system to differentiate it from other objects. Analysis of the visual scene uses a database of object colour signatures. In Figure 8.2, for example, the low saturation of the pig is most prominent in the yellow-green colour area. In Figure 8.3, on the other hand, the saturation of the tiger spread in the red hue is clearly distinct from that of the pig. The algorithm for object recognition enables real-time object recognition with low demands on processing (for more technical details, see Franzel & Newman (2009)).

Sensor data transmitted wirelessly improves the accuracy of object localisation beyond the

visual recognition. For example, accelerometers that measure rotations in different axes provide some information about the movement of a tangible object in three-dimensional space. If the object moves through a position in which it is partially obscured, the sensor data contributes to the object localisation task through data fusion techniques. Other location data collected within the learning space include wireless TOA, audio and vision processing by combining views of the scene from multiple cameras. Estimations of body positioning provide constraints on the possible movements.

8.7 What does location information provide?

The use of optimised algorithms such as these allow for real-time tracking and localisation, enabling us to create lessons that incorporate the movements of real-world objects. When learning one of our mini-stories, for example, a learner may move an animal from one location to another. The system can locate the toy animal in regards to the other objects of interest, and when the learner makes a correct gesture with the objects, the system provides positive feedback. If the toy is moved into a particular location, feedback relates to the system of understanding that is being constructed for that location: for example, (1) haptic feedback in the form of vibration; (2) a noticeable sound emerges; and (3) the visual appearance of the toy or its representation in the virtual environment becomes noticeably changed. This feedback, which varies from situation to situation and location to location, becomes part of the learner's understanding. What is on its face a lesson in human communication is indeed a shared perception of the infinite possibilities of human existence manifested as stories, and thus the language of communicating about events and situations is realised as a bridge between entities such as humans and machines.

In conjunction with object localisation is the expression of those locations in relation to other tangible and virtual objects and locations. Configurations of objects are determined through natural language training of the system. In creating a language-learning lesson, the lesson designer trains the system to recognize those locations of the tangible object that correspond to the relative phrases.

For example, take the phrase with the general meaning: "the animal is in the water". The lesson designer must indicate those configurations of objects that are in agreement with the phrase, and there are additionally innumerable variations of the phrase that are also synonymic or holonymic with the phrase, such as: "the creature is in the water", "the big animal has jumped in the lake", or even "the foamy seawater readily engulfs the eager tiger". The building blocks of speech (such as adjectives and adverbs) add richness to an interaction experience that conveys, through language, more fine-grained and deeper meaning. Conversely, lending such tools for expression to the lesson designer complicates the task of training the com-

puter to identify agreement with a sentence that conveys meaning that corresponds to complex interrelationships between concrete objects and abstract concepts in space and time.

Additionally, the system must recognise when configurations agree. In our previous example, if the learner evidenced relationships between objects according to the sentence: "it went into and then out of the water", then it would not be construed as an acceptable understanding of that aspect of the multimedia lesson. The Thinking Head system provides feedback to the learner to reduce misunderstandings and to promote rapid comprehension and assimilation of ways of communicating.

8.8 Mini-Story

To provide the reader with a clear example that explicates language learning through mulsemedia, we here elaborate the short mini-story lesson called Spot the Hungry Tiger. This mini-story teaches the new phrases and words: "loaf of bread", "very dirty", and "wash up". We have elected to use several real world objects for the lesson: A toy tiger, a plastic banana, a real loaf of bread, and a manipulable LCD screen. Although this story is short, it incorporates the storytelling techniques of building toward a climax to maintain interest and promote emotional attachment with the movements of the story. Initially, it is our assumption that the learner knows a fair number of the words in the story, but not all of them. The first part of the lesson is vocabulary acquisition exercises, conducted through images, question and answer, repetition drills and mnemonics. Learners can associate gestures with phrases as a physical mnemonic technique. For example, the learner enacts the phrase "wash up" by an imaginary opening of a faucet and rubbing hands under the stream of water. By the simple monitoring of changes in hand positions, the learner is quizzed on their learning of this hand movement, and research into Total Physical Response (as presented in the literature review) has demonstrated its effectiveness—learners are able to naturally associate the kinaesthetic mode of expression with the words and phrases (Asher, 2000). Once the learner is confident with the set of three or four words and phrases, a procedure that takes up to five minutes, they proceed to learn the story. A brief version of the story is as follows: "Spot is a tiger. Spot goes to the market and eats a loaf of bread and six pieces of chocolate. The shopkeeper laughs because the chocolate makes Spot's face very dirty. Spot runs to the lake to wash up." This story, though short and simple, can be a memorable experience for a learner in the early stages of language acquisition. Although the story can be written as it was above, the focus is on the meanings and interrelationships, not on the sentences themselves. Therefore, as there are many possible ways to tell any story, there are also many versions of the story. Through repeated retellings, the story becomes more elaborate in accordance with the comprehension of the learners. For example, at the beginning of the story, the learner discovers that the tiger is small and lives with his aunt. This morning, the small tiger woke up in his blue and yellow bedroom feeling extremely hungry.

Later, learners discover that the market's name is Willie's Marvellous Grocery Store, and that Spot runs away to a unique location, perhaps the Rock of Gibraltar. Learners will not quickly forget these stories, in part because of their interesting and vivid details, nor will they forget many of the ways to describe the details. This is essential for language learning—the learner gradually builds a meaningful mental system of expression in the new language. Furthermore, as the system is adaptive, the learner adds some of the details in response to the system's questions. Giving the learner a voice in the process means that the learner feels a part of the process, and because the computer can store the details of learning in its persistent memory, subsequently these personalisations are incorporated into later language learning lessons.

8.8.1 Procedures of telling the stories

8.8.1.1 Vocabulary learning

The learner first learns what the new phrases mean through a series of physical actions. This is where we need the mulsemedia user interface. The learner moves the real objects around. When learners are learning the verb "laugh", the computer asks the learner to really laugh. This builds a strong association with the verb and a natural memory of it.

8.8.1.2 Comprehension checks

As the computer tells the story, it performs comprehension checks throughout. This means that the animated teaching head can ask a question. For example, there are a number of ways that the system can ensure complete understanding at the part of the story where Spot eats the piece of bread. For example, in one part of the story it asks the comprehension check questions: "What was it that Spot ate a piece of" or "Is Spot a dog or is he a tiger?" If the learner does not give the right answer, then an inadequate level of language understanding is apparent. In order to facilitate later retellings of the story, the system would then attempt to better acquaint the learner with the understanding of bread. For example, since a loaf of bread was prepared for the lesson, the learner can really take a bite of a piece of bread. By associating the taste of bread with the foreign language, we believe that the learner acquires the new word, grounded in experience. As the computer tells the story, the learner acts it out, as shown in Figure 8.4. The figure on the left illustrates the learner, who is holding a toy tiger. The colour signature of the tiger, derived from the hue and saturation, determines the yellow bounding box. The figure on the right shows the tiger running away from the shopkeeper.

Figure 8.4: The view of the tiger from the webcam. Note that the camera is looking from the front, so the virtual image seems reversed. Spot the Tiger heads for the lake in the virtual world, movements controlled by the actions depicted in the picture on the left.

8.8.1.3 Retelling the story

A significant goal of the TPR Storytelling system of instruction is to teach a learner to a point where they are able to produce unique utterances in the new language. In our view, language is a thought provoking and motivating tool. In addition to physical acting, the learner interacts with the learning space through speaking. Learners use multiple avenues to express the stories, including word webs, oral retellings, and reframing. Word webs provide a safe way for learners to reflect upon the interrelationships inherent to the story and the intrinsic relationships between characters and places. Word webs change over time to reflect the different parts of the story leading up to the climax. The learner tells the story in portions or in whole to the Thinking Head or to other learners. As stories naturally take place over time, reframing allows the learning of different verb tenses. Through repeated retellings of the story from the perspective of different characters in the story, learners acquire an understanding of the first person, second person and third person. To act out the story, the learner uses a device with a small screen to tell the story. The learner holds the moveable device that represents the shopkeeper, who is the manager of the grocery store known as Marvellous Market. The learner sees three types of facial expression appear on the small movable screen, and by selecting the appropriate one, the grocer laughs heartily in response to the dirty face of the tiger.

8.8.1.4 Writing on the whiteboard

Finally, to indicate their accomplishment learners write their versions of the story. Our system requires low-cost webcams, calibrated to acquire words written on a standard whiteboard using a marker, with both off-line and on-line writing recognition algorithms designed for this purpose (c.f. Gustainis (2009); Sydlowski (2009)). The system records the writing of learners, and a natural language system could be used to recognise common errors made by second language learners. Learners in the early stages of language acquisition may simply copy the majority of their mini-story, while replacing key phrases with their own words. For example, a learner may write a story about a small white dog that goes to the market. This story is similar to the story "Spot the Thief" that was learned in the lesson, but allows the creative expression within a familiar framework. As learners become more confident in their language abilities, their stories in the new language become creations that are more individual.

8.9 Discussion

Despite the advantages of new ways to interact with the computer, it is important to keep the educational goals in sight. Learning a new language is learning a view of the new language as a system that represents changing relationships within the real world. Accordingly, to provide learning experiences that are transferable, it is relevant to consider new ways of interacting with educational media, since learning improves, particularly when more senses are involved. Not all learning must occur through the strict processes of formal schooling. As (Papert, 1980) suggests, microworlds serve as "incubators of knowledge". Learning the particular details of a language in the certain context of a specific story implies that learners are gaining their knowledge through direct experience. Learning the details of the story facilitates a natural acquisition of the language of the story, and provided that the designers choose the language that is frequently useful for conveying these stories, learners gain facility with the language of the stories and they also naturally gain the rudiments necessary for fluency in the language. For efficient and natural teaching, we involve the learners physically. The physical response of a learner becomes an anchor on which the learner can hang more language. For example, we might ask a learner to associate a certain action with a word that they are learning. For example, if they are to remember the word "mobile phone", they would put their hand to their ear as though they were talking on the phone. When they respond with their body movements, what they are envisioning is a certain kind of object. This type of memory is much deeper than the simple memorization of how to spell the word phone. The learner can reach out to show that they know what the phone means, and the computer system can also know that the learner understands. In this manner, it is possible to hold the language learner accountable to internalise the language through the muscle movements. Likewise, expansion of

our system affords interaction with learners in other learning spaces. As learners improve their language skills, their lessons will include virtual tours of real locations in real-time or historical time, which allows for interaction with humans outside the learning spaces and with embodied conversational agents.

8.9.1 Challenges

As mentioned in the literature, some research explores giving learners the opportunity to create their own stories. A significant challenge that arises is that the space of understanding expands beyond the initial parameters. In the StoryRooms of Alborzi et al. (2000), for example, children would bring their own toys from home to augment the physical environment. In terms of implementations, the sensors of the room must have the capability to learn new objects without a great deal of effort on the part of the lesson designer. One might argue that the best way to design a lesson would be to teach a new story to the system, which would subsequently connect the details it learned from the story to the stories it had previously learned, interacting with objects, and asking questions of more knowledgeable experts to confirm its knowledge. In the same way that the system provides feedback to the learner to create a seamless experience for the learner, the lesson designer provides a story and sufficient feedback to create a suitable story environment in the system. On the other hand, unforeseen circumstances of language arise when the system is to produce novel language and interactions between objects, even when it is in the framework of previously known stories. Adding to the challenge, learners that incorporate ideas from outside the framework of existing stories create unforeseen situations that are difficult to test, and are therefore unknown until the problems have arisen. For example, plural nouns in English typically take an –s suffix. However, if a student wishes to have a story about four sheep, the system must be able to deal with this situation. We see this as one of the significant challenges to scalability of our system. Whether it is beneficial to give options to learners if the system occasionally produces incorrect language as a result is an unanswered research question for future research.

8.10 Our Related Work

Our work with the interactive Thinking Head and hybrid world harnesses the tactile aspect of handling real-world objects within three-dimensional virtual scenes. We seek to understand how modes of physical world kinaesthetic interactions can enhance human-machine interactions, and more specifically, how real-world experiences of users connect with gains in second language learning. This chapter reflects on the Thinking Head for language learning. Others in the lab have similar goals and brief descriptions of the work of the Flinders Artificial Intelligence and Language Technology Laboratory were included in the published chapter. Some members of

the lab collaborate with the Flinders University Medical Devices in addition to our work in autonomous robot teaming. In addition to the work presented in the current chapter for a mulsemedia environment for storytelling, we here present a number of our related projects of interest to the field of mulsemedia.

8.11 Conclusion

Our quick sweep through the larger picture of hybrid reality and our detailed focus on a language teaching application illustrates how the manipulation of tangible real-world objects, including toys and mini-screens, provides a powerful learning experience. Learners use our mulsemedia system to ground their newly learned language skills in compelling stories that engage their senses and involve them in the retelling. Comprehension checks, expressed bimodally in natural language descriptions and in virtual reality as relationships between tangible objects and places, ensure maximum learning of subject matter. A key to success is the telling of stories that are interesting and novel. Through sensors and appropriate questions, the system must be able to maintain the comprehension levels of the learner at high levels. In the future, media will improve along with advances in computer-related technologies and input and output devices, and accordingly, we expect to see an increase in development of tangible interfaces for computer-based story environments.

8.12 Acknowledgements

We would like to thank the following for their review and insight: Fatemeh Khazar and Sean Fitzgibbon. Additional thanks to William Newman, Daniel Franzel, David Gustainis, and Adrian Sydlowski for their contributions in their Honours projects.

Chapter 9

An Approach to Accent Visualisation for the Reduction of Vowel Pronunciation Errors

9.1 Statement

The dubbing activity in Chapter 7 revealed that learners are keen to receive feedback on their pronunciation. Speech recognition works when the learners have acquired sufficient language capability, but speech recognition alone does not let learners know about the quality of their speech. Recognising this gap, the KNPT research from earlier in this thesis may be situated in an environment that allows for pronunciation feedback. Thus, bringing together the research in the first part of the thesis with respect to the language learning considerations of the second part culminated in the production of the paper in this chapter. It is envisioned that this could be used to extend the storytelling environment described in Chapter 8 or be integrated into video game based learning that targets specific pronunciation issues.

The system framework was developed by Tom Anderson in consultation with David Powers. This paper was written by Tom Anderson with a section on Form Focused Instruction written by Barry Lee Reynolds. Tom Anderson edited with proofreading and feedback from Barry Lee Reynolds and David Powers. The short paper was accepted for the International Conference on Computers in Education which occurred in December 2017.

9.2 Authors

Anderson, T. A. F., Reynolds, B. L. and Powers, D. M. W.

9.3 Abstract

In this paper, we introduce a novel mechanism for pronunciation evaluation for use in a computer-based application for the reduction of vowel errors. Unlike modern pronunciation feedback, which relies on the charts used by linguists or simply highlighting of incorrect segments, our two-dimensional maps situate the utterances produced by the learner in the context of the phonemes of the target accent.

9.4 Keywords

Pronunciation, phonemes, technology-enhanced language learning

9.5 Introduction

The concept behind this study is towards enhancing the activity of pronunciation practice. Learners know that they should devote time to pronunciation practice; however, efforts to substantially improve pronunciation require not only many sessions of practice but also integrated feedback. Learners need to understand how their pronunciation is improving, and the speaking quality needs to be reflexive. This means that learners should be able to practice the same thing many times with feedback that encourages the learner to alter their behaviour. Such practice induces the frequency effect, whereby "the greater the practice, the greater the performance" Ellis (2012).

A human teacher can provide targeted feedback to a learner, but there would be many advantages to computer-based training, including cost and repeatability (Chan et al., 2006). In mobile apps for language learning, the aspect of pronunciation is often restricted to reading (both alphabetic and phonetic) and listening. Reading and listening are necessary components for language learning; however, allowing users the opportunity to produce speech and providing visualisations that aid their understanding of their pronunciation errors may enhance the model of the student's current communicative abilities.

9.6 Literature Review

9.6.1 Pronunciation

The typical feedback for pronunciation provided by language learning software has room for improvement. A common mechanism, adopted in software such as Rosetta Stone, a leading

language learning software, is providing spectrographs, voice contours for visual inspection of the voice patterns (Witt, 2012), but learners are not experienced linguists, so may miss the point of what such representations provide. Feedback based on automatic speech recognition provides an indication of which words were not pronounced correctly, so this gives learners a better idea of what to focus on.

Most state-of-the-art pronunciation modules use automatic speech recognition with posit (ASR) (Golonka et al., 2012). Typically, these systems, such as EnglishCentral or Spexx, identify those sections of utterances that are less than ideal (see Witt (2012) for a more comprehensive review of these programs and more). However, due to the nature of current pronunciation training systems, they often do not provide an indication of how the sounds of the language relate to each other, or how the current utterance may relate to other similar utterances. With an aim to improve computer-based pronunciation instruction, in this paper we present a biofeedback method that helps the learner to visualise their accent and how it fits into the context of the phonemic inventory of the language they are learning.

In contrast to systems that do not embed phonemes in their context, we have developed a neural network approach for visualising pronunciation. A Kohonen neural phonetic typewriter (KNPT) learns to represent the sounds of speech on a two-dimensional map with similar sounds located near one another Kohonen (1988, 2013). The underlying principle of the self-organising map is that information is arranged topographically, such that neurons that are near one another represent similar information. This allows viewing phonemes in relation to one another, and if necessary, by using an ensemble of maps, narrower contexts can also be visualised.

A two-dimensional map display allows the learner to get comfortable with a new representation of their voice. Although training a KNPT system can take a significant amount of time, once the values have been learned, it is very quick to classify the different sounds in a stream of speech. The learner can say something and the display will show the current state of their voice, along with the trajectory of the recent path of their voice. As neurons corresponding to similar inputs are located on nearby regions, a trajectory will generally pass from one zone to another through a transition zone. By tuning the maps, representations of voices and accents in the context of an individual speaker or an accent group are generated.

9.6.2 Form Focused Instruction

In recent years, isolated form-focused instruction has received renewed interest and a welcome response by second language acquisition researchers (Spada & Lightbown, 2008). Isolated form-focused instruction for pronunciation errors could be helpful in deterring first language interference with second language pronunciation. In terms of the kinds of mistakes that occur in pronunciation, Witt (2012) differentiated between phonemic and prosodic errors, the former being our current focus. Phonemic mistakes may arise in two forms: (1) severe - a phoneme is

replaced by another, omitted, or an extra phoneme is produced; or (2) accented - a phoneme is pronounced with an accent. Its sound is thus different than a native speaker would produce. Both types of errors may affect the intelligibility of the learner.

Usually during form-focused instruction on such pronunciation errors, the classroom teacher provides oral corrective feedback in the form of recasts (implicit feedback) or metalinguistic explanation (explicit feedback). Although both are frequent occurrences in the second language classroom, explicit feedback has been shown to have a greater effect on language acquisition (Ellis et al., 2006). However, in the limited class time a teacher has with students, all pupils cannot be given corrective feedback on their oral language production.

The effects arising from limited time for pronunciation feedback are sometimes alleviated by pairing language learners with tutors outside the classroom, which can result in not only the learners perceiving themselves as having improved but also showing said improvement on formal assessments (Lynch & Maclean, 2003). Still, this is not always a practical option since it cannot be guaranteed that a more capable peer or tutor can be secured for every language learner.

Form-focused explicit feedback given by the computer is a probable solution to this conundrum. In addition, one of the benefits of oral corrective feedback provided by the computer is the reduction in the potential of affective damage that can occur when language learners receive feedback from a teacher in front of their classroom peers. Language learners have an emotional response to the feedback delivered by teachers and when this response induces anxiety, the potential for negative effects on language learning increases (Agudo & De Dios, 2013). The computer can make accessible the type of feedback needed by all language learners within a comfortable context and environment.

9.7 Overview

A mobile application is used for accent reduction. A system flowchart of the system is depicted in Figure 1. For directed learning, a prompt is selected for a learner based on the system learner model. Note that learners may be involved in selecting the order of prompts, or produce spontaneous speech. Speech is evaluated in the context of the accent of the speaker, with a goal to provide feedback quickly, within 300 milliseconds. The learner model is updated as interaction increases with the system.

9.7.1 Improving the Pronunciation of Vowels

As vowels are produced by an uninterrupted outflow of air, the sounds of vowels appear on the maps in continuous trajectories (averaged as traces with thickness reflecting variance).

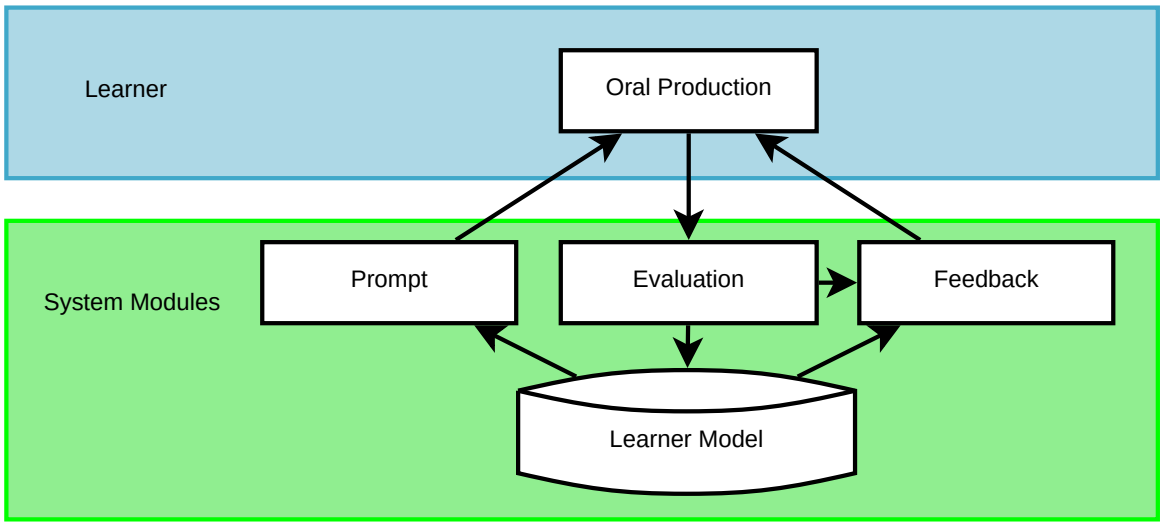


Figure 9.1: System Flowchart: Interactions between system modules and learner.

There are multiple ways to get similar sounds. However, the trace of a target accent for the pronunciation of the vowels is narrower than the trace for the corresponding foreign accented pronunciation. In other words, the sounds of the vowels that the foreign speakers produce are naturally and consistently out of the target zone. A speaker's first task is thus to pronounce the vowel closer to the target. The pronunciation of steady-state vowels can be changed and shaped by moving various articulators. The idea is to get the speaker to produce vowel sounds in isolation that are more similar to the target pronunciation. Next, speakers should produce the vowel sounds accurately in the context of isolated words.

In continuous speech, when many words are strung together, the pronunciation of each particular sound is much less important. In contrast, when a single word is pronounced in isolation, each word and each of its constituent phonemes is expected to be pronounced clearly and appropriately. The eventual goal is to aid the speaker to produce speech that is more intelligible or less accented. It is not guaranteed that reduction in accentedness in the pronunciation of isolated words will cross over to the regular speech patterns. However, during the process of learning how to make the sounds of an individual word be closer to the target pronunciation, the learner will gain an understanding of how to position their articulators to produce certain sounds that were previously less familiar to them. The next step would be to help learners to understand how the sounds of their own voice, in continuous speech, can be shaped to produce speech that is more like the target, and to give them an ability to practice shaping spoken words with less of an accent.

9.7.2 Speech Representation

Ensembles of self-organizing maps (SOMs) were trained on the voices of native speakers (general Australia, educated Melbourne) and a target group (Chinese background) using data from the

AusTalk corpus (Burnham et al., 2009, 2011;Wagner et al., 2010). The maps provide a visual representation of the speech of the learner in phonemic context. The speech of the learner is pre-processed into 39-feature mel-frequency cepstral coefficients (MFCCs), commonly used in automatic speech recognition. The MFCCs are then compared to codebook vectors. Although the initial training of the system must be performed offline, the evaluation and updates to the learner model can be performed near real-time. For more details, see Anderson & Powers (2016).

9.7.3 Implementation

- **Client** —A microphone is used to obtain audio input. This may be the device microphone of the mobile computing device (often an array microphone in modern phones and laptops), but an external microphone may result in better sound quality.
- **System** —As the learner speaks, their voice is analysed using the system and feedback is presented. Speech is shown on a map using a selection of pronunciation samples, as in Figure 2. Learners interactively explore the differences between how their speech and pre-recorded utterances are rendered, thereby improving the understanding of their speech in the context of the speech of others.
- **Gamification** —Games can increase motivation. In a meta-analysis on digital game based learning for English as a foreign language, Chui et al. (2011) found that language learning games positively affect learning but meaningful and engaging games can yield a greater positive effect than drill-and-practice games. When learners are immersed in digital game based learning environments for more than a month, they gain automisation of language knowledge (Kao, 2014). We feel that automisation is a necessary aspect for improving pronunciation. In terms of future directions for our research, we aim to apply our SOM-based phoneme visualisation to game-based learning. The difficulty level can be increased by chaining multiple utterances together, providing fewer hints, or by narrowing the tolerance, which requires a learner to produce speech closer to the target.

9.8 Conclusion

In this paper, a mobile learning environment for pronunciation was proposed. Learners receive real time feedback about their speech and achieve goals in a video game by modifying their speech. Learners are presented with lessons sequenced so that they may gradually improve toward the target. In future research, the usability of the system for accent reduction would be evaluated, along with additional aspects for promoting learning such as gamification.

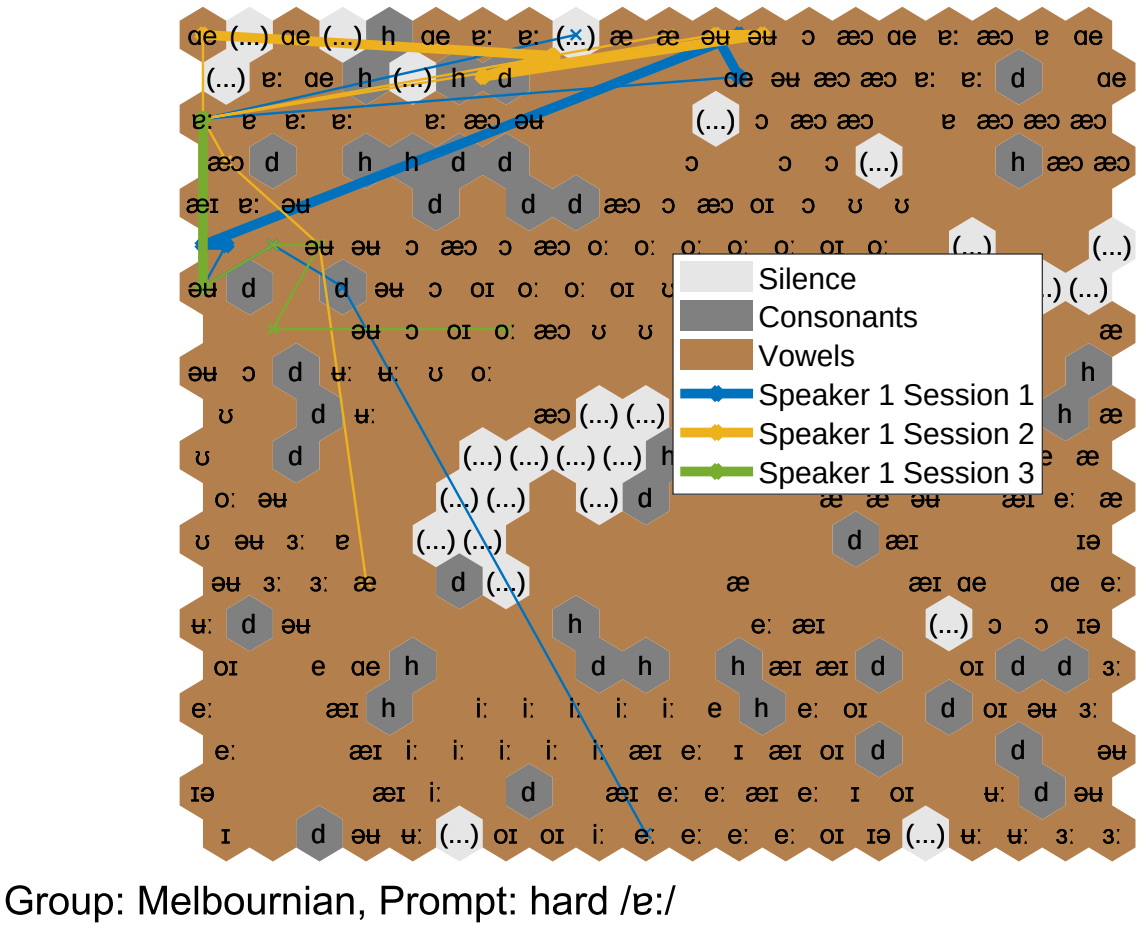


Figure 9.2: Trajectory analysis. The map depicts the trajectory of the learner's voice in the context of the different phonemes of the lesson (currently /ɐ/ as in "hard").

9.9 Acknowledgements

The AusTalk corpus was collected as part of the Big ASC project (Burnham et al., 2009, 2011;Wagner et al., 2010), funded by the Australian Research Council (LE100100211). See: https://austalk.edu.au/ for details.

Chapter 10

Conclusion

10.1 Summary

The work presented in this thesis was the result of travel along many roads that were laid out by previous researchers. The use of the Self-Organising Map (SOM) algorithm for vowel classification and analysis is considered, as are Embodied Conversational Agents (ECAs) and games for Computer Aided Language Learning (CALL). In the course of this research, a number of novel findings were brought out and explored.

A considerable challenge is how to present a cohesive understanding of factors that are important for this area of research. As speech underlies our verbal communication as humans, many have conducted related investigations into what makes our sounds fit into one dialect or another. The field of SOMs for speech is surveyed in Chapter 3. In addition to providing a literature review, this thesis is comprised of two main parts. The focus of the first part was on using the Kohonen Neural Phonetic Typewriter (KNPT). I explored the single layer KNPT architecture in Chapter 4 and a multiple layer architectures in Chapter 5. In Chapter 6, I investigated the capabilities of KNPT for voice activity detection (VAD), which clarifies the results of previous chapters and establishes it as an important consideration for KNPT practitioners. Finally, I presented some thoughts on data management for speech with KNPT in MATLAB. The second part of the thesis was aimed at enhancing the field of CALL through the use of ECAs. In Chapter 7, I addressed pronunciation learning with the activity of dubbing for ECAs. Next, in Chapter 8, I described a holistic and multisensory approach to CALL with an ECA. Finally, in Chapter 9, I explored the most recent work that ties together the visualisation of pronunciation using KNPT with CALL.

The general literature review introduced the concepts of speech corpora and data management. A main approach to the analysis of speech is the unit of phonemes. I introduced how speech can be captured with a microphone and then transformed into features for analysis.

The approach I used was to divide the speech stream into overlapping windows, then apply an algorithm to each window to arrive at a feature representation. In this thesis, I reported on speech as spectrograms and mel frequency cepstral coefficients (MFCCs). Using this approach allows the sounds of speech to be compared with a distance measure. I described the statistical tests that were used in this thesis: (1) bookmaker scores, a statistical measure (developed and championed by my PhD adviser, David M. W. Powers), and (2) cross-fold validation. These unbiased statistics allow us to make fair comparisons between different tests.

I used an operational definition of the KNPT as the frame level analysis of phonemes by means of SOMs, introducing the acronym KNPT. I conducted an extensive literature review of this area that covers most relevant work conducted in this area since the 1980s. I described how KNPT works and different factors that have been shown to affect its results, such as size and shape. I then addressed how KNPT has been used, which has had applications for automatic speech recognition (ASR) (primarily phoneme recognition and phoneme alignment), for dysphonia recognition, as well as for non-phoneme vocalisations and VAD. I covered the different architectures that have been used: single layer, ensembles, and hybrids. I provided an overview of the languages that KNPT has been used with. Finally, I found that normalisation for KNPT is often overlooked or not reported in the research literature.

To enable the analysis of speech with a single layer KNPT architecture, I used the Microsoft Kinect audio array to create several single-speaker corpora with a Californian accent for the purpose of evaluating vowel phonemes. I showed how spectrograms may be used with KNPT, i.e. using speech in the spectrogram feature representation, unsupervised KNPT is able to classify corner vowels, with different effects from map size, initial/final neighbourhood size, and normalisation.

Next, I showed that for a 25x25 map, the MFCC feature representation clearly outperforms spectrograms on all measures for all phonemes and silence. I showed, through experimental results, the effects of normalisation of MFCCs using KNPT for vowel analysis, as well as the feature selection of MFCC deltas. Although the results specifically indicated that adding MFCC deltas deteriorate the classification abilities, the approach I presented can be followed more generally to perform the evaluation of other feature representations.

I evaluated the KNPT for monophthongs and diphthongs of a single speaker, presenting classification results at the frame level and utterance level for a range of map sizes from 2x2 to 100x100. It was found that the inclusion of diphthongs deteriorates the classification performance of all vowels, but that maps 36x36 and above were able to successfully classify all phonemes at the utterance level for both a subset of just the monophthong vowels and also for a full vowel inventory.

I also briefly described how monophthongs and diphthongs can be shown on KNPT maps, which can also be performed in real time for the purposes of CALL. Although I did not specifi-

cally investigate non-verbal communications in the work for this thesis, I suggest that the findings (map size, hierarchies, normalisation, and inclusion of silence) should also have significant applicability to the wider domain of KNPT speech research that addresses infant vocalisation and animal communications.

Moving towards a multi-layer architecture, I presented a comparison of two approaches for improving the classification performance of maps. One approach that has been suggested for improving the results obtained with SOMs is to use auxiliary maps to disambiguate those phonemes that have the most confusion. I proposed a different method, which uses the first map only to classify /(...)/, /h/, /d/ or vowels. An experiment with the Austalk corpus showed that both methods significantly outperform a single layer SOM. My method was also marginally better than the method of using auxiliary maps to disambiguate confused vowels.

Next, I investigated whether silence detection could account for this difference in performance. In particular, I revealed that VAD in the context of SOMs is a very important consideration. Yet in the literature, it often seems to be overlooked. Many researchers used fully annotated datasets, allowing them to omit all pauses between utterances. Others used VAD algorithms to perform a similar function. And finally, perhaps because this step is not clearly described, there are also researchers who have not performed VAD and encountered difficulty with the classification results. The results showed that VAD can be performed by KNPT even with small 6x6 maps.

The second part of my thesis addressed computer aided language teaching with an emphasis on ECAs. First, I presented a system for dubbing an ECA. This system enabled language learners to perform the activity like video dubbing, but instead of using YouTube videos, we used an ECA. This allows language learners to practice synchronising their speech. This ECA dubbing system was tested by my colleague, revealing that those who used the system appreciated its value for pronunciation practice. However, they also revealed that automatic pronunciation evaluation would be a very desirable enhancement to the system. Indeed, it was this finding that motivated much of the first part of this thesis into the evaluation of pronunciation using KNPT.

Next, I described a holistic overview of how ECAs in a hybrid world (incorporating aspects in both the virtual and the real world) can mediate multiple sensorial media (mulsemedia) learning of language. I introduced my implementation of this and described how a foreign language is learnt through acting out mini-stories with real-world objects. The system uses Microsoft Kinect video sensors to detect the movement of people and objects to enable hybrid language learning. Such a multisensory environment affords a new type of situated experiential language learning with a computer through storytelling.

The chapter on mulsemedia for CALL was published prior to much of the other work in this thesis. It therefore emphasised receptive language (listening) with the student primarily

communicating understanding via the interaction with objects. KNPT pronunciation evaluation would be a natural addition to the capabilities of the mulsemedia system, allowing it to better harness feedback to improve both receptive and productive language. To address this, I introduced a system for pronunciation learning that brings learners to better understanding through KNPT visualisation of pronunciation.

In conclusion, I have presented my research that has been directed towards enabling Computer Assisted Pronunciation Teaching (CAPT). I addressed gaps in the literature for making KNPT operational for the evaluation of pronunciation. I furthermore expanded on knowledge of how KNPT and ECA can be used in language learning contexts.

10.2 Directions for future research

In addition to the work described in this thesis, a number of experiments using SOMs in the phoneme context were conducted that were not particularly successful. I will briefly summarise them here.

I tried rescaling trajectories so that they would be equal length to determine the difference between two utterances. However, a good method of classification of utterances using the interpolation technique was not determined. Beyond this, using a map to determine the difference between two utterances is no more than using maps for vector quantisation. This opens the question: Why not just do the differencing directly? The self-organising map is a useful tool for this task only if the self-organisation reveals patterns in the original encoding of the data. If the vector quantisation of the self-organising map is a useful intermediate step, then to achieve speed without losing accuracy, Learning Vector Quantisation (LVQ), may be a better choice. Where SOMs would be useful is in the fast labelling of data, but it seems that this benefit is not immediately practical for trajectory comparison, at least as far as those experiments got.

10.2.1 Consonants

A number of consonantal sounds in other languages differ from English. This causes difficulty for language learners. This thesis has, for the most part, restricted attention to vowels. This approach for accents is taken by many researchers and phonologists. However, it clearly avoids the difficulties of consonants. This is appropriate for accent reduction through the shaping of vowels, but not for learners who have not acquired the phonemic inventory of English.

The basic philosophy of focusing on vowels is rather than attempting to solve the bigger problem of unsupervised learning of pronunciation, to restrict the attention to a smaller class of phonemes that is well known to be a starting point for many researchers. Speech may be regarded as a stream of vowels, over which consonants are laid (Fulop, 2011). Accent (in

English) is primarily affected by the underlying stream. Although the proper production of consonants is essential to intelligibility, deviations from the target accent in the pronunciation of a consonant is seldom a subtle change, i.e. a person has it or they don't. For example, the voiceless dental fricative "th" /θ/ is often produced with the tongue between the upper and lower teeth. Yet in some accents, it may be replaced by a different sound, such as a voiceless labiodental fricative /f/, which made by restricting the air by moving the lower lip to the upper teeth. Since it a complete replacement of one phoneme for another that is necessary, my suggestion is that such a mapping is more complex than a two-dimensional visualisation would allow. The spectrum of vowels, on the other hand, approach the target relatively smoothly.

The vowels are produced with an unrestricted airstream, with gradual changes caused by movements of the mouth and lips. According to intuition, vowels are continuously similar to one another. I can produce any vowel and change the sound to another vowel smoothly. This differs from consonants, which are interruptions. Therefore, though I expect vowels to be represented smoothly on a map, I do not expect consonants to be arranged in such a way (in other words, I expect that the choice of feature representation will primarily determine their placement). The fundamental difference between the continuous airflow of vowels past the tongue and lips and the discontinuity of consonants suggests a different form of analysis.

The International Phonetic Association uses a chart for the consonants, but a quadrilateral diagram for the vowels. As I described in section 2.5, it was through the work of Joos, who brought the spectrogram as a tool for phonetics, the vowel triangle of classical phonetics became enhanced by the knowledge, allowing the vowel quadrilateral to be informed by formant maps. However, the phoneticians at the International Phonetic Association do not provide a two-dimensional visualisation of the space of consonants. The current method for the teaching of the production of consonants is generally to display a cross-section of the mouth with indication of which articulators are to move where. Perhaps a consonant map developed using KNPT technology could one day influence phonetic practice away from simply displaying the prototypical movements of the articulators necessary for the production of the target consonant. Extending the current work to provide evaluation of consonants is clearly an important consideration for future research, whether accomplished via KNPT visualisation or another technique.

10.2.2 Applying lessons from neural networks to KNPT

One approach I pursued with only limited success was to use a KNPT network as an input to Extreme Learning Machines (ELM) (Huang et al., 2012) and multi-layer perceptrons (MLPs). Although some researchers have reported results of SOM as a lower layer in hybrid architectures as I described in section 3.6.5.1, I suggest that these attempts seem to be more of a solution seeking a problem. SOMs may sacrifice classification performance, but they have the benefit of

producing topologically structured maps for visualisation. Efforts to use the SOMs algorithm for its classification performance without utilising the topological structure are a little puzzling to me. I see the value in pursuing hybrid KNPT approaches, particularly if the topology is maintained or improved upon.

As suggested in (Kohonen, 2014, p. 25), the SOM algorithm can be used alongside other techniques, such as k-means clustering, to achieve self-organisation. Furthermore, it is not necessary to continually run the the SOM algorithm towards such an end; for example, for k-means clustering, it is only necessary to run the SOM algorithm once every 50th iteration. The maps will self-organise despite running the self-organising map algorithm only occasionally. Additionally, beyond the SOM algorithm, unsupervised approaches to dimensionality reduction exist, such as t-Distributed Stochastic Neighbor Embedding (t-SNE) (Van Der Maaten, 2009), that may find suitable use in hybrid architectures, either alongside or in the place of KNPT.

Despite the long history of SOMs in phoneme recognition, the topology of the resulting maps has similarity with the topology of representations in the human brain. The function of topological representations in the brain is not well understood, as better classification results can be obtained if topology is foregone. I propose that a possible direction for research to explore topology is through the use of deep learning. Recent advances in deep learning have demonstrated the capabilities of many layers of neural networks in many areas, including ASR. Thus, one way to better understand KNPT might be to explore the representations in the lower layers of deep neural networks that are trained to produce the same outputs that were determined by SOMs.

10.2.2.1 Learning feature representations

It is possible to generate feature representations for speech using neural networks such as autoencoders and convolutional neural networks (CNNs). Both of these approaches can learn to represent the input data without requiring supervised training. Thus, one avenue for future research would be the consideration of feature representations that are learnt rather than produced by a time-tested, hand-crafted feature representation such as MFCCs that I used in this thesis.

An advantage of MFCCs over learning features is that the results are consistent, rather than consistent with the training data. In other words, autoencoders and CNNs take the input data and learn a representation for that data. When using such approaches, the question arises whether the transfer of learning is sufficient. If we learnt that these approaches worked for one dataset or language, we would not know that it would work for others. An algorithm such as MFCCs has a certain level of tried-and-true nature in that it has been used for many years for speech applications. The advantage is that for MFCC, we can provide any size input, down to the frame level, and receive a consistent result. An autoencoder must first be trained, using

many frames, and results will be based on the results of training. However, it seems clear that the future of machine learning is through learnt feature representations (Bengio et al., 2013), so understanding how to apply such perspectives on representation learning to KNPT (or whether KNPT is to be supplanted) is an open question. Unsupervised learning of feature representations is also a desirable for moving towards a better understanding of human language and cognition.

As I mentioned in 2.6.4, a significant disadvantage of MFCCs is that they are lossy, meaning that reconstruction of the original signal cannot be perfect. As a result, human-sounding speech cannot be synthesised from MFCCs alone (the results sound flat). A constant-Q transform such as reported in Schörkhuber et al. (2014) could be used in place of MFCC, although the MFCC algorithm is computationally less expensive and can run in real time. Thus efforts towards feature representations may involve striking a balance between the speed of processing and the quality of speech synthesis.

10.2.2.2 Learning vector quantisation

As I discussed in 3.6.2.1, LVQ is a supervised method. All work for KNPT that I have seen uses LVQ as a supervised technique. I would propose a method by which LVQ can be used as an essentially unsupervised technique: to use the neighbouring frames with LVQ to improve the classification; thus, to use the sequence information to guide training. Such an approach could be beneficial by guiding map neighbourhoods towards reflecting the time sequence.

Previous work with LVQ and KNPT used labelled data. A data point, associated with a known phoneme, was shown to the network. The phoneme label associated with the best matching unit (BMU) was compared with the known phoneme. If the labels matched, the codebook vector was moved towards the data point; otherwise, it was moved away from it.

An assumption that we may have is that within tens of ms, a neighbouring window will have significant information for the current window. As a result, couldn't we use the proximal frames to improve the SOM classification? This may not be a successful mechanism for fast changing signals, such as are generated by sounds of the stopping and starting of air in consonants. However, vowels are by definition produced with an open vocal tract. If changes in vowels take tens of ms to develop, without changing the other parameters of the KNPT system, it would seem possible to use the temporal information to tune SOM networks via an unsupervised LVQ technique.

There are a number of mechanisms that temporal information can provide for unsupervised LVQ in the context of KNPT. Firstly, there is an assumption that trajectories in the map should change slowly for normal vowel vocalisations. Hammer et al. (2005) outlines a number of SOMs with time series, including recurrent processing of time signals, but concludes that

there is no canonical model for such. The term "sequential LVQ" is typically used in contrast with "batch LVQ", and describes the order in which the reward and penalty steps are applied.

As I reported in 3.6.2.1, Kangas (1991) applied time-dependency to phoneme recognition in SOMs (which we call KNPT) by training a second map on a sequence classified by the first. LVQ was applied to improve the results, but it is understood that LVQ-tuning in the context of KNPT has been supervised method based not on time-dependency, but on known phoneme labels. The goal of such a system is to recognise the classes, not to ensure that phoneme clusters (which comprised of multiple neurons) have good qualities such as convexity and separability from other phonemes. However, I expect that some portions of phonemes should overlap. I speculate that it may be an improvement in terms of recognition of phonemes based on subtle distinctions but deleterious for highly similar sounds to be mapped together.

The LVQ can be understood as a system of penalties and rewards. A neuron is penalised if the classification is determined to be bad by moving its codebook vector away from the data point. On the other hand, if the classification is good, then is rewarded by moving its codebook vector towards the data point. The question is then, for an unsupervised LVQ, as the class information is not known for a given data point, how can the good and bad decisions be determined?

Firstly, we have some desirable properties of the resulting maps. We would like trajectories to move smoothly and gradually. Penalties can be incurred for those sequence classifications that introduce sudden shifts. Another property that may be desirable is straightness. We may consider a trajectory to have momentum. If the path is wiggly, it may simply be due to suboptimal matching. By incurring a slight penalty in such cases, the map can be shifted to better respond to sequential vowel information.

One interpretation of this method of unsupervised LVQ is that it works towards finding the classes (through temporal information) rather than separating known classes (as in supervised LVQ). This concept can also be extended to larger windows to promote movement of the speech trajectory. Just as neighbouring frames should correspond to small neighbourhoods, wider time windows should be encouraged to transit larger neighbourhoods.

In a way, this idea of unsupervised LVQ applied to MFCCs is an extension of the concept of deltas, which are approximated between subsequent frames. Similarly, deltas can also be determined from the two subsequent codebook vectors associated with those frames. When there is a disjoint between the derivatives of the data with the derivatives of the vector quantisation, it indicates that the classification has resulted in a different trajectory. A penalty applied in this case may thus aid the fit of the model.

10.2.2.3 Mini batch training

A recent advance in the related area of feed-forward neural networks is the usage of mini batches, which are subsets of the training data, as opposed to batches, which consist of the full training set. Mini batches are random subsamples of the entire distribution. As mini batches are smaller than the entire training set, which may contain redundancy, many of the characteristics of the dataset may be revealed in mini batches. The main advantage of mini batches is faster convergence, which means that training requires significantly less processing time.

The usage of mini batches for SOMs was alluded to in Vatanen et al. (2015), but as far as I am aware, an exploration of how mini batch updates can improve training has not been conducted for SOM or in the KNPT space. In this current thesis work, emphasis was not placed on improving training, so an exploration of how mini batch might improve training time was not undertaken. However, given that learning of normalisation parameters using mini batches has been demonstrated to be a powerful technique in feedforward networks citepIoffe2015, mini batches in SOM may not only increase convergence speed but also provide additional avenues for research.

10.2.3 Comparing maps

Can we compare different maps that were trained on different data? In addition to its use in classification, SOM also clusters the data. The SOM can appear in any rotated form (Kohonen, 2014). Different input data used for training will also result in different results in terms of maps, though we might be able to expect that similar input data should result in similar maps.

In order to evaluate different maps fairly, then, I propose that we should use a mechanism for rotating the maps to similar positions. In the classification of vowels using KNPT, we might take as the desirable configuration something similar to the standard vowel quadrilateral that is based on formants. Those types of rotations of the neurons of the map that do not affect the mapping of the input space to the neurons in the map change only how we define the positions of the neurons. As the positions of neurons in the map do not affect the mapping from the input space, this procedure does not necessarily change the maps. However, in practice, the maps currently in use do not have qualities that make them amenable for such fitting. In particular, a rectangular hexagonal map with constant width, has rows that are offset. I suggest that for hexagonal units, a hexagonal map would enable the kind of rotation that I am suggesting. Alternatively, a fine tuning procedure can be performed after a rotation that ensures any errors induced by the rotation are minimised.

To perform such a rotation we must first train the network. We then project a number of vowels, possibly those vowels that are most in correspondence with the corners of the vowel

quadrilateral, onto the map. The actual vowels to be used will vary from language to language, dialect to dialect. From there, we observe the locations of the vowels and compare them to the desired orientation. The MATLAB SOM Toolbox lacks the capability for rotating maps, though if we are content with losing data edge and row alignment, it is a simple matter to rotate a codebook. As far as I am aware, such a feature has not been implemented for KNPT, but I suggest that it could enhance our ability to evaluate clustering.

10.2.4 Visual speech and enhancing ECAs

The research presented in this thesis has always been envisioned to be further combined with the visual aspects of speech. In particular, as the Austalk corpus matures, we are gaining better access to video data from Australian speakers. Video processing techniques are being developed to identify aspects of visible speech, including jaw, lip, and tongue movements. These aspects can be combined with the audio analysis techniques presented in this thesis, to enhance and supplement the evaluation of speech.

In addition, I propose that KNPT, particularly with the addition of visual data, could be applied to efforts that enhance the quality of performance of ECAs. Additionally, improving upon the pedagogical agent that combines the usage of KNPT with ECAs for accent reduction has a number of open questions for future research, including: an evaluation of the effects of providing KNPT visualisations to language learners, the development of effective pedagogies for improving accent beyond that of dubbing presented in Chapter 7.

Appendix A

Data Management

In order to conduct experiments with speech, it is necessary to manage the data. The Austalk corpus was used in the experiments presented this appendix. It was created as a large corpus of Australian speech. The entire corpus contains many different utterances produced by around a thousand speakers. Austalk stores the data in a nested folder, one folder per speaker. Approximately one thousand speech files were created for each speaker. The structure and grouping of files is described in Cassidy et al. (2017). A speaker's folder contains one folder per speech file. In this thesis, only the 16 kHz speech files were used, but . Each of these folders contains the speech file and a text file that contains metadata that describes that file, such as who said it and what was said. Additionally, some (but not all) of the speech files are associated with annotation data stored as Praat TextGrids, also stored alongside the speech file.

The SOM data structure allows a label to be associated with each data vector. However, with a volume of data that Austalk, it is necessary to keep be able to keep track of more information, i.e. the data with its metadata, such as who spoke and what they said. In the remainder of this appendix, I will introduce my approach to managing the Austalk data files for processing within MATLAB, my method for importing and accessing Praat TextGrids, my work towards creating a phonetic lexicon of Australian English.

A.1 Tidy Data

In order to handle the speech data and annotations, I implemented data management in MATLAB. Due to the lengthy nature of the data management class and its specialised purpose, the functions of the custom object I implemented for data handling is not included in its entirety in this thesis; only its class definition and a description of the functions that it provides will be presented. The complete source code is provided on my GitHub repository at https://github.com/twocs/alveo-matlab.

A significant portion of the work was getting the speech data into a condition that would enable me to run the experiments. A programming environment such as MATLAB is designed for the general problems of matrix multiplication and plotting data, leaving the user to manage the details of managing the data. For speech data there are various units of analysis, including speakers, utterances, annotations, phonemes, and temporal signal data (e.g. frames). It is necessary to use the data frames in different subsets, such as training and testing sets, or different demographic groups, while preserving access to the details, such as which speaker produced them, what was the target utterance, and for some of the data, phoneme annotations.

The Austalk data structure contains which audio files were stored in one folder per utterance in a folder for the speaker. As I described in section 5.2, for speech analysis, the continuous speech data at 16 kHz is broken into overlapping frames. The audio data may also be annotated. The phonemic annotation contains Speech Assessment Methods Phonetic Alphabet (SAMPA) characters that indicate which phonemes were produced and the times at which those phonemes start and end. Thus, to determine for each frame of audio data, it is necessary to cross reference the start and end times to determine which phoneme annotation is relevant to the frame. Although the storage capacity of our computers is quite large, if each phonemic annotation is copied once per frame, a dataset can quickly grow very large. Additionally, different feature representations may be required for each experiment. Therefore, I found that it was practical to retain the speech data and associated files in the original structure, but to enable access to that data on demand in parallel arrays of feature representation or annotations, with one observation per row. Additionally, I enabled the reverse procedure, which transforms results of the classification to be reshaped back into the original data structure. In such a manner, it is possible to maintain an association of data frame with annotations or classification results.

After working with Austalk data in MATLAB for a considerable amount of time preparing data for processing, I realised that I had basically reinvented a specific type of database. Reading about the struggles of others in using data for machine learning helped me to refine my understanding of the methods I created for handling audio data. Although I created these procedures for data manipulation on my own over months of iterative improvement, I subsequently reviewed similar procedures for machine learning data. Working in the language R, a matrix oriented language like MATLAB, (Wickham, 2014, p. 2) calls this type of data structure "tidy datasets provide a standardized way to link the structure of a dataset (its physical layout) with its semantics (its meaning)." In terms of database structure, the data is stored in Codd's 3rd normal form.

In contrast with my method, which uses parallel matrices of rows of observations, Wickham argued for using a single table. Using parallel matrices simply means that different datatypes (numbers or phonemes) can be stored independently, allowing for slightly more efficient procedures in MATLAB. It also allows for a reduction of certain information (such as filenames), because if copies of all data such as the filename were included with the observations at the

frame level, the data size could exceed the RAM working capacity. By allowing for converting back and forth between the original data structure and the structure for machine learning, the semantic information about the data could continue to be associated with the data frames but without the overhead of multiple copies of information that is seldom used.

A.2 Phonetic annotations

A significant portion of my research uses the speech of Australians that was collected in the Austalk speech corpus. In particular, I used for the labelling process some of the Austalk speech files that were fully manually transcribed with time alignments. Manual transcription of the Austalk corpus was a laborious and challenging process (Cassidy et al., 2014, p. 9). Indeed, time-aligned transcription of speech is a lengthy process, which even with specially designed software takes professionals significantly longer than it took the speakers to produce the speech. An active area of research effort is the automatic annotation of Austalk, with ongoing efforts described in Cassidy et al. (2017) and Cassidy et al. (2014).

SAMPA is a phonetic alphabet that may be entered into a computer entirely in ASCII. Prior to the widespread use of encoding systems like UTF-8, ASCII input enabled a way to represent International Phonetic Alphabet (IPA) symbols by using the typewriter characters, "a-z, A-Z" (alphabet, both lowercase and uppercase), "0-9" (digits), and symbols including "@", ":", "ˈ, and "". For the phonemes /ɐ/ and /ɐː/, Austalk uses the X-SAMPA (Extended-SAMPA) character "6".

Additionally, the symbol <p:> was used as an undocumented addition in Austalk annotations to indicate silence. In IPA, a short, medium, or long pause may be represented as /(.)/, /(..)/, or /(...)/, respectively. As the pauses in the speech files that were used for the main portion of this research were not located between words, but were at the beginning and end of a speech file, I used the long pause symbol /(...)/ exclusively as it may be understood as not communicating information, as some pauses in speech may convey, but rather preceding or following canonical form hVd utterances. Therefore, all silence was consistently represented with the same symbol.

To allow for making conversions from SAMPA (and a few other symbols) to IPA in MATLAB programs, I created ta_sampa2ipa_map.m, a MATLAB map object, which enables access to the mappings, shown in Appendix B.2. The primary use of this object is for the creation of figures to show KNPT maps with IPA labels. It may be of interest to other researchers who need to make similar conversions in MATLAB. As far as I am aware, no similar mapping for MATLAB has been released, but there are several in other languages that could be used from MATLAB as an alternate method. I also created the reverse mapping from IPA to Austalk SAMPA, which is virtually identical to ta_sampa2ipa_map.m but in reverse, and may be found on my

GitHub repository at https://github.com/twocs/alveo-matlab.

A.3 Praat TextGrids in MATLAB

Praat is a popular software application that was designed for the speech sciences (Boersma & Weenink, 2017). Praat (and for my purposes, more specifically Praat TextGrids) are tailored to use with Praat and are not immediately accessible in MATLAB, the primary programming language used for the KNPT research presented in this thesis.

Praat Textgrids were created from the manual annotation of Austalk speech files, one TextGrid per file. In order to process these files, I created the MATLAB code. Although this code is short, it may be of interest to other researchers who need to access Austalk annotations in MATLAB. I have included it in Appendix B.4. The loaded data is stored as a object of class ta_praat_textgrid, which can then be stored in a cell or table structure.

To use, ensure that the Praat TextGrid files are encoded in UTF-8. The specific structure of TextGrid files is fixed for Austalk, so if other datasets were used, the strings would need to be modified appropriately.

At the time that I explored the use of Austalk annotations, I was unable to locate any suitable solutions for loading Praat TextGrids into MATLAB; hence, I programmed a class definition myself. TextGridTools (Buschmeier & Wlodarczak, 2013) is free software that enables loading of Praat TextGrids in the Python programming language. It enables loading, manipulating, and saving TextGrids. My approach is similar, allowing access to tiers, intervals, and items (with start and end times), but it works in MATLAB rather than Python.

After I had created and refined my data storage and handling procedures for manipulating Praat TextGrid files in MATLAB as I reported above, Bořil & Skarnitzl (2016) released an interesting package for MATLAB called mPraat that allows for access to data that was stored in TextGrid annotation format. This tool also provides the ability to write TextGrid files as well as an interface with Praat itself, which can enable built in tools that are useful for the speech sciences.

In contrast with the narrow approach to that I used for my data handling, which enables specific capabilities that are tailored to the KNPT application, mPraat takes a more general approach. Annotation in Austalk took a rigid approach resulting in TextGrid files that were consistent (namely, they always had three tiers). Future work could enable the use of mPraat (or TextGridTools via a Python bridge) with my KNPT workflow, which would be of particular benefit if annotation files were not as consistent.

Appendix B

MATLAB Code

In this appendix, which consists of some of the MATLAB code that I authored towards the publication of this thesis, I include those scripts that are directly appropriate to discussions as referenced in the body of the thesis. More up-to-date and comprehensive MATLAB scripts and data objects that are useful for the manipulation of speech files and metadata in the Alveo database are maintained at my GitHub repository (https://github.com/twocs/alveo-matlab). Additionally, a few other repositories that I maintain contain relevant work to this thesis: a small number of bug fixes to SOM Toolbox (https://github.com/twocs/SOM-Toolbox) and the creation of an Australian pronunciation dictionary (https://github.com/twocs/australian-lexicon).

B.1 Object for storing and accessing Austalk corpus

```
1  classdef ta_data_class < handle % hgsetget
2    %TA_DATA_CLASS Object for storing data from the Austalk corpus
3    %  Contains helper functions so that MATLAB idiosyncracies don't cause
       problems
4    %  For struct2table functions differently, depending on whether there are
       one or two records
5    %  All the properties are parallel
6    % Author: Tom Anderson
7
8    properties
9      json_metadata     % all the json data loaded from file
10     speaker_id   % the speaker_id for the given speaker
11     dataset_path % a path to the alveo folder containing the data
12     mat_filename_cellstr % list of .mat files that the data comes from
13     textgrid     % PRAAT TextGrids
14     wav_file    % wav files
```

```
15      mfcc39   % mfcc39 data
16      userdata     % stores current state and any extra data needed
17    end
18
19    % convert adhoc SAMPA labels to IPA
20    methods(Static)
21      [ sampa2ipa ] = ta_SAMPA_to_IPA()
22      [ prompt2ipa ] = ta_hVd_prompt_to_IPA()
23      gender = ta_gender_from_speaker_id(speaker_id)
24    end
25
26    methods     %% signatures of externally defined methods
27      ta_add_labels( self, variable_name, labels_to_be_added, varargin)
28      sMap = ta_batchtrain_lininit(self, SOMwidth, varargin)
29     clone_of_self = ta_clone(self, varargin)
30      [ train_data, test_data ] = ta_crossval( self, K )
31      [ Train_indices, Test_indices ] = ta_crossval__one_left_out___indices(
       self )
32      clone_of_self = ta_data_add(self, to_be_added_ta_data)
33      retVal = ta_flat_stack(self)
34      gender_cell = ta_get_gender_cell( self )
35      my_clone = ta_get_only( self, varargin )
36      out_self = ta_get_PER( self, vocabulary)
37      phoneme_to_prompt_map = ta_get_phoneme_to_prompt_map(self, varargin)
38    labels = ta_get_phonemes_categorical(self, vocabulary)
39      prompt_to_phoneme_map = ta_get_prompt_to_phoneme_map(self, varargin)
40      target_prompts = ta_get_prompts_categorical(obj, target_regex)
41      binary_label_cell = ta_get_vowel_phoneme_from_word__binary_label_cell(
       self, prompt_to_phoneme_map )
42      retVal = ta_label( self, labels, varargin)
43      ta_label_mfcc39_from_TextGrid(self)
44      ta_load_mfcc39(self)
45      output = ta_mask_prompts(obj,arg1,arg2)
46      ta_normalise(self, mode);
47      ta_prune(self, varargin)
48      [mfcc39_retVal, label_retVal] = ta_stack_mfcc39(self, property)
49      [mfcc39_stack, speaker_id_stack, prompt_stack] =
       ta_stack_mfcc39_and_speaker_id(self)
50      output_args = ta_stack_predicted_binary_labels( self,
       predicted_binary_labels, varargin)
51      retVal = ta_subset(self, indices_cell, varargin)
52
53      %% constructor
54      function self = ta_data_class(varargin)
55        % empty constructor
56        if nargin == 0
```

```
57          return
58        end
59
60        % not empty
61        p = inputParser; % check the input
62        p.FunctionName = 'TA_DATA_CLASS constructor';
63        p.addParameter('clone', false);
64        p.addParameter('dirpath', false);
65        p.addParameter('prompt', false, @ischar);
66        p.addParameter('TextGrid', false, @islogical);
67        p.addParameter('mfcc39', false, @islogical); % only perform this step
      if it is specified
68        p.addParameter('label', false, @islogical);
69        p.addParameter('mat_label', false, @ischar);
70        p.addParameter('sampa_to_ipa', true, @islogical); % use SAMPA lookup?
71        p.parse(varargin{:});
72        pp = p.Results;
73
74        if isa(pp.clone, 'ta_data_class')
75          self = pp.clone.ta_clone();
76        end
77
78        if pp.dirpath
79          self.get_all_mat_files_in_a_folder_cellstr(pp.dirpath);
80          self.load_mat_files();
81
82          % store the dirpath that was used
83          self.userdata.dirpath = pp.dirpath;
84        end
85
86        % retain the mat_label that is used when outputting to describe the
      dataset
87        if pp.mat_label
88          self.userdata.mat_label = pp.mat_label;
89        end
90
91        %% are we pruning based on 'prompt'?
92        if pp.prompt % prune using regex
93          % prune data that doesn't match the target
94          self.ta_prune('prompt', pp.prompt);
95        end
96
97        %% are we loading TextGrid files?
98        if pp.TextGrid
99            self.ta_get_TextGrids();
100       end
```

```
101
102         %% are we creating mfcc39? Create them after pruning.
103         if pp.mfcc39
104           self.ta_load_mfcc39();
105         end
106
107         if pp.label
108           %% use the textgrid to create labels for the mfcc39_data
109           self.ta_praat_textgrid_to_mfcc39_label('sampa_to_ipa', pp.
    sampa_to_ipa);
110
111           % add the binary labels that are used for comparing actual labels
    with predicted labels
112           self.ta_mfcc39_category_labels_to_binary_labels();
113         end
114       end
115
116       % load the data stored as wav files with filenames in
    mat_filename_cellstr
117       function load_mat_files(self)
118         mat_files = cellfun(@load, self.mat_filename_cellstr);
119         self.json_metadata = {mat_files.json_table}';
120         self.speaker_id = {mat_files.speaker_id}';
121         self.dataset_path = {mat_files.dataset_path}';
122       end
123
124       function get_all_mat_files_in_a_folder_cellstr(obj, mat_dirpath)
125         % get a list of all .mat files in the folder
126         mat_filenames = dir(fullfile(mat_dirpath, '*.mat'));
127         mat_filenames_name = {mat_filenames.name}';
128         mat_fullfilename = cellfun(@(x) strcat(mat_dirpath, '\', x),
    mat_filenames_name, 'UniformOutput', false);
129
130         % cellstr of filenames
131         obj.mat_filename_cellstr = mat_fullfilename;
132       end
133
134       % prune data structure to a subsample of speakers
135       function ta_prune_speakers(self, speaker_selection)
136
137         % we assume that selection is a subset of row numbers
138         assert isa(speaker_selection, 'double')
139
140         % end if there is an error
141         if max(speaker_selection) > self.ta_length()
142           error('@ta there are more items in the subset than in the data
```

```
structure');
143       end
144
145       % iterate through each of the properties
146       for property = properties(self)'
147         % store the value of the property
148         temp = self.(property{:});
149
150         % skip empty properties
151         if isempty(temp)
152           self.(property{:}) = temp;
153         else
154           % the item's property is equal to the subset of speaker data
specified by selection
155           self.(property{:}) = temp(speaker_selection);
156         end
157       end
158     end
159
160     function retVal = ta_length(self)
161       if isfield(self.userdata, 'state') && strcmp(self.userdata.state, 'flat
')
162         retVal = length(self.mfcc39);
163       else
164         % assumption: the length of all the parallel arrays is equal, so we
return the length of one of them
165         retVal = length(self.json_metadata);
166       end
167     end
168
169     function retVal_bool = ta_has_textgrid(self)
170       retVal_bool = ~isempty(self.textgrid);
171     end
172
173     % get the set of all sorted, unique label tokens for a given speaker
174     function complete_vocabulary = ta_get_complete_vocabulary(self)
175       % isolate the mfcc39 data
176       all_mfcc39_table = cat(1, self.mfcc39{:});
177
178       % isolate the labels
179       temp_labels1 = all_mfcc39_table.labels; % only works for labelled data
180       temp_labels2 = vertcat(temp_labels1{:}); % note vertcat(
all_mfcc39_table.labels{:}) does not give the same result
181
182       % get the unique labels
183       complete_vocabulary = unique({temp_labels2{:}}'); % results of unique
```

```
      are sorted
184     end
185   end
186 end
```

B.1.1 Data object for storing and accessing Austalk corpus data

```
1  function [retVal] = ta_flat_stack(self)
2  % TA_FLAT_STACK internal method to ta_data_class
3  % stacks all the data to windows for data processing,
4  % flatten into single row all round
5  % maintains access to speaker_id and prompt
6  % Author: Tom Anderson
7
8    retVal = ta_data_class();
9    properties_temp = properties(self);
10   for property_index =1:length(properties_temp)
11     property = properties_temp{property_index};
12     %return stack of speaker_id parallel with mfcc39 data
13     if strcmp(property, 'speaker_id')
14       speaker_id_cell = cell(self.ta_length(), 1);
15
16       % get height of each speaker_id_stack
17       for i = 1:self.ta_length()
18         total = sum(cellfun(@length, self.mfcc39{i}.mfcc39));
19         speaker_id_cell{i} = repmat(self.speaker_id(i), total, 1);
20       end
21       prompts_categorical = cellfun(@(x) x.prompt, self.json_metadata, '
     UniformOutput', false);
22       prompt_cell = cell(self.ta_length(), 1);
23
24       for i = 1:self.ta_length()
25         current_prompt_categorical = prompts_categorical{i};
26         assert(length(current_prompt_categorical) == height( self.
     json_metadata{i} ))
27
28         temp = cellfun(@length, self.mfcc39{i}.mfcc39);
29         prompt_temp_cell = cell(length(current_prompt_categorical), 1);
30         for j = 1:length(current_prompt_categorical)
31           prompt_temp_cell{j} = repmat(current_prompt_categorical(j), temp(j)
     , 1);
32         end
33         prompt_cell{i} = vertcat(prompt_temp_cell{:});
34         assert(length(prompt_cell{i}) == sum(temp));
35       end
```

```
36
37        speaker_id= vertcat(speaker_id_cell{:});
38        prompts = vertcat(prompt_cell{:});
39        group = cellfun(@(x, y) sprintf('%s_%s', x, y), speaker_id, cellstr(
      prompts), 'uni', false);
40
41        retVal.speaker_id = speaker_id;
42        retVal.userdata.group = table(categorical(speaker_id), prompts,
      categorical(group), 'VariableNames', {'speaker_id', 'prompts', 'group'});
43        retVal.userdata.state = 'flat';
44      elseif strcmp(property, 'mfcc39') % simply flatten mfcc39 data (which
      also contains labels)
45          temp1 = vertcat(self.mfcc39{:});
46          mfcc39_cell = temp1.mfcc39;
47          retVal.mfcc39 = vertcat(mfcc39_cell{:});
48      else % leave the data alone if already filled
49        if isempty(retVal.(property))
50          retVal.(property) = self.(property);
51        end
52      end
53    end
54  end
```

B.1.2 Data object for pruning data

```
1  function ta_prune(self, varargin)
2  %  self: the json data structure for each participant
3  %  prompt: a regex used for search, all non-matching prompts are deleted
4  %  mfcc39: if false, remove mfcc39 data (for compact storage of data)
5  % Author: Tom Anderson
6  p = inputParser;
7  p.addParameter('prompt', false, @ischar);
8  p.addParameter('mfcc39', true, @islogical);
9  p.parse(varargin{:});
10 pp = p.Results;
11
12 if pp.prompt
13   %% MARK
14    target_prompts_categorical = self.ta_get_prompts_categorical(pp.prompt);
15
16   %% marked_prompts is parallel to mat_files.json_metadata.prompts with T/F
       for whether it is included
17    marked_prompts_cell = cellfun(@(x) ismember(x.prompt,
       target_prompts_categorical), self.json_metadata, 'UniformOutput', false);
18
```

```
19    %% PRUNE
20     for index = 1:self.ta_length()
21       marked_prompts_cell_temp = marked_prompts_cell{index};
22
23       % assign the value to the return value table
24       temp = self.json_metadata{index};
25       self.json_metadata{index} = temp(marked_prompts_cell_temp,:);
26       if self.mfcc39
27         mfcc_temp = self.mfcc39{index};
28         self.mfcc39{index} = mfcc_temp(marked_prompts_cell_temp,:);
29       end
30     end
31   end
32
33   if ~pp.mfcc39
34     for index = 1:self.ta_length()
35       mfcc_temp = table(self.mfcc39{index}.autolabels, self.mfcc39{index}.
         prompts, 'VariableNames', self.mfcc39{index}.Properties.VariableNames
         (2:3));
36       self.mfcc39{index} = mfcc_temp;
37     end
38   end
39   end
```

B.1.3 Method for obtaining a subset of the data object

```
1  function retVal = ta_subset(self, indices_cell, varargin)
2  %% returns a data structure that contains only those items matching
       indices_cell
3  % input:
4  %  indices_cell: the items to retain
5  % optional input:
6  %  type: the type of the subset, where 'file' which is standard method and
       takes file by file or 'flat' and all the data is considered to be
       flattened
7  % output:
8  %  retVal: the ta_data_class to return
9  % Author: Tom Anderson
10
11 p = inputParser(); % parse optional arguments
12 p.addRequired('indices_cell', @(x) isa(x, 'cell') || isa(x, 'double'));
13 p.addParameter('type', 'file', @ischar)
14 p.parse(indices_cell, varargin{:});
15
16 if strcmp(p.Results.type, 'file')
```

```
17    % return value starts as a clone of self
18    retVal = ta_data_class('clone', self);
19
20    % get a list of all the properties0
21    properties_temp = properties(self);
22
23    % need to iterate through each property
24    for property_index = 1:length(properties_temp)
25
26      % set current property
27      property = properties_temp{property_index};
28
29      % do not modify speaker_id or dataset_path, but go through all the other
        properties and reduce them
30      if isempty(self.(property))
31        % do nothing
32      elseif strcmp(property, 'speaker_id')
33        % retain data
34      elseif strcmp(property, 'dataset_path')
35        % retain all data
36      elseif strcmp(property, 'mat_filename_cellstr')
37        % retain all data
38      elseif strcmp(property, 'userdata')
39        % retain all data
40      elseif strcmp(property, 'textgrid')
41        % loop through each speaker
42        for i = 1:retVal.ta_length()
43          if isa(indices_cell, 'cell')
44            retVal.textgrid{i}.praat_textgrid = self.textgrid{i}.praat_textgrid
      (indices_cell{i});
45          elseif isa(indices_cell, 'double')
46            retVal.textgrid{i}.praat_textgrid = self.textgrid{i}.praat_textgrid
      (indices_cell(i));
47          end
48        end
49      elseif strcmp(property, 'mfcc39') || strcmp(property, 'json_metadata') ||
        strcmp(property, 'wav_file')
50        % loop through each speaker
51        for i = 1:retVal.ta_length()
52
53          % get handle on the data
54          temp = retVal.(property){i};
55
56          % reduce the data
57          if isa(indices_cell, 'cell')
58            retVal.(property){i} = temp(indices_cell{i},:);
```

```
59          elseif isa(indices_cell, 'double')
60            retVal.(property){i} = temp(indices_cell(i),:);
61          end
62        end
63      else
64        error('@ta property %s unaccounted for', property);
65      end
66    end
67  elseif strcmp(p.Results.type, 'flat')
68
69    % return value starts as a clone of self
70    retVal = ta_data_class('clone', self);
71
72    % get a list of all the properties0
73    properties_temp = properties(self);
74
75    % need to iterate through each property
76    for property_index = 1:length(properties_temp)
77
78      % set current property
79      property = properties_temp{property_index};
80
81      % do not modify speaker_id or dataset_path, but go through all the other
        properties and reduce them
82      if isempty(self.(property))
83        % do nothing
84      elseif strcmp(property, 'speaker_id')
85        % retain data
86      elseif strcmp(property, 'dataset_path')
87        % retain all data
88      elseif strcmp(property, 'mat_filename_cellstr')
89        % retain all data
90      elseif strcmp(property, 'userdata')
91        % retain all data
92      elseif strcmp(property, 'json_metadata') || strcmp(property, 'wav_file')
        || strcmp(property, 'textgrid')
93        % do nothing, because we assume that every single file is retained, but
        only some data from each file is retained
94      elseif strcmp(property, 'mfcc39')
95        % indices correspond to the flattened data, so need to project it to
        see the proper mfcc data
96
97        % indices_cell is Kx1 data row indexes, but our data is shaped as cells
        of speakers of files of mfcc39
98        % number all the mfcc39 then use those numbers
99        start = 1;
```

```
100
101         % loop through each speaker
102         for i = 1:retVal.ta_length()
103           mfcc39_numbering = cell(retVal.ta_length(), 1);
104           for j = 1:height(retVal.mfcc39{i})
105             current_height = size(retVal.mfcc39{i}.mfcc39{j}, 1);
106
107             finish = start + current_height- 1;
108             % get handle on the data
109             mfcc39_numbering(j) = {[start:finish]'};
110             start = finish + 1;
111
112             mfcc39_temp = retVal.mfcc39{i}.mfcc39{j};
113             retVal.mfcc39{i}.mfcc39{j} = mfcc39_temp(ismember(mfcc39_numbering{
      j}, indices_cell), :);
114
115             if ismember('labels', retVal.mfcc39{i}.Properties.VariableNames) &&
       ~isempty(retVal.mfcc39{i}.labels{j})
116               mfcc39_labels_temp = retVal.mfcc39{i}.labels{j};
117               retVal.mfcc39{i}.labels{j} = mfcc39_labels_temp(ismember(
      mfcc39_numbering{j}, indices_cell), :);
118             end
119           end
120         end
121       else
122         error('@ta property %s unaccounted for', property);
123       end
124     end
125   else
126     error('@ta type not defined');
127   end
128
129   end
```

B.2 Map structure for SAMPA to IPA mapping

```
function [ sampa2ipa ] = ta_get_sampa2ipa_map()
%% TA_SAMPA_TO_IPA Convert SAMPA to IPA
% returns a map structure that enables annotation symbol conversions
% Output:
%  sampa2ipa: map container
%   use by querying like: sampa2ipa('dZ') that will return the string '\
    textdyoglig'
% Author: Tom Anderson

% SAMPA tags
A = [{'p', 'p'}; {'b', 'b'}; {'t', 't'}; {'d', 'd'}; ...
 {'tS', 'ʧ'}; {'dZ', 'ʤ'}; {'k', 'k'}; {'g', 'g'}; ...
 {'f', 'f'}; {'v', 'v'}; {'T', 'θ'}; {'D', 'ð'}; ...
 {'s', 's'}; {'z', 'z'}; {'S', 'ʃ'}; {'Z', 'ʒ'}; ...
 {'h', 'h'}; {'m', 'm'}; {'n', 'n'}; {'N', 'ŋ'}; ...
 {'l', 'l'}; {'r', 'ɹ'}; {'w', 'w'}; {'j', 'j'}; ...
 {'W', 'ʍ'}; {'x', 'x'}; {'a:', 'aː'}; {'i:', 'iː'}; ...
 {'I', 'ɪ'}; {'e', 'e'}; {'3:', 'ɜː'}; {'{', 'æ'}; ...
 {'a:', 'aː'}; {'a', 'a'}; {'O', 'ɔ'}; {'o:', 'oː'}; ...
 {'U', 'ʊ'}; {'}:', 'ʉː'}; {'@', 'ə'}; {'I', 'ær'}; ...
 {'Ae', 'ɑe'}; {'oI', 'oɪ'}; {'@}', 'əʉ'}; {'O', 'æɔ'}; ...
 {'I@', 'ɪə'}; {'e:', 'eː'}; {'U@', 'ʊə'}; {'j}:', 'jʉː'} ];

% silence tag
A = [A; {'⟨ p:⟩', '(...)'}];

% also some X-SAMPA
A = [A; {'6', 'ɐ'}];

% assuming the 6: is just r-coloured 6
A = [A; {'6:', 'ɐː'}];

% * means multiple labels
A = [A; {'*', '*'}];

% null value labels
A = [A; {'', ''}];

sampa2ipa = containers.Map(A(:, 1), A(:, 2));

end
```

B.3 Spectrogram wrapper

```
function [ output_spectrogram_data ] = ta_spectrogram( wav )
%TA_SPECTROGRAM prepares a spectrogram
% input:
%  wav: single channel 16 kHz wav data
% output:
%  output_spectrogram_data: 129 channel spectrogram
% Author: Tom Anderson
   window = 256; %0.020 * Fs; % 20 ms
   overlap = .5;
   noverlap = window * overlap;
   nfft = 256;
   [~, ~, ~, P] = spectrogram(wav, window, noverlap, nfft, 16000);
   output_spectrogram_data = P';
end
```

B.4 Data object for storing Praat annotations

```
classdef ta_praat_textgrid
%ta_read_textgrid Read a PRAAT TextGrid
% input:
%  filename: the name of the PRAAT TextGrid
% output:
%  textgrid_table: a table containing the data from the PRAAT TextGrid
% Author: Tom Anderson

properties
  file_type
  object_class

  % start of the file in seconds
  xmin

  % end of the file in seconds
  xmax

  % number of tiers
  % the PRAAT file data structure used for the Austalk corpus consists of 3
    cells of data named ORT, OBJ and MAU
  tiers

  % number of items
  size

  % the annotation data itself
  items
end

methods
  % method for scanning file (externally defined)
  return_value = ta_textscan_1(obj, fid, search_string)

  %% return a table of the labels and timings
  function interval = get_interval(self)
    if self.size == 0 % manual annotation does not exist
      interval = [];
    else
      interval = self.items{3}.interval;
    end
  end

  %% return a list of all labels used in the annotation
```

```
44    function all_labels = get_all_labels(self)
45      if self.size == 0
46        all_labels = {};
47      else
48        all_labels = self.items{3}.interval.text;
49      end
50    end
51
52    %% reading in the data
53    % create the lowest level structure
54    function obj = ta_praat_textgrid(filename)
55
56      % if no filename, just return an empty obj
57      if nargin == 0
58        obj.size = 0;
59        return
60      end
61
62      %% initialise the object with data from a TextGrid file
63      fid = fopen(filename, 'r');
64
65      % get data, line by line
66      obj.file_type = obj.ta_textscan_1(fid, 'File type = %q');
67      obj.object_class = obj.ta_textscan_1(fid, 'Object class = %q');
68      obj.xmin= obj.ta_textscan_1(fid, 'xmin = %f');
69      obj.xmax= obj.ta_textscan_1(fid, 'xmax = %f');
70      obj.tiers = obj.ta_textscan_1(fid, 'tiers? %q');
71      obj.size = obj.ta_textscan_1(fid, 'size = %d');
72
73      obj.items = obj.ta_items(fid);
74
75      % close the file
76      fclose(fid);
77    end
78
79    % read in the items
80    function items = ta_items(obj, fid)
81      items = cell(obj.size, 1);
82
83      % read in unneeded PRAAT file structural information
84      obj.ta_textscan_1(fid, 'item []:');
85
86      % read in the (three) intervals
87      for current_item_index=1:obj.size
88        items{current_item_index} = obj.ta_interval(fid);
89      end
```

```
90    end
91
92    % read in the intervals
93    function current_item = ta_interval(obj, fid)
94        % read in unneeded structural information
95        % note: the value of %f here is the same as the index i
96        obj.ta_textscan_1(fid, 'item [%f]:');
97
98        current_item.class = obj.ta_textscan_1(fid, 'class = %q');
99        current_item.name = obj.ta_textscan_1(fid, 'name = %q');
100       current_item.xmin= obj.ta_textscan_1(fid, 'xmin = %f');
101       current_item.xmax= obj.ta_textscan_1(fid, 'xmax = %f');
102       current_item.intervals_size = obj.ta_textscan_1(fid, 'intervals: size =
        %f');
103       current_item.interval = cell(current_item.intervals_size, 1);
104       for interval_index = 1:current_item.intervals_size
105         current_interval = struct();
106
107         % read in unneeded structural information
108         % note: the value of %f here is the same as interval_index
109         obj.ta_textscan_1(fid, 'intervals [%f]:');
110
111         current_interval.xmin= obj.ta_textscan_1(fid, 'xmin = %f');
112         current_interval.xmax= obj.ta_textscan_1(fid, 'xmax = %f');
113         current_interval.text = obj.ta_textscan_1(fid, 'text = %q');
114
115         current_item.interval{interval_index} = current_interval;
116       end
117       current_item.interval = struct2table([current_item.interval{:}]);
118   end
119 end
120
121 end
```

Listing B.1: Helper function for reading PRAAT TextGrid files

```
1  function return_value = ta_textscan_1(obj, fid, search_string)
2  % Helper class for the ta_praat_textgrid class used for reading in data
3  % input:
4  %  fid: file id of open file for reading with textscan
5  %  search_string: a string containing the text for scanning
6  %     e.g. 'File type = %q', where q is a string inside quotes like "
       ooTextFile"
7  % Author: Tom Anderson
8
9  % read one line of file
10 return_value = textscan(fid, search_string, 1);
```

```
11
12 % unwrap cells
13 while ~isempty(return_value) && iscell(return_value)
14   return_value = return_value{:};
15 end
16
17 end
```

B.5 SOM Toolbox

Some of the scripts that I created that use the SOM Toolbox are presented in this section.

Listing B.2: Split files into training and testing sets

```
%% divide audio files into training and testing sets
N = height(data_table);
K= 1/8; % how much data to hold out for each test

%% Generate indices for current train and test
group = data_table.word;

% hold out one of each word
[train, test] = crossvalind('holdout', group, K);
1
% create the data tables (train and test)
train_table = data_table(train, :);
test_table = data_table(test, :);
```

Listing B.3: Training maps

```
%% train sMap using rough and fine training
tic,
SOM_width_4 = ceil(parameters.SOM_width/4);
sMap_roughed_no = som_batchtrain(sMap_inited_no, sData_no, 'radius', [
    SOM_width_4 2], 'trainlen', 30, 'tracking', 0);
sMap_fined_no = som_batchtrain(sMap_roughed_no, sData_no, 'radius', [2 1], '
    trainlen', 20, 'tracking', 0);

sMap_roughed_word = som_batchtrain(sMap_inited_word, sData_word, 'radius', [
    SOM_width_4 2], 'trainlen', 30, 'tracking', 0);
sMap_fined_word = som_batchtrain(sMap_roughed_word, sData_word, 'radius', [2
    1], 'trainlen', 20, 'tracking', 0);

sMap_roughed_set = som_batchtrain(sMap_inited_set, sData_set, 'radius', [
    SOM_width_4 2], 'trainlen', 30, 'tracking', 0);
sMap_fined_set = som_batchtrain(sMap_roughed_set, sData_set, 'radius', [2 1],
     'trainlen', 20, 'tracking', 0);

toc
% SOM_width: 18 -> Elapsed time is 35.857142 seconds.
% SOM_width: 25 -> Elapsed time is Elapsed time is 30.198208 seconds.
```

Listing B.4: Procedures to display labelled SOMs using SOM Toolbox

```
    figure
    som_show(sMap,'empty','');%, 'colormap', colormap([ 1 1 1]));
    som_show_add('label', sMap);
end
```

Listing B.5: Labelling maps

```
%% Create labels each word for no/set/word-level normalisation
for i = 1:height(train_table)
  % don't copy more labels than windows
    len = size(train_table.MFCC39{i}, 1);
    train_table.sData_word(i).labels = labels(1:len);
end

temp_labels = arrayfun(@(x) x.labels, train_table.sData_word, 'uni', false);
sData_word_labels = vertcat(temp_labels{:});

%% collect sData_no for no normalisation
temp_data = vertcat(train_table.MFCC39{:});
labelled_training_sData_no = som_data_struct(temp_data, 'labels',
    sData_word_labels);

%% label each word for the word-level normalisation
temp_data = arrayfun(@(x) x.data, train_table.sData_word, 'uni', false);
temp_data = vertcat(temp_data{:});

labelled_training_sData_word = som_data_struct(temp_data, 'labels',
    sData_word_labels);

%% label each word for the set-level normalisation
labelled_training_sData_set = som_data_struct(sData_set.data, 'labels',
    sData_word_labels, 'comp_norm', sData_set.comp_norm);
```

References

Agudo, M. & De Dios, J. (2013). An investigation into how EFL learners emotionally respond to teachers' oral corrective feedback. *Colombian Applied Linguistics Journal*, 15(2), 265–278.

Alam, M. J., Ouellet, P., Kenny, P., & O'Shaughnessy, D. (2011). Comparative evaluation of feature normalization techniques for speaker verification. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7015 LNAI, 246–253.

Albeverio, S., Krüger, N., & Tirozzi, B. (1997). An extended Kohonen phonetic map. *Mathematical and Computer Modelling*, 25(2), 69–73.

Alborzi, H., Druin, A., Montemayor, J., Platner, M., Porteous, J., Sherman, L., Boltman, A., Taxén, G., Best, J., Hammer, J., Kruskal, A., Lal, A., Schwenn, T. P., Sumida, L., Wagner, R., & Hendler, J. (2000). Designing StoryRooms: Interactive Storytelling Spaces for Children. *Proceedings of the 3rd Conference on Designing Interactive Systems*, 02, 95–104.

Anderson, T. A. F. (2009). *Design of an English Vocabulary Acquisition System with Minigames for Elementary School Students in Taiwan*. Master's, National Central University.

Anderson, T. A. F., Chen, Z.-H., Chan, T.-W., & Juan, C.-H. (2009). Implementing a vocabulary acquisition system with minigames. In S. Kong, H. Ogata, H. Arnseth, C. Chan, T. Hirashima, F. Klett, J. Lee, C. Liu, C. Looi, M. Milrad, A. Mitrovic, K. Nakabayashi, S. Wong, & S. Yang (Eds.), *Proceedings of the 17th International Conference on Computers in Education (ICCE)* (pp. 751–753).: Asia-Pacific Society for Computers in Education.

Anderson, T. A. F., Chen, Z.-H., Wen, Y.-F., Milne, M., Atyabi, A., Treharne, K., Matsumoto, T., Jia, X.-B., Luerssen, M., Lewis, T., Leibbrandt, R., & Powers, D. M. W. (2011). Thinking head MulSeMedia: A storytelling environment for embodied language learning. In G. Ghinea, F. Andres, & S. R. Gulliver (Eds.), *Multiple Sensorial Media Advances and Applications: New Developments in MulSeMedia* (pp. 182–203). IGI Global.

Anderson, T. A. F., Hwang, W.-Y., & Hsieh, C.-H. (2008a). A study of a mobile collaborative learning system for Chinese language learning. In *Proceedings of the 16th International Conference on Computers in Education (ICCE)* (pp. 217–222).

Anderson, T. A. F. & Powers, D. M. W. (2016). Characterisation of speech diversity using self-organising maps. In *16th Australasian International Conference on Speech Science and Technology (SST2016)* Sydney, Australia: Australasian Speech Science and Technology Association.

Anderson, T. A. F., Reynolds, B., Yeh, X.-P. Y. X.-P., & Huang, G.-Z. H. G.-Z. (2008b). Video Games in the English as a Foreign Language Classroom. In *2008 Second IEEE International Conference on Digital Game and Intelligent Toy Enhanced Learning* (pp. 188–192). New York, NY: IEEE.

Anderson, T. A. F. & Wen, Y.-F. (2008). An Approach to Learning Research with a Wireless Sensor Network in an Outdoor Setting. In *Proceedings - ICCE 2008: 16th International*

Conference on Computers in Education (pp. 1–4).

Anderson, T. R. (1991). An Auditory Model and a Neural Network. *Aerospace*, (pp. 149–152).

Arous, N. & Ellouze, N. (2003). Cooperative supervised and unsupervised learning algorithm for phoneme recognition in continuous speech and speaker-independent context. *Neurocomputing*, 51, 225–235.

Asher, J. J. (2000). *Learning another language through actions: The complete teacher's guidebook*. Los Gatos, CA: Sky Oaks Productions, eighth edition.

Barbedo, J. G. A., Ribeiro, M. V., Von Zuben, F. J., Lopes, A., & Romano, J. M. T. (2002). Application of Kohonen self-organizing maps to improve the performance of objective methods for speech quality assessment. In *European Signal Processing Conference* (pp. 2–5).

Behi, T., Arous, N., & Ellouze, N. (2012). Spike Timing Dependent Competitive Learning in Recurrent Self Organizing Pulsed Neural Networks Case Study: Phoneme and Word Recognition. *IJCSI International Journal of Computer Science Issues*, 9(4), 10.

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8), 1798–1828.

Birsh, J. R. (2011). *Multisensory teaching of basic language skills*. Baltimore, MD: Brookes Publishing Company.

Boersma, P. & Weenink, D. (2017). Praat: doing phonetics by computer.

Bořil, T. & Skarnitzl, R. (2016). Tools rPraat and mPraat. In P. Sojka, A. Horák, I. Kopeček, & K. Pala (Eds.), *Text, Speech, and Dialogue. TSD 2016. Lecture Notes in Computer Science*, volume 9924 (pp. 367–374). Springer.

Brune, M. (2004). *Total Physical Response Storytelling: and analysis and application*. Bachelor of arts, University of Oregon.

Burnham, D., Ambikairajah, E., Arciuli, J., Bennamoun, M., Best, C. T., Bird, S., Butcher, A. R., Cassidy, S., Chetty, G., Cox, F. M., Cutler, A., Dale, R., Epps, J. R., M, J., Goecke, R., Grayden, D. B., Hajek, J. T., Ingram, J. C., Ishihara, S., Kemp, N., Kuratate, T., Lewis, T. W., Loakes, D. E., Onslow, M., Powers, D. M. W., Rose, P., Togneri, R., Tran, D., & Wagner, M. (2009). A Blueprint for a Comprehensive Australian English Auditory-Visual Speech Corpus. *Science And Technology*, (pp. 96–107).

Burnham, D., Estival, D., Fazio, S., Viethen, J., Cox, F., Dale, R., Cassidy, S., Epps, J., Togneri, R., Wagner, M., Kinoshita, Y., Goecke, R., Arciuli, J., Onslow, M., Lewis, T., Butcher, A., & Hajek, J. (2011). Building an audio-visual Corpus of Australian English: Large Corpus collection with an economical portable and replicable black box. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, (pp. 841–844).

Buschmeier, H. & Wlodarczak, M. (2013). TextGridTools: A TextGrid Processing and Analysis Toolkit for Python. In *Tagungsband der 24. Konferenz zur Elektronischen Sprachsignalverarbeitung (ESSV 2013)* (pp. 152–157). Bielefeld, Germany.

Cassidy, S., Estival, D., & Cox, F. (2014). *AusTalk Annotation report*. Technical report, Big Australian Speech Corpus (ASC).

Cassidy, S., Estival, D., & Cox, F. (2017). Case study: The AusTalk corpus. In N. Ide & J. Pustejovsky (Eds.), *Handbook of Linguistic Annotation* (pp. 1287–1301). Dordrecht: Springer Science+Business Media.

Chan, A. M., Dykstra, A. R., Jayaram, V., Leonard, M. K., Travis, K. E., Gygi, B., Baker, J. M., Eskandar, E., Hochberg, L. R., Halgren, E., & Cash, S. S. (2014). Speech-Specific Tuning of Neurons in Human Superior Temporal Gyrus. *Cerebral Cortex*, 24(10), 2679–

2693.

Chan, T.-W. (2010). How East Asian classrooms may change over the next 20 years. *Journal of Computer Assisted Learning*, 26(1), 28–52.

Chan, T.-W., Roschelle, J., Hsi, S., Kinshuk, K., Sharples, M., Brown, T., Patton, C., Cherniavsky, J., Pea, R. D., & Norris, C. (2006). One-to-one technology-enhanced learning: an opportunity for global research collaboration One-to- one technology-enhanced learning: an opportunity for global research collaboration. *World Scientific Publishing*, 1(1), 3–29.

Chang, S.-B., Lin, C.-J., Ching, E., Cheng, H. N. H., Chang, B., Chen, F.-C., Wu, D., & Chan, T.-W. (2007). EduBingo: Developing a content sample for the one to one classroom by the content first design approach. *Educational Technology & Society*, 12(3), 343–353.

Chappell, G. J. & Taylor, J. G. (1993). The Temporal Kohonen Map. *Neural Networks*, 6(3), 441–445.

Chen, Z.-H., Anderson, T. A. F., Cheng, H., & Chan, T.-W. (2009). Character-Driven Learning: Facilitating Students' Learning by Educational Virtual Characters. In S. Kong, H. Ogata, H. Arnseth, C. Chan, T. Hirashima, F. Klett, J. Lee, C. Liu, C. Looi, M. Milrad, A. Mitrovic, K. Nakabayashi, S. Wong, & S. Yang (Eds.), *Proceedings of the 17th International Conference on Computers in Education* (pp. 748–750). Hong Kong: Asia-Pacific Society for Computers in Education.

Chihi, H. & Arous, N. (2012). Recurrent Neural Network Learning by Adaptive Genetic Operators. *6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT)*, (pp. 832–834).

Chiraz, J., Arous, N., & Ellouze, N. (2015). Cooperative Growing Hierarchical Recurrent Self Organizing Model for Phoneme Recognition. *Int J Comput Neural Eng.*, 02(1), 11–15.

Chiu, Y.-h., Anderson, T. A. F., & Powers, D. M. W. (2012a). Dubbing of virtual embodied conversation agents for improving pronunciation. In *Fifteenth International CALL Conference: The Medium Matters* (pp. 188–193). Taichung, Taiwan.

Chiu, Y.-h., Kao, C. W., & Reynolds, B. L. (2012b). The relative effectiveness of digital game-based learning types in English as a foreign language setting: A meta-analysis. *British Journal of Educational Technology*, 43(4), 104–107.

Chou, C. Y., Chan, T. W., & Lin, C. J. (2003). Redefining the learning companion: The past, present, and future of educational agents. *Computers and Education*, 40(3), 255–269.

Chu-Carroll, J. (2000). MIMIC. In *Proceedings of the sixth conference on Applied natural language processing* - (pp. 97–104). Seattle, WA: Association for Computational Linguistics.

Cooke, M., Green, P. D., & Crawford, M. D. (1994). Handling missing data in speech recognition. *Proc. ICSLP*, (pp. 1555–1558).

Crawford, C. (2003). *Chris Crawford on game design.* Berkeley, CA: New Riders Publishing.

Crawford, C. (2004). *Chris Crawford on interactive storytelling.* Berkeley, CA: New Riders Publishing.

Crystal, B. & Crystal, D. (2014). *You say potato: A book about accents.* London: Macmillan.

Dahl, O.-J., Myhrhaug, B., & Nygaard, K. (1968). No Title. In *Proceedings of the Second Conference on Applications of Simulations* (pp. 29–31). New York, NY: Winter Simulation Conference.

Dalsgaard, P. (1992). Phoneme label alignment using acoustic-phonetic features and Gaussian probability density functions. *Computer Speech and Language*, 6(4), 303–329.

Danielson, S. (1990). Recognition of Danish phonemes using an artificial neural network. In *1990 IJCNN International Joint Conference on Neural Networks* (pp. 677–682 vol.3). San Diego, CA: IEEE.

De Luna-Ortega, C. A., Mora-González, M., Álvarez-Medina, M. A., Romo, J. C. M., Rosas, F.

J. L., & Viillar, V. E. G.-d. (2009). Analysis of Kohonen's Neural Network with application to speech recognition. In *1st Workshop on Computer Vision and Pattern Recognition2* (pp. 1–12). Mexico City, Mexico: Sociedad Mexicana de Inteligencia Artificial.

Deng, D. & Kasabov, N. (2003). On-line pattern analysis by evolving self-organizing maps. *Neurocomputing*, 51, 87–103.

Derwing, T. M., Munro, M. J., & Wiebe, G. (1998). Evidence in Favor of a Broad Framework for Pronunciation Instruction. *Language Learning*, 48(3), 393–410.

DeSieno, D. (1988). Adding a conscience to competitive learning. In *IEEE International Conference on Neural Networks (ICNN)* (pp. 117–124). New York, NY: IEEE.

Domingos, P. (2015). *The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World*. UK: Penguin Books Limited.

Druin, A., Montemayor, J., Hendler, J., Mcalister, B., Boltman, A., Fiterman, E., Plaisant, A., Kruskal, A., Olsen, H., Revett, I., Thomas Plaisant Schwenn, L. S., & Wagner, R. (1999). Designing PETS: A personal electronic teller of stories. *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, (pp. 326–329).

Ellis, N. C. (2012). What can we count, and what counts in language acquisition, cognition, and use? In S. Divjak (Ed.), *Frequency effects in language learning and processing* (pp. 7–33). Germany: Walter de Gruyten GmbH & Co.

Ellis, R., Loewen, S., & Erlam, R. (2006). Implicit and explicit corrective feedback and the acquisition of L2 grammar. *Studies in Second Language Acquisition*, 28(2), 339–368.

Eng, G. K. (2006). *Self-Organizing Map and Multilayer Perceptron for Malay Speech Recognition*. Master of science (computer science), Universiti Teknologi Malaysia.

Fix, E. & Hodges, J. L. J. (1951). *Nonparametric Discrimination: Consistency Properties*. Technical Report Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas.

Flach, P. (2012). *Data, Machine Learning: The Art and Science of Algorithms that Make Sense of*. Cambridge University Press.

Fornberg, B. (1988). Generation of finite difference formulas on arbitrarily spaced grids. *Mathematics of Computation*, 51(184), 699–699.

Franzel, D. & Newman, W. (2009). *Virtual world: Hybrid reality for computer learning and teaching*. PhD thesis, Flinders University of South Autralia.

Fulop, S. A. (2011). *Speech Spectrum Analysis*. Signals and Communication Technology. Berlin, Heidelberg: Springer Berlin Heidelberg.

Gardón, A. P., Vázquez, C. R., & Illarramendi, A. A. (1999). Spanish Phoneme Classification by Means of a Hierarchy of Kohonen Self-Organizing Maps. In V. Matousek, P. Mautner, J. Ocelikova, & P. Sojka (Eds.), *Text, Speech and Dialogue - TSD '99 Proceedings of the Second International Workshop on Text, Speech and Dialogue* (pp. 181–186). Berlin Heidelberg: Springer-Verlag.

Gauthier, B., Shi, R., & Xu, Y. (2007). Learning phonetic categories by tracking movements. *Cognition*, 103(1), 80–106.

Gebert, H. & Bothe, H.-H. (2010). LIPPS – A Virtual Teacher for Speechreading Based on a Dialog-Controlled Talking-Head. In K. Miesenberger, J. Klaus, W. Zagler, & A. Karshmer (Eds.), *Computers Helping People with Special Needs. ICCHP 2010. Lecture Notes in Computer Science*, volume 6179 (pp. 621–629). Berlin, Heidelberg: Springer.

George, D. & Hawkins, J. (2009). Towards a Mathematical Theory of Cortical Micro-circuits. *PLoS Comput Biol*, 5(10).

Ghinea, G., Andres, F., & Gulliver, S. R., Eds. (2012). *Multiple Sensorial Media Advances and Applications*. Hershey, PA: IGI Global.

Godino-Llorente, J. & Gomez-Vilda, P. (2004). Automatic Detection of Voice Impairments by Means of Short-Term Cepstral Parameters and Neural Network Based Detectors. *IEEE Transactions on Biomedical Engineering*, 51(2), 380–384.

Golonka, E., Bowles, A., Frank, V., Richardson, D., & Freynik, S. (2012). Technologies for foreign language learning: A review of technology types and their effectiveness. *Computer Assisted Language Learning*, 27(1), 70–105.

Grashey, S. (2003). A New Approach to Voice Activity Detection Based on Self-Organizing Maps. In *Eurospeech* (pp. 1733–1736).

Guenther, F. H. & Gjaja, M. N. (1996). The perceptual magnet effect as an emergent property of neural map formation. *The Journal of the Acoustical Society of America*, 100(2 Pt 1), 1111–1121.

Guimarães, G., Lobo, V. S., & Moura-Pires, F. (2003). A Taxonomy of Self-organizing Maps for Temporal Sequence Processing. *Intelligent Data Analysis*, 7(4), 1–52.

Gustainis, D. (2009). *Hybrid character recognition using a visual input.* PhD thesis, Flinders University of South Australia.

Halkidi, M., Batistakis, Y., & Vazirgiannis, M. (2001). On Clustering Validation Techniques. *Journal of Intelligent Information Systems*, 17(2), 107–145.

Hammer, B., Micheli, A., Neubauer, N., Sperduti, A., & Strickert, M. (2005). Self-Organizing Maps for Time Series. *WSOM 5th Workshop On Self-Organizing Maps*, (pp. 1–8).

Hawkins, J. & Ahmad, S. (2016). Why Neurons Have Thousands of Synapses, a Theory of Sequence Memory in Neocortex. *Frontiers in Neural Circuits*, 10(March), 23.

Hecker, J. C. (2006). *Unsupervised Neural Pattern Classifiers for Oral Vowel Pronunciation Training of Foreign-Accented Speech.* Master's, Clemson.

Honkela, T. (1996). *On the Use of Self-Organizing MAP (SOM) in Linguistic Visualization.* Technical report, Helsinki University of Technology, Helsinki, Finland.

Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2003). *A Practical Guide to Support Vector Classification.* Technical report, National Taiwan University, Taipei, Taiwan.

Huang, G.-B., Zhou, H., Ding, X., & Zhang, R. (2012). Extreme learning machine for regression and multiclass classification. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, 42(2), 513–29.

Huckvale, M. (2004). ACCDIST: A metric for comparing speakers' accents. *International Conference on Spoken Language Processing (INTERSPEECH)*, (pp. 1–4).

Humphries, C., Liebenthal, E., & Binder, J. (2010). Tonotopic organization of human auditory cortex. *Neuroimage*, 50(3), 1202–1211.

Hwang, J., Jung, J., & Kim, G. J. (2006). Hand-held Virtual Reality: A Feasibility Study. In *Proceedings of the ACM symposium on virtual reality software and technology* (pp. 356–363). New York, NY: ACM.

Inchauste, F. (2010). Better user experience with storytelling. *Smashing Magazine.*

Jurafsky, D. & Martin, J. H. (2009). *Speech and Language Processing: An introduction to natural language processing, computational linguistics, and speech recognition.* Upper Saddle River, NJ: Prentice Hall, 2nd edition.

Kalashnikova, M., Goswami, U., & Burnham, D. (2016). Mothers speak differently to infants at-risk for dyslexia. *Developmental Science*, (pp. 1–15).

Kangas, J. (1991). Phoneme recognition using time-dependent versions of self-organizing maps. In *[Proceedings] ICASSP 91: 1991 International Conference on Acoustics, Speech, and Signal Processing* (pp. 101–104). New York, NY: IEEE.

Kangas, J. (1994). *On the Analysis of Pattern Sequences by Self-Organizing Maps.* PhD thesis,

Helsinki University of Technology.

Kangas, J., Kohonen, T. K., & Laaksonen, J. (1990). Variants of self-organizing maps. *IEEE Transactions on Neural Networks*, 1(1), 93–99.

Kasabov, N., Kozma, R., Kilgour, R., Laws, M., Watts, M., Gray, A., & Taylor, J. (1998). Speech data analysis and recognition using fuzzy neural networks and self-organising maps. In N. Kasabov & R. Kozma (Eds.), *Neuro-Fuzzy Tools and Techniques for Information Processing*. Heidelberg: Physica Verlag.

Kaski, S. & Lagus, K. (1996). Comparing self-organizing maps. *Proceedings of ICANN96, International Conference in Artificial Neural Networks, Lecture Notes in Computer Science*, 112, 809–814.

Kaskiy, S., Kangas, J., & Kohonen, T. K. (1998). Bibliography of self-organizing map (SOM) papers: 1981-1997. *Neural Computing Surveys*, 1(3), 1–176.

Kellerman, S. (1990). Lip Service: The Contribution of the Visual Modality to Speech Perception and its Relevance to the Teaching and Testing of Foreign Language Listening Comprehension1. *Applied Linguistics*, 11(3), 272–280.

Knagenhjelm, P. & Brauer, P. (1990). Classification of vowels in continuous speech using MLP and hybrid net. *Speech Communication*, 9, 31–34.

Kohonen, T. K. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43(1), 59–69.

Kohonen, T. K. (1988). The 'neural' phonetic typewriter. *Computer*, 21(3), 11–22.

Kohonen, T. K. (1990). The self-organizing map. *Proceedings of the IEEE*, 78(9), 1464–1480.

Kohonen, T. K. (1995). *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Heidelberg: Springer-Verlag Berlin, second edition.

Kohonen, T. K. (2001). *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Heidelberg: Springer-Verlag Berlin, third edition.

Kohonen, T. K. (2013). Essentials of the self-organizing map. *Neural Networks*, 37, 52–65.

Kohonen, T. K. (2014). *MATLAB Implementations and Applications of the Self-Organizing Map*. Helsinki, Finland: Unigrafia Oy.

Kohonen, T. K. & Hari, R. (1999). Where the abstract feature maps of the brain might come from. *Trends in Neurosciences*, 22(3), 135–139.

Kohonen, T. K., Hynninen, J., Kangas, J., & Laaksonen, J. (1995). *SOM PAK: The Self-Organizing Map Program Package Version 3.1.* Technical report, Helsinki University of Technology, Rakentajanaukio.

Kohonen, T. K. & Somervuo, P. (1998). Self-organizing maps of symbol strings. *Neurocomputing*, 21(1-3), 19–30.

Kohonen, T. K., Torkkola, K., Shozakai, M., Kangas, J., & Venta, O. (1988). Phonetic typewriter for Finnish and Japanese. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 607–610). New York, NY: IEEE.

Koreman, J. C., Andreeva, B., & Barry, W. J. (1998a). Do Phonetic Features Help To Improve Consonant Identification in ASR. In *Proceedings of the 5th International Conference on Spoken Language (ICSLP)*.

Koreman, J. C., Andreeva, B., & Strik, H. (1999). Acoustic parameters versus phonetic features in ASR. *14th Int. Congress of Phonetic Sciences (ICPhS-99)*, (pp. 719–722).

Koreman, J. C., Barry, W. J., Andreeva, B., Box, P. O., & Saarbrücken, D. (1998b). Exploiting Transitions and Focussing on Linguistic Properties for Asr. In *Proceedings of the 5th International Conference on Spoken Language (ICSLP)* Sydney.

Krashen, S. (1987). *Principles and practice in second language acquisition*. New York, NY: Prentice Hall International.

Krashen, S. (1997). A conjecture on accent in a second language. *Applied Linguistic Studies in Central Europe*, 1.

Kurimo, M. (1997). *Using Self-Organizing Maps and Learning Vector Quantization for Mixture Density Hidden Markov Models*. PhD thesis, Helsinki University of Technology.

Ladefoged, P. (2007). Handbook of the International. In *Handbook of the International Phonetic Association*. Cambridge: Cambridge University Press, eighth pri edition.

Ladefoged, P. & Broadbent, D. (1957). Information Conveyed by Vowels. *The Journal of the Acoustical Society of America*, 29(1), 98–104.

Ladefoged, P. & Johnson, K. (2010). *A course in phonetics*. Wadsworth, Cengage Learning, Inc., sixth edition.

Laleye, F. A. A., Ezin, E. C., & Motamed, C. (2014). Weighted Combination of Naive Bayes and LVQ Classifier for Fongbe Phoneme Classification. In *Signal-Image Technology and Internet-Based Systems (SITIS)*: IEEE.

Langner, G., Sams, M., Heil, P., & Schulze, H. (1997). Frequency and periodicity are represented in orthogonal maps in the human auditory cortex: evidence from magnetoecephalography. *J. Comp. Physiol. A.*, 181(6), 665–676.

Lee, C.-B. & Chang-Young, L. (2011). Classification of consonants by SOM and LVQ. *The Journal of the Korea Institute of Electronic Communication Sciences*, (pp. 34–42).

Lee, J., Marsella, S. C., & Rey, M. D. (2006). Nonverbal Behavior Generator for Embodied Conversational Agents. In *Proceedings of the Sixth International Conference on Intelligent Virtual Agents* (pp. 243–255). Marina del Ray, CA: IVA.

Lee, S., Potamianos, A., & Narayanan, S. (1997). Analysis of children's speech: Duration, pitch and formants. In *Fifth European Conference on Speech Communication and Technology (EUROSPEECH)*, volume 1 (pp. 473–476).

Leinonen, L., Kangas, J., Kangas, K., & Juvas, A. (1991). Pattern recognition of hoarse and healthy voices by the self-organizing map. In T. Kohonen, K. Makisara, O. Simula, & J. Kangas (Eds.), *Artificial Neural Networks* (pp. 1385–1388). North-Holland: Elsevier Science Publishers B. V.

Levis, J. (2007). Computer Technology in Teaching and Researching Pronunciation. *Annual Review of Applied Linguistics*, 27(2007), 184–202.

Lorenz, M. (1993). *Object-oriented software development*. Englewood Cliffs, NJ: PTR Prentice Hall.

Luerssen, M., Lewis, T., & Powers, D. (2011). Head X: Customizable audiovisual synthesis for a multi-purpose virtual head. In J. Li (Ed.), *Lecture Notes in Computer Science*, volume 6464 of *Lecture Notes in Computer Science* (pp. 486–495). Berlin, Heidelberg: Springer Berlin Heidelberg.

Luerssen, M. & Lewis, T. W. (2009). Head X: Tailorable audiovisual synthesis for ECAs. In *Proceedings of HCSNet Summerfest 2009*.

Lundberg, J. (2005). *Classifying Dialects Using Cluster Analysis*. Master's thesis in computational linguistics, Goteborg University.

Lynch, T. & Maclean, J. (2003). Effects of Feedback on Performance: A Study of Advanced Learners on an ESP Speaking Course. *Edinburgh Working Papers in Applied Linguistics*, 12, 19–44.

Malsburg, C. (1973). Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*, 14(2), 85–100.

Mäntysalo, J., Torkkola, K., & Kohonen, T. K. (1993). Handling Context-Dependecies in Speech by LVQ. *Proc. ICANN'93, International Conference on Artificial Neural Networks*, (pp. 389–394).

Marsland, S. (2015). *Machine Learning: An Algorithmic Introduction*. CRC Press, 2nd edition.

Massaro, D. W. (2004a). A framework for evaluating multimodal integration by humans and a role for embodied conversational agents. *Proceedings of the 6th International Conference on Multimodal Interfaces (ICMI)*, (pp.24̃).

Massaro, D. W. (2004b). From multisensory integration to talking heads and language learning. In *Handbook of Multisensory Processes* (pp. 153–156). Cambridge, MA: MIT Press.

Massaro, D. W., Bigler, S., Chen, T., Perlman, M., & Ouni, S. (2008). Pronunciation training: The role of eye and ear. In *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)* (pp. 2623–2626).

Mcloughlin, I. (2009). *Applied Speech and Audio Processing with MATLAB Examples*. Cambridge.

McTenery, T., Xiao, R., & Tono, Y. (2006). *Corpus-based language studies: An advanced resource*. Taylor & Francis.

Mesgarani, N., David, S., & Shamma, S. (2007). Representation of Phonemes in Primary Auditory Cortex: How the Brain Analyzes Speech. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 765–768). New York, NY: IEEE.

Mitra, V., Vergyri, D., & Franco, H. (2016). Unsupervised learning of acoustic units using autoencoders and Kohonen nets. *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*, (pp. 1300–1304).

Munro, M. J. & Derwing, T. M. (2015). A prospectus for pronunciation research in the 21st century A point of view. *Journal of Second Language Pronunciation*, 1(1), 11–42.

Nelson, K. (1998). *Language in Cognitive Development: Emergence of the Mediated Mind*. New York, NY: Cambridge University Press.

Nti, A. A. (2009). *Studying Dialects to Understand Human Language*. PhD thesis, Massachusetts Institute of Technology.

Ohman, S. (2001). Why current speech technology is false phonetics. *Working Papers in Linguistics*, 49, 180–183.

Oja, M., Kaski, S., & Kohonen, T. K. (2002). Bibliography of Self-Organizing Map (SOM) Papers: 1998-2001 Addendum. *Neural Computing Surveys*, 3, 1–156.

O'Malley, C. & Fraser, S. (2004). *Literature review in learning with tangible technologies*. Technical report, Futurelab, Bristol, UK.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books, Inc., second edition.

Pike, K. L. (1947). *Phonemics: a technique for reducing languages to writing*. Ann Arbor: University of Michigan Press, 8th edition.

Plamondon, R. & Srihari, S. N. (2000). On-Line and Off-Line Handwriting Recognition : A Comprehensive Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1), 63–84.

Poggi, I., Pelachaud, C., De Rosis, F., Carofiglio, V., & De Carolis, B. (2001). Greta. A believable embodied conversational agent. In O. Stock & M. Zancanaro (Eds.), *Multimodal intelligent information presentation* (pp. 3–25). Dordrecht: Springer.

Polla, M., Honkela, T., & Kohonen, T. K. (2009). *Bibliography of self-organizing map (som) papers: 2002-2005 addendum*. Technical report, Helsinki University of Technology, Helsinki, Finland.

Powers, D. M., Leibbrandt, R., Pfitzner, D., Luerssen, M., Lewis, T., Abrahamyan, A., & Stevens, K. (2008). Language teaching in a mixed reality games environment. *Proceedings of the 1st ACM international conference on PErvasive Technologies Related to Assistive*

Environments - PETRA '08, 282.

Powers, D. M., Leibbrandt, R. E., Santi, M., & Luerssen, M. H. (2007). A multimodal environment for immersive language learning - space, time, viewpoint and physics. In *Proceedings of Joint HCSNet/Hxl Workshop on Human Issues in Interaction and Interactive Interfaces*.

Powers, D. M. W. (2011). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies*, 2(1), 37–63.

Pozzi, L., Gamba, M., & Giacoma, C. (2012). Artificial Neural Networks: A New Tool for Studying Lemur Vocal Communication. In J. Masters, M. Gamba, & F. Génin (Eds.), *Leaping Ahead* chapter 34, (pp. 305–313). New York, NY: Springer.

Rabiner, L. R. & Schafer, R. W. (2011). *Theory and Applications of Digital Speech Processing*. Upper Saddly River, NJ: Pearson Higher Education, Inc.

Ranjard, L. (2009). *Computational Biology of Bird Song Evolution*. PhD thesis, The University of Aukland.

Ranjard, L. & Ross, H. a. (2008). Unsupervised bird song syllable classification using evolving neural networks. *The Journal of the Acoustical Society of America*, 123(6), 4358–4368.

Ray, B. & Seely, C. (1997). *Fluency through TPR Storytelling: Achieving real language acquisition in school*. Berkeley, CA: Command Performance Language Institute.

Rei, F. (1996). TIPA : A System for Processing Phonetic Symbols in L TEX. *TUGBoat*, 17(2), 102–114.

Rei, F. (2001). *Vowel package manual*. Technical report, University of Tokyo, Tokyo, Japan.

Reiss, M. & Taylor, J. G. (1991). Storing temporal sequences. *Neural Networks*, 4(6), 773–787.

Ribeiro, C., Coutinho, C., & Costa, M. (2009). Robotics in Child Storytelling. In *Science for All, quest for excellence: 6th International Conference on Hands-on Science*, volume 9 (pp. 198–205). Ahmedabad, India.

Rigoll, G. (1990a). Information theory principles for the design of self-organizing maps in combination with hidden Markov modeling for continuous speech recognition. In *1990 IJCNN International Joint Conference on Neural Networks* (pp. 569–574 vol.1). New York, NY: IEEE.

Rigoll, G. (1990b). Neural network based continuous speech recognition by combining self organizing feature maps and Hidden Markov Modeling. In L. B. Almeida & C. J. Wellekens (Eds.), *Neural Networks, EURASIP Workshop* (pp. 205–214). Sesimbra, Portugal: Springer-Verlag Berlin.

Rigoll, G. (1991). Information theory-based supervised learning methods for self-organizing maps in combination with hidden Markov modeling. In *ICASSP 91: International Conference on Acoustics, Speech, and Signal Processing* (pp. 65–68 vol.1). New York, NY: IEEE.

Rihkanen, H., Leinonen, L., Hiltunen, T., & Kangas, J. (1994). Spectral pattern recognition of improved voice quality. *Journal of Voice*, 8(4), 320–326.

Roy, D. K. (1999). *Learning from sights and sounds: a computational model*. Doctor of philosophy, Massachusetts Institute of Technology.

Rumelhard, D. E. & Zipser, D. (1985). Feature discovery by competitve learning. *Cognitive Science*, 9, 75–112.

Sakai, T. & Doshita, S. (1962). The Phonetic Typewriter : Its Fundamentals and Mechanism. *Proceedings of IFIP Congress*.

Salhi, M. S., Arous, N., & Ellouze, N. (2013). Ability of Evolutionary and Recurrent SOM model GA-RSOM in Phonemic Recognition Optimization. *International Journal of Mathematics Trends and Technology*, 4(6), 97–106.

Samarasinghe, S. (2006). *Neural Networks for Applied Sciences and Engineering*. Taylor & Francis Group, LLC.

Schörkhuber, C., Klapuri, A., Holighaus, N., & Monika, D. (2014). A Matlab Toolbox for Efficient Perfect Reconstruction Time-Frequency Transforms with Log-Frequency Resolution. *Proc. 53rd AES Conference on Semantic Audio*, (pp. 1–8).

Sek, A. & Moore, B. C. J. (1995). Frequency discrimination as a function of frequency, measured in several ways. *The Journal of the Acoustical Society of America*, 97(4), 2479–2486.

Skrípal, P. (2006). *Parrots' acoustic communication analysis*. PhD thesis, Czech Technical University in Prague.

Somervuo, P. & Kohonen, T. K. (1999). Self-Organizing Maps and Learning Vector Quantization for feature sequences. *Neural Processing Letters*, 10, 151–159.

Spada, N. & Lightbown, P. M. (2008). Form-Focused Instruction: Isolated or Integrated? *TESOL Quarterly*, 42(2), 181–207.

Srivastava, M., Muntz, R., & Potkonjak, M. (2001). Smart Kindergarten: Sensor-based wireless networks for smart developmental problem-solving environments. *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking - MobiCom '01*, (pp. 132–138).

Stastny, J., Skorpil, V., & Fejfar, J. (2013). Audio data classification by means of new algorithms. In *2013 36th International Conference on Telecommunications and Signal Processing (TSP)* (pp. 507–511). New York, NY: IEEE.

Świetlicka, I., Kuniszyk-Jóźkowiak, W., & Smołka, E. (2009). Artificial Neural Networks in the Disabled Speech Analysis. In *Computer Recognition Systems 3* (pp. 347–354). Springer Berlin Heidelberg.

Sydlowski, A. (2009). *Offline and online character recognition using a visual input*. PhD thesis, Flinders University of South Australia.

Szczurowska, I. (2006). The application of kohonen and multilayer perceptron networks in the speech nonfluency analysis i. szczurowska. *Acoustics*, 31(4(S)), 205–210.

Tashan, T. & Allen, T. (2011). Two stage speaker verification using Self Organising Map and Multilayer Perceptron Neural Network. In *Research and Development in Intelligent Systems XXVIII* (pp. 109–122). London: Springer London.

Tashan, T., Allen, T., & Nolle, L. (2014). Speaker verification using heterogeneous neural network architecture with linear correlation speech activity detection. *Expert Systems*, 31(5), 437–447.

Tiwari, M. & Tiwari, M. (2012). Voice - How humans communicate? *Journal of Natural Science, Biology and Medicine*, 3(1), 3.

Togneri, R., Alder, M. D., & Attikiouzel, Y. (1990). Speech processing using artificial neural networks. In *Proceedings of the Third Australian International Conference on Speech Science and Technology (SST)* (pp. 304–309). Melbourne: The University of Western Australia.

Torkkola, K. (1988). Automatic alignment of speech with phonetic transcriptions in real time. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 611–614 vol.1).

Torkkola, K. (1990). A Combination of Neural Network and Low-Level AI-Techniques to Transcribe Speech into Phonemes. In *Proc. COGNITIVA 90* (pp. 1–12). Madrid, Spain.

Torkkola, K. & Kokkonen, M. (1991). Using the topology-preserving properties of SOFMs in speech recognition. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 261–264). New York, NY: IEEE.

Torkkola, K., Kokkonen, M., Kurimo, M., & Utela, P. (1991). Improving Short-Time Speech

Frame Recognition Results by Using Context. *Proceedings of the Second European Conference on Speech Communication and Technology (EUROSPEECH)*, 2, 793–796.

Tsai, P.-H. (2006). Bridging pedagogy and technology: User evaluation of pronunciation oriented CALL software. *Australasian Journal of Educational Technology*, 22(3), 375–397.

Vallabha, G. K., McClelland, J. L., Pons, F., Werker, J. F., & Amano, S. (2007). Unsupervised learning of vowel categories from infant-directed speech. *Proceedings of the National Academy of Sciences of the United States of America*, 104(33), 13273–13278.

Van Der Maaten, L. (2009). Learning a Parametric Embedding by Preserving Local Structure. *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics (PMLR)*, 5, 384–391.

Varsta, M., Heikkonen, J., Lampinen, J., & del R. Millan, J. (1998). On the Convergence Properties of the Temporal Kohonen Map and the Recurrent Self-Organizing Map. *Proceedings of ICANN98, the 8th International Conference on Artificial Neural Networks*, 2(1), 687–692.

Vasquez, D., Gruhn, R., & Minker, W. (2013). *Hierarchical Neural Network Structures for Phoneme Recognition*. Signals and Communication Technology. Berlin, Heidelberg: Springer Berlin Heidelberg.

Vatanen, T., Osmala, M., Raiko, T., Lagus, K., Sysi-Aho, M., Orešič, M., Honkela, T., & Lähdesmäki, H. (2015). Self-organization and missing values in SOM and GTM. *Neurocomputing*, 147(1), 60–70.

Veletsianos, G., Heller, R., Overmyer, S., & Procter, M. (2010). Conversational agents in virtual worlds: Bridging disciplines. *British Journal of Educational Technology*, 41(1), 123–140.

Venkateswarlu, R., Kumari, R. V., & Nagayya, A. (2011). Novel Approach for Speech Recognition by Using Self-Organized Maps. *International Journal of Computer Science & Information Technology (IJCSIT)*, 3(4), 199–210.

Verhaegh, J., Fontijn, W., & Jacobs, A. (2008). On the Benefits of Tangible Interfaces for Educational Games. In *Proceedings of the Second IEEE International Conference on Digital Game and Intelligent Toy Enhanced Learning (DIGITEL)* (pp. 141–145). Los Alamitos, CA: IEEE Computer Society.

Vesanto, J. & Alhoniemi, E. (2000). Clustering of the self-organizing map. *IEEE Transactions on Neural Networks*, 11(3), 586–600.

Vesanto, J., Himberg, J., Alhoniemi, E., & Parhankangas, J. (2000). *SOM Toolbox for Matlab 5*. Technical Report 0, Helsinki University of Technology, Helsinki, Finland.

Vij, A. (2014). Some remarks regarding English transcription practices in Taiwan. *Taiwan Journal of Linguistics*, 12(1), 25–62.

Vinayagamoorthy, V., Gillies, M., Steed, A., Tanguy, E., Pan, X., Loscos, C., & Slater, M. (2006). Building Expression into Virtual Characters Building Expression into Virtual Characters. *The Eurographics Association*, (pp. 1–42).

Vygotsky, L. (1964). *Thinking and Speaking*. Boston, MA: MIT Press.

Wagner, M., Tran, D., Togneri, R., Rose, P., Powers, D. M. W., Onslow, M., Loakes, D., Lewis, T., Kuratate, T., Kinoshita, Y., Kemp, N., Ishihara, S., Ingram, J., Hajek, J., Grayden, D., Göcke, R., Fletcher, J., Estival, D., Epps, J., Dale, R., Cutler, A., Cox, F., Chetty, G., Cassidy, S., Butcher, A., Burnham, D., Bird, S., Best, C. T., Bennamoun, M., Arciuli, J., & Ambikairajah, E. (2010). The Big Australian Speech Corpus (The Big ASC). In M. Tabain, J. Fletcher, D. Grayden, J. Hajek, & A. Butcher (Eds.), *13th Australiasian International Conference on Speech Science and Technology* (pp. 166–170). Melbourne: ASSTA.

Wakita, H. (1976). Instrumentation for the study of speech acoustics. In N. J. Lass (Ed.),

Contemporary Issues in Experimental Phonetics (pp. 3–40). New York, New York, USA: Academic Press, Inc.

Warlaumont, A. S., Oller, D. K., Buder, E. H., Dale, R., & Kozma, R. (2010). Data-driven automated acoustic analysis of human infant vocalizations using neural network tools. *The Journal of the Acoustical Society of America*, 127(4), 2563–77.

Wen, Y.-F. & Anderson, T. A. F. (2008). Distributed resource and routing assignment algorithms for multi-channel WMNs. In *Proceedings of the 4th International Conference on Wireless and Mobile Communications (ICWMC)* Athens, Greece: IEEE.

Wickham, H. (2014). Tidy Data. *Journal of Statistical Software*, 59(10).

Witt, S. M. (2012). Automatic error detection in pronunciation training: Where we are and where we need to go. *Proceedings of the International Symposium on Automatic Detection of Errors in Pronunciation Training (IS ADEPT)*, (pp. 1–8).

Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data Mining: Practical Machine Learning Tools and Techniques*. Burlington, MA: Morgan Kaufmann, third edition.

Wojcicki, K. (2011). HTK MFCC MATLAB [Computer software]. Retrieved from http://au.mathworks.com/matlabcentral/fileexchange/32849-htk-mfcc-matlab/content/mfcc/compare.pdf.

Yilmaz, A., Javed, O., & Shah, M. (2006). Object tracking. *ACM Computing Surveys*, 13(4), 1–45.

Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X. A., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., & Woodland, P. (2013). *The HTK Book*. Technical Report 9, Cambridge University Engineering Department, Cambridge.

Zhao, Z. & Rowden, C. (1992). Use of Kohonen self-organising feature maps for HMM parameter smoothing in speech recognition. *IEE Proceedings F Radar and Signal Processing*, 139(6), 385.

Zissman, M. A. & Berkling, K. M. (2001). Automatic Language Identification. *Speech Communication*, 35(1), 115–124.

Zue, V. & Cole, R. (1979). Experiments on spectrogram reading. In *ICASSP '79. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 4 (pp. 116–119). New York, New York, USA: IEEE.

Zue, V., Seneff, S., & Glass, J. (1990). Speech database development at MIT: Timit and beyond. *Speech Communication*, 9(4), 351–356.

Zue, V. W. (1985). The use of speech knowledge in automatic speech recognition. *Proceedings of the IEEE*, 73(11), 1602–1615.