Musical Instrument Identification with Feature Selection Using Evolutionary Methods

Róisín Bernadette Loughran

Submitted in partial fulfillment of the requirements for the degree of Ph.D.

The University of Limerick.

Supervised by Dr. Jacqueline Walker and Dr. Niall Griffith.

Submitted to the University of Limerick, September 2009.

Abstract

Musical Instrument Identification with Feature Selection Using Evolutionary Methods

Róisín Loughran

Musical instruments may be identified using machine learning methods, but it is not clear which aspects of the sound or *features* are best used in such methods. Classification experiments using Principal Component Analysis (PCA) and Multi-Layered Perceptrons (MLP) in this thesis find that the addition of extra features may not necessarily be beneficial — optimisation of the features is required. This optimisation is implemented using Evolutionary Computation methods as they have yet to be extensively applied in musical sound analysis.

A Genetic Algorithm (GA) with a new instrument-clustering fitness function based on PCA is applied to optimise a set of 95 features for classification with an MLP. With this method, the number of features used to classify an instrument is reduced from 95 to as low as 22 with a classification accuracy reduction of less than 0.3%. This method is tested against another evolutionary method that has not yet been applied to instrument identification — Genetic Programming (GP). GP is used to evolve a classifier program that can identify unseen samples with an accuracy of 94.3% using just 14 of the 95 original features. Though not as high as the MLP or the GA-MLP, it is found that GP is more consistent with its choice of features, offering a possible insight into timbre and the nature of sound recognition.

In both EC methods it is found that the first principal component of the envelope of the centroid, a new measure of this feature, is the most important among all 95 features. It is also seen that each classification method performs significantly better when tested with a general set of samples, than with a one-octave sample set common to each instrument. The classifiers are compared to a set of human listening tests on particularly troublesome samples. It is seen that although the GA and GP are accurate at identifying general unseen samples, the human ear performs significantly better than both methods at identifying these difficult samples.

Declaration

Title: Musical Instrument Identification with Feature Selection Using Evolutionary Methods

Author: Róisín Loughran

Award: Ph.D.

Supervisors: Dr. Jacqueline Walker, and Dr. Niall Griffith.

I hereby declare that this thesis is entirely my own work, and does not contain material previously published by any other author, except where due reference or acknowledgement has been made. Furthermore, I declare that it has not previously been submitted for any other academic award.

Róisín Loughran

Acknowledgments

I would first of all like to thank my supervisors Jacqueline Walker, Niall Griffith and Michael O'Neill in UCD for all of their help and guidance throughout the project. I would also like to thank the other students on this project Cornelia Kreutzer and Sébastien Piccand.

Special thanks to Dr. James McDermott in UCD for reading through this thesis and offering many insights to a number of chapters. I would also like to thank Dr. Marion O'Farrell for her help with the early classification experiments in this thesis.

The work carried out for this thesis was reliant on certain open source software. Most notably I would like to thank the creators of the MIR Toolbox and the GPLAB Toolbox for Matlab for sharing their work with the programming community.

Thank you to all those that volunteered for and participated in my listening tests.

Thanks to all in the CSIS and ECE departments at the University of Limerick. Thanks to all in lab A2-012 and everyone — in particular Karen — who were willing to discuss the pros and cons of pursuing a PhD over many cups of tea.

A special thank you to my family; to mum whose ability to deal with whatever comes along is a constant inspiration and to dad whose last advice was to *just stick with it.*

A final thank you to Kevin, who always said I could do it.

For Dad

Contents

List of Tables

List of Figures

Chapter 1

Introduction

As humans, we possess an astute ability to recognise sounds aurally. Although the human ear may *hear* a sound, this sound is *perceived* by the human brain or mind. Sounds are presented to and collected by the ear, but it is the brain that organises the aural inputs so that they may be categorised and hence recognised by the conscious mind. Thus the recognition of a sound is dependent on a number of processes:

- the presentation of aural stimuli
- the collection of these stimuli by the ear
- the transmission of this stimuli data to the brain
- the organisation of the transmitted data by the brain
- the conscious recognition of the organised data as a particular sound by the mind

Though we may not realise it when identifying a sound source — such as a particular musical instrument — our minds are carrying out the above processes. Chapter 8 of this thesis demonstrates that the human mind is very accurate in identifying familiar musical instruments. Even people with minimal musical experience were found to be able to identify musical instruments quite easily, even at the extremes of their pitch range. This thesis aims to determine which aspects of a sound make it distinguishable as a particular musical instrument. This problem is approached by examining the recognition of musical instrument sounds with machine learning methods.

1.1 Motivation

The motivation for the work carried out in this thesis was based on the need to develop a classifier for judging the quality of a synthesised sound. The idea for a synthesiser produced from evolutionary methods was proposed in Loughran et al. (2007). Such a model would require a method of evaluating the quality of each synthesised sound. Human evaluations may be used with evolutionary methods using *Interactive GA* (McDermott, 2008) but such evaluations may be time-consuming and costly. Using an accurate automatic classifier would eliminate the need for a human observer. From examining recent literature on automatic instrument identification, it became apparent that although many instrument classifiers have been developed, no one specific classifier has emerged as better than all others. Thus the focus of this thesis turned to developing a robust accurate musical instrument classifier — one that could judge the 'realness' of a synthesised sound regardless of pitch or dynamic. Creating such a classifier required a close examination of what it is that makes a particular sound identifiable — what are the attributes, features or aural qualities that inform our mind what instrument a note was played on? It is hoped that such a study may offer some insight into musical quality or timbre.

1.2 Problem Definition

This thesis questions what aural qualities distinguish a particular musical instrument. The process of aural recognition may be described as above, but it is not clear what aspects of the aural stimuli are most important for sound recognition.

1.2.1 Timbre

Timbre is the distinctive aspect of a sound: it is the quality of a note that distinguishes it from other notes of a similar pitch and loudness. Two notes of the same pitch played on different instruments may sound distinctly different from one another implying that recognition of an instrument is dependent on its timbre. Therefore to correctly identify an instrument, we must find a way to accurately measure its timbre. However, as discussed in Chapter 3, timbre is not a simple quality to measure. Timbre is a multi-dimensional quality that cannot be captured in one specific attribute or feature. Studies in timbre have used multi-dimensional scaling (MDS) in an attempt to gain some insight

into the different aspects of timbre (Grey, 1977; Wessel, 1979). These studies reduce the dimensions of timbre and attempt to assign each dimension to a specific measurable attribute. Obtaining the specific descriptors of enough dimensions should allow us to accurately describe timbre, but there remains disagreement as to what properties of timbre some of the dimensions represent (Donnnadieu, 2007). Thus in order to accurately identify an instrument, we must first decide which attributes or features to use in order to describe any given timbre.

1.2.2 Instrument Classification

Studies using machine learning methods have been used to address the problem of musical instrument recognition since the mid-1990s. These studies have created musical instrument classifiers using methods such as multi-layered perceptrons (Nielson et al., 2007), support vector machines (Essid et al., 2006), k-nearest neighbours (Livshin and Rodet, 2004) and self-organising maps (Cosi et al., 1994) among others. They use a number of features calculated from a selection of sound samples to train and test these classifiers. A detailed review of these studies is given in Chapter 4. It is clear from looking at these studies that there is no consensus as to which is the best method of classification to use. The studies quote various results from various classifiers. More importantly, their experiments vary in terms of instruments examined and features used. Some studies quote a large number of instruments but only classified between instrument families, whereas others only attempted to classify between instruments of the same family. The number of samples used to both train and test these classifiers also differed greatly between studies. Some studies only tested common pitches between instruments, thus limiting the scope of the classifier. Limiting the classifier in these ways may introduce a *bias* into the classification system (Herrera et al., 2000).

Such limitations in pitch — or in dynamic or playing style — are not indicative of real instrument sounds; real instruments can be played at a wide range of pitch and dynamic, with vibrato or other effects. One of the aims of this thesis is to create a more robust instrument classifier, one that could identify an instrument sound regardless of the pitch or dynamic of the note played. In doing so, we question how to define the accuracy of a developed classifier — how much does the accuracy of the classifier depend on the range of samples used to test it? Is the classifier better at classifying sounds within a smaller pitch range covering one octave or a more general pitch range of samples? How does it handle samples that are not within the 'typical' pitch

range of a given instrument?

The majority of previous studies compared the classification results of two or more classifiers, or compared the results of one classifier with those from separate studies in the literature. However, this only verifies the results of one artificial classifier against another. A true measure of accuracy would be to compare an artificial classifier against the original classifier — the human ear. When dealing with real instrument sounds we may ask: can we create a classifier that is as accurate as the human ear?

1.2.3 Feature Selection

The number and type of features used in the studies discussed above also vary widely. No consensus has been reached as to which set of features are the most important for recognising a sound, resulting in further variation in the accuracy of classification between studies. The selection of features is arguably the most important step in creating an automatic instrument classifier. Regardless of which type of classifier is used, it has no a priori knowledge of the instruments it must recognise — it can only learn from the features it is given. Thus it follows that if an appropriate set of features from a sufficient set of samples is not used as training data, then no classifier can be trained to its full potential. The question is: what is the best selection of features for successfully training an instrument classifier?

1.3 Approach to the Problem

The problems addressed above are examined by creating and testing a series of musical instrument classifiers. All of the experiments throughout this thesis were implemented using Matlab (MATLAB7, 2006). These experiments were conducted using a large range of samples from a small range of instruments: over 2000 samples from 3 instruments in Chapter 5 and over 3000 samples from 5 instruments in Chapter 6 and Chapter 7. These samples contained many instances of each pitch across the whole range of each of the instruments played at several dynamic levels. This allowed the testing of various different pitch ranges — pitch ranges common to all instruments, ranges that were more general covering the entire range of each instrument and ranges that had high proportions of pitches at the extremes of the instruments. Testing in this manner can give a more conclusive indication of how accurate and robust the developed classifier is at identifying instrument sounds.

To determine the accuracy of the final developed classifiers, they were compared against a set of human subjects. Although most of the studies described above and in Chapter 4 only compared results against other classifiers, Martin (1999) compared the results of his developed classifiers against a set of human experiments. A similar human-evaluation test is undertaken here. In this thesis however, the classifiers are first evaluated on more typical samples before being pitted against human classifiers in experiments containing a high proportion of samples in atypical pitch and dynamic ranges.

To determine the best set of features for instrument classification, a number of temporal and spectral features used in previous studies were implemented in a series of classification experiments. Various combinations of these features were used in classification experiments using principal component analysis (PCA) and multi-layered perceptrons (MLP). The differences in the results of these experiments indicated that certain features were more valuable to the systems than others. As exhausting all combinations of the available features to find the best set was impractical, a method of feature selection was required.

Methods of optimising audio features have been proposed using hierarchical classifiers with inertia ratio (Peeters and Rodet, 2003) and binary genetic algorithms for use with a k-nearest neighbour classifier (Fujinaga, 1998). This issue of optimal feature selection was addressed in this thesis using Evolutionary Computational (EC) methods. EC methods such as Genetic Algorithms (GA) have been shown to be a powerful tool for large scale feature selection in comparison to other methods (Ferri et al., 1993; Siedlecki and Sklansky, 1993b). Despite this, from the review of previous classification experiments in Chapter 4, it is clear that EC methods have not been applied extensively in the domain of musical instrument recognition. Experiments were conducted using GAs that further the work in Fujinaga (1998) by using floating point genomes with an originally developed fitness function to determine the best set of features to use with an MLP classifier. The fitness function developed is based on the clustering of the data once it has been reduced using PCA.

Although the GA may offer some insight into which features are useful for classifying instruments, the evolved result (the genome) cannot be used as a classifier directly — it still required an MLP. Thus another evolutionary method was considered that can evolve the full solution to the problem — Genetic Programming (GP). Experiments were conducted to create a musical instrument classifier using GP with a limited number of functions available to it. Like GAs, GP combines the available features, placing emphasis on

some and ignoring others. GP has an added benefit over GA for this purpose however, in that it can evolve the entire solution to the problem without the need for an external classifier such as the MLP.

Both the GA and the GP indicate which of the features examined are best used for musical instrument classification. By analysing the results from these experiments, it may be possible to find common features that occur more frequently among the results. These common features would thus be more 'important' for instrument identification. It may then be concluded that these features are more important in the description of timbre. Thus by conducting these experiments it is hoped to gain further insights into the dimensions and nature of timbre.

1.4 Contributions

The experiments undertaken as part of this thesis contribute a number of findings to the fields of timbre analysis, musical instrument recognition and evolutionary computation. Details of these contributions are outlined below.

Centroid Envelope Various statistical measures of the centroid have been included as features in many previous studies on instrument classification. This thesis used a new measure of the centroid known here as the *Centroid Envelope*. Described in Chapter 3, this envelope is a measure of the changes in the centroid throughout the duration of a note. Its dimensionality was reduced using PCA and incorporated into a number of experiments in Chapters 5, 6 and 7. The first principal component of this centroid envelope was found to be the most consistent feature to emerge as important among all the features used throughout this thesis.

Range of Training Samples The training samples used for experiments in Chapters 5, 6 and 7 of this thesis include notes from the entire pitch range of each instrument at three dynamic levels played with and without vibrato, where possible. These samples were taken from at least two different manufacturers of each instrument. This consists of more samples per instrument than previous studies in the literature, which should result in the development of a more robust classifier.

Selecting and Reducing Features with GA A floating-point GA was used in Chapter 6 to determine how much of each feature should be used for optimal instrument recognition. Despite the success of GAs in feature

selection, very few studies have incorporated them in musical instrument classifiers. In these experiments, the value of each gene within the evolved genomes indicated the relative importance of each feature. Using a number of cut-off values the features of lower valued importance were removed from the data set. This resulted in a smaller data set requiring a reduced number of computations with minimum reduction in classification accuracy.

New GA Fitness Function A new clustering fitness function was developed for the GA in Chapter 6. This function multiplied the current genome by the data set and then transformed the variance of this data using PCA. The fitness of the genome was determined according to how well the lower principal components of the data set separated and formed distinct instrument clusters.

GP Instrument Classifier From the review of classification techniques, it may be seen that GP has as yet to be applied to the problem of musical instrument identification. GP was used to evolve simple instrument classifiers as described in Chapter 7. The results from these GP runs indicate that a simple instrument classifier using just a selection of features and a small number of mathematical and logical operators may be developed with quite accurate results. The result from the most successfully evolved program is comparable with those from much more complex classifiers.

Independent Listening Tests A series of independent listening tests were conducted, as described in Chapter 8 to evaluate how recognisable the instrument samples used throughout this thesis were to candidates of varying musical experience. Most of the samples were found to be easily recognised with the most problematic samples being in the high pitch ranges of the instrument. A number of trumpet samples at high pitches proved to be particularly problematic. On average, candidates with musical experience were found to perform slightly better than those without.

Human versus Machine Evaluation The classifiers developed in Chapters 6 and 7 were tested in Chapter 8 using the same set of listening test samples as the human candidates in this chapter. This provided a validation measure for the classifiers other than comparison to those in the literature and offered a unique comparison between the success of the automatic classifiers and the human ear when tested with particularly difficult samples.

Range of Testing Samples A number of different test sets were compiled to test the effect of the choice of testing samples on the developed classifiers. The smallest set consisted of a one-octave test set, common to each instrument, with samples played at the same dynamic. Ten cross-validation sets that consisted of one tenth of the entire set of samples from all make, range, presence of vibrato and dynamic were created. The classifiers were tested using both one octave and more general tests sets in experiments in Chapter 5, 6 and 7. In addition, the listening test set consisted of 219 samples selected to have a high proportion of problematic samples. The developed classifiers were tested with this set of samples in Chapter 8. The classification results varied widely between the different test sets used to validate the classifier. Notably, tests using the more general test set offered significantly better results than tests with the more specified one-octave test set. This demonstrates the importance of selecting a test set that offers a true indication of a classifier's accuracy.

1.5 Summary of the Thesis

An overview of a musical sound is given in Chapter 2. This chapter introduces the instruments examined throughout this thesis. A discussion on how a musical sound is produced, synthesised, heard and perceived is given in this chapter. Chapter 3 discusses timbre and the way in which it is measured by experiments in this thesis. Previous measures of describing and defining timbre are discussed. This chapter also introduces and describes the timbral features measured in an attempt to classify sounds in the work carried out in this thesis. Chapter 4 describes previous work carried out in the fields of research examined in this thesis. A detailed look at previous studies in musical instrument recognition is given. This chapter also introduces the field of Evolutionary Computation and describes areas in which it has been applied to music and sound production.

The main experimental work of this thesis is described in Chapters 5 through 8. Chapter 5 describes experiments carried out in sound sample identification using PCA and MLPs. A number of experiments are described that classify between a piano, violin and flute using a selection of the features described in Chapter 3. Chapter 6 uses a GA to select the best set of features for musical instrument recognition. The fitness function developed for this GA is explained. The results of the genomes evolved and the classifications by an MLP using these genomes are discussed. This work is

furthered in Chapter 7 by implementing a GP to select the best features to use. The advantage of using GP over a GA is that GP can evolve the entire solution to the classification problem. The best evolved strings are tested and discussed. Chapter 8 describes a series of listening tests carried out with a group of human participants. The experiments in this chapter compare the results of different groups of participants with varying musical experience on a wide range of sample notes. The classifiers developed using the GA and GP from the previous chapters are compared against the results of these listening tests. Finally Chapter 9 discusses the conclusions made as a result of the experiments undertaken throughout the thesis.

Chapter 2

Musical Sound

This thesis examines the recognition of musical instrument sounds. To identify such sounds we must first understand what a musical sound or note consists of and the aspects of particular sounds that make them recognisable. This chapter defines a musical sound; how it is produced, how it can be synthesised and how it is perceived. The aural recognition of a sound is dependent on processes from three distinct disciplines: physics, biology and psychology. The combination of these three disciplines for sound perception results in the field of *psychoacoustics*. Psychoacoustics is concerned with *auditory perception*; sounds as experienced by the listener, regardless of how they were produced. It draws on the physical nature of a sound in combination with how the ear *hears* it to determine how the brain *perceives* it.

This chapter introduces the different ways in which sound may be produced in Section 2.1. A brief description of the instruments used in this study and the way in which they produce sound is given. Some basic aspects of how to describe such sounds temporally and spectrally are also given in this section. An introduction to recent methods used to synthesise musical instrument sounds is given in Section 2.2. A description of the anatomy of the ear and how it functions is given in Section 2.3 along with a number of psychological phenomena that may affect how a sound is perceived.

2.1 Sound Production

Physically a sound is produced by a waveform displacing molecules in any medium, most typically air. This waveform arrives at the ear and is perceived and translated by the brain into an identifiable sound. The current study considers the perception of sounds created by acoustic musical instruments. Musical instruments have three essential functions: to produce the sound, to

manipulate the frequency content of the sound and to project the sound. Thus every acoustic instrument may be defined in terms of a *generator*, *resonator* and *radiator* (Johnston, 1994).

2.1.1 Instruments

The instruments included in the studies in this thesis are the piano, violin, flute, trumpet and guitar. These instruments were chosen as they represent different families of instruments and are among the most common instruments played in western music. A detailed examination of any of these instruments is beyond the scope of this study, but may be found in the relevant chapters in Benade (1990). The following sections offer a brief introduction to each of these instruments from a physical perspective. Fundamentally, pitched instruments are based on creating a standing wave typically either on a stretched string or within a column of air. Such standing waves have points of minimal displacement (nodes) and points of maximum displacement (anti-nodes). For a stretched string there is a node at each end where the string is attached (and therefore cannot move). The string may oscillate in different *modes of vibration* between these two nodes. The mode with the lowest frequency is called the *fundamental* and those with higher frequencies are called *overtones* (Johnston, 1994). The first three modes of vibration for a stretched string are illustrated in Figure 2.1. Exciting a string at an anti-node will cause maximum displacement of that mode of vibration of the string. The strength and combination of these overtones control the characteristic sound of each musical instrument.

The Piano

The internal mechanisms of a piano consist of a row of felt-tipped hammers against an array of steel strings attached to a large sounding board. The hitting of the hammer against the strings is the generator, the sound board is the radiator, and the resonator is comprised of both the strings and the sound board (Johnston, 1994). The hammers hit the strings when a note on the keyboard is pressed. The mass and stiffness of each string dictates how quickly the string vibrates which in turn determines the pitch of the note played. The stiffness in the wire affects the overtones in the sound produced, causing them to be slightly sharp. This sharpness leads to the numerical frequency between two notes an octave apart on a piano being slightly larger than that between those on other instruments. This is know as a *stretched*

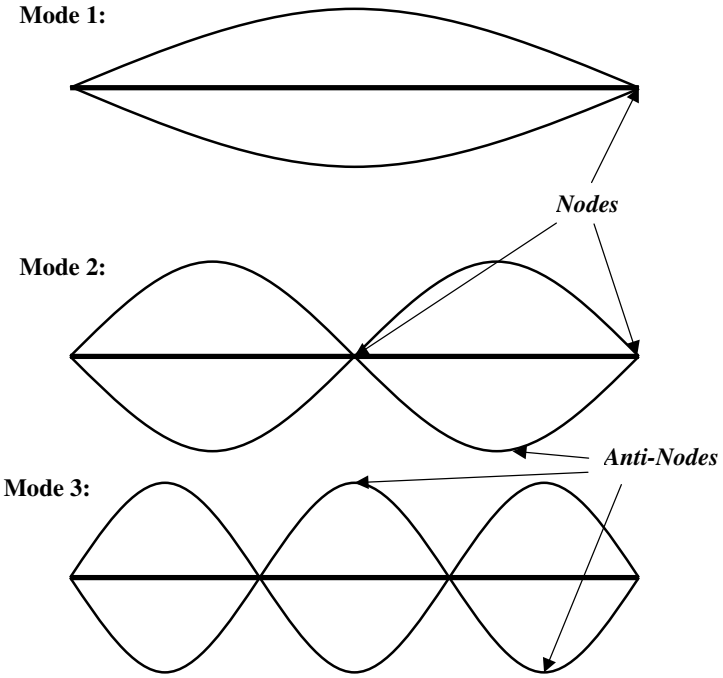


Figure 2.1: *First three modes of vibration on a stretched string*

octave. In addition to this, each hammer hits more than one string. Generally the strings are grouped in threes, each slightly de-tuned from each other. The effect of multiple strings on the tone of a piano is threefold: it increases the volume of a note, it increases the audibility time of a played note and it affects the timbre of the note (Benade, 1990). The slight de-tuning of the strings has been shown to create a pleasant sound that is preferred by listeners. A skilled piano tuner will be able to ensure these pitch anomalies are accounted for, giving the piano its distinctive sound. Such frequency discrepancies, however, make the piano a particularly difficult instrument to synthesise realistically (Fletcher et al., 1962; Keane, 2004). The piano has the widest pitch range of all commonly played acoustic instruments, spanning up to eight octaves.

The Violin

The violin is the most commonly played instrument within the string family. It consists of four metal strings (pitched to G_3, D_4, A_4 and E_5) separated

from the hollow body via the *bridge*, that run along the *fingerboard* and are tightened around the *pegs*. The top plate of the body is the main vibrating surface, made of a soft wood and containing the two distinctive *f-holes*. The strings may be plucked or bowed, although only sustained bowed violin strings are considered in this study. The bow consists of a flat bundle of hairs stretched across the stick which is arched slightly towards the hairs. This design allows the player excellent control over the amount of pressure to exert on the strings (Johnston, 1994). The resonance of the sound created by the bow on the strings is controlled by the vibration of the top plate and the air inside the body, which emerges through the f-holes. The pitch of the note played is controlled by the musician by *stopping* a string at any point with their finger.

The Flute

The flute is a member of the woodwind family. It differs from other woodwind instruments in that it is played by blowing *across* the edge opening located at one end of the instrument rather than blowing directly into the instrument. The air blown across the mouthpiece sets up a standing wave within the hollow tube of the body. The wave created by the edge opening produces a side-to-side movement of air which is symmetric. This creates a note that tends to have few overtones, giving the 'pure' tone characteristic to the flute. The pitch of a note on the flute is controlled by a series of keys that open and close holes along the body of the instrument. The opening of these holes change the effective length of the encased standing wave, thus changing the pitch of the note. The modern flute is approximately 60 cm long, with a fundamental note of D_4 (Johnston, 1994).

The Trumpet

The trumpet is a member of the family of brass instruments. In contrast to the woodwind family, the pitch of a brass instrument is somewhat controlled by the manner in which it is blown into. Original brass instruments were pitched by the players vibrating their lips at the correct frequency (thus playing the fundamental frequency of the cylinder of the instrument). Overtones of this fundamental could be pitched by vibrating their lips at higher frequencies. This limited brass instruments to a very select number of pitches. This problem was rectified in modern brass instruments though the inclusion of *valves*. These valves open up extra sections of tubing within the instru-

ment, changing the fundamental frequency and allowing a greater range of pitches to be played (Johnston, 1994).

The Guitar

The guitar is a six-stringed fretted instrument, with a hollow sound box and a long fingerboard. Guitars may be Classical (or Spanish), Acoustic or Electric, but only Classical guitar samples are considered in this thesis. The Classical guitar consists of nylon strings and has a wider neck than the Acoustic guitar. It may be played with a plectrum or the fingers, depending on the sound required. Although not always included as part of an orchestra, the guitar is arguably the most widely played instrument in the Western world due to its popularity in modern music (Johnston, 1994). The pitch of each note is controlled by stopping the string between two adjacent frets. The frets serve two purposes: they instruct the player where to press down making intonation less problematic than on non-fretted instruments, and they provide a hard surface for the string to vibrate against causing less absorption of the sound than a soft finger would cause.

The tone of a note produced on any of these instruments is dependent on both the quality of the instrument and the quality of the performer. A note played by a novice performer would sound different to that played by a virtuouso on the same instrument; likewise a performer may only be as good as the instrument they play. Thus there is more to the quality of a tone of any instrument than merely the physical specifications of how it was made. The remainder of this section discusses ways in which a sound may be described by its temporal and frequency content.

2.1.2 ADSR Model

The *temporal envelope* of any sound is a measure of the energy changes throughout the duration of the sound. This envelope can be modelled as having four sections: Attack, Decay, Sustain and Release (ADSR), (Sethares, 1998). These four sections are displayed on a trumpet note of pitch C5, as an ADSR envelope in Figure 2.2. The illustrated note displays a classic ADSR formation: the sound rises to a peak (the Attack), reduces slightly while remaining strong (Decay), maintains a relatively even level (Sustain) and then reduces to zero as the note is stopped (Release). While some notes with a strong attack and steady sustain may fit into such an envelope, the majority of real sounds do not. This can be for a number of reasons. The note may be

played quietly, thus reducing the attack as in Figure 2.3(a) which displays a quiet violin note on pitch D5. The envelope may not fit notes played at pitch extremes of the instrument such as the high-pitched A5 note played on a trumpet as shown in Figure 2.3(b). Furthermore, the instrument in question may not exhibit all sections, such as the lack of steady sustain in the piano note in Figure 2.3(c). As this thesis examines instrument sounds at a wide range of pitches and dynamics, this ADSR model will not fit a large number of samples used. Hence it is not included in the features used in this study (as described in the next chapter).

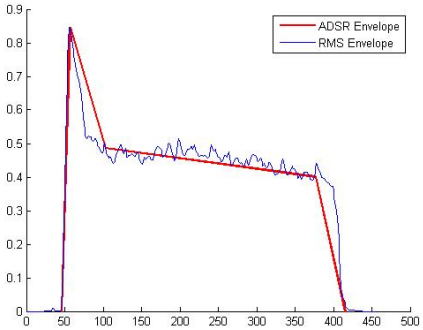
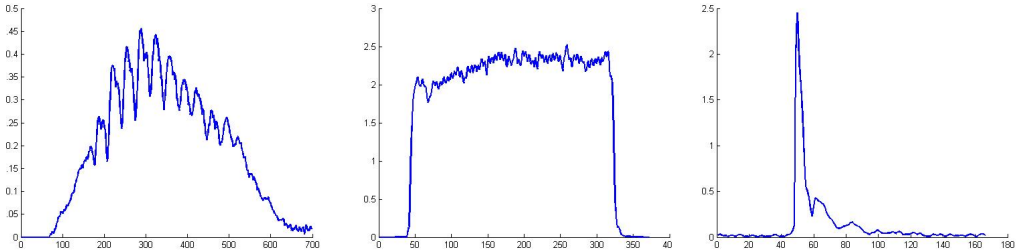


Figure 2.2: *ADSR model of the temporal envelope of C5 played on a trumpet*



(a) Soft Violin (b) High-pitched Trumpet (c) Piano

Figure 2.3: *Temporal Envelopes that do not fit the typical ADSR shape*

While all four sections of the model may not be evident in a given sound, every instrument sound does consist of a steady state section and transients. For instruments that are plucked or struck, the steady state may be difficult to define, but in general it refers to when the sound is spectrally stable. Transients are changes within the note, most notably at the beginning (attack) and at the end (release). It has been shown that both steady state and transient attributes are aurally important in identifying a sound (Martin and Kim, 1998).

This ADSR envelope models the temporal changes in the energy of the sound. The following sections discuss methods of measuring the frequency or spectral content of a sound.

2.1.3 Frequency Content

The simplest of all waveforms is the sine wave. Although pure sine waves rarely occur in nature as real sounds, Fourier's Theorem states that any complex periodic sound can be decomposed into a series of sine waves. Thus any naturally occurring or artificially produced pitched sound can in theory be completely described by the amplitude, frequency and phase of its constituent sine waves or harmonics. The Fast Fourier Transform (FFT) (Cooley and Tukey, 1965), an implementation of the Discrete Fourier Transform (DFT) is most commonly used to determine the frequencies within a sound. Analysing the strength and frequency of partials within a sound is known as *spectral decomposition*.

Fourier methods have a drawback however in that there is a trade off between frequency resolution and time resolution, depending on the size and shape of the window used. An alternative to these Fourier methods is to use Wavelets which decompose the sound using a variable window size (Teolis, 1998). Although a number of studies do employ Wavelets, the DFT is still the most commonly used method. Fourier analysis is used in many of the feature extraction methods described in the next chapter and hence is used in this thesis rather than Wavelets. Fourier analysis allows the extraction of specific frequency information from a sound such as the fundamental frequency as described next.

2.1.4 Fundamental Frequency

As discussed above, a standing wave may vibrate at a number of vibrational modes. The fundamental mode corresponds to the lowest or fundamental frequency (F_0) of the note produced. The overtones vibrate at integer multiples of this frequency. These overtones or *harmonics* produce the rich timbre of a complex harmonic sound. Thus F_0 of a real pitched sound can be calculated from the ratio between the frequencies of the upper harmonics during the sustained part of the sound. For real sounds, however, the relationship between the harmonics is rarely this exact. Harmonics that deviate from a

perfect integer multiple of the fundamental are referred to as *upper partials*[1]. This imperfect relationship means that obtaining the fundamental frequency for a real sound can be a difficult task. A method used for simple sounds is to measure the zero-crossing rate (the number of times the signal goes from positive to negative), but this is not particularly accurate for more complex sounds.

Obtaining F_0 in the time domain may be done through the process of *autocorrelation* as originally proposed in Licklider (1951). This process compares a sound with a delayed version of itself. A periodic signal repeats itself every cycle. Thus the point at which maximum correlation is found indicates the value of F_0. Mathematically this can be stated as: given a signal *f(t)*, the continuous autocorrelation $R_{ff}(\tau)$ is the continuous cross-correlation integral of *f(t)* with itself at lag τ ,

$$R_{ff}(\tau) = \int_{-\infty}^{\infty} f(t+\tau)\bar{f}(t)\,dt \tag{2.1}$$

where $\bar{f}$ represents the complex conjugate of f (for real functions $\bar{f} = f$).

Spectral methods of estimating F_0 are often based on the *cepstrum* of the sound (Oppenheim and Schafer, 1989). The real cepstrum may be considered as a projection of the spectrum on a set of cosine functions. It is defined as the inverse Fourier transform of the logarithm of the magnitude of the Fourier transform,

$$c_x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} log|X(e^{j\omega})|e^{j\omega n}\,d\omega \tag{2.2}$$

Both the autocorrelation and the cepstrum can be implemented using the signal processing toolbox in Matlab (MATLAB7, 2006).

A number of algorithms have been developed based on a modification of one of the above methods in an attempt to estimate the fundamental frequency of a given real sound. A F_0 estimator for quasi-harmonic sounds is proposed in Maher and Beauchamp (1994). This method, known as the *'Two-way Mismatch'* method estimates F_0 by minimising the discrepancies between measured partial frequencies and upper frequencies calculated from trial values of F_0. This method achieves encouraging results and is used in a number of applications including Spectral Modelling Synthesis (see next section). Another popular F_0 estimator is the YIN algorithm (de Cheveigne and Kawahara, 2002). This is based on the autocorrelation method, modified

[1]In the literature, the terms harmonics, overtones and partials are often interchangeable; here we consider 'harmonics' to be always perfectly harmonic, whereas 'partials' may be harmonic or not.

to reduce errors. It is suitable for high-pitched voices and music and gives error rates about three times lower than competing methods. An F_0 estimator based on the salience of a number of potential F_0 values as a weighted sum of the amplitudes of its harmonic partials was proposed in Klapuri (2006). A combination of the autocorrelation and the cesptral frequency estimation methods were used for a pitch representation algorithm in Peeters (2006).

A fundamental problem with all F_0 estimation methods, however, is that only periodic sounds have a fundamental frequency. For most if not all real acoustic instruments, upper partials deviate from this ideal to some degree making the sound only pseudo-periodic. As discussed above, instruments such as the piano rely on stretching the relationship between upper partials and the fundamental (particularly at the pitch extremities) for their rich distinctive tonal quality (Fletcher et al., 1962). In addition to this, elements such as noise and vibrato can interfere with any F_0 estimation algorithm. These problems may lead to inaccuracies in finding F_0 for problematic sounds, such as the low-pitched piano samples used within this study. A comprehensive review of models to estimate F_0 is offered in de Cheveigne (2005).

2.1.5 Resonance and Formants

As discussed at the beginning of this section, every instrument is comprised of a generator, resonator and radiator. The resonator within an instrument is vital to the sound quality produced by that instrument. Each instrument filters the sound in a unique manner, certain frequencies are boosted whereas others are attenuated, leading to a resonance curve. A specific resonance curve can be approximated by a number of *formant peaks*, each centered around a particular frequency. The amount of effect this formant has on the frequencies it is boosting is dependent on its damping factor; the less damped a system (instrument) is, the more the resonant frequencies will be boosted. In addition to this, the range of frequencies affected by the formant peak is given by its quality, Q, factor. In low Q systems a wide range of frequencies are affected, giving a wide formant peak. Conversely, formant peaks with a high Q factor are narrow, affecting only a few surrounding frequencies. The resonance curve of a particular instrument affects the timbre of the instrument by boosting the strength of specific frequencies played by the instrument. This curve remains the same regardless of the fundamental frequency of the note. Thus to maintain a similar timbre when a note is transposed, the individual levels of the harmonics must be adjusted with respect to the resonance or formant shape.

2.2 Sound Synthesis

Although the current thesis concentrates on sound analysis rather than synthesis, a brief discussion on the main methods of sound synthesis is given in this section. Synthesis may be considered as the *opposite* process of the analysis performed throughout this thesis, and hence understanding it may be beneficial in designing an accurate classifier. A musical identifier developed on real instrument sounds may be used to classify a synthesised musical instrument sound or to evaluate the 'realness' of such artificial sounds. Analysis-synthesis techniques reduce a given sound into its relevant components so that it may be reconstructed by synthetic means. The two main types of sound synthesis are spectral (or signal) modelling and physical modelling. Spectral models are based on the spectral content of a sound as it reaches the ear, regardless of how it was produced. Spectral modelling employs the same representation in synthesising a sound as used here in analysing it; a note is decomposed spectrally regardless of the physical parameters that may have created it. Physical models on the other hand are based on the behaviour of the physical sound source. These physical models are usually specific to one type of instrument, as the physical parameters of each instrument vary. As physical modelling is more computationally expensive than spectral modelling, attention has been focussed on spectral modelling in the past. The increase in computer speed has led to a new interest in physical modelling in recent years.

2.2.1 Additive and Subtractive Synthesis

Additive synthesis is one of the simplest yet most powerful synthesis methods. Through Fourier analysis, the steady state of any sound can be analysed into its harmonics. These harmonics can then be individually re-created and added together to form an approximation of the original sound. Additive synthesis is the basis for a specific type of spectral synthesis known as *sinusoidal modelling*. In this method a signal is windowed and a spectral algorithm is used to track the peaks of the signal from one frame to the next. The calculated frequency, amplitude and phase of each partial in each frame can then be calculated and used to resynthesise the partials, thus recreating the original sound.

Subtractive synthesis uses a harmonically rich sound such as white noise and filters it until only the desired components are left, imitating an acoustic instrument's *generator-resonator* method of producing sound. A simple

example of subtractive synthesis is the vocoder, which consists of a bank of narrow band-pass filters excited by a broadband signal. Linear Predictive Coding is similar to the vocoder, but replaces the bank of simple filters with a more complex filter that typically approximates the frequency response of some instrument (Garcia, 1996). Subtractive synthesis can be less computationally expensive than additive synthesis but it can also be less accurate in its output.

2.2.2 SMS and TMS

While additive methods can produce accurate harmonic representations of sounds, the sounds produced are not realistic as they contain no noise. Real sounds have a noise element that give the sound 'naturalness', but this naturalness is not easy to quantify. Spectral Modelling Synthesis (SMS) is a technique developed by Julius Smith and Xavier Serra that synthesises tones by modelling them as two parts: a relatively stable sinusoidal (the spectral or deterministic) part and a noise (stochastic) part (Serra, 1990). The sinusoidal part is produced using sinusoidal modelling. This part is subtracted from the original signal leaving a residual noisy signal. This noise signal can be analysed and modelled using filtered white noise and finally added to the sinusoidal part to give a complete synthesised model of the original waveform.

One problem with the SMS model is that it does not model transients efficiently. As transients are short-lived signals, it is not efficient to model them with sinusoids. On the other hand, modelling transients with filtered noise causes them to lose some of their attack. Without realistic synthesis of the transients, the synthesised signal tends to sound dull. This problem is overcome in Verma (2000) by synthesising the transients on their own and combining Transient Modelling Synthesis (TMS) with SMS. The synthesised sinusoidal signal is calculated and again subtracted from the original signal, leaving the residual noise plus transient signal. As transients are frequency-rich impulsive sounds, this signal is transformed into the frequency domain for analysis using the discrete cosine transform (DCT). Once this signal has been analysed it can be synthesised and re-transformed back into the time domain. This can then be subtracted from the residual (noise plus transients) signal, allowing the noise signal to be synthesised as before.

2.2.3 Further Synthesis Methods

While spectral modelling may be of relevance to this work due to the similarities in sound wave representation, other approaches to sound synthesis have been examined. Physical modelling uses mathematical approximations of the physical excitation and resonant responses of systems such as musical instruments. The aim of physical modelling is to implement the physically and perceptually important properties of an instrument using signal processing algorithms. While this may be more computationally expensive than spectral modelling, it has the advantage that all of the parameters obtained in the analysis stage are intrinsic to the instrument itself. A number of studies have modelled string and wind instruments with physical methods involving the use of wavetables and filters (Bank and Valimaki, 2003; Holm and Toiviainen, 2004; Karjailnen et al., 2001; Karplus and Strong, 1983; Nackaerts et al., 2003). Further synthesis methods such as *granular synthesis* (Roads, 1996) and FM synthesis (Chowning, 1973) have been used in the modelling and synthesis of instrument sounds.

So far this chapter has concentrated on the physical aspects of sounds — how a sound may be described spectrally and how it may be synthesised. A sound as we hear it may be more than this however. Human perception often results in a sound or event being greater than the sum of its parts. The following section describes how a sound is perceived by the human ear.

2.3 Hearing Sound

Hearing is the process by which aural information is presented to our brain via our ears. The function of an instrument identifier is to 'mimic' the ability of the ear (and brain) to hear and identify a given sound. In developing such an identifier using Artificial Intelligence techniques, it is worth looking at the biological and perceptual ways in which the human ears and brain perform the same task, as described in this section.

2.3.1 The Ear

The ear is a complex organ consisting of outer, middle and inner sections. The outer ear consists of the external pinnae and the auditory canal. The main acoustic purpose of the outer ear is for resonance and sound localisation. The middle ear consists of three small bones situated between the tympanic membrane (ear-drum) and the oval window of the cochlea. The function of

the middle ear is to transmit sounds from the tympanic membrane to the cochlea and to protect the ear from damage from loud sounds. The inner ear consists of a coiled snail-shaped structure known as the cochlea, one end of which is called the base, the other the apex. It is a fluid filled tube divided into three sections by Reissner's membrane and the basilar membrane. The basilar membrane carries out a frequency analysis of any incoming sound. The basilar membrane is narrowest at the base and steadily widens towards the apex. It vibrates in response to an aural stimulus acting as a resonator system in which a specific area is excited depending on the frequency of the sound presented. Thus the frequency of a sound is detected by the place at which this membrane vibrates. This is known as the place theory of pitch detection. A more detailed description of the workings of the ear is given in Mathews (1999).

The following sections on critical bands, pitch and consonance perception are all based on which parts of the basilar membrane are excited by a given sound.

2.3.2 Critical Bands

A pure sine tone does not just excite a single location on the basilar membrane. There is a resonance bandwidth of approximately a minor third or three semitones that is excited by a pure tone stimulus. If two tones very similar in frequency are presented to the ear simultaneously, two overlapping regions exhibit a response. When the difference between these tones is extremely small, about 12.5 Hz or less, it leads to constructive and destructive interference between the two tones which causes the perception of beats. If the tones are further apart but still overlapping on the basilar membrane, there may be difficulty in discerning the frequency of the sounds. This leads to perception of roughness of the tones. If the tones are separated further, a point is reached whereby the roughness is diminished and two separate smooth tones are heard. The distance at which two tones are heard separately is known as the 'critical band' (Howard and Angus, 1996). The critical band is more formally defined in Scharf (1970) as *'...that bandwidth at which subjective responses rather abruptly change'*. The critical band may be used in developing a series of perceptually equal auditory filters. A filter of such design with an ideal rectangular frequency response curve is known as the *equivalent rectangular bandwidth* (ERB). Classical theory of the critical bandwidth as a function of frequency states that it is approximately constant below 500Hz and increases above that frequency roughly in proportion to the

centre frequency (Zwicker and Terhardt, 1980). The bandwidths for such a filter are calculated as (Glasberg and Moore, 1990):

$$ERB = 24.7(4.37f_c + 1)Hz \tag{2.3}$$

where f_c is the filter centre frequency in Hz. Alternative models for the ERB have been proposed in Moore and Glasberg (1983).

2.3.3 Pitch

Pitch is the subjective measure that is related to the frequency of the sound. The basic interval of pitch detection is the octave. This is the only naturally occurring interval corresponding to an exact doubling of the frequency. Due to its simplicity, the octave is used in music from most cultures, but it can be further segmented in many different ways. Western music uses the 'Equal Tempered Scale', splitting the octave into 12 equal steps or notes. The frequency of each successive note is calculated by multiplying the previous note by the *twelfth root of two*. Music can be transposed easily using this method of tuning, as all pairs of adjacent tones have identical frequency ratios (Howard and Angus, 1996). As discussed above, real tones consist of a fundamental frequency and a number of harmonics. The fundamental often corresponds to the pitch of the note, but it is not necessarily the strongest partial in the sound. Even if the fundamental is not present it is possible to hear the pitch as being that of the fundamental and not that of the lowest frequency present. It is the relationship between the partials, rather than one specific partial, that dictates the perceived pitch of a note. The case of the missing fundamental is sometimes known as 'virtual pitch' (Howard and Angus, 1996). It may be used in sound systems and instruments that do not have the ability to produce the low fundamental frequencies of the pitches played.

2.3.4 Consonance and Dissonance

As a complex tone is composed of a number of different harmonics, such a sound causes a number of areas of excitation on the basilar membrane. Consider two tones played simultaneously; if the partials of these tones are close to each other (but not overlapping) they interfere with each other on the basilar membrane. Thus the combined sound will be perceived as rough or *dissonant*. If on the other hand the partials of the two sounds overlap each

other, the sound will be perceived to be more pleasant or *consonant*. The pitch relationship or *interval* between these two tones affects the consonance perceived by combining them. The octave is the most natural sounding interval with frequency ratio of 2:1[2]. Because of this ratio the harmonics from two tones an octave apart should all overlap each other. From examining other intervals it is apparent that the next most consonant (or pleasant) interval is the perfect fifth which has a ratio of 3:2 between the two pitches. In general the ear tends to favour intervals that are based on simple integer ratios (Sethares, 1998) owing to more overlap of their partials. Simultaneous tones that have less simple ratios can cause more interference on the basilar membrane resulting in the sound being perceived as more dissonant.

In addition to being frequency selective, the ear will only perceive sounds that are presented at a certain loudness. The following sections describe how sounds that are physically loud enough to be heard are not heard for sub-conscious or conscious reasons.

2.3.5 Masking

In sound classification studies, such as those undertaken in this thesis, it is worth noting that not all frequencies present in the sound will necessarily be heard. The threshold of audibility is different for each frequency. The equal threshold curve displays the sound pressure level at which each frequency must be present for it to be audible. One such set of curves depicting sound pressure levels for equal loudness are the Fletcher-Munson curves, (Fletcher and Munson, 1933). These curves demonstrate that the human ear has a loudness peak sensitivity at approximately $3kHz$.

Masking is the phenomenon whereby one strong tone can diminish the audibility of a simultaneous or neighbouring weaker tone. The presence of a strong tone lifts the loudness contour up around it, creating a new curve as depicted in Figure 2.4. From this diagram it is clear that the $1kHz$ sine tone would be heard as it reaches above the threshold of audibility. The effect of this tone adjusting the curve however, results in making adjacent tones, that may have otherwise been clearly audible, inaudible. Thus the strong $1kHz$ tone masks any weaker tones that are close to it in frequency. Masking curves are asymmetrical in shape, being more gradually sloped on the high frequency side. Thus a masking tone is more prominent over tones that are higher in frequency that itself. Masking has a practical use in audio compres-

[2]It has even been shown that animals can detect this interval.

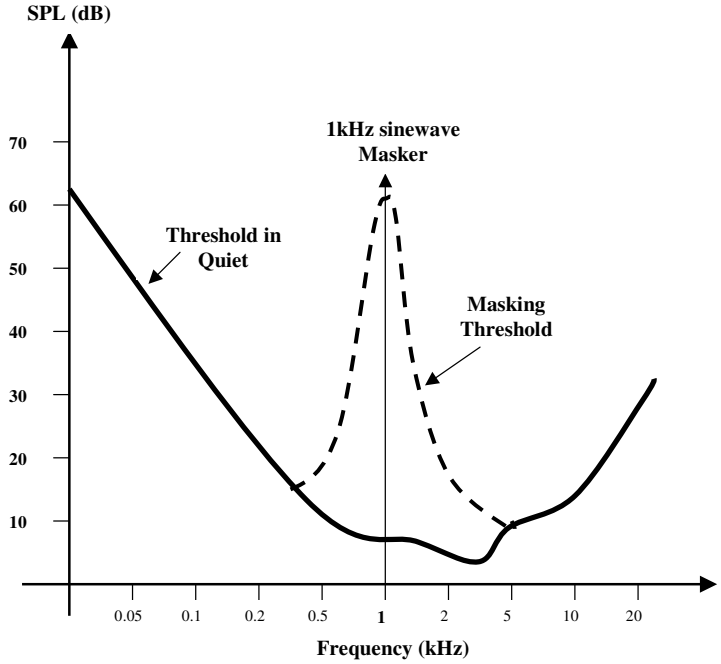


Figure 2.4: *Effect of a 1kHz Masking tone on the threshold of audibility in quiet (adapted from Haritaoglu (1997))*

sion technologies such as MPEG standards (Watkinson, 2004). Redundant sounds that are masked by adjacent stronger frequencies can be identified and removed, thus reducing the amount of data that is to be stored. Masking has also been used in the measurement of frequency bands such as verification of the critical bandwidth (Greenwood, 1961a).

2.3.6 Hearing versus Listening

We hear sounds presented at our ears, regardless of whether we intended to or not. Unlike our eyes, our ears are not easily closed and so if we wish to block out sounds we must take measures, such as ear-plugs, to do so. To *hear* something implies a passive response to an aural stimuli which is neither intentional nor focussed. To *listen* to a sound implies a more active, intentional concentration on a particular sound that may interest us. Sounds from various sources in our surroundings combine together and are presented as one complex wave, yet we are able to distinguish one sound from another. The manner in which we organise the sounds that we hear in a coherent scene is known as auditory scene analysis (Bregman, 1990). In a crowded place with music and background noise, we are still able to hold a conversation with a

person next to us. This phenomenon is known as the *cocktail party effect* and demonstrates the brain's ability to discern what aural information is important and what is superfluous. Sounds are grouped together, in music and in everyday scenes, according to similarities in aspects of the sound such as pitch, timbre, loudness or location. This phenomenon is known as auditory streaming (Van Noorden, 1975).

There are several different types of listening. The ear may be required to discern the typical source of a sound. If sounds from particular sources were always identical, then it would be a matter of memorising every sound (Handel, 1989). Due to variation within each sound source and its interaction within the environment, the source of a sound may not be 'heard' as the same each time it is presented. Thus the ear must be able to contextually recognise a sound source each time a complex sound is presented to it. This type of listening is referred to as *everyday listening*. Everyday listening involves the perception of sound-producing events, associating the sound heard with an object or incident that may have caused it (Gaver, 1993). *Musical listening* is the experience of the sounds themselves, the frequencies present, the relationship between partials and the melodies created.

In the context of the experiments carried out in this thesis, it is important to be aware of such differences in listening as opposed to hearing. Chapter 8 of this thesis describes listening tests undertaken by a number of participants. In identifying instruments in these tests, everyday listening is required but in a musical context. Although the participants are listening to musical notes, it is the source of the note that they are asked to identify. The results of these tests are directly compared to those from the developed automatic instrument classifiers. Thus we compare the accuracy of the developed classifier against a group of candidates actively listening to as opposed to passively hearing the sounds presented to them.

2.4 Conclusion

From a signal processing perspective a sound is a time-varying waveform, but in this thesis it is considered to be more than that. This chapter introduced musical sounds; how they may be produced, synthesised and heard. The aim of this study is to gain a deeper understanding of how we recognise the sounds produced by specific instruments. Attributes of musical sounds were described here as an introduction to concepts used in later experiments and discussions.

Section 2.1 described how notes are produced, in particular by the instruments used in this study: the piano, violin, flute, trumpet and guitar. This section introduced how a sound may be described temporally (such as by the ADSR model) or spectrally (by frequency content and resonance). Such temporal and spectral methods are the basis of a number of sound features and descriptors used within this study. Section 2.2 briefly introduced methods by which musical sounds have been artificially produced using sound synthesis. Analysis-synthesis methods such as SMS were discussed as an opposite view of the analysis procedure undertaken in this study. Although such methods are not used further in this study, they were considered as they offer a perspective into the construction and analysis of a given sound. The human ear and the way in which it decomposes a sound was introduced in Section 2.3. We examined the place theory of pitch perception and the way in which it affects our perception of complex sounds. Finally we noted the way in which people consciously listen to as opposed to subconsciously hear sounds.

As humans we may be adept at recognising sounds occurring around us, but such an ability is not easily explained, let alone reproduced artificially. From a qualitative, descriptive perspective to a quantitative, physical perspective, a given sound may be inherently difficult to define. In trying to discover what makes a sound unique or identifiable, we must first clarify the aspects of sound that affect our understanding and perception of the sound portrayed to us. The concepts discussed in this chapter such as pitch and frequency, temporal envelopes, resonance and loudness curves may *physically* describe a sound, but what identifies a sound perceptually has not yet been fully considered. The following chapter attempts to define what is it that makes a note recognisable by examining the concept of *timbre*.

Chapter 3

Timbre and its Measurement

The previous chapter discussed properties used to describe and analyse musical instrument sounds. While such attributes may offer some explanation as to what physically constitutes a sound, they do not necessarily explain how we recognise the source of specific sounds. This thesis attempts to determine the most important aspects of a sound to identify it as played on a specific instrument: what distinguishes A5 played loudly on a trumpet from A5 played equally loudly on a violin? We approach this problem by examining the concept of *timbre*.

The purpose of this chapter is to introduce timbre and establish the way in which it is described in this thesis. The concept and definition of timbre are discussed in Section 3.1 along with methods that have attempted to describe and quantify timbre. In particular a number of Multi-Dimensional Scaling techniques are described that have been used in the past to attempt to give meaning to the dimensions of timbre. Section 3.2 introduces the timbral features and descriptors used in experiments throughout this thesis. Temporal and spectral features that have been developed for this study are included along with those obtained from specific toolboxes. Finally Section 3.3 gives an overview of the chapter and proposes further directions in the work.

3.1 Defining Timbre

A sound can be completely described by four characteristics, namely its pitch, volume, duration and timbre. Duration refers to the length of the sound and can easily be measured in seconds. Volume refers to the perceived loudness of a sound and can again easily be measured as the amplitude of the sound. Pitch is related to the frequency of the sound. Although it can be more

troublesome to measure than duration or loudness[1], for pitched sounds it can generally be measured, as described in the previous chapter, from the relationship between the harmonic partials of the sound or the fundamental frequency. Perceptually, pitch is quite easy to hear; it is that quality of a sound that allows a listener to name the 'note' (C4, A5 etc.). Timbre, on the other hand, is not so easily described or measured. It is often referred to as the 'quality' or 'colour' of a sound, but this is not particularly helpful in measuring it[2]. Considering two notes both at pitch C4 of 2 seconds duration played at equal loudness on a violin and a flute, it is reasonable to say that most people would be able to decide which instrument each note was played on. Thus it must follow that our aural recognition of musical instruments is dependent on timbre. If we consider the sound as a waveform, then physically the duration is a measure of the length of this waveform, volume a measure of its amplitude and pitch a measure of its frequency — but what physical quality does timbre rely on? Conceptually it could be considered as the variation or diversity of the sound, although by process of elimination it refers to *everything that has not been described already.* The most often quoted definition of timbre describes it as such (ANSI, 1973):

> Timbre is that attribute of auditory sensation in terms of which a listener can judge two sounds similarly presented and having the same loudness and pitch as being dissimilar.

This definition is flawed in that it describes what timbre is not, as opposed to what it is, and yet a significantly improved definition has yet to be proposed. This lack of clear definition leads to the difficulty in measuring timbre — how can one measure what one cannot define? Nevertheless, obtaining an objective way of defining or describing timbre is paramount in gaining a deeper understanding of what timbre actually is. This in turn would lead to more consistent methods of describing sounds and hence creating more effective automatic instrument classifiers. Handel (1995) gives a further discussion and offers insights into the timbre of an event or object.

3.1.1 Describing Timbre

The lack of a clear definition of timbre makes it difficult to describe verbally. With no objective measures available, a reliance on subjective descriptors is

[1] In that in certain cases it may be dependant on other features, or be somewhat ambiguous perceptually, for example the Shephard Scales (Shephard, 1964)

[2] In fact if we consider an analogy with light 'colour' would represent frequency

inevitable. A large number of verbal characteristics have emerged in the attempt to describe the quality of a sound. A list of adjectives used to describe timbre include: mellow, rich, covered, open, dull, bright, dark, strident, grating, harsh, shrill, sonorous, sombre, colourless and lacklustre (Howard and Angus, 1996), although many more are likely to have been used. The meaning of such descriptors in an aural sense may be unclear. Terms such as 'open' have no clear, specific meaning in relation to sound, which may cause ambiguities between candidate's results in choosing such words. The meaning of such verbal descriptors, especially when discussing timbre may differ somewhat between candidates, particularly if the candidate considers them to be context-dependent. More specifically, language translations may cause significant changes in the underlying or true meaning portrayed by such descriptors, leading to difficulties in globally defining or describing timbre with such attributes. Furthermore, a problem may arise where candidates could attribute some physical meaning to their own understanding of a given sound. Deciding that a particular note sounds 'a bit like a trumpet' for example, attaches an inherent meaning for the candidate that may not be present for others. For purely objective descriptions, it is imperative that candidates maintain an impartial, context-free opinion of each sound, although such measures are inevitably almost impossible to enforce.

A number of studies have been conducted using a scale between a descriptor and its opposite such as *soft-hard* and then asking subjects to judge the placement of the sound on this scale (Kendell and Carterette, 1993; von Bismarck, 1974). In choosing the descriptors used, this method (known as semantic differential (Osgood et al., 1957)) asks subjects to first decide on a number of descriptors that would be useful in timbre description. Subjects then listen to a tone and rate it along one of the decided scales. More recently, a series of listening tests were conducted to determine the effectiveness of communicating assessment judgements regarding timbre (Darke, 2005). They conducted tests on 22 subjects listening to 15 sounds. These sounds were from different instruments but were corrected to be of the same volume and approximately the same pitch. Subjects were asked on a scale of 0-5 how much each of the following attributes fitted to the sound: bright, harsh, clear, thin, hard, full, nasal, muted, reedy, brassy, metallic, wooden. The subjects were split into two groups with one group hearing the sounds played in inverse order to the second group. The results showed significant variance in responses from different subjects. Thus the experiment concludes that one subject, after hearing a sound once, will not necessarily rate each

sound similarly to other subjects. Differences in the judgements made by the two groups were also found, indicating that in such listening experiments judgement of a current sound depends somewhat on that of sounds recently heard. Many similarity tests ask the users to gauge the difference in sound according to a certain descriptor. The results from Darke's experiment indicate there may be a lack of consistency as to how much each candidate considers a particular descriptor to be indicative of a given sound. Nevertheless, tests based on similarity judgements have been the most common method of quantifying timbre used in recent years.

Description experiments such as those described above are inherently subjective. More objective tests examining the multi-dimensionality of timbre were conducted to explore the idea of a 'timbre space'. Such experiments are described in the next section.

3.1.2 Multi-Dimensional Scaling and Creation of a Timbre Space

From the number and variety of descriptors that can be used to describe timbre, it is evident that any quantitative method of describing it must be multi-dimensional. It is not clear from such descriptive studies however, the number of dimensions that are necessary and sufficient for such a description. As such, a statistical technique known as multidimensional scaling (MDS) (Borg and Groenen, 2005), used in information visualisation, was applied to the analysis of timbre. The application of this technique (first used in Plomp (1970)), involves presenting a group of subjects with a number of pairs of tones differing in timbre but alike in all other aspects. The subjects are asked to consider how dissimilar each pair of tones are from one another and to rate this difference on a scale. The results are stored in a matrix and subjected to analysis with a multi-dimensional scaling algorithm. This can then be mapped onto a geometric structure in Euclidean space in which each timbre is represented by a point. Sounds that are similar in timbre are thus close to each other in this space, whereas sounds that are perceived to be very different are far away from each other (McAdams and Cunible, 1992). This method was used in a number of studies during the 1970's (Grey, 1977; Grey and Gordon, 1978; Wessel, 1979). These early experiments developed models on either two or three dimensions. It was then attempted to verbally connect these common dimensions to timbral attributes. Two dimensions were described in Wessel (1979): one related to the spectral energy distribution and the

other to the nature of the onset transients. Grey found that one dimension related to the spectral energy distribution, one related to the presence of low-amplitude, high frequency energy and the third represented the presence of synchronicity in the transients of the higher harmonics and level of spectral fluctuation (Grey, 1977). These results were supported in Grey and Gordon (1978) by exchanging the shape of the spectral energy distributions between the pairs of tones and noting a corresponding swap in the order on the spatial axis. In general, it is accepted that the two most prominent axes to emerge from such MDS studies on timbre relate to spectral energy distribution and the nature of the onset transients. There remains disagreements however as to what property or attribute of timbre the third (or indeed further) dimension may be most representative of (Donnnadieu, 2007; Hajda et al., 1997).

The MDS techniques thus create an interpretable geometric representation of musically useful timbres, or 'timbre space' in which each timbre has a distinct position. The experiments above looked at two and three dimensional spaces. While there is the the obvious advantage of being able to visualise and plot up to three dimensions, it is possible to consider many more dimensions with Euclidean space, although most studies tend to keep the dimensionality low. These timbre spaces can be used to interpolate between different specific timbres of instrument to create novel hybrid instruments. Further studies have used these timbre spaces to examine the perceived effect of altering one property of a sound. MDS and vector models were used to test subjects' abilities to perceive timbral relations and to judge their similarity in terms of magnitude and direction in McAdams and Cunible (1992). MDS was used to study the effect of musical training on the perceptual structure of musical timbre in McAdams et al. (1995). In this study, the model was modified by dividing a large number of subjects with varying musical experience into a number of latent classes — latent meaning it was not known in advance which class a subject belonged to. These latent classes were used in an effort to determine meaningful dimensions common to the whole population as well as some that may be specific to certain classes. Experiments were designed to use MDS to examine the perceptual relevance of timbral attributes by varying these attributes on synthesised tones in Caclin et al. (2005). Likewise, in Ilmoniemi et al. (2004), specific aspects of cello tones were modified to note the effect on the perceived difference in timbre using a timbre space. They conclude that changes in the centroid frequency (which is related to brightness) gave the most notable difference. Instead of applying MDS to listener judgements, in Hourdin et al. (1997) it was applied to

physical description of the sound to classify these physical features in order of decreasing importance. They found the spectral sound energy to be the most significant feature, agreeing with previous subjective experiments. Although MDS has been used for numerous timbre studies, problems with the method have been discussed, such as those in Terasawa et al. (2005). This study proposes two problematic issues with MDS: that axes are not initially labelled and that it does not help find a new sound that has a needed distance between other sounds. They suggest an alternate inverse method of looking at timbre by starting with dimensions, synthesising sounds and then noting the perceived differences as measured by human subjects. In doing this they attempt to develop a quantitative mapping between a physical description of timbre and its human perception.

From the discussion above it is clear that MDS has played an important role in the development of our understanding of timbre. Whereas initially it was introduced as a method of investigating timbre, the use of MDS has progressed to become a useful tool in confirming the relevance of specific attributes. While it is accepted that the onset transients and spectral distribution are important, other aspects of sound have emerged as possible useful measurable descriptors of a sound.

This section described the way in which timbre has been described in experiment in the past. The following section describes the measures by which timbre is described throughout this thesis.

3.2 Timbre Features

Musical instrument classifiers have been developed that use certain auditory features to describe timbre and hence identify a specific instrument. Details of these studies are given in the following chapter. The number and range of these features varies between each study. Both temporal and spectral features are used in such experiments. This thesis aims to determine the optimum choice of such features for automatic classification. In doing so, a number of these features were calculated and used in experiments described later in this thesis. This section offers an introduction to and description of the various spectral and temporal features and details how they were calculated in these experiments.

3.2.1 Temporal Features

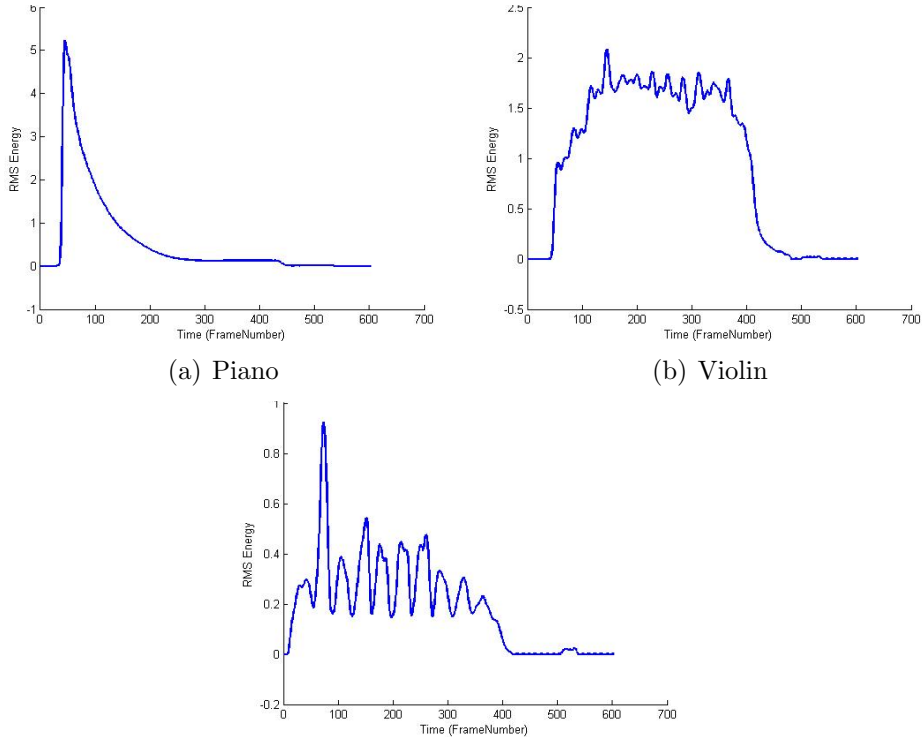
Sound varies in both time and frequency. The temporal features used here describe the way in which aspects of the sound vary with time throughout a given note.

Root Mean Square Temporal Envelope

The Temporal Envelope is a simple measure of the temporal 'shape' of a sound (Kaminskyj and Materka, 1995; McKinney and Breebart, 2003). It is a measure of the energy present throughout the duration of a note. In this thesis, it was calculated from the root mean square (RMS) energy envelope of each sound. The sound was separated into frames with a framesize of 512 samples and a timeshift (the distance between frames) of 256 samples. The RMS energy within each frame was found and combined into an envelope which was smoothed using a 3^{rd} order low pass Butterworth filter. This envelope was calculated over the length of each note to include temporal information on how the energy within the sound changes over time. Thus the envelope incorporates information regarding the attack and release time, which have been shown to be of high importance to instrument classification (Grey, 1977; McAdams et al., 1995). Typically the Temporal Envelope is indicative of the instrument it is played on. A plot of the Temporal Envelope for C4 played on a violin, piano and flute is shown in Figure 3.1

Temporal Residual Envelope

As described in the previous chapter, SMS creates a realistic sound by separating the residual from the stable part of the sound (Serra, 1990). While the Temporal Envelope displays the stable large fluctuation in energy throughout the sound, smaller fluctuations in the envelope indicate the noise or residual part of the sound. The Temporal Residual Envelope was found from the absolute value of the difference between the original unfiltered RMS envelope and the smoothed filtered RMS envelope. This envelope displays the smaller amplitude fluctuations within the energy of the note. Such small variations may be due to the noise from the instrument, player error, vibrato or beats produced by the instrument. A plot of the Temporal Residual Envelope for the piano, violin and flute is shown in Figure 3.2.



(a) Piano (b) Violin

(c) Flute

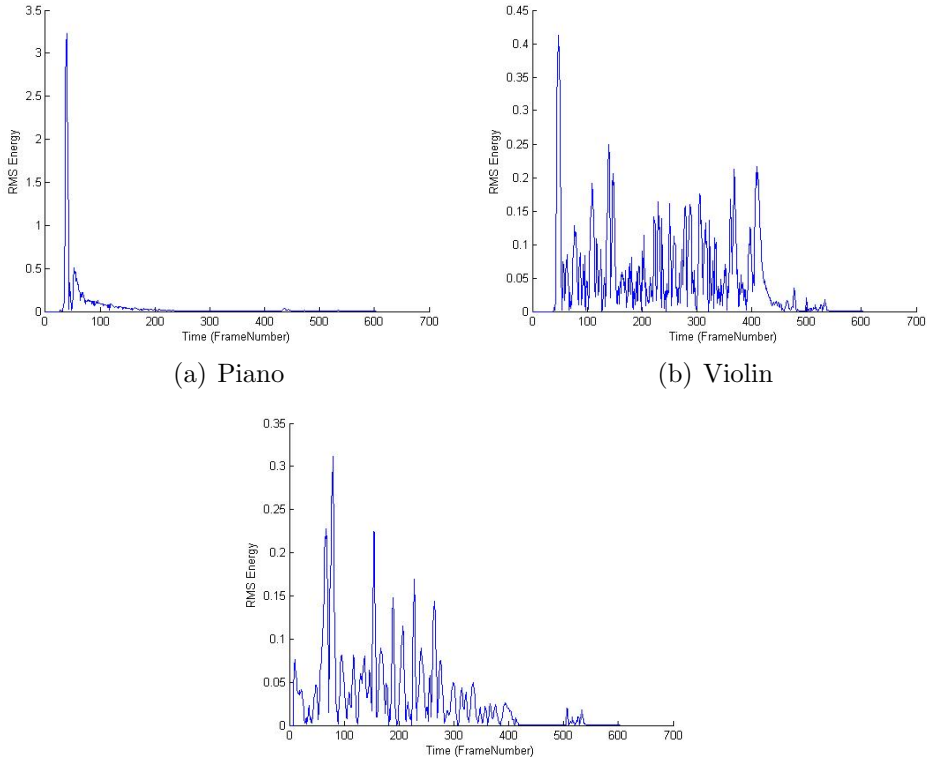
Figure 3.1: *Temporal Envelope of pitch C4 played on the three instruments*

3.2.2 Spectral Features

While temporal features describe changes in the sound over time, spectral features describe changes in the sound in relation to frequency. These features describe aspects of the sound spectra, calculated from the Discrete Fourier Transform (DFT) of each sound sample.

Spectral Envelope

The Spectral Envelope is a measure of the energy within successive frequency frames of the spectrum. It was calculated by applying the DFT to each sample to obtain spectral information from the sound implemented using the `fft` function (MATLAB7, 2006). To calculate the Spectral Envelope, the resultant DFT was separated into (spectral) frames with a framesize of 64 and a timeshift of 32. This resulted in 342 frames. The envelope was found by calculating the RMS energy in each frame. The Spectral Envelope for C4 played on the piano, violin and flute is shown below in Figure 3.3. In this study the Spectral Envelope is described by the average shape of this envelope rather than the actual values present at each frequency; note that the x-axis is marked in *framenumbers* and the amplitude range varies consider-



(a) Piano

(b) Violin

(c) Flute

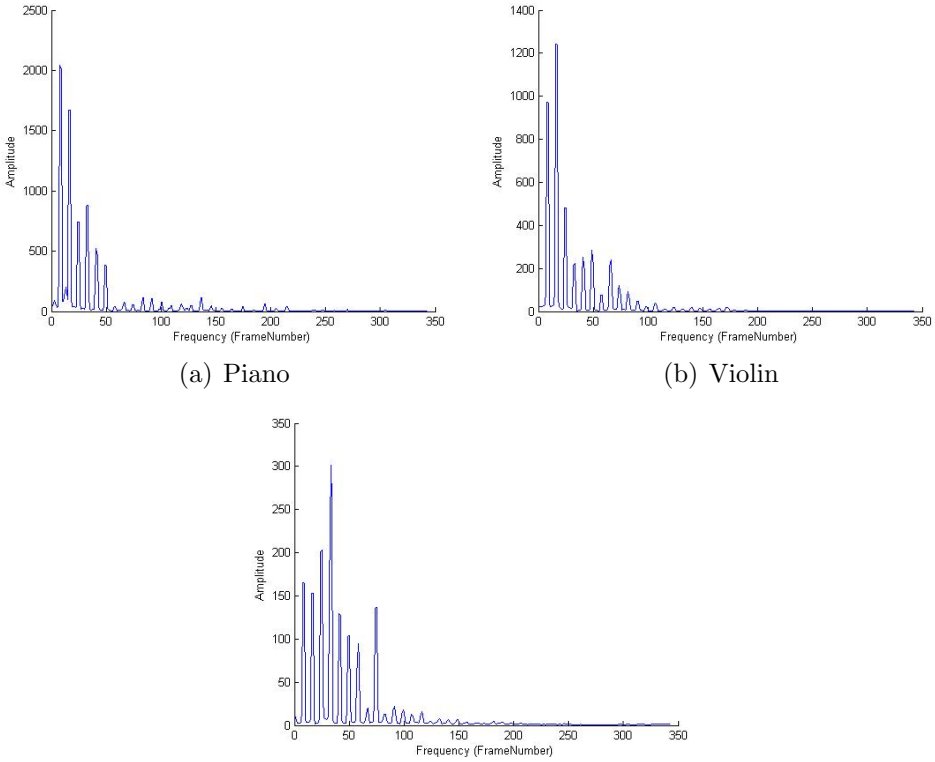
Figure 3.2: *Residual Evelope of pitch C4 played on the three instruments*

ably between the instruments. Although this envelope does not specify the frequencies of the partials present, it describes the relationship between the partials for each instrument.

Centroid Envelope

Perceptually the Centroid has been linked to the perceived quality of brightness (Jensen, 1999). Physically it is a measure of the power distribution of a sound: how much energy is in the upper frequencies. If a sound contains many strong high frequency partials the centroid will be at a high value, producing a bright sound. While a number of previous experiments examined the average centroid, the evolution of the centroid over the duration of each note is considered to be more informative. The sound was separated temporally into frames with a framesize of 4096 and a timeshift of 1024. Each frame was then decomposed into its frequency content using a DFT. The centroid of each frame was then calculated from:

$$\text{Centroid} \quad = \frac{\sum_{f=1}^{f_s/2} f|X_f|}{\sum_{f=1}^{f_s/2} |X_f|} \tag{3.1}$$



(a) Piano

(b) Violin

(c) Flute

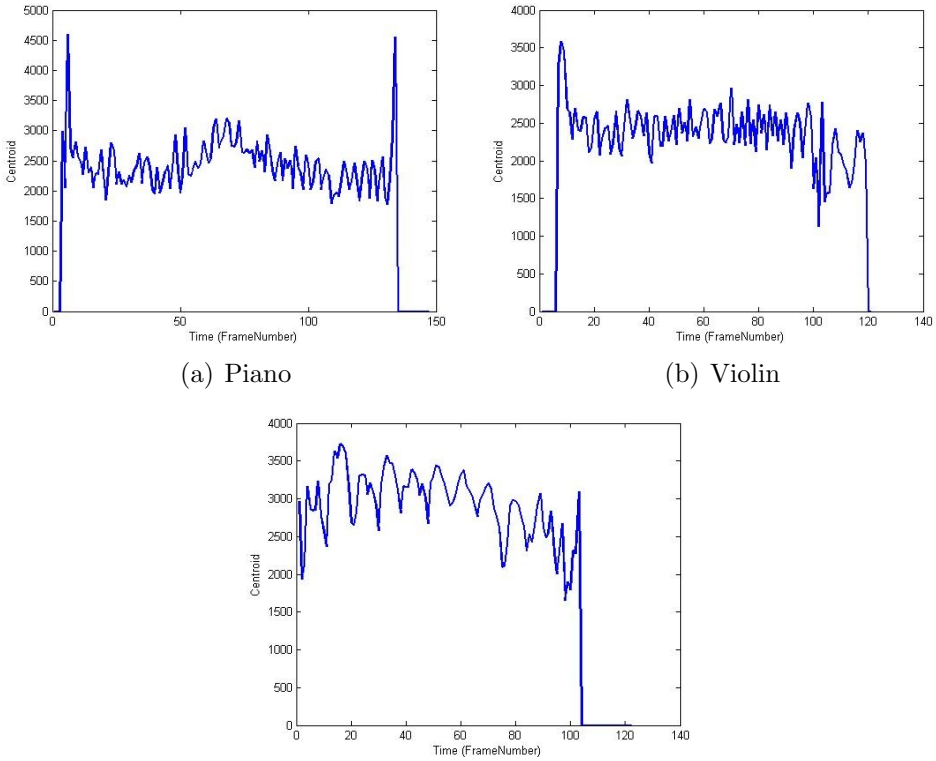
Figure 3.3: *Spectral Envelope of pitch C4 played on the three instruments*

where $|X_f|$ is the strength of the signal (in this frame) at frequency f.

A plot of changes within the Centroid as played on C4 on the violin, piano and flute is shown in Figure 3.4. This envelope describes the changes of a spectral quantity, the Centroid, in a temporal domain. Noticeable peaks in the centroid may be observed at the start and end of the piano sound and at the start of the violin sound. All three sounds are played at dynamic level f making the transients in these instrument strong (produced by a strong strike of the piano key or strong start of the violin bow). These peaks in the centroid envelopes may be explained by the high pitched noisy content of these transients, which is consistent with the high energy at the start of the corresponding residual envelopes in Figure 3.2. Although the flute is also played loudly, it is clear from listening to this particular sample that it does not start with a particularly strong attack.

Inharmonicity

As discussed in the previous chapter, the ear favours intervals with simple integer ratios. This is true also for the internal harmony within a note. For



(a) Piano

(b) Violin

(c) Flute

Figure 3.4: *The Centroid Envelope on pitch C4 played on the three instruments*

a note to be purely harmonious, all of its upper partials would be exact multiples of the fundamental frequency. In reality this is rarely the case. Many instruments, such as the piano, rely on their upper partials being slightly 'detuned' to add warmth and character to their tone. Inharmonicity is a measure of the deviation of upper partials from being perfect integer multiples of the fundamental frequency. As this quality is distinctive to each instrument, Inharmonicity may be used as a quality to identify instruments. An FFT is performed on the data to find the frequency values of the spectral peaks. The Inharmonicity of partial k is calculated from (Beauchamp, 2007):

$$I_k = \frac{f_k}{k f_1} - 1 \tag{3.2}$$

It is evident from this formula that a perfectly harmonic sound will have an Inharmonicity of 0. Sounds which have mostly sharp upper partials (higher than integer multiples) will have a positive value for Inharmonicity and those that contain mostly flat partials (lower than integer multiples) will have a negative value. Thus the Inharmonicity value for any given partial will be in the range -0.5 to 0.5.

One problem with this attribute is that it is dependent on knowing the fundamental frequency of the note. As discussed in the previous chapter, the fundamental frequency of a real instrument sound can be difficult to find. This is particularly true for a timbre rich instrument such as the piano, especially at pitch extremities. This is compensated for somewhat here by considering the first three strong partials as the possible fundamental and comparing the inharmonicity calculated from each possible fundamental. The true fundamental will give a lower value than the other two possibilities, thus the Inharmonicity is calculated as the lowest of the three values. Even with this compensation, however, it must be noted that there may be inaccuracies in the calculation of this feature, particularly for pitches in the extreme ranges of the instruments.

Number of Spectral Peaks

Instruments with a rich timbre contain more partials than those with a more pure tone. The previous chapter discussed how instruments such as the piano produced complex tones with many partials from a number of slightly detuned strings, whereas instruments such as the flute created more simple sounds with fewer partials or overtones. Thus measuring the number of peaks in a spectrum may estimate the number of partials present in a given sound. We take this measurement as an indication of the amount of strong partials within a sound regardless of their frequency value. For this thesis the Number of Peaks is defined as the number of spectral peaks that are at least one-tenth the strength of the strongest spectral peak (whether the strongest peak is the fundamental or not). This was calculated by performing an FFT on the sound, and examining the spectral peaks produced.

Spectral Irregularity

Spectral Irregularity (or Spectral Variance) is a measure of how much the actual spectral envelope varies in comparison to a smoothed version of itself. The calculation of the Spectral Irregularity (SIR) here is based on the log-based formula originally proposed in Krimphoff et al. (1994). This is described in Donnnadieu (2007) as the log of the standard deviation of component amplitudes to a smoothed envelope comprised of a running mean of three adjacent harmonics. Thus a smoothed envelope $\hat{A}_k$ was calculated at each partial k by averaging it with its two surrounding partials:

$$\hat{A}_k = (A_{k-1} + A_k + A_{k+1})/3 \tag{3.3}$$

Then the Spectral Irregularity was calculated from the log of the standard deviation from each measured amplitude point to this smoothed envelope:

$$SIR = log(std(\hat{A}_k - A_k)) \quad k = 1 : \text{no. of partials} \tag{3.4}$$

A high value SIR means that the amount by which the real envelope differs from the averaged envelope varies widely across the partials, giving a jagged spectrum.

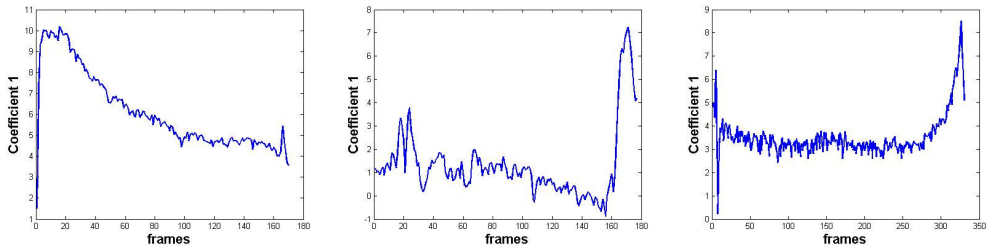
Mel-frequency Cepstral Coefficients

The Mel-frequency Cepstral Coefficients (MFCCs) are a set of coefficients that can represent the spectral quality within a sound according to a scale based on human hearing. MFCCs have been widely used in speech recognition for some time (O'Shaughnessy, 1987). MFCCs were more recently examined and found to be useful in the area of music analysis (Logan, 2000). Obtaining the MFCCs involves analysing and processing the sound according to a number of steps (Logan, 2000). The first step is to divide the signal into frames. This is carried out by applying a windowing function (typically Hamming) to the signal at fixed intervals. A cepstral feature vector is calculated for each frame. The next step is to obtain the amplitude spectrum of each frame by applying the DFT. Only the log of this amplitude spectrum is retained as the phase is considered less important. The signal is then converted to the perceptually derived *mel-scale*. The mel-scale, proposed in Stevens et al. (1937) is a perceptual scale of pitches judged by listeners to be equal in distance to one another. The mel-scale has been popular for speech recognition for many years as it is considered to approximate critical-band spacing. A frequency f is converted to mels m according to:

$$m = 1127.01048 \ln(1 + f/700) \tag{3.5}$$

The components of these mel-spectral vectors within each frame are highly correlated. To reduce the number of parameters, these vectors are decorrelated using the Karhunen-Loeve (KL) transform or equivalently Principal Component Analysis (see Chapter 5). Experiments in speech analysis have shown that the KL transform can be approximated with the Discrete Cosine Transform (DCT) (Logan, 2000; Merhav and Lee, 1993). Thus the final step in calculating the MFCCs is to apply the DCT to the components of the mel-scale vectors. After this algorithm has been applied the result is a matrix of

values for each sample sound that is the number of coefficients by the number of frames in size. This was implemented in MATLAB7 (2006) using the `melcepst` function from the *Voicebox Toolbox* (Brookes, 2009). These calculated coefficients change from frame to frame. The changes in these values can be plotted as an envelope across the sound. Such envelopes are distinctive to the instrument as illustrated below. Figure 3.5 shows the changes in the first MFCC for C5 on a piano, violin and flute.



(a) Piano (b) Violin (c) Flute

Figure 3.5: *Changes in MFCC1 on pitch C5 played on the three instruments*

Although the MFFCs have been widely used in speech and more recently music analysis, there remains some debate as to their suitability and accuracy in this type of analysis. After proposing the original model, Stevens discovered a methodological flaw in that subjects' results depended on whether notes were played in ascending or descending order (Stevens and Volkmann, 1940). That the mel-scale only approximates (and arguably not accurately) critical-band frequency spacing has been proposed a number of times over several decades by Greenwood (Greenwood, 1961b, 1997). Such discussions indicate that a more accurate perceptual scale is necessary for a set of spectral coefficients such as the MFCCs to be truly accurate. Even though such arguments have persisted since the introduction of the mel-scale, MFCCs remain a prominent feature in speech (and more recently music) analysis.

3.2.3 MIRToolbox Features

The features described above were calculated using Matlab. A number of toolboxes have been developed in recent years to calculate auditory features from a given sound, such as MUSCLE (Information Society, 2009), IPEM (Leman et al., 2005) and CLAM (Amatriain, 2007). A widely used Matlab toolbox for calculating such features is the MIRToolbox (Lartillot and

Toiviainen, 2007). This toolbox is designed on a modular framework to offer an overview of computational approaches in the area of music information retrieval. In the experiments described in Chapters 5 and 6 the following features were calculated using this toolbox.

Zero-crossing Rate

A sound wave oscillates between positive and negative values. A simple measure of noisiness of a sound signal would be a measure of how often it crosses the X-axis. The Zero-crossing Rate was here implemented with the `mirzerocross` function as the number of sign-changes in the signal per second.

Spectral Rolloff

Spectral Roll-off is a measure of energy distribution within a sound. Specifically it measures the frequency below which a certain percentage of the energy in the sound is contained (Tzanetakis and Cook, 2002). It was calculated using the `mirrolloff` function as the frequency below which 85% of the sound energy occurs.

Brightness

Although Brightness has been associated with the centroid, it is considered here as a different measure of the high frequency content within a sound. It was calculated using the `mirbrightness` function as the proportion (denoted between 0 and 1) of the energy within the sound above 1500Hz.

Regularity

The spectral Regularity is calculated in contrast to the spectral Irregularity as defined above. The MIRToolbox calculates Regularity using `mirregularity` according to Jensen (1999) as the sum of the square of the difference in amplitude between adjoining partials.

Onsets and Attack

As described above, the envelope of a sound describes the changes in energy throughout the sound. The MIRToolbox offers a number of functions that can be calculated from these energy changes. Onsets are successive energy bursts within a sound. While they are generally used in determining the

tempo of a piece of music, they may also be of interest for individual sounds in determining the presence of temporal bursts of energy within a note. We propose that such patterns of energy bursts may be indicative of a specific instrument. For this feature, an energy envelope object was first calculated with the `mirenvelope` function. The `mironsets` function used this envelope object to calculate the Onsets. Both the Number of Onsets within a sound and the Onset Distance (average distance between the onsets) were calculated as features in this thesis.

The Attack Time is a measure of how long the signal takes to reach its maximum energy after the note starts. It indicates the perceived sharpness or abruptness of the beginning of a note. It is calculated in the MIRToolbox using the function `mirattacktime` again from the energy envelope, giving a value for each of the onsets of energy. The Attack Time, in seconds, of the initial onset was included in the data. The slope of the attack is another measure of how quickly the signal reaches its maximum value, thus indicating the abruptness of the sound. As with the Attack Time, the Attack Slope is calculated from the envelope value with the function `mirattackslope`.

Central Moments

It is clear that the spectral content of a sound is highly relevant to timbre. Statistical values from the spectrum may be calculated to indicate how the spectrum behaves across all frequency values. These measures, calculated in relation to the centre or mean of the spectrum are known as the central moments (Dougherty, 1998). The MIRToolbox calculates a number of features related to the central moments as discussed below (Lartillot, 2008).

The first moment represents the mean of the data. The mean of the spectrum is here calculated as the geometric centre or Centroid of the distribution of the spectrum. Unlike the Centroid calculated above, the MIRToolbox calculates the Centroid with the function `mircentroid` as a measure of central tendency. The mean of N discrete values may be calculated as (Quinn, 2009):

$$\bar{x} = \frac{1}{N} \sum_{j=1}^{N} x_j \tag{3.6}$$

The second central moment is the variance or spread of the spectrum,

which may be calculated as (Quinn, 2009):

$$\delta(x) = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(x_j - \bar{x})^2} \tag{3.7}$$

This is a measure of the standard deviation of the spectrum from its mean value. The Spread of the spectrum is calculated as the standard deviation of the sound by the function `mirspread` (Koch, 2000).

The third central moment, called the Skewness, is a measure of the symmetry of the distribution of the spectrum around the mean. The spectrum may be positively skewed or negatively skewed. A positively skewed spectrum has a long tail to the right of the spectrum, owing to a few values that are much larger than the mean. Similarly, a negatively skewed distribution has a long tail to the left of the spectrum. The Skewness is calculated from:

$$Skew(x) = \mu_3 = \frac{1}{N}\sum_{j=1}^{N}(\frac{x_j - \bar{x}}{\delta})^3 \tag{3.8}$$

The function `mirskewness` returns a value for the *coefficient of skewness*. This is calculated as the ratio of the skewness to the standard deviation raised to the third power,

$$Coefficient\ of\ Skewness = \frac{\mu_3}{\delta^3} \tag{3.9}$$

as this has more convenient units than the skewness (Koch, 2000).

Kurtosis is the fourth standardised moment of the spectrum is calculated as (Quinn, 2009):

$$Kurt(x) = \frac{1}{N}\sum_{j=1}^{N}(\frac{x_j - \bar{x}}{\delta})^4 - 3 \tag{3.10}$$

The subtraction of 3 in the equation is not always used, but is included to make the normal distribution of the Kurtosis equal to zero. Kurtosis is a measure of how short and wide or tall and thin a spectrum is. It is calculated using the `mirkurtosis` function.

3.3 Conclusion

The aim of the work presented in this thesis is to examine the best way in which to describe sounds in order to distinguish between musical instruments. Accurately describing a sound is largely dependent on being able to measure

or describe the timbre of that sound. All other aspects of a sound (duration, intensity, pitch, placement) are relatively simple to measure in comparison to this aspect of timbre. Although the term *timbre* is regularly used in the literature on sound, it remains an ill-defined concept that is inherently difficult to measure.

This chapter focussed on the definition and measurement of timbre, as used in the past and as proposed for this thesis. The lack of definition of timbre in Section 3.1 was discussed together with the attempts to form an accurate and consistent verbal description of timbre. It is evident from studies undertaken that due to differences in candidate's perception, understanding and communication of what they hear, a consistent verbal description of timbre may not be possible to obtain. More objective experiments were discussed that examined the multi-demensionality of timbre using MDS. Such experiments helped to gain some insight into the dimensions of timbre and proposed perceptual meanings for the dimensions found.

Section 3.2 introduced a number of temporal and spectral measures that have been used to describe timbre. These measures are used throughout this thesis to quantify timbre for automatic instrument classification. A selection of these features are used in instrument classification studies in Chapter 5 and in optimisation experiments in Chapter 6 and Chapter 7. The selection of features listed here is not an exhaustive list of all features used in previous audio recognition experiments. The following chapter discusses a number of experiments that incorporated alternative features or different aspects or variations on the features listed above. As with all classification experiments of this kind, there is no guarantee that the selection of features used is ideal or sufficient for accurate instrument recognition. We propose in the experiments in this thesis, not to use as many features as possible, but to determine which are the best timbral features or descriptors to use for identifying a musical instrument. Determining which are the most effective features for instrument classification may offer some insight into the nature of the timbre. In identifying the most useful timbral descriptors, we hope to gain further understanding of what makes a sound, in particular an acoustic instrument sound, so distinctive to the human ear.

Chapter 4

Sound Identification and EC

The experiments discussed in later chapters use a number of natural computing techniques in describing and identifying musical instrument sounds. Chapter 5 describes experiments on the automatic classification of musical instruments, whereas Chapters 6 and 7 investigate timbre by optimising the features used in such experiments with evolutionary methods. This chapter describes relevant work that has been undertaken in the past in relation to these experiments. Section 4.1 introduces the problem of automatic instrument recognition. A selection of the most important studies on instrument recognition, incorporating various combinations of features and classification methods, are described. The features used in these previous experiments are discussed and summarised.

There has been little justification for the inclusion of any specific features in these previous experiments. Ideally there would be a method of selecting which features would be most important or relevant for an automatic instrument classifier. The process of choosing a set of optimal features from an initial set of candidates is referred to as *feature selection*. Some relevant work and reviews on this idea of feature selection are discussed in Section 4.2. It was seen in the previous chapter that timbre is a multi-dimensional quality that has proven to be difficult to quantify or even describe. It is proposed that exploring the optimal selection of features for instrument classification will indicate which features are necessary for accurate classification, thus providing an insight into the nature of timbre. Although there are many methods for optimising features, the current thesis uses natural computing methods inspired by genetic evolution. This field of Evolutionary Computation (EC) is introduced in Section 4.3. Although not extensively used so far for sound classification, EC methods have been applied to other areas of sound and music production such as melody composition and sound synthesis as described

in this section. Finally, a summary of the chapter is given in Section 4.5.

4.1 Automatic Instrument Classification

Previous chapters have examined human hearing and the perception of timbre. It is this human ability to distinguish between timbres that allows us to identify musical instruments. Thus creating an artificial method of identifying instruments must be based on describing musical timbre. Systems such as the MPEG-7 Standard (Chang et al., 2001) have been developed in recent years for sound content description. While such a system is useful for search, organisation and management of audio and multimedia data it is not ideal for individual instrument recognition. Classifiers based on timbral features can be used to automatically identify a specific instrument from a single note or from a short solo passage. A number of such automatic instrument classifiers have been created using Artificial Intelligence (AI) or Machine Learning (ML) techniques developed from observing human traits such as neural processing or natural evolution.

The recent interest in identifying musical instruments has concentrated on two distinct objectives, namely:

- to identify musical instrument sounds from monophonic sources
- to identify musical instruments from polyphonic sources, when one instrument is playing within a group

Much research has been undertaken on the first task above. It may be seen as a simpler task then analysing polyphonic music and as such techniques and algorithms for solving this problem have been developed, possibly with the view that they may be extended to undertake the second task. It is becoming clearer, however (Herrera-Boyer et al., 2003), that the two may not be as interchangeable as once thought. Segmentation of instruments from a complex auditory stream is a problem that is far from being solved. And although many studies have been undertaken on it, the first task of identifying monophonic sounds still merits further study. This task in itself can further be broken down into two categories:

- identification of solo passages containing numerous notes
- identification of isolated instrument sounds

A musical passage contains many auditory cues not present in single notes. Effects from a particular playing style or the transition between notes can all contribute to the recognition of a particular instrument. The analysis of single notes however, relies purely on the timbre of that specific note. The classifier being developed here is intended to be able to determine the quality of a synthesised sound, this sound being synthesised as a single note. Hence, although previous classification studies on both categories are discussed, the second scenario listed above of identifying isolated instrument sounds is the main focus of investigation throughout this thesis.

Correct classification of an instrument is dependent on the following distinct processes:

- instrument selection
- feature selection
- mathematical representation of the features into a data set
- reduction of this data set
- classification method of the reduced data

Selecting the number — and indeed which instruments to classify is the first step in designing a classifier. By selecting the instruments for the classifier to choose between one defines the ultimate objective of the classifier. Feature selection here refers to the high level attributes that are known to be indicative of the sound from a psycho-acoustical perspective. Most of the studies undertaken on this topic have included a large number of features, although few gave reason for their selection.

The mathematical representation of this data refers to the way it is collected or portrayed. For example when examining the amplitude envelope, the root mean square of the energy across the sound is often calculated. Other statistical measurements such as the mean or standard deviation are also calculated to give measurements of various features. The reduction of data is not always implemented. It is used in certain studies when the data set is particularly large. Statistical reduction techniques such as principal component analysis (PCA) or more recently state predictive models such as hidden markov models (HMM) have been employed for this process. Finally the merits of many different classification techniques such as artificial neural networks, bayesian classifiers or binary decision trees have been compared and contrasted in numerous studies.

This section describes a number of experiments that deal with the processes defined above in instrument classification. A number of empirical studies that give an overview of the experiments undertaken so far have emerged over the last decade, (Herrera et al., 2000, 2006; Herrera-Boyer et al., 2003; Simmermacher et al., 2006). As discussed in these studies (and as pointed out below), the issue with many of these specific experiments undertaken is that while they may compare differences in one of the steps defined above, they do not consider the merits of each method when combined with a change in another step. Most notably, many studies picked a data set calculated from defined features from sounds and then focussed on comparing different classification techniques on this dataset. This is not surprising due to the interest in developing and validating such new techniques.

4.1.1 Instrument Classification Studies

Interest in musical instrument classification as described above gained substantial momentum from the mid-90s onwards. A number of the early studies concentrated on a small collection of audio features for classifying instruments. Kaminskyj and Materka (1995) looked at the root mean square (RMS) envelope across a one octave range (C4-C5) of four instruments, namely the piano, guitar, marimba and accordian. They then reduced this data using PCA and compared classification results using a multi-layered perceptron (MLP) and a nearest neighbour classifier (NNC). They obtained quite accurate results with the NNC performing slightly better than the MLP. Cemgil and Gurgen (1997) performed a similar experiment by calculating harmonic envelopes derived from the short-time fourier transform (STFT) of sampled sounds to classify these sounds as belonging to a specific timbral family. They then compared the classification as performed by a MLP, a time-delay neural network (TDNN) and a hybrid self-organising map radial basis function network (SOM-RBF). Of these they found that the TDNN model performed slightly better than the other two models at this specific task. Cosi et al. (1994) modelled sample sounds essentially by using the joint Synchrony/Mean-rate (S/MR) model of auditory speech processing developed in Seneff (1990). They used this method to discern between 12 musical instruments using a self-organising map (SOM) to define a hypothetical timbre space. Fragoulis et al. (1999) examined timbre recognition of five musical instruments using an ARTMAP network. This ARTMAP is a neural network comprised of two unsupervised Adaptive Resonance Theory (ART) networks, one of which (the Master) controls the mapping of the input to the other (the

Slave) (Carpenter et al., 1991). They analysed the instruments according to the slope of the attack, the degree of synchrony of the attack transients and the amount of energy in the higher frequencies. The training and test size in this experiment were small (60 and 12 respectively) although they did report good results.

Wold et al. (1996) created a classifier at Muscle Fish for classifying a large range of sounds. This system was developed for analysing everyday sounds, speech sounds and sound effects, although some musical sounds were included. They extracted features such as loudness, pitch, brightness, bandwidth, and harmonicity and arranged vectors for sounds according to the mean, variance and auto-correlation of these features. They reported good classification results on sounds such as laughter, female speech and touch-tones and suggested a number of applications for such a device. De Poli and Prandoni (1997) explored the possibility of creating a timbre space that would map acoustic similarity to the concept of distance. In doing this they looked to automatic speech recognition and in particular Mel Frequency Cepstral Coefficients (MFCCs) as a feature for instrument recognition. As noted in previous chapters MFCCs had widely been applied in speech recognition, but until this point had rarely been used in musical sound analysis. They explored the use of MFCCs in creating a timbre space and use PCA and SOMs in judging their effectiveness. They found these coefficients to be well suited to the representation of sounds and compared favourably with previous work on developing a timbre space.

MFCCs were later employed in numerous experiments by Antti Eronen and Anssi Klapuri. In Eronen and Klapuri (2000), a large number of features, including MFCCs and temporal features, were used to identify musical instruments. This work looked at a wider set of instruments (30 in total) than the previous studies and aimed to create a classifier general enough to deal with these different instruments by including more types of features. Along with the cepstral coefficients they calculated numerous temporal measurements derived from features of the sound. The rise-time, decay-time, crest factor and strength and frequency of amplitude modulation were calculated from the RMS energy envelope. The spectral centroid was calculated as both an absolute and a normalised value. The onset and steady state intensities were included along with a measure of the fundamental frequency. The mean and standard deviations of a number of these measurements were incorporated along with error and variation measures. These features were used to form a tree hierarchy with a Gaussian or k-NN classifier at each node. They obtained

favourable results in comparison to previous studies, thus demonstrating the benefits of combining numerous different features in such instrument classification tasks. Eronen (2001) furthered this work by introducing new cepstral features based on either linear prediction or filterbank analysis (or warped linear prediction). These were combined with the temporal and spectral features described in the previous experiment. The usefulness of these features were compared using a k-NN classifier. It was found that the MFCCs gave the best instrument recognition accuracy of all the features on their own, but that the best overall accuracy was determined by combining these features describing the sounds brightness, modulations, excitation, synchronicity and fundamental frequency. A further study (Eronen, 2003), attempted to classify instrument and drums sounds using just MFCCs. These coefficients were reduced using independent component analysis (ICA) and classified with a hidden markov model (HMM). They found applying an ICA-based transform to the set of features to be beneficial to the systems performance.

In his doctoral thesis, Martin (1999) used a large set of features in developing an automatic instrument identifier. This feature set included: the spectral centroid, average relative spectrum — overall and by partial number, high-frequency rolloff rate and cutoff frequency, spectral irregularity and number of 'zeros', relative energy in odd and even partials, pitch range, tremelo, centroid modulation, individual harmonic amplitude modulation, relative onset time by partial frequency and 'rise likelihood' by frequency and post-onset time. He also considered, but did not implement features based on pitch 'wobble', pitch jitter, number of 'blips' in the attack, explicit onset skew, rise rates, inharmonicity, note-to-note transitions, explicit identifications of resonance and cognitive cues. These features were calculated and grouped to a number of different classifiers to form a hierarchy. In a similar study Martin and Kim (1998) extracted 31 features based on those listed above. These were classified with both Fisher multiple discriminant analysis and a k-NN with good success.

Fujinaga conducted a number of studies to try to determine the most useful features to include in such instrument identification tasks (Fraser and Fujinaga, 1999; Fujinaga, 1998; Fujinaga et al., 1998). In these experiments a number of features were used in instrument identification using k-NN classifiers. Some of these features consisted of a number of musical moments used to describe numeric quantities related to the spectral shape of the sound (Dubnov et al., 1997). Those included were the mass or integral of the curve, centroid, standard deviation, skewness, kurtosis, higher order central mo-

ments, fundamental frequency and amplitudes of the harmonic partials. This generated hundreds of features. Rather than using a dimensional reduction technique, they used a genetic algorithm (GA) to determine which were the most important features to use (see next section). Once the features were selected, their optimum weights were selected by a similar method. The results showed that the optimum subset of features to use for accurate instrument identification were the fundamental, integral of the spectrum, centroid, standard deviation, skewness and the first two harmonics.

Brown (1999) used cepstral coefficients based on the constant Q transform to analyse solo passages played on the oboe and saxophone. A k-means algorithm was used to form clusters from the obtained features. Test samples of individual notes were then assigned to a cluster according to a Bayes decision rule with very good results that were comparable to human judgement. This work was extended in Brown et al. (2001) to include more features to identify more instruments, namely the oboe, saxophone, clarinet and flute. The features examined included the cepstral coefficients, constant-Q coefficients, spectral centroid, average spectral energy, autocorrelation coefficients and a number of time moments. Marques and Moreno (1999) again created a classifier for identifying instruments from solo pieces of music. The instruments classified included the bagpipes, clarinet, flute, harpsichord, organ, piano, trombone and violin. As in Brown (1999) they looked to speech analysis for feature representations and based their features on linear prediction coefficients, FFT based cepstral coefficients and FFT based mel-cepstral coefficients. They classified instrument notes using Gaussian Mixture Models (GMM) and Support Vector Machines (SVM). Their best result (30% error rate) was achieved using 16 mel-cepstral features with a SVM.

Agostini et al. (2001) and Agostini et al. (2003) evaluated the performance of various different classifiers in identifying instruments based on a number of spectral characteristics. Using a dataset of 1007 tones from 27 instruments they calculated the mean and standard deviation of the zero-crossing rate, spectral centroid, band-width, percentage of harmonic energy in first 4 partials, inharmonicity and harmonic energy skewness. They classified these using quadratic discriminant analysis (QDA), canonical discriminant analysis (CDA), nearest neighbour and support vector machines (SVM). They found that QDA performed the best with comparable results from the SVM with poorer performance from the other two methods. They also reported that they found that the strings to be the most mis-classified family of instruments with the best results obtained with brass and woodwinds.

Kitahara et al. (2003) created a classifier examining the pitch dependency of timbre in musical instruments. They represented the tone features in the feature space using an F0-dependent multivariate normal distribution with two parameters: the F0-dependent mean function and F0-normalised covariance. They implemented many of the spectral, temporal and modulation features described in previous experiments. They also included some novel non-harmonic component features to give a total of 129 different features. They reduced their data set to 18 dimensions using PCA and linear discriminant analysis (LDA) before using a Bayes decision rule to classify the instruments. They found that using this F0-dependent multivariate normal distribution improved results compared to those achieved through the usual multivariate normal distribution. Non-suprisingly, this was most evident for instruments with a wide range of pitch. Chetry and Sandler (2006) examined a linear predictive model for instrument identification. They included information about the evolution of the signal over time by appending the delta (speed) and delta-delta (acceleration) coefficients from the features from LPC analysis. Nielson et al. (2007) examined two models for describing sounds. The first assumed a constant spectrum envelope — calculated from the MFCCs, and the second assumed constant relation among the amplitude of the harmonics (harmonic representation). By comparing results from a MLP and kernel orthonormalised partial least squares (KOPLS) followed by a single layered perceptron, they determined the model based on the constant spectrum envelope to perform better than that based on the harmonic representation.

Although the current thesis examines instruments from single note sounds, we consider briefly a small number of studies that use similar methods of instrument classification, but on solo passages of music. Livshin and Rodet (2004) introduced an instrument classifier for solo music passages to discern between a bassoon, clarinet, flute, guitar, piano, cello and violin. They used features as calculated from the Cuidado project developed at IRCAM (Vinet et al., 2002). They reduced these features with LDA before classifying with a k-NN. They reduced their feature set using gradual descriptor elimination (GDE) allowing them to perform solo recognition in real-time. They found these results to be comparable with those of the complete feature set. Ihara et al. (2007) used LPC along with line spectral frequencies (LSFs) to distinguish between solos from 8 instruments collected from commercial CDs. A combination of PCA and LDA was applied to reduce the dimensionality of the data after which SVM were used as a classifier. Their results compared favourably with similar contemporary studies. Essid et al. (2006) and Essid

et al. (2004) conducted studies in choosing features and processes for classifying musical instruments from solo musical phrases. In these studies a large number of features including previously used temporal, spectral, cepstral and modulation features, along with new features based on octave band signal intensities were initially examined for classifying the phrases. These features were reduced using GAs and inertia ratio maximisation using feature space projection. These reduced feature sets were then used in classifications incorporating a gaussian mixture model (GMM), pairwise coupling classification and SVMs. Their results found their new features on octave band signal intensities to be very useful as was using pairwise class feature selection. Furthermore they found the SVM with a radial basis kernel function gave better more accurate classification than the GMM.

4.1.2 Summary of Experiments

It is clear from the list of experiments described above that many different methods and combination of methods have been exploited in the attempt to create a robust musical instrument classifier. Nevertheless, no one method or technique has emerged as the definite best choice for this task. It is proposed in Herrera et al. (2000) that such studies may be inherently *biased* in some way — such as in instrument selection or sample selection. The sound samples used above include the MUMS (Opolko and Wapnick, 1987), RWC samples (Goto, 2004), University of Iowa samples (Fritts, 1997), SHARC (Sandell, 1994) along with many recordings made specifically for the experiment at hand. The number of instruments classified has ranged from 2 to over 30 (with a wide range of instances of each) and the number of features extracted has ranged from 1 to hundreds. Combining these differences in ranges with the various data reduction and classification techniques result in a huge variety of experimental set-ups — all with the same goal of classifying musical instruments. A summary of the main points of each of the studies above, including the number of instruments/classes used, the number of samples used for training, the method(s) of classification and the main success rates is shown in Table 4.1.

This table displays the variety in set-up across the range of studies. Despite these differences, each individual experiment reports encouraging results. Thus selecting the best combination to proceed with is somewhat difficult. The final accuracy of any classifier is clearly dependent on the features chosen, the number and type of instruments included and the number of training samples for each instrument. There may be some discrepancies in

Table 4.1: *Summary of Previous Studies*

Study	**Instruments**	**Samples**	**Classifier**	**Accuracy (%)**
Kaminskyj and Materka (1995)	4	240	PCA & MLP	97.7
			PCA & kNN	98.1
Cemgil and Gurgen (1997)	10	480	MLP	97.5
			TDNN	100
			SOM-RBF	94.17
Cosi et al. (1994)	12	12	SOM	
Fragoulis et al. (1999)	5	60	ARTMAP	83
De Poli and Prandoni (1997)	40	40	PCA & SOM	
Eronen and Klapuri (2000)	30	1498	k-NN	80.6
Eronen (2001)	29	5286	k-NN	77
Eronen (2003)	27	5895	HMM	68
Martin and Kim (1998)	14	1023	k-NN	67.5
			Fisher	71.6
Martin (1999)	27	1500+	Hierarchy	75.9
Fujinaga (1998)	23	1338	GA & k-NN	50
Fujinaga et al. (1998)	23	1338	GA & k-NN	46
Fraser and Fujinaga (1999)	23	1338	GA & k-NN	64
Brown (1999)	2	1 min.	k-means	84.1
Brown et al. (2001)	4	1 min.	k-means	79.84
Marques and Moreno (1999)	8	1024	GMM	63
			SVM	70
Agostini et al. (2003)	27	1007	QDA	92.2
			SVM	88.7
			k-NN	90.4
			CDA	90
Kitahara et al. (2003)	19	6247	Bayes	79.73
Chetry and Sandler (2006)	6		SVM	86.3
Nielson et al. (2007)	17		MLP	74.7
			KOPLS	75.9
Livshin and Rodet (2004)	7	108	k-NN	88.13
Ihara et al. (2007)	8	3592	PCA & LDA	91
Essid et al. (2006)	10	15 mins.	SVM	87

the way in which the accuracy of some experiments in Table 4.1 are reported; a number of the results quoted are according to correct identification within instrument family, rather than as a specific instrument.

As discussed in the previous chapter, this thesis examines just five instruments, one from each instrument family, but uses over 3000 individual samples to ensure all possible instances of these instruments are considered. The main focus of this thesis is concentrated on the initial steps of the process — specifically on feature selection. Without optimum feature selection it is unlikely that accurate classification could ever be achieved, regardless of the classifier used. Hence the following section gives an overview of all the features used in the experiments described above.

4.1.3 Summary of Features

The experiments listed categorise features in a variety of ways. Some merely refer to features individually, others classify them as temporal and spectral, whereas others categorise them as a variety of types of features including temporal, spectral, cepstral, energy, modulation, harmonic or perceptual. Attempting to discern all the different features used can be difficult as the definition of a given feature may differ between experiments. For example, the attribute of the centroid is widely quoted as being used in many studies above, however the manner in which it is used is by no means consistant. The mean, maximum and standard deviation of the normalised and un-normalised spectral centroid have been used in Eronen and Klapuri (2000). Martin and Kim (1998) quote the average spectral centroid, spectral centroid delta ratio[1], the variance of spectral centroid, spectral centroid variance delta ratio and the spectral centroid modulation, modulation strength or modulation heuristic strength[2], in both normalised and un-normalised forms. Yet many other studies for example Agostini et al. (2003) just include the 'spectral centroid'. Furthermore certain attributes or features may be named differently among different experiments, for example the duration of the attack is referred to as rise time (Eronen and Klapuri, 2000) or onset duration (Martin and Kim, 1998). The details of the calculation of such features are rarely included in the write-ups and so methods of numerical measurements most likely vary between experiments.

[1] the delta ratio was here defined as the ratio of the feature during the transition from onset to steady state to the feature value after the transition period

[2] the heuristic strength was here defined as the peak height of the feature divided by the average value surrounding the peak

Bearing these complications in mind a list of the feature measurements used throughout these experiments has been compiled. They have been loosely divided into two categories — temporal features and spectral features, though these are not objective or physical classifications. It is not always immediately obvious as to which category a feature should belong — many are technically frequency-dependent aspects of a sound examined in a temporal manner. Where such ambiguities arise the most appropriate classification was chosen. A summary of the features used is shown below in Table 4.2.

Table 4.2: *Summary of Temporal and Spectral Features used in Previous Instrument Identification Experiments. EOF refers to Error of Fit, SS refers to Steady State and int refers to Intensity*

Temporal Features	**Spectral Features**
RMS energy envelope	Centroid (all measurements)
Attack time	Pitch (or fund freq)
Slope of att	Pitch range
Slope of line fitted to RMS after att	Std dev of fund freq
Degree of synchrony of att transients	Average pitch
Slope of att. harmonic skew	Average pitch delta ratio
Intercept of att. harmonic skew	Pitch Variance
Relative attack time by partial freq	Pitch Variance delta ratio
Rise likelihood by freq, post att time	Odd/Even harmonic ratio
Crest factor	Relative energy in odd/even partials
Decay time	Average relative spectrum
Autocorrelation coefficients	Avrg relative spectrum by partial no.
Strength of amplitude modulation	Integral of spectrum
Frequency of AM	Std dev of spectrum
Heuristic strength of AM	Skewness
Variations of intensity at each band	Kurtosis
EOF between each SS int and mean SS int	Bandwidth
EOF between each att int and mean att int	Inharmonicity
Vibrato frequency	% Energy in first partials
Vibrato amplitude	Zero-crossing rate
Vibrato heuristic strength	Spectral irregularity
Tremelo frequency	High freq roll-off rate and cut-off freq
Tremelo amplitude	Spectral spread
Tremelo heuristic strength	Relative power of fund component
Moments of the time wave	Harmonic energy
Gradient of line approximating power env	Noise part energy
Avrg differential of power env during att	Sharpness
	Spectral Width
	Spectral Asymmetry
	Spectral Flatness
	MFCCs
	Avrg cepstral coeff during att
	Avrg cepstral coeff after att
	Linear prediction cepstral coeff
	Delta cepstral coeff
	Constant Q coeff

4.1.4 Discussion

The list of features summarised in Table 4.2 is not necessarily exhaustive. A number of studies, such as Kitahara et al. (2003) do not explicitly list all of the features included. As mentioned above it can be difficult in some cases to determine exactly what is being calculated for many of the features given. Very few studies justified their choice of features, making it difficult to say which of these features are necessary or most suitable for timbre description. Conversely, there is no evidence to say that even with including all of the above, one may be able to completely describe a timbre. Including too many features in classification problems ultimately leads to problems caused by the *Curse of Dimensionality* (Bellman, 2003). This refers to over-complication of the system by adding too many variables or dimensions, some of which may not be necessary. In other words, simply adding more features and hoping for the best could actually be exacerbating the problem of classification, rather than alleviating it. Although using standard dimensional reduction techniques such as PCA may help reduce dimensionality, these techniques ultimately alter the variance within the selection of features, often in an unintuitive way. A more decisive manner in which to select which features to use in a classifier is needed.

4.2 Automatic Feature Selection

There have been a number of studies on the most effective manner to optimise feature selection for classification tasks. In 1977 Cover and Campenhout determined that for a feature set of size n, to determine the best m features to use, all possible subset feature combinations of size m must be considered (Cover and Van Campenhout, 1977). From a practical perspective however, an exhaustive trial of each subset of features is rarely feasible. In reality the best subset of features ideally can produce suboptimal but efficient and robust methods of classification (Siedlecki and Sklansky, 1993b). Thus a number of studies have focussed on optimising the most appropriate set of features for their problem. A recent review of such methods (Guyon and Elisseff, 2003) examined variable and feature selection for large scale classifiers or predictors. They defined three distinct objectives to feature selection: to improve the prediction performance, to provide faster and more cost effective predictors and to provide a better understanding of the underlying processes within the data. The work in this thesis considers each of these objectives. Ideally, it is hoped that by determining and selecting only the most important

timbral features the prediction performance and the cost effectiveness of a musical instrument classifier may be improved. In addition to this, it is hoped that determining and understanding the relative importance of the individual features may offer an insight into the nature of timbre and the important aspects in defining and describing timbre.

A review of the selection of features and examples in machine learning was given in Blum and Langley (1997). They discussed the merits of various embedded, filter and wrapper approaches to feature selection. Of particular note is their discussion on the relevance of particular features. The meaning of 'relevance' has not been consistent in machine learning as it often depends on the goal of any given algorithm. A variety of definitions of relevance were proposed in regard to what a set of features may be relevant *to*, such as the target, distribution or complexity. The current thesis is an attempt to find features that are relevant to timbre, or more specifically to discerning between timbres of individual instruments. Some of the features used in the experiments in the previous section may be of musical interest, but their relevance for the task at hand has not been determined.

The various functions of feature selection algorithms were described in Dash and Liu (1997). The purpose of feature selection as defined by a number of authors were discussed. They split the purpose of selection into four distinct concepts: an idealised approach to find the minimal subset that is necessary and sufficient to the target, a classical approach to select a subset to optimise a criterion function, an approach to improve prediction or decrease size without decreasing prediction accuracy, and a subset selection that maintains the original distribution of the features. The approach chosen is dependent on the manner in which the fitness of each selection is calculated.

The reviews above discussed feature and variable reduction methods using many types of algorithms applied to a number of studies in different fields. It is clear from these studies that the means of selection is dependent on the type of data used and the reason for reducing the given data or feature set. Regardless of the methods used, reducing the amount of features inputted to any system will undoubtedly alter the accuracy of any classifier to some extent. The impact of an error introduced by feature selection was examined in Sima et al. (2005) for three Gaussian models. This study determined that the while error estimation may work quite well with large sample sets, it did not work as well for small sample sets whereby feature selection could cause significantly worse classification accuracy than the optimal feature set. The sample size of the data in this thesis is quite extensive, containing many in-

stances of each instrument at a wide range of pitches and dynamics. Thus we need to include all features that are necessary to identify any note played under these conditions, rather than just 'typical' notes played at comfortable pitch and dynamic ranges. From the range of features discussed in the previous section, it is not evident which of these features would be most relevant for instrument classification with these conditions. We employ Genetic Algorithms (see experiments in Chapter 6) and Genetic Programming (see experiments in Chapter 7) to determine the most important auditory features to use for automatic instrument recognition. These algorithms are part of a specific field of natural computing known as Evolutionary Computation (EC).

A number of studies have found Genetic Algorithms to be a powerful tool for large-scale feature selection in comparison to contemporary methods (Ferri et al., 1993; Siedlecki and Sklansky, 1993a; Tseng and Yang, 1997). EC methods may evolve the optimum selection of features by combining different features and determining how good a particular selection is according to a user-defined *fitness function*. The advantage of EC methods is that they work on a population of solutions, and so many different combinations of features may be tried at once. The following section introduces the field of EC, and describes a number of ways in which it has been applied in the field of audio analysis and synthesis.

4.3 Evolutionary Computation

The field of Evolutionary Computation (EC) or Evolutionary Learning (EL) incorporates a number of algorithms that imitate the process of natural selection as found in nature. Evolution according to Darwin (1859) is based on the principal of survival of the fittest; that within any population of organisms, those that perform best are likely to succeed to the next generation and weaker specimens are more likely to die out. This principal can be applied in a computational manner for problem solving. The main difference between these evolutionary algorithms and other search algorithms is that evolutionary algorithms are initialised with a population of solutions to the given problem. The fitness of each of these solutions is defined by how well they solve the given problem. The population is then 'evolved' over many generations by allowing those solutions with high fitness to be passed to successive generations while removing solutions with low fitness values.

4.3.1 Search Space

Algorithms such as these that search for a solution have necessitated the idea of a *search space*. In these applications a search space refers to a collection of possible solutions to the given problem whereby the distance between the solutions is measurable in some manner. Consider a specific solution S_1 the success of which is known. As this solution has a specific measurable location within the search space it may now be moved to a neighbouring solution S_2. If this solution is more successful at solving the problem than S_1 it may continue in this direction, if not it may move in a different direction. Hence it may traverse the search space looking for the best solution (Mitchell, 1996).

The success of S_1 and S_2 at solving the problem above is measured as their *fitness*. Thus each solution in the search space has a corresponding fitness value. This in turn leads to a *fitness landscape*. This fitness landscape is imperative to the success of the algorithm. A typical fitness landscape consists of a number of peaks and troughs representing solutions of high and low fitness. These peaks represent local maxima (or minima) in the fitness. These maxima may represent 'good' solutions to the problem but there will only be one global maximum in the fitness landscape corresponding to the best solution available (Goldberg, 1989).

4.3.2 Optimisation

As the algorithm traverses the search space, it looks for the optimal solution to the problem. As described above, there are most likely a number of sub-optimal solutions at the local maxima or minima. Whether the algorithm is looking for a maximum or a minimum is dependent on the fitness function (if the problem being solved is to be minimised or maximised). The application of evolutionary algorithms in this thesis involves minimising the fitness function. Hence from this point minimisation is considered to be optimal. One traditional method of traversing a search space is described by the analogy of a *hill climber* (Goldberg, 1989). In this analogy a person standing at a point in the search space examines the slope at each direction away from them and follows the path of steepest descent, thus heading towards a minimum. While the hill walker will find a minimum in this manner, he has no idea when sitting in the bottom of the trough if it is the deepest trough in the whole space or if he has merely found a local minimum. Ideally he would need to note how deep that trough is and then 'jump' to another point in the space and start again, each time noting how low each trough is. The number of 'walks' he

needs to do is dependent on how complex the fitness landscape is; if the landscape has a large number of troughs he will need to perform a large number of walks to ensure he reaches the global minimum. Evolutionary algorithms work on a population of solutions rather than a single solution. Hence in this analogy we can start with a team of hill walkers starting at different random points on the search space. Those that appear to have found a trough can keep walking downwards (by keeping this solution in the next generation) and those that have not can be moved to a different point (changing this solution for the next generation).

In this thesis, the optimising property of EC algorithms are used to find the best selection of timbre features to use in automatic instrument classification. Two specific algorithms, Genetic Algorithms (GA) and Genetic Programming (GP) are described in more detail and employed for this purpose in later chapters. Artificial intelligence techniques have been applied to musical problems such as instrument recognition as detailed above, but also to problems in other musical areas such as pitch perception, tonal analysis, chord classification, sound synthesis, algorithmic composition, music cognition, musical schema and artificial listening (Balaban et al., 1992; Leman, 1996; Todd and Loy, 1991). Although EC methods have not been applied as much as other AI techniques, in recent years they have gained popularity in a number of musical applications. The remainder of this chapter reviews the use and success of EC methods in music and sound analysis and production.

4.4 Applications of EC in Sound and Music Production

Although EC has not been applied yet extensively in the area of sound analysis, a number of recent studies have started to use evolutionary methods to examine sounds. The following sections described the use of EC methods in sound analysis, music composition, sound synthesis and a number of other musical applications.

4.4.1 EC Methods in Speech and Sound Analysis

A number of studies have used evolutionary methods for speech recognition and analysis. Conrads et al. (1998) examined whether genetic programming could find programs that could discriminate certain spoken vowels and consonants. They conclude that a simple approach is successful even at speaker-

independent discrimination. More recently, GP was applied in Day and Nandi (2007) in the creation of an automatic speech recognition system. This system incorporated spectral, cepstral, perceptual and temporal features to create a program that may recognise a speaker with encouraging results. GP was also applied for speech quality estimation in Raja et al. (2008). This employed a GP using symbolic regression to evaluate speech samples subjected to distortion. This study found that GP compared favourably to existing approaches. GAs were used to reduce the computational costs for hearing aids in Cuadra et al. (2008). In this study, GAs were used to reduce a set of features in distinguishing between speech and non-speech finding that it could reduce the features set from 76 to 21 values with 'adequate' results. GAs are used in Rho et al. (2007) to develop a content-based music retrieval system. Using this system a user may look for a particular song using a number of different querying techniques and then give feedback according to whether or not the given sound is considered relevant to what they were searching for. They observed an improvement in querying results from incorporating a GA in such a system.

It is evident that although EC may have been applied for speech analysis and recognition, it has as yet to be applied for musical instrument recognition. They have, however, been applied extensively to other areas of music such as composition and synthesis. Such studies are discussed in the following sections as the details of the use of EC in other areas of sound and music production may be informative in applying it to sound analysis and identification.

4.4.2 EC Methods in Music Composition

Evolutionary methods have been applied to musical compositions in a number of studies over the past two decades. One of the first such applications was in evolving the musical transitions between two musical statements, know as thematic bridging. This problem was investigated in Horner and Goldberg (1991) with GAs using two fitness measures based on the similarities between the two segments being bridged. A review of evolutionary methods as applied to composing musical sequences is undertaken in Burton and Vladimirova (1999). For this task they define three main areas that must be considered for an efficient algorithm to be applied to this task. These areas are:

- Search Domain
- Input Representation

- Fitness Evaluation

The range of pitches, rhythms, melodies and harmonies available to a composer is essentially endless. The search domain limits the range of notes available to the algorithm by imposing constraints regarding the notes it may choose. These are dependent on the type of composition undertaken but may, for example, be related to rhythm, key or pitch range. The input representation must define how the music is represented by the algorithm, and hence how the music may evolve between successive generations. The fitness evaluation defines how successful any given composition is and thus whether it survives to the next generation or not. They propose that fitness evaluation methods for a GA may be further subdivided as: deterministic, formalistic, user-defined or neural. They discuss a number of studies that use such GA fitness methods along with those that employ GP for music composition finding that although successful sequences had been composed, these methods suffered from a 'fitness bottleneck' as each melody has to be individually evaluated. In addition to this, musical taste and the subjectivity of musical composition may prevent evolved compositions from appealing to all listeners. In general they found that GP techniques outperformed GA techniques for musical composition.

A number of studies to compose specific styles or aspects of music have incorporated evolutionary techniques. An algorithm known as GENJAM (Biles, 1994) used a GA to evolve jazz solos. It built solos from pre-generated MIDI sequences that were judged by a user to determine the fitness measure. GENJAM has been developed into a real-time performance system that can interact with a human mentor and a human performer (Biles, 2001). A composition tool named GeNotator was presented in Thywissen (1999) that used a modified GA to manipulate a musical composition using a hierarchical and generative grammar. VoxPopuli is an interactive compositional tool developed that used evolutionary methods in real-time algorithmic music composition using notes and chords (Moroni et al., 2000). Other studies have used evolutionary techniques in generating rhythms (Horowitz, 1994) and harmonies (Horner and Ayres, 1995) rather than melodies. Melody and rhythm were both evolved to create a simple melody in Goksu et al. (2005). This method used a GA to evolve melody and rhythm which were evaluated separately using two MLPs. These evolved melodies and rhythms were then mixed to produce verses and whole songs. Wiggins et al. (1999) discussed the use of GAs in generating four part harmonies for user-defined melodies and in generating instrument solos. Although they found that their method worked

'up to an extent', they concluded that composing using a GA ultimately lacked the subtleties and nuances of a human composer.

Miranda (2004) examined three distinct approaches to using evolutionary methods in music. The engineering approach refers to the use of GA and GP techniques in the field of sounds synthesis. The creative approach refers to the manner in which evolutionary techniques have been used in musical compositions. They discussed one further approach in applying evolutionary methods to music: the musicological approach. This approach refers to the search for the origins of music by means of computer simulations. In particular they discussed a model proposed in Todd and Werner (2000) which took its inspiration from the mating songs of birds. This model co-evolved male composers that composed songs along with female critics who decided, based on the songs composed, who to choose as a mate to produce the next generation. The critics judged the fitness of each song by encoding a Markov-like chain to rate the transitions from one note to the next. Each critic picked only one song, thus all critics may have only one mate but a composer may have many (or no) mates. This model demonstrated that co-evolving such male composers with female critics can lead to the evolution of pleasant tunes while maintaining diversity among the melodies throughout many generations.

The common weakness in all of the above studies lies in the calculation of the fitness of the individuals. Arguably, a user-defined fitness evaluation will give the most natural set of results, but it causes the most severe bottlenecks and only small populations over limited generations may be reasonably considered. Deterministic fitness calculations perform a mathematical function such as a pattern-matching scheme to evaluate a phrase or melody. While more individuals may be evaluated this way, such methods leave little room for artistic variation, which is often desirable in a musical composition. Formalistic fitness functions determine the fitness of an individual according to a set of pre-defined rules. Although rules are necessary in music, not being able to bend or break such rules would again lead to somewhat dull compositions. Employing a neural network as a fitness function allows more variety within compositions as these networks may be trained to produces a more general decision than rule-based methods. These generalisations may cause their own problems, however, in that a network may not be strict enough to deem an outlier note or chord as being unacceptable for the melody, thus ruining the compositions (Todd and Werner, 2000). These may not be such problematic issues in the sound analysis proposed in this thesis however, as the evaluation of a quality of a sound tone is more objective than that of a musical composi-

tion. Thus EC techniques should be as suitable if not more suitable to sound identification as it is for the music composition studies described above.

4.4.3 EC Methods and Sound Synthesis

The application of evolutionary methods in the area of sound synthesis has become more popular in the past decade. As discussed in Section 2.2 sound synthesis may be considered the inverse process to sound analysis, making the application of EC to sound synthesis of particular relevance to the work in this thesis. A synthesiser known as *Chaosynth* was developed in Miranda (1995, 2000). This program used Cellular Automata (Wolfram, 2005) to control the parameters of a synthesiser based on granular synthesis. An interactive GA was used in Johnston (1999) to examine the timbral qualities of synthesised sounds. The sounds were implemented in *CSound* using granular synthesis and the Karplus-Strong plucked sound algorithms. The parameters of these algorithms were evolved with the resultant tones evaluated by a human observer. They proposed that such a method may lead a user towards the production of genuinely novel sounds. A synthesiser known a *ESSynth* that integrates a mathematical approximation of a sound to a GA was proposed in Manzoli et al. (2001). GP was used in Garcia (2001) to automate the design of a Sound Synthesis Technique (SST) to generate sounds similar to those of acoustic instruments along with novel sounds. The SST was decomposed into a functional form and internal parameters and represented with expression trees. The resultant sound from the generated SST was compared to a target using a selection of parameters, resulting in an error (or fitness) value. A model called *Genophone* was developed in Mandelis (2001) that used a GA to control a dataglove that interfaced with a computer and synthesiser for sound synthesis and performance. GAs were used to optimise the parameters of a plucked string synthesis model in Riionheimo and Valimaki (2003). The fitness of each individual in this case was calculated from the perceptual distance between the synthesised sound and the desired sound.

EC was applied to control sound synthesis in a number of experiments described in McDermott (2008). Interactive EC was used for sound synthesis with a Graphical User Interface (GUI) developed to allow real-time user-controlled interpolation within the population. This addressed the problem of the user-bottleneck as it can speed up the audition and evaluation of the population. Synthesis experiments using such GUIs were compared to other interactive EC and non-EC synthesisers with comparable results.

Evolutionary computation has been used on a number of occasions to

evolve the parameters for Frequency Modulation (FM) synthesisers. FM synthesis is highly efficient for synthesising rich timbres although the results are not easily controlled. Originally proposed in Chowning (1973), FM when applied to synthesis of musical instruments generally relied on trial and error. One of the earliest studies to apply evolutionary techniques to such a synthesiser was Horner et al. (1993a). In this study, a tone was synthesised with FM and then compared to a target tone using short-time spectral analysis. The calculated fitness of this tone was used with a GA to evolve the parameters of the FM synthesiser, thus developing a method for efficient determination of FM parameters. A number of studies (Horner, 1996; Lim and Tan, 1999) have used evolutionary methods to optimise the features for double modulator FM models (Schottstaedt, 1977). Horner also applied a GA to nested modulator and feedback FM matching (Horner, 1998). A more detailed discussion of the merits of these evolved FM synthesis techniques is given in Horner (2003). A variety of fitness functions used to control a GA evolving the parameters of a FM synthesiser were compared in McDermott et al. (2005). This study compared synthesised sounds to target sounds using fitness functions based on a uniform metric, a pointwise metric, and DFT metric, a perceptual metric and a composite metric. They concluded that the composite metric (comprising of the summed weights of several simpler measures) performed the best as a fitness function for this task.

As with FM synthesis above, evolutionary methods have been applied to optimise the use of wavetable synthesis. This type of synthesis involves storing one period of a waveform in an oscillator table and scaling the output of this table as necessary. The waveform is created using a sum of simple harmonic sine waves thus creating a static spectrum. Spectral interpolation may be achieved with wavetables by splitting the spectrum into a number of basis spectra and crossfading the wavetables of these spectra. A GA was used in Horner and Beauchamp (1996) to select the number of basis spectra to use for spectral interpolation using wavetables. GAs have also been used to match wavetable models by selecting spectral snapshots of the original tone as the basis spectra (Horner et al., 1993b). These snapshots can then be compared to the synthesised tone. A simpler yet equally effective version of this was proposed in Horner (2001) whereby a best match was found for a subset of the spectral snapshots rather than each snapshot of the original tone. Various forms of genetically evolved wavetable and FM synthesis methods are discussed and compared in Horner (2003). They conclude that the best choice of synthesis is dependent on the complexity and memory constraints of the

given situation.

4.4.4 Further EC Applications

Although most of the evolutionary applications within the field of music have been applied to either composition or synthesis as noted above, there have been a number studies on various other music related issues. Three case studies that used evolutionary computing to examine the evolution of musicality, including musical tastes and the emergence of grammars were discussed in Miranda et al. (2003). The co-evolved mating calls described in Todd and Werner (2000) were discussed as a demonstration of how Darwinian systems with survival strategies can evolve coherent sets of melodies within a population. A further study by Miranda (Miranda, 2002) proposed a *mimetic model* to force candidates to form social bonds by 'remembering' a number of melodies, thus showing that a community may evolve a shared repertoire of musical melodies. Finally, Miranda et al. (2003) designed a model to study the emergence of musical grammars. They found that developing more general musical grammar rules could result in a more stable musical culture. In Takala et al. (1993) GAs were used in the composition of sound signals synchronised to animated motion. Sound compositions were represented using 'Timbre Trees' which were then evaluated so the resultant sound may be rated by a user. Parameters that define the instance of the sound along with environmental effects and morphing were included in the model.

Section 4.1.1 above discussed machine learning methods for developing musical instrument classifiers. Studies by Essid and Fujinaga incorporated binary GAs in selecting features to use in such classifiers (Essid et al., 2004, 2006; Fraser and Fujinaga, 1999; Fujinaga, 1998; Fujinaga et al., 1998). An alternative method of optimising features for classifying musical instruments was proposed in Peeters (2003); Peeters and Rodet (2003). Rather than using evolutionary methods, these studies used an algorithm based on inertia ratio maximisation, finding that good feature selection may increase the recognition rate of a classifier. From examining the previous literature on musical sound recognition and evolutionary computation, the current thesis will further the work already undertaken:

- to create a more robust musical instrument identifier that may be trained and tested using a large number of sound samples
- to remove any pitch, recording or dynamic bias in the classifier by including samples from the entire pitch range of each instrument played

on a number of different makes of each instrument at various dynamic levels

- to concentrate on the features used by the classifier rather than the classifier itself as no classifier, regardless of how powerful it is, can accurately classify a sound on which it does not have enough information
- to apply a floating point GA, with a purpose-defined fitness function to decide which and how much of each feature to use with a classifier for musical sound recognition
- to apply a GP to attempt to develop a simpler musical sound classifier than those described in this chapter
- to compare the results of the features chosen by the GA and the GP to determine if any emerging features can offer any insight into the understanding, description or definition of timbre

4.5 Conclusion

This chapter discussed numerous studies from areas within which the current thesis is based. We have seen in Section 4.1 that although numerous studies have been carried out over the past two decades on musical instrument classification from single note samples, there are significant differences in the ways in which these studies were carried out. A wide variety in the number of instruments, number of samples, feature selection and classification methods incorporated in these experiments has led to some ambiguity in their results. For the majority of studies, the number of samples quoted would not cover multiple instances for each instrument studied. Hence it is unlikely that these classifiers could be generalised to recognise the instrument played or recorded under different conditions. This thesis concentrates on identifying five instruments, but includes over 3000 individual samples played on these instruments. This ensures that any classifier developed will be capable of identifying an instrument regardless of pitch or dynamic.

It was seen that from the wide range of features used, the choice of features included was rarely explained in individual studies. Simply incorporating more features into a classifier may not increase its accuracy or performance. Thus we look to optimise the choice of features through the process of feature selection as explained in Section 4.2. We propose to implement this feature

selection in later chapters using Evolutionary Computation. A brief introduction to the field of EC was given in Section 4.3 along with a discussion on how it has been applied within the area of music and sound synthesis. Further details on evolutionary algorithms and how they are used in this thesis are given in later chapters. The following chapter details some experiments on creating automatic instrument identifiers similar to those described in Section 4.1.

Chapter 5

Data Reduction and Classification

The previous chapters discussed the difficulties in defining and describing the timbre of musical instruments. Chapter 3 discussed some previous attempts at describing timbre whereas Chapter 4 described numerous methods that have been applied to the problem of instrument identification. These involved many different classification techniques applied to a wide variety of calculated timbral attributes. In a number of these experiments, the number of attributes used led to very large sets of data. In order to classify such large data-sets it is useful to be able to reduce the data in some way before applying the chosen classification technique. The reduction technique chosen for this thesis is Principal Component Analysis (PCA). The classification technique employed for a large part of this thesis is Multi-layered Perceptrons (MLP). This chapter describes both PCA and MLP and discusses results obtained from applying these techniques to classification experiments involving real instrument sample sounds.

This chapter is laid out as follows: Section 5.1 discusses the need for a reduction in data, introducing PCA and its implementation within this project. Section 5.2 discusses the development of Artificial Neural Networks, in particular MLPs. Section 5.3 discusses some experiments undertaken using these methods as part of this work, as presented in Loughran et al. (2008a,b,c). Finally Section 5.4 presents conclusions that may be drawn from these experiments and the implications they may have for further work.

5.1 Data Reduction

Many timbral features employed in instrument classification experiments are described by multiple data points. Sound is inherently a time-varying process. A number of the features used in this and previous studies are examined as they change over time. For example, one of the most commonly used features is the Root Mean Square (RMS) envelope, which measures the change in amplitude throughout the sound. Depending on the window size and hop size used in calculating this envelope (the amount of data used in taking this measurement and the distance between each measurement), this envelope could have several hundred points of data for each sample. Much of this data is redundant however. The information relevant for the features is in the overall shape of the envelope, not in the individual points. Hence, in using such features it is better to be able to reduce this data down to a more manageable number of data points without losing any of the vital information contained in the whole envelope. The method used for such data reductions in this study is PCA.

5.1.1 Principal Component Analysis

PCA, first introduced in Hotelling (1933), is a standard technique commonly used in statistical pattern recognition and signal processing in performing dimension reduction. Also known as the Karhunen-Loève transformation, it transforms the data orthonormally so that the variance of the data remains constant, but is concentrated in the lower dimensions. Data from higher dimensions can then be discarded without losing much detail of the original data.

In Rojas (1996), the calculation of the principal components based on linear associators is described as an iterative process. The first principal component of a set of n dimensional vectors (x_1, x_2, ..., x_m) is given by vector **c** that maximises the expression

$$\frac{1}{m}\sum_{i=1}^{m} \parallel c.x_i \parallel^2 \tag{5.1}$$

Figure 5.1 shows an example data distribution centered at the origin, where the diagonal runs in the direction of maximum variance of the data. If we consider the orthogonal projection of each point onto this diagonal we can represent each point as a single number as opposed to the x_1 and x_2 coordinates. To statistically analyse this data, the coordinate axis is rotated by 45

degrees to maximise the content of the new **x** coordinate. The new direction of the x_1 coordinate is the direction of the first principal component. The second principal component is calculated by subtracting the projection onto the first principal component of each vector x_i from the original vector x_i. Each successive component is then calculated recursively. The result is a set of orthogonal components with decreasing variance.

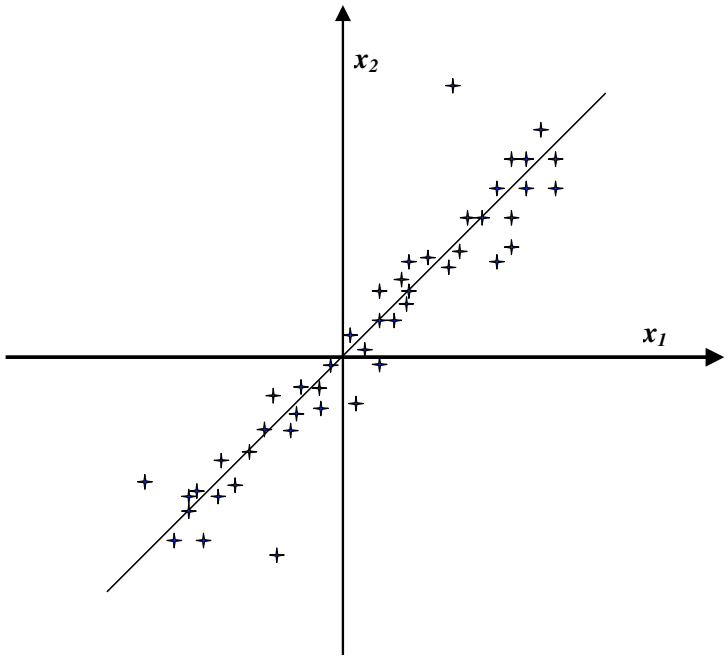


Figure 5.1: *Distribution of input data around diagonal running in direction of maximum variance, adapted from Rojas (1996)*

PCA algorithm

The PCA algorithm is explained in full in Haykin (1999). A brief synopsis of this algorithm is given below. Suppose we have an m-dimensional matrix **X** that we wish to reduce to $l < m$ dimensions. Assume **X** to be zero mean and let **q** denote a unit vector of length m onto which **X** will be projected giving **P**:

$$P = X^T q = q^T X \tag{5.2}$$

As the mean value of **P** is zero, its variance is the same as its mean-square value, giving

$$\sigma^2 = q^T C q \tag{5.3}$$

where the m-by-m correlation matrix $\mathbf{C}$ is

$$C = E[XX^T]q \tag{5.4}$$

As the variance of the projection P is a function of $\mathbf{q}$ we may define $\psi(\mathbf{q})$ as a variance probe

$$\psi(q) = q^T C q \tag{5.5}$$

By using eigenstructure analysis to locate the local maxima and minima along $\psi(\mathbf{q})$ we find that any changes in $\mathbf{q}$ must be orthogonal to $\mathbf{q}$ and therefore only changes in direction are allowed. Thus for a local maximum or minimum

$$Cq = \lambda q \tag{5.6}$$

where λ are the eigenvalues of the correlation matrix $\mathbf{C}$ and the associated unit vectors $\mathbf{q}$ are known as the eigenvectors of $\mathbf{C}$. Thus we find that

$$CQ = Q\Lambda \tag{5.7}$$

where

$$\begin{aligned} Q &= [q_1, q_2, ...q_j, ...q_m] \\ \Lambda &= diag[\lambda_1, \lambda_2, ...\lambda_j, ...\lambda_m] \end{aligned} \tag{5.8}$$

As $\mathbf{Q}$ satisfies the *conditions of orthornormality* and the inverse of $\mathbf{Q}$ is the same as its transpose we may re-write 5.7 as the *orthogonal similarity transformation*

$$Q^T C Q = \Lambda \tag{5.9}$$

Now $\mathbf{C}$ may be expressed in terms of its eigenvalues and eigenvectors as

$$C = \Sigma \lambda_i q_i q_i^T \tag{5.10}$$

Thus Equations 5.9 and 5.10 give two equivalent representations of the *eigendecomposition* of the correlation matrix $\mathbf{C}$. These equations demonstrate that eigendecomposition and principal component analysis are effectively the same process viewed in different ways, as the variance probes and the eigenvalues are equal. Thus the eigenvectors of the correlation matrix $\mathbf{C}$ of the original zero mean data vector $\mathbf{X}$ define the unit vectors q_j representing the principal directions along which the variance probes $\psi(q_j)$ have their minimum and maximum values. In addition to this the associated eigenvalues define the

minimum and maximum values of these variance probes.

5.1.2 Implementation of PCA

The Statistics Toolbox in Matlab (MATLAB7, 2006) contains a function `princomp` to calculate the principal components of a set of given data. This function returns

- coefs - the eigenvalues of the correlation matrix
- scores - the values of the original data mapped onto the new coordinate system defined by the principal components
- variance - the variance explained by each principal component
- t2 - Hotellings T^2, a statistical measure of the multivariate distance of each observation from the centre of the data (Jolliffe, 2002)

Thus the principal components of a set of data are easily computed in Matlab in experiments such as those described in Section 5.3.

5.2 Artificial Neural Networks

Once the data has been reduced, this reduced dataset may be used to identify a musical instrument. Such tasks can be undertaken by a number of classification algorithms. In this chapter a classification structure known as an Artificial Neural Network (ANN) was used. This network is formed from interconnecting artificial neurons, designed to mimic the first order characteristics of the biological neuron. The computational ability of these neurons was initially proposed in McCullock and Pitts (1943). The collaboration of a neuroanatomist (McCulloch) and a mathematician (Pitts) resulted in this pioneering work that described how a network of such simple units and connections could theoretically compute any computable function.

There are many different types of ANNs but they all share a common trait in that they are modelled on the functioning processes of the human brain. The fundamental operation of an artificial neuron is that a vector $\mathbf{X}$ is applied to a network corresponding to the way in which a signal is inputted to the synapses of a biological neuron. This input vector is then multiplied by an associated weight matrix, $\mathbf{W}$, before being summed to give the **NET** output. This **NET** is then passed through an activation function, which produces the output signal of the neuron. The activation function may differ between

networks; the simplest of them, a hard limiter, delivers a '1' if the output is above a threshold value, and a '0' or '-1' if it is below that value (Wasserman, 1989).

For a network to be able to classify a given pattern sample it must first learn how to respond to similar patterns through network *training*. Typically a large data set (the training set) is used to train the network so that it is modified to perform in a certain way. Training involves updating the weights and biases of each neuron to help the network classify the training samples as required. This trained network can then be used to classify another set of data (the test set) according to the manner in which it was trained. Trained ANNs have been shown to be a particularly powerful tool in data classification. The training of ANNs can be split into three broad categories (Duda et al., 2001):

- Supervised Training — a teacher (or target vector) provides a category or cost for each pattern in the training set. The correct classification of each training sample must be known beforehand so that the training process may reduce the sum of these costs. MLPs (Haykin, 1999) and Support Vector Machines (Cristianini and Shawe-Taylor, 2000) are examples of supervised networks.

- Unsupervised Training — no targets are associated with the training set. These networks operate by classifying data into clusters or groups according to how close they are within a given feature space. Depending on the algorithm used, the number or size of the clusters may be defined. k-Nearest Neighbours (Dasarathy, 1990) and Self-Organizing Maps (Kohonen, 2001) are examples of unsupervised networks.

- Reinforcement Training — partial information about the correctness of the network's response is available. In the extreme case the result is whether the response is right or wrong with no quantifiable measure of how much is mistaken by. This training is sometimes known as 'learning with a critic' (Duda et al., 2001).

The manner in which the weights are updated is dependent on the given network *learning rule*. There are a number of different learning rules available, some of which are discussed later in the chapter. A summary of a wider selection of these rules is presented in Zurada (1995). This chapter uses MLPs as a classification tool. As the name suggests, MLPs are formed from layers of interconnecting layers of artificial neurons known as *perceptrons*. The following sections discuss perceptrons and MLPs.

5.2.1 The Perceptron

Perceptrons were one of the first types of ANN, developed in the 1950's by Frank Rosenblatt (Rosenblatt, 1958) as the first model with supervised training. Perceptrons are the simplest form of an ANN used for classification, consisting of a single neuron connected by weights to a set of inputs. Despite their simplicity, the theory of perceptrons is the foundation for many other forms of ANN. The operation of a perceptron is as described above where a linear combination of the inputs to the node is combined with an external bias and then subjected to a hard-limiter. The illustration in Figure 5.2 denotes the inputs to neuron j as $x_1, x_2, ...x_m$ and the corresponding weights as $w_1, w_2, ...w_m$ with the bias denoted by b. From this it can be seen that the input to the activation function, or the *induced local field* of neuron j is

$$v = \sum_{i=1}^{m} w_i x_i + b \tag{5.11}$$

The perceptron decides which of two classes ε_1 and ε_2 each data element

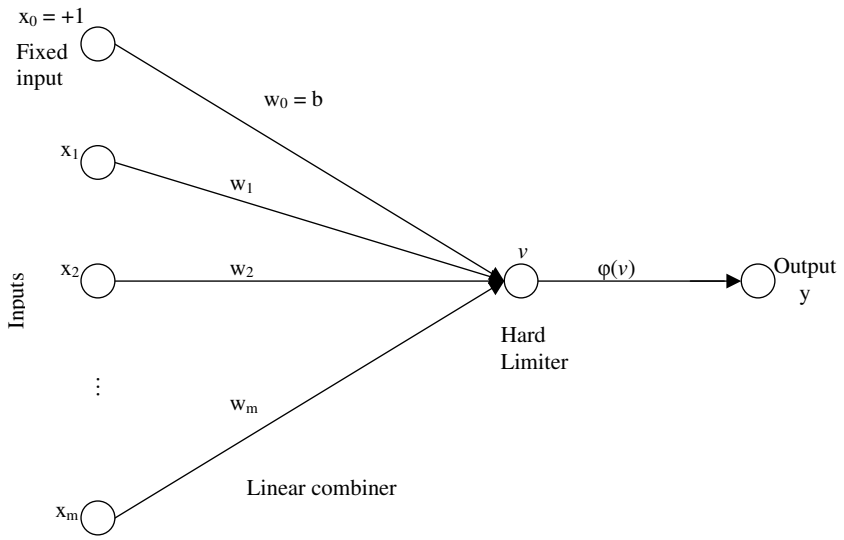


Figure 5.2: *Signal flow through a Perceptron, adapted from Haykin (1999)*

$x_1, x_2, ...x_n$ belongs to according to the value given by the limiter function. Training the network involves updating the weights and bias at each iteration so that the network correctly classifies all of the training data. Adapting the weights is performed according to the error correction-rule known as the perceptron convergence algorithm (Haykin, 1999). This algorithm defines the *error-correction learning rule* which updates the weight vector at each

iteration according to

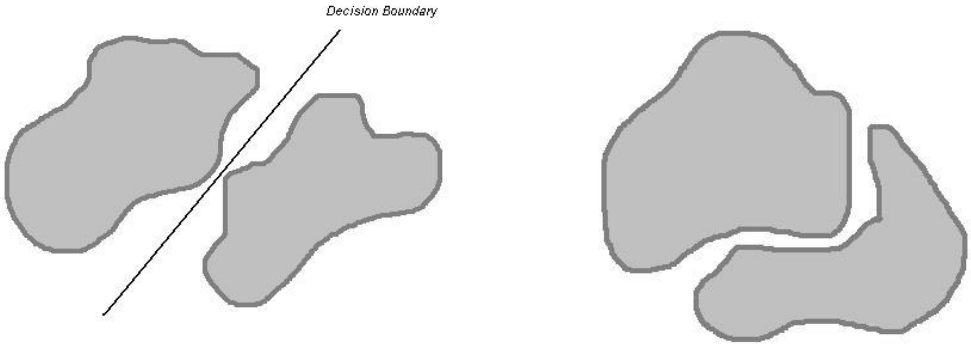
$$w(n+1) = w(n) + \nu[d(n) - y(n)]x(n) \tag{5.12}$$

where ν is the *learning-rate parameter* — a constant in the range $0 < \nu \leq 1$, and *d(n)* is the *quantized desired response* defined by

$$d(n) = \begin{cases} +1 & \text{if } x(n) \text{ belongs to class } \varepsilon_1 \\ -1 & \text{if } x(n) \text{ belongs to class } \varepsilon_2 \end{cases}$$

The choice of ν is clearly important to the training of the network. When choosing ν one must bear in mind that it needs to be large enough to provide fast adaptation of the network, but small enough to provide stable weight estimates.

Although the perceptron is a powerful yet simple tool, it is only useful for classifications involving data sets that are *linearly separable*. Patterns that are linearly separable can be separated by a hyperplane as illustrated in Figure 5.3. This problem of dealing with linearly inseparable data sets meant that a single layered perceptron would never be able to represent such problems as the Exclusive-OR Boolean function as shown in Minsky and Papert (1969). This apparent limitation of the perceptron led to a decline in the interest in the field of ANNs for almost two decades (Wasserman, 1989). To solve this problem the idea of using multiple layers of interconnecting perceptrons was introduced.



(a) (b)

Figure 5.3: *Two sets that are (a) linearly separable and (b) linearly inseparable*

5.2.2 Multi-layered Perceptron

Multi-layered Perceptrons (MLPs) consist of a number of layers of interconnected perceptrons. There must be at least three layers of perceptrons — an input layer, an output layer and one or more hidden layers. In general, this type of network is trained using the backpropagation algorithm which is based on the *error-correction learning rule* described in the previous section. Backpropagation involves two passes of information through the layers of the network — a forward pass and a backward pass. In the forward pass the input vector is applied and propagated through the layers, leading to the term *feed-forward network*. In the backward pass the error function is calculated as the error between the expected output and the achieved output. This error is then propagated back through the layers of the network and the synaptic weights of each neuron are adjusted accordingly. A diagram of an interconnected network is illustrated in Figure 5.4.

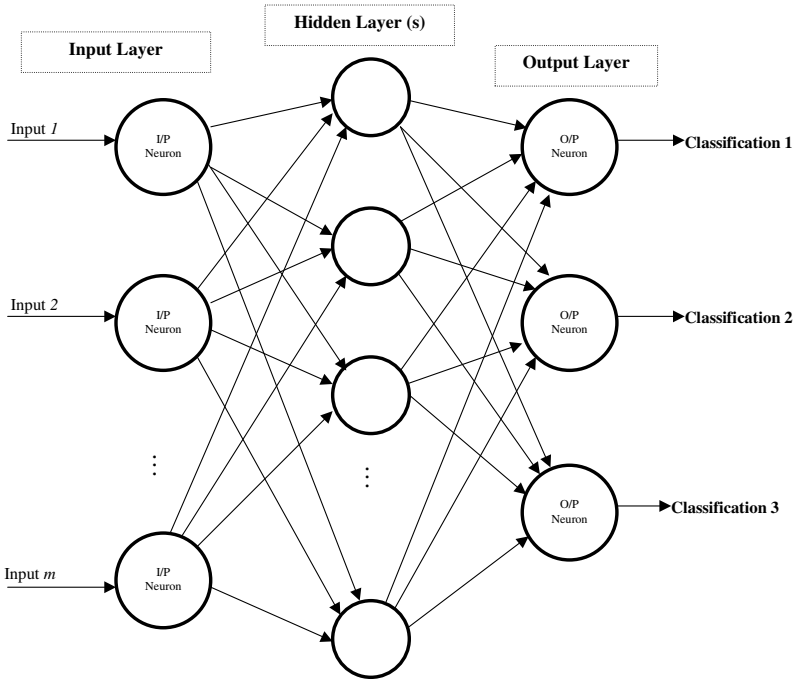


Figure 5.4: *Illustration of an MLP with inputs 1-to-m, one hidden layer and three output nodes*

To function successfully, an MLP must have the following characteristics (Haykin, 1999):

- The network must contain at least one hidden layer of neurons.

- Each neuron in the hidden layers must have a *non-linear activation function* associated with it.
- The network must exhibit a high degree of connectivity.

The presence of highly connected hidden layers of neurons with non-linear activation functions is what enables an MLP to deal with the non-linear data sets that single perceptrons cannot represent. If the activation function is not non-linear, the two connecting layers could be reduced to a single layer of perceptrons. The backpropagation algorithm may be described as a more general form of the least-mean-square (LMS) algorithm, which when applied to single error networks can estimate the new weight vector at iteration $(n+1)$ as

$$w(n+1) = w(n) + \nu x(n)e(n) \tag{5.13}$$

where $e(n)$ is the error signal at time n (Haykin, 1999). Many other training models may be seen as modifications of this backpropagation model (Duda et al., 2001). Backpropagation is a powerful yet simple training model because of its intuitive graphical representation as discussed below.

The Backpropagation Algorithm

The backpropagation algorithm works by calculating an error signal at the output of a network and propagating that error back through the network, layer by layer, adjusting the weights of each neuron accordingly. A detailed description of the workings of the backpropagation algorithm is given in Haykin (1999). This description is summarised here.

The error signal at the output of neuron j (where j is an output node) after n presentations is defined by

$$e_j(n) = d_j(n) - y_j(n) \tag{5.14}$$

The instantaneous value of the total error is calculated by summing the instantaneous errors of all neurons in the output layer C

$$\varepsilon(n) = \frac{1}{2}\sum_{j \in C} e_j^2(n) \tag{5.15}$$

The average error energy over all neurons ε_{av} represents a cost function of the performance of the network. The instantaneous error energy at the output of the network is independent of all synaptic weights and bias levels within the network. The objective of the learning process is to adjust the weights and

bias values to minimise this cost function. This is achieved by updating the network weights on a pattern-by-pattern basis until one epoch (presentation of the entire training set) has been completed.

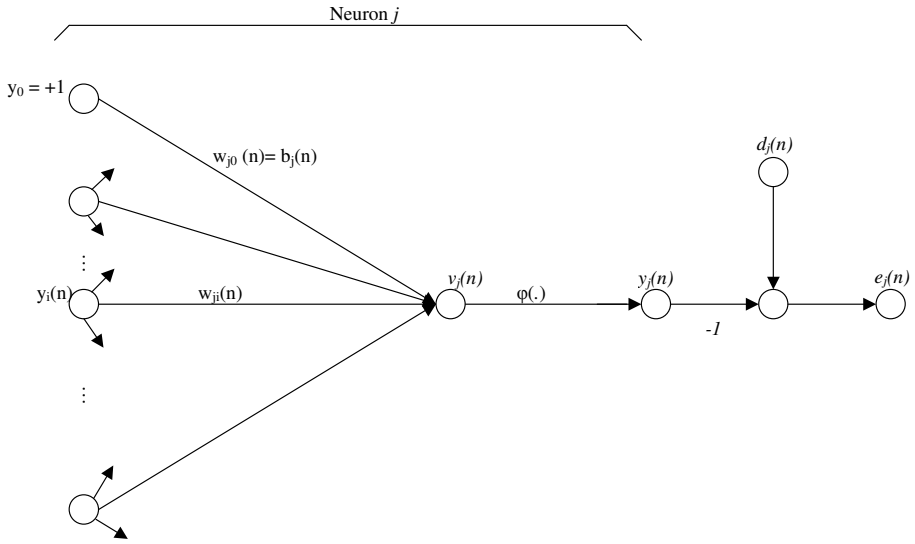


Figure 5.5: *Signal flow through neuron j within the MLP, adapted from Haykin (1999)*

Consider an individual neuron j being fed signals from a layer of m neurons as shown in Figure 5.5. The induced signal or local field being presented to the activation function of j is therefore

$$v_j(n) = \sum_{i=0}^{m} w_{ji}(n) y_i(n) \tag{5.16}$$

It can be shown that the change in weight Δw_{ji} is given by

$$\Delta w_{ji} = \nu \delta_j(n) y_i(n) \tag{5.17}$$

where ν is the *learning-rate* of the algorithm and the *local gradient* $\delta_j(n)$ is defined by

$$\delta_j(n) = e_j(n) \varphi_j^{'}(v_j(n)) \tag{5.18}$$

Equation 5.18 above shows that the local gradient, which indicates the necessary changes in the weight at neuron j is dependent on the error signal and the derivative of the activation function for that neuron. Thus the error signal for the neuron must be found to calculate the necessary change in its synaptic weight. There are two types of neurons, output neurons and hidden

neurons. In the case of the output neuron, the error signal is easy to obtain as the desired output of this neuron is known (as backpropagation involves *supervised* learning). Thus for an output neuron the error signal $e_j(n)$ can be calculated from Equation 5.14 and hence the local gradient $\delta_j(n)$ can be calculated from Equation 5.18. If neuron j is in one of the hidden layers, its error signal may not be directly calculated, but it still has a bearing on the error signal produced at the output of the network. Backpropagating the error through the network determines the change in synaptic weight of the hidden neuron j according to the error signal of all the neurons it is directly connected to.

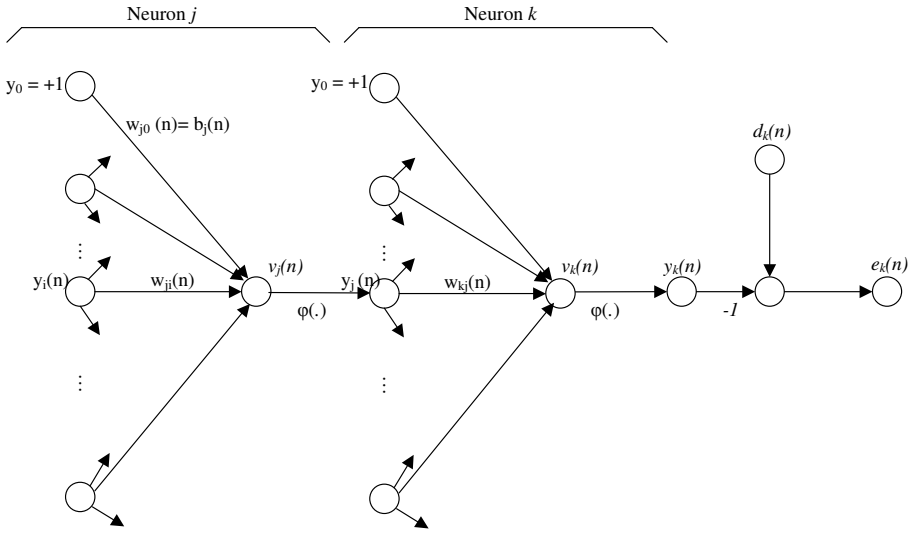


Figure 5.6: *Signal flow through hidden node j to output node k, adapted from Haykin (1999)*

Consider the hidden node j depicted in Figure 5.6. The local gradient for j is now

$$\delta_j(n) = -\frac{\partial \varepsilon(n)}{\partial y_j(n)}(n)\varphi_j^{'}(v_j(n)) \tag{5.19}$$

From examining the instantaneous energy and differentiating, it can be shown (Haykin, 1999) that the local gradient at j is given by

$$\delta_j(n) = \varphi_j^{'}(v_j(n)) \sum_k \delta_k(n) w_{kj}(n) \tag{5.20}$$

and again the necessary change in weight Δw_{ji} is given by the delta rule

$$\Delta w_{ji}(n) = \nu \delta_j(n) y_i(n)$$

Improvements to the Backpropagation Algorithm

Although the multiple layers and high level of connectivity of MLPs is what gives them their powerful processing abilities, this is also the cause of one of their biggest flaws, that being the difficulty in designing an optimum MLP. As the network is trained iteratively in two directions, it is difficult to determine how many neurons, or even how many layers of neurons are necessary and sufficient for the network to learn a specific task. As such, MLPs offer a 'black box' solution to a given problem and their design is often based on trial and error. The operation of the backpropagation algorithm as described above is dependent on a number of factors which the user has control over and must decide upon before learning commences. These factors include choice of activation function, learning-rate parameter, modes of training and stopping criterion (Haykin, 1999).

The activation function controls the local gradient at each neuron and hence has an effect on the operation of each layer of neurons in updating the synaptic weights. As it is the derivative of the activation function that is required, it is necessary for the function chosen to be continuous. Any non-linear continuous function may be used as an activation function. Two of the most commonly used functions are the *Logistic Function*:

$$\varphi_j(v_j(n)) = \frac{1}{1 + e^{-av_j(n)}} \quad a > 0 \text{ and } \infty < v_j(n) < -\infty \tag{5.21}$$

and the *Hyperbolic Tangent Function*:

$$\varphi_j(v_j(n)) = a \tanh bv_j(n), \quad (a, b) > 0 \tag{5.22}$$

Both of these functions are forms of *sigmoidal nonlinearity functions*.

The backpropagation algorithm searches the weight space for the direction of the decrease in error (gradient descent). The choice of learning-rate parameter ν controls how rapid this descent is. The smaller the value of ν the less change there will be in the weights at each iteration and so learning may be quite slow. Alternatively, if ν is too high the changes in synaptic weights may be too drastic and training of the network could oscillate and become unstable. A balance between these extremes may be reached by generalising

the delta rule to include a feedback component to maintain stability:

$$\Delta w_{ji}(n) = \alpha \Delta w_{ji}(n-1) + \nu \delta_j(n) y_i(n) \tag{5.23}$$

where the *momentum constant*, α, controls the amount of feedback. This constant may vary from 0, where the new weight is purely dependent on the gradient to 1 where it is equally dependent on the gradient and the previous weight. The use of momentum may help prevent the network from becoming stuck in a local minimum.

As a network is being trained, it is typically presented with many examples of data from a large data set. Each complete presentation of the data is known as an epoch. There are two modes of training, *sequential training* and *batch training*. Sequential training involves updating the weights of each node as each data training example is applied. In batch training, however, the weights of the neurons within the network are not updated until one full epoch has been applied, whereby the gradients calculated at each training example are summed to determine the new weights and biases. Although the batch mode of training may appear to (and often does) require less computation, it is worth noting that for online operations or when a large training data set is largely redundant, sequential training can often be more efficient.

There is no absolute method of deciding when the network has finished learning. In practice, the algorithm may be stopped once a specified number of epochs has been presented to the network or a specified amount of time (in seconds) has passed. Alternatively the network could be stopped when the improvement in the error or the gradient of the descent has reached a specified limit or when the error is no longer decreasing. These criteria imply that the network has converged and will improve no more, although it is possible that this is due to the network finding a local minimum rather than the optimum solution. The following section outlines the practical choices such as these that have to be considered when implementing an MLP.

5.2.3 Implementation of MLPs

MLPs were implemented in Matlab (MATLAB7, 2006) by creating a network object using the function `newff` from the Neural Network Toolbox. The number of layers of hidden neurons and the amount of neurons contained in each layer were specified using this function. At initialisation, the maximum and minimum permissible values for each element of training data were specified. The choice of activation function was between the *Log-Sigmoid*, *Tan-Sigmoid*

or *Linear* transfer function for each layer of neurons. The *Linear* transfer function may be useful at the output layer if the output is to be outside the region of [-1, 1]. Once the network was designed, choices regarding the method of training and presence of momentum were made. Matlab offers a number of methods of implementing faster training such as a variable learning rate, resilient backpropagation or conjugate gradient algorithms. Once the network, activation and training function were decided on, the stopping criteria were specified as described in the previous section. Details of the choices of these specifications are given in the descriptions of the experiments in the following section. The trained network was then simulated with the test data to determine how successful it is at classifying new data.

5.3 Classification Experiments

The method of reducing data using PCA and then classifying this reduced data with an MLP as described in this chapter was used in a number of early experiments in musical instrument identification. These experiments were undertaken on the grounds of the success noted in similar work as described in the previous chapters. Although such previous studies appear to produce good results, it is inherent in this type of study that the experiments may be biased in some way, as discussed in Herrera-Boyer et al. (2003). This bias may be in the choice or number of instruments used. Some of the studies classify all the instruments in the orchestra, whereas others classify a smaller group of instruments, some of which are in the same instrument family. A bias may also arise in the range of pitches examined. A number of these studies examined the same range of notes across all instruments, while the natural playable range of each instrument studied may be much larger. The data reduction and classification methods chosen may also affect the outcome of the instrument classification. Finally the features chosen in the analysis of the instrument sounds as well as the sample sounds themselves will have a large bearing on the outcome. As discussed in Herrera-Boyer et al. (2003), these types of choices or 'biases' can lead to inconsistent or misleading results.

Although a number of previous studies incorporated a large number of instruments, it is not evident if enough samples were taken from each instrument to classify them accurately. From the number of samples quoted in some of these experiments it is unlikely that the instruments were sampled at different dynamic levels, or that different models of each instrument were sampled. In the experiments described here, the choice of instruments was

limited to the piano, violin and flute. This was to ensure an exhaustive coverage of each instrument. Samples were taken from the RWC Music Database (Music Instrument Sound) (Goto, 2004) of the selected instruments. Three models of piano, Yamaha, Bosendorfer and Steinway were each sampled at dynamic levels *f*, *mf* and *p* across their range (RWC, 2001a). Violins manufactured by J.F. Pressenda, Carcassi and Fiumebianca were sampled at these three dynamic levels with vibrato and at level *mf* without vibrato across their range (RWC, 2001b). Flutes manufactured by Louis Lot and Sankyo were sampled at the three dynamic levels both with and without vibrato (RWC, 2001c). In total, this produced a training set of 2004 samples across the entire pitch range of the three instruments. In contrast, many of the previous studies discussed in the previous chapter used approximately 1000 samples ranging across 15 to 30 instruments.

The samples that constitute the test dataset in these experiments were taken from the MUMS (Opolko and Wapnick, 1987) database. This smaller dataset consisted of samples of the three instruments played at the one dynamic level. In total, this dataset consisted of 45 violin samples, 37 flute samples and 88 piano samples. A separate dataset was chosen as a test set to realistically examine the generality of the trained classifier. Ideally, a classifier such as the neural network described here, when trained properly, should be general enough to be able to recognise a sample regardless of the source of the sound. Using separate datasets recorded under different conditions ensured that it was the tonal quality of the instrument being categorised, and that superfluous qualities such as those arising from recording conditions or playing style should not have an effect on the results.

5.3.1 Pitch Range

The first experiment undertaken examined the classification of the instrument samples described above by extracting a small selection of features. Specifically, this experiment compared results obtained from training and testing across different pitch ranges of the instruments. The natural pitch range of each instrument was first examined. These ranges differ in size for each instrument but consist of all playable pitches from each of the instruments. Next a single octave, C5 to C6, present in all three instruments was used to distinguish between the instruments. The classification accuracy over these two pitch ranges were then compared. The results shown here were presented in Loughran et al. (2008a). The features first examined in this study included the Temporal Envelope, Residual Envelope, Spectral Envelope and

the Centroid Envelope, as described in Chapter 3.

Results from PCA Data Reduction

As each of the features described above are time-varying envelopes, they all have multiple data points; for example the Temporal Envelope of C4F (pitch C4 played at volume f) on a Yamaha piano consists of 603 data values, whereas the Temporal Envelope of C7M played on the same instrument has 724 data values. An Excel file was created for each feature storing all 2004 instrument sample values for that particular feature. The temporal feature calculations described above involved splitting the sound into temporal frames. As the sounds were of various lengths (each note was held for a different amount of time), the number of frames in each sound varied. To store these envelopes together a maximum length was found for each feature with the corresponding feature envelopes padded with zeros as appropriate. Thus the size of these sets of features ranged from 2004 x 300 values for the Spectral Envelope to 2004 x 1400 for the Centroid Envelope. Each feature was reduced using the PCA algorithm as described earlier in this chapter. For example the 2004 x 1350 Temporal Envelope data matrix could be reduced to determine its eigenvalues λ_i according to Equation 5.10:

$$C = \Sigma \lambda_i q_i q_i^T$$

where $\mathbf{C}$ is the correlation matrix of the data. This matrix is of dimensions 1350 x 1350 representing the resultant principal components ordered according to the percentage of explained variance from highest to lowest. The determined eigenvalues in turn were used to calculate the projections of the original data mapped onto the new co-ordinate system. This is equivalent to the `scores` output of the `princomp` function in Matlab. Up to three instances (dimensions) of this data may be plotted within this new co-ordinate system. Thus it is possible to observe how efficiently the first three principal components of each feature separate the original dataset into individual instrument groups. The observed separation of this data can give an indication as to how well the MLP will be able to categorise the samples.

PCA Plots Across Range of Instruments A 3-dimensional plot of the first three principal components from the Temporal Envelope data of the entire training set of data is shown in Figure 5.7. This plot is encouraging as the three instruments can clearly be seen to segregate from each other. The

piano samples have segregated themselves into a distinct group. This is not surprising as the strong attack in the envelope of the piano is very distinct from the other two more sustained instruments. The violin and flute samples also segregate, but there is some overlap between the two. Hence another feature is needed to distinguish these instruments distinctly.

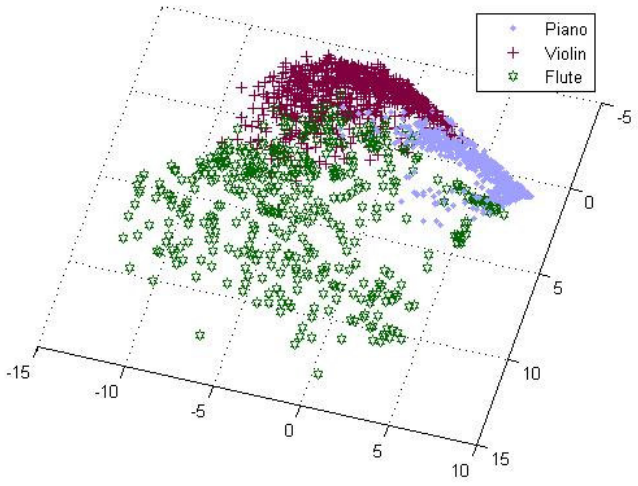


Figure 5.7: *Plot of the first 3 principal components of the Temporal Envelope data across the physical range of each instrument*

A plot of the projection of the Residual Envelope by the first three components is shown in Figure 5.8. Although there appears to be some clustering between the instruments, the boundaries between the instrument groups are less clear than those formed by the Temporal Envelope as the data is more tightly clustered. This plot shows a large overlap between the Violin and Piano groups and some overlap with the Flute samples. This lack of definition between the instrument groups implies that this feature would not be useful as an input to an MLP to train it to distinguish between these instruments. As such the Residual Envelope was not used any further within this experiment.

A similar plot of the projection obtained from the first three components of the Spectral Envelope is shown in Figure 5.9. Again this shows a lot of overlap between the instrument groups. The separation between the instruments does not appear to be sufficient to aid the MLP in its classification. Hence the Spectral Envelope was also not considered further in this experiment.

Figure 5.10 displays the plot obtained from three principal components of the Centroid Envelope data. This plot shows distinct segregation between the instrument groups. Although there is a slight overlap between the Piano and

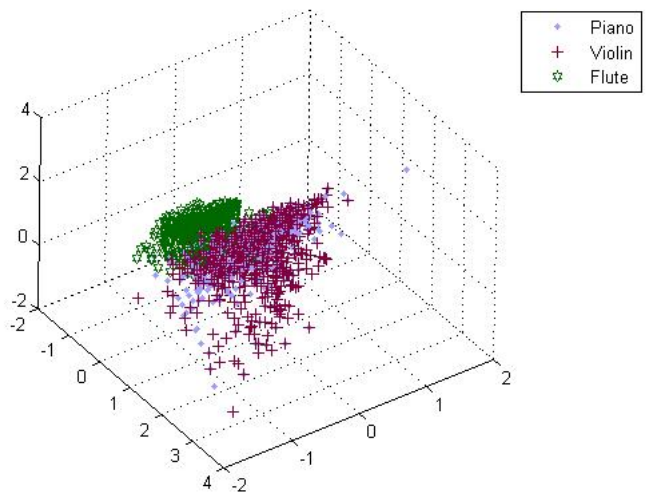


Figure 5.8: *Plot of the first 3 principal components of the Residual Envelope data across the physical range of each instrument*

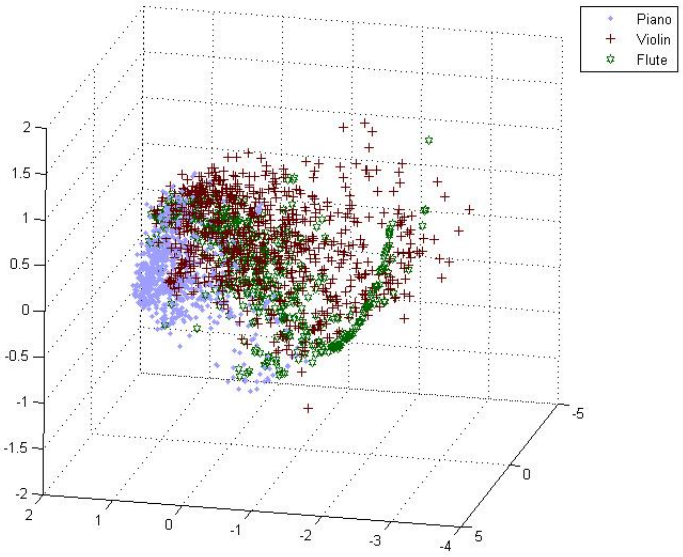


Figure 5.9: *Plot of the first 3 principal components of the Spectral Envelope data across the physical range of each instrument*

Violin and the Flute and Violin, for the most part each instrument clusters within a specific region. This distinct separation between the instruments encourages the use of the Centroid Envelope data for classification between the instruments. Thus the remainder of this experiment considered data from the Temporal Envelope and Centroid Envelope only.

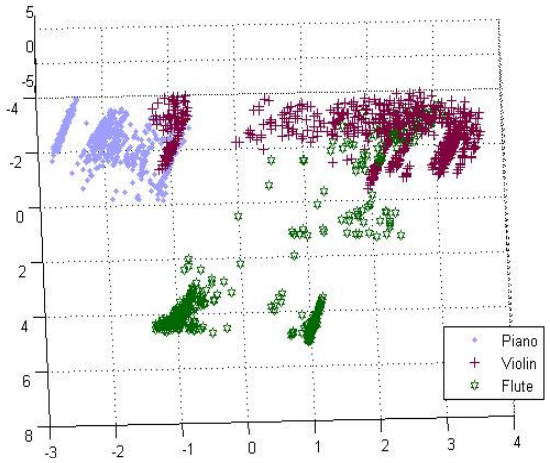


Figure 5.10: *Plot of the first 3 principal components of the Evolution of the Centroid data across the physical range of each instrument*

PCA Plots Across One Octave A 3-dimensional plot of the first three principal components from the Temporal Envelope across the range C5 to C6 can be seen in Figure 5.11. This plot displays the principal components of both the training and the test data on the same plot. The piano once again is the most easily segregated instrument, although the flute and violin do also show good separation. A plot of the principal components extracted from the training and test samples across one octave for the Centroid Envelope data is shown in Figure 5.12. Again this shows quite good separation between the instruments. In particular, the flute samples are distinctly segregated from the rest of the samples. Although the plots are more sparse due to the smaller number of samples used, the data in these plots separates in a similar manner to that of the corresponding larger datasets for these features. In particular, the test data included in these plots appears to cluster with the corresponding training instrument groups. This clustering indicates that data from these features would be useful as input to the MLP.

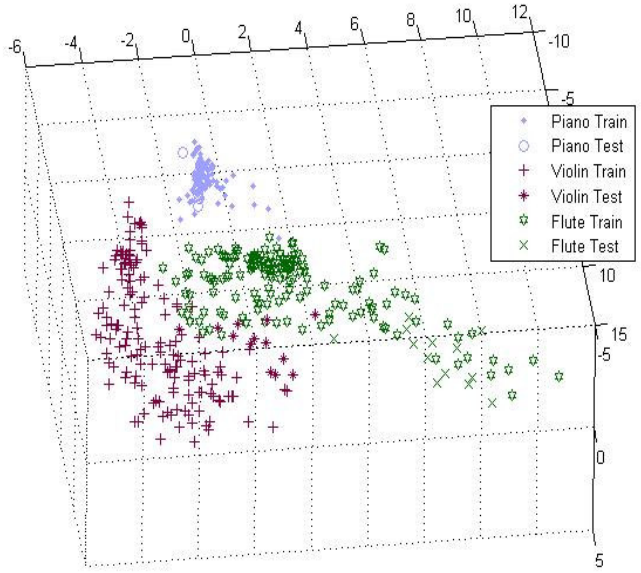


Figure 5.11: *Plot of the first 3 principal components of the Temporal Envelope data across one octave of each instrument*

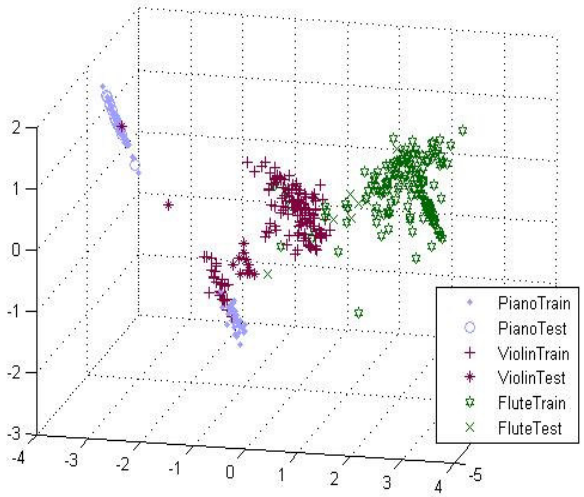


Figure 5.12: *Plot of the first 3 principal components of the Centroid Envelope data across one octave of each instrument*

MLP Classification Results

Once the data has been reduced and the scores from the principal values extracted, these values were used to train a MLP. The MLP used for this experiment was implemented with the `newff` function in MATLAB7 (2006) as described above. As described in Section 5.2.2 the network was trained to classify a sound by being presented with training samples and updating the synaptic weights of each neuron w_j according to the delta learning rule:

$$\Delta w_{ji}(n) = \alpha \Delta w_{ji}(n-1) + \nu \delta_j(n) y_i(n) \tag{5.24}$$

where α is the momentum constant and ν is the learning rate. The network in this experiment was trained by presenting it with three to five principal components of the Temporal Envelope and Centroid Envelope data from the 2004 training samples (in the case of the full range of samples) and updating the weights according to Equation 5.24 with ν set to 0.95 and α set to 0.1. It was batch trained with a Quasi-Newton, BFGS (Broyden, Fletcher, Goldfarb and Shanno) algorithm. This algorithm updates an approximate Hessian matrix of the performance of the current weights at each iteration of the algorithm (Dennis and Schnabel, 1996). It was implemented in MATLAB7 (2006) using the `trainbfg` function. The stopping conditions for training were set to a goal of 0.001 and 1000 epochs, that is this network will continue to train using the data set until it achieves an error of 0.0001 or below, or until it has tried to train the network to that goal 1000 times and fails. With this set up it was found that a network with 57 neurons in the first layer and two hidden layers containing 22 and 8 neurons respectively would be sufficient to train the larger data set. As these classification experiments discern between three instruments, there are three output neurons in each of these experiments. This number of neurons and layers were found through trial and error. A smaller network would most likely train the one-octave set, but for ease of comparison the same network was used for both sets.

Classification Over One Octave The classification results over one octave of each instrument are shown in Table 5.1. These results display the percentage of times the network trained on the 2004 training samples correctly identified a new test sample. The experiment was initially run using the scores data from the first 3 principal components, as plotted on the previous PCA plots in Figures 5.11 and 5.12. Although the PCA plots may only display data from three principal components, inputs to the MLP are *not*

limited in this way. Hence the experiment was repeated to include the scores data from the 4^{th} and 5^{th} principal components. It is evident from Table 5.1 that choosing four principal components from the Temporal Envelope data produced the most accurate results. Conversely, varying the number of principal components for the Centroid Envelope data did not have a significant effect, as the results were consistently high. These results may appear inconsistent, in that increasing the amount of principal components can increase accuracy in one instance yet not in another. The manner in which PCA reduces data is quite unintuitive however. It is not known what physical aspect each component relates to, if it does relate to one. It is possible that higher components extracted from the Temporal Envelope data may correlate to some playing style or alternative aspect of the note not necessarily reliant on the instrument's timbre. This lack of intuitiveness is a drawback of PCA. The reductional ability of this algorithm with minimal loss of data, however, compensates for this drawback. The consistently high results in the Centroid data may be due to the small data set tested. The next section gives the classification results of the more generalised dataset.

Table 5.1: *Classification Results for samples ranged across one octave*

No. PCs	Temporal Envelope (%)	Centroid Envelope (%)
3	69.23	92
4	76.92	92
5	74.36	92

Table 5.2: *Classification Results for samples ranged across the natural pitch range of each instrument*

No. PCs	Temporal Envelope (%)	Centroid Envelope (%)
3	82.94	67.06
4	81.76	78.82
5	73.53	74.14

Classification Over Range of Instrument A network of the same structure was trained with all 2004 RWC samples across the range of the three instruments and then tested with the entire range of MUMS samples. The results are shown in Table 5.2. It is evident from this table that the classification accuracy from the Centroid Envelope data has diminished from that

achieved across the smaller range of pitches. The accuracy from the Centroid Envelope increased from 67.06% to 78.82% by increasing the number of principal coefficients included from three to four, but it did not achieve the 92.31% accuracy obtained using the smaller dataset. This reduction in accuracy is not surprising considering the increase in the search space was from one octave to over eight octaves in the case of the piano. In contrast to this, the results obtained from the Temporal Envelope data were more accurate for the full dataset than for the smaller one-octave dataset. For this larger dataset the accuracy of the classification decreased as more principal components were included. The highest accuracy achieved for the Temporal Envelope was 82.94% from three principal components on the full dataset, as opposed to 76.92% from four components on the smaller dataset.

Conclusions

From the PCA plots obtained in this experiment it can be concluded that of the features examined, the Temporal Envelope and the Centroid Envelope produced the most accurate instrument separation. This agrees with previous literature that has found these features to be very important perceptually (Jensen, 1999). The Residual Envelope and Spectral Envelope did not facilitate instrument clustering when analysed using PCA. Further experiments such as those explained in the following sections investigate more features in this manner. The results here have shown that the most accurate classification was obtained using the Centroid Envelope data across the one-octave range. Increasing the range decreased the accuracy in classification but still gave encouraging results for pursuing classification across the physical range of instruments. The accuracy of the classifier was not automatically decreased by increasing the pitch range examined. In the case of the Temporal Envelope data the accuracy of the classifier was higher across the range of the instrument than when limited to the one-octave range. A number of preceding studies in this area have purposely constricted the range of notes examined so that only a common pitch range is studied across the instruments. These results show that widening the search space to a more realistic range may increase classification accuracy as a more varied range of notes are presented to the classifier, thus making it easier for it to recognise new samples. Hence, in further experiments, classification is performed across the actual physical range of each instrument.

5.3.2 Mel-frequency Cepstral Coefficients

The experiment in the previous section examined a limited number of features for classifying musical sounds. This section uses the same methods of classification in exploring the use of another feature in musical instrument identification — that of Mel-frequency Coefficients (MFCCs). Until recently the field of audio research has largely been dominated by speech analysis rather than its musical counterpart. This is most likely due to the practical uses of speech analysis in modern technology such as voice recognition or security systems. It is not surprising then that many researchers in musical analysis would look to the features and methods employed in speech analysis when examining musical tones. MFCCs have been used extensively in speech analysis over the past few decades (O'Shaughnessy, 1987) and have more recently received attention in music analysis (Eronen, 2001). This experiment attempts to distinguish between musical instruments using only MFCCs and looks at how many of these coefficients are necessary and useful for accurate instrument identification. A number of studies have looked to MFCCs in sound identification. De Poli and Prandoni (1997) used MFCCs in their study of timbre space. Brown (1999) distinguished between oboes and saxophone sounds by calculating cepstral coefficients and applying a *k-means* algorithm to form clusters. Eronen and Klapuri (2000) included MFCCs as one of their features in examining a wide range of orchestral instruments.

As discussed in Chapter 3, Logan (2000) examined some of the finer points of the MFCC in music analysis as opposed to speech analysis and determined that it is useful in this domain. The MFCCs were calculated as described previously in Chapter 3. As with the features in the previous experiment described above, the MFCCs are here calculated across the range of each note.

Results

PCA was used on the time-varying envelope of each mel-coefficient to reduce the amount of data necessary to represent each coefficient. The results compared were from changes in

- the number of MFCCs used
- the number of principal components used to represent each MFCC

Each experiment was conducted across the entire pitch range of each instrument. The MLP used for this experiment was again created in Matlab with

three layers of neurons containing 50 neurons in the first layer, 18 in the second and 15 neurons in the third layer again with three output neurons. As in the previous experiment it was trained with a momentum constant of 0.1 and a learning rate of 0.95. It was batch trained using the `trainbfg` function, with a goal of 0.001 to a maximum epochs of 400.

Results From the First Three Principal Components As before, it was possible to visualise the separation obtained from each of the mel-coefficients by plotting the scores from the first three principal components of each individual mel-coefficient. One such plot for the second mel-coefficient can be seen in Figure 5.13 in which clustering of each instrument can be observed. Similar plots can be created for the other MFCCs. Initially classification was performed on the first three principal components calculated on each coefficient. A preliminary experiment was run to determine the range of MFCCs to examine. This involved training and simulating the network once on the first three principal components from 2-12 MFCCs to get an indication of how well it might perform. The results of this are displayed in Figure 5.14. These results indicate that for high recognition accuracy, at least 6 MFCCs should be used.

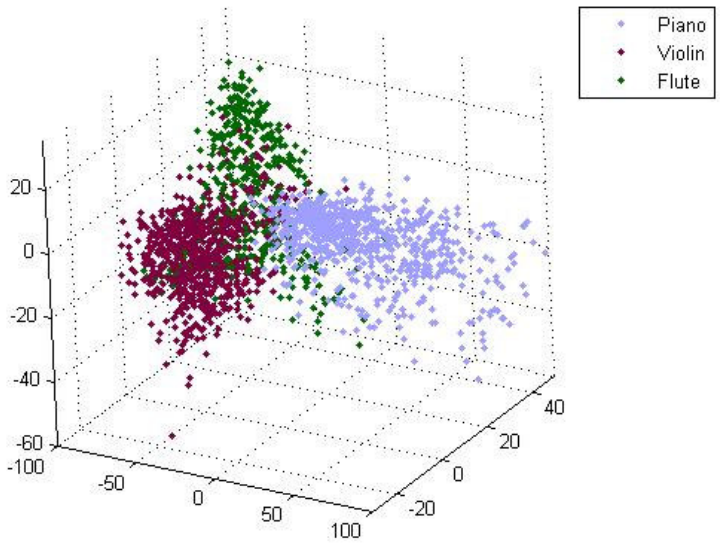


Figure 5.13: *Plot of the first 3 principal components of MFCC2 for the 3 instruments*

To obtain a more reliable result the network was trained and tested 10 times using the first 6 to 16 MFCCs. The average of these test results can be seen in Figure 5.15. These results indicate that once more than 10 MFCCs

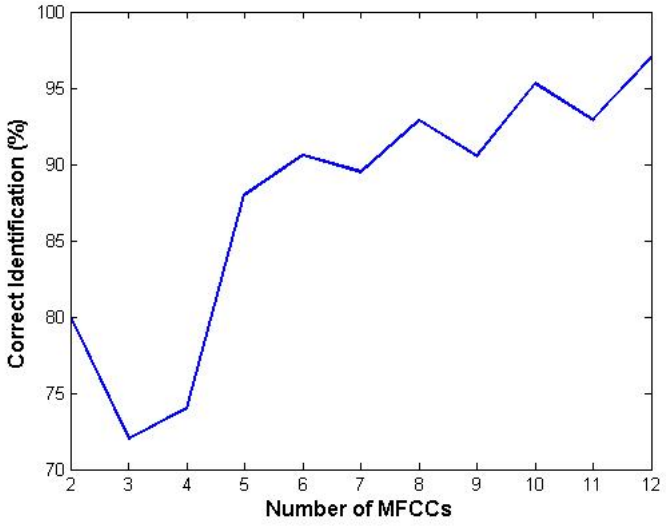


Figure 5.14: *Preliminary results over 1 run for classification accuracy for network trained on 2 to 12 MFCCs*

were used, the recognition results were consistently high. Using 15 MFCCs gave the highest recognition rate of 94.59%. It is worth noting however that using 12 MFCCs gave almost as high an accuracy of 94.35%. This may be of interest as the more MFCCs used in such a system, the more computationally expensive the calculations become.

Results From More Principal Components As in the previous experiment, more principal components for each mel-coefficient were included in the experiment to see if this would increase the accuracy of the classifier. Test results for a network trained on the first three, four and five principal components of the first 11 to 16 MFCCs are shown in the bar chart in Figure 5.16. These results clearly indicate that for each number of MFCCs used, including four principal components increased the accuracy of the classification. Including the fifth, on the other hand, reduced the accuracy of the result. As discussed in relation to the Temporal Envelope in the previous experiment, this may be due in part to the unintuitive way in which PCA reduces the data, possibly incorporating frequency or dynamic aspects that are not specific to each instrument. From this bar chart it can be seen that using the first 4 principal components from 15 MFCCs resulted in the most accurate classification of 95.88%. This is an encouraging result as this classifier is based only on MFCCs.

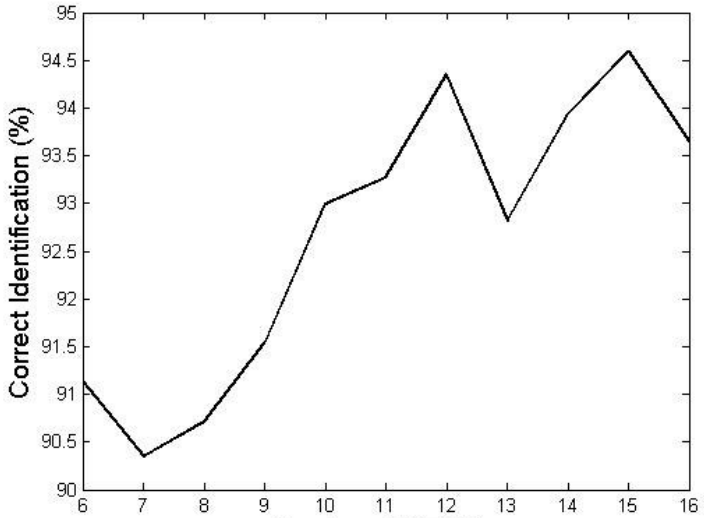


Figure 5.15: *Classification results, averaged over 10 runs, for network trained on 6 to 16 MFCCs*

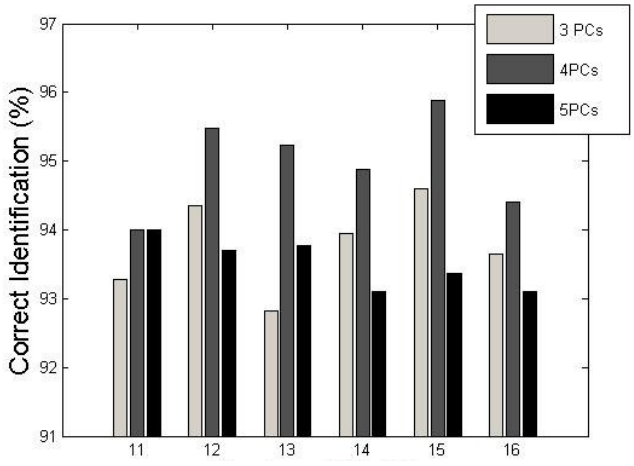


Figure 5.16: *Comparison of results for different number of principal components of 11 to 16 MFCCs*

Conclusions

This experiment demonstrated that MFCCs are a powerful tool in musical instrument identification. Using only this feature an accuracy rate of 95.88% was achieved in identifying novel sounds across the natural range of any of the three instruments, Piano, Violin and Flute. From the results it can be concluded that for high recognition accuracy at least 10 MFCCs should be used. For each number of MFCCs included, it was observed that taking 4 principal components gave the best classification accuracy and that the highest result was obtained from using 15 MFCCs. Thus we can conclude that future classification experiments of this type involving MFCCS should ideally use 15 MFCCs. This optimum number of MFCCs can be combined with more spectral and temporal features to create a more robust classifier. The results from this experiment were presented in Loughran et al. (2008c).

5.3.3 Feature Combination

This experiment combines the features used in the previous two experiments along with a selection of other features used in previous studies on instrument identification. The features incorporated here were the Temporal Envelope and Centroid Envelope as described in Section 5.3.1 and the MFCCs as described in Section 5.3.2 above. These were combined with the spectral features of *Inharmonicity*, *Spectral Irregularity* and *Number of Peaks*. These features were calculated as described earlier in Chapter 3. All features were normalised before being inputted to the MLP.

Results

Data reduction was again performed on the Temporal Envelope, Centroid Envelope and MFCC data using PCA as described in the previous experiments. As determined in the previous experiments, the first four principal components of each feature were used. This reduced data was then combined with the other calculated features and presented to a MLP for training. The network used for identifying all of these features updated its weights with a learning rate (ν) of 0.1 and a momentum constant (α) of 0.95. It was batch trained with the Resilient Backpropagation Algorithm `trainrp`. This algorithm is a variation on the Backpropagation Algorithm that modifies the update-weight values for each weight according to the error function (Riedmiller and Braun, 1993). It was trained to a goal of 0.001 up to a maximum number of epochs of 1000. With this set up it was found that a network with

three layers containing 57, 28 and 15 neurons respectively would be sufficient to train the dataset.

Table 5.3 displays the results from the combinations of the four principal components from the Temporal Envelope, Centroid Envelope and the first 15 MFCCs. Again this table indicates the percentage of accurate classifications of test data from a network trained on these features. Each column of this table represents an experimental set-up — each 'X' indicates that this feature was used in this particular run of the experiment. These results indicate that as individual features, the MFCCs resulted in the most accurate classification, but also that more accurate results could be obtained by combining features. The highest result of 99.41% accuracy was obtained from using all three features. The rest of the features were combined with this best result to determine if it could be improved further. The classification results for these features are shown in Table 5.4. It can be seen from these results that rather than increasing the accuracy of the classifier, including these features actually decreased performance. Regardless of which combination of these features was used with the three initial features, the accuracy was somewhat reduced. This is surprising, as these features have been used in numerous previous studies. Features such as Inharmonicity however are dependent on an accurate fundamental frequency estimation, which as described in Chapter 2 may prove to be problematic for extreme pitched samples such as those included in this experiment.

Table 5.3: *Classification results from training on the Temporal Envelope, Centroid Envelope and MFCC*

Feature							
Envelope	X			X	X		X
Centroid		X		X		X	X
MFCC			X		X	X	X
% Correct:	84.71	78.82	94.71	91.76	95.88	95.29	99.41

An important point to note regarding these features is that they do not all provide the MLP with equal amounts of data for classification — some features are more computationally expensive than others. Inharmonicity, Spectral Irregularity and Number of Peaks all have only one data value each whereas the Temporal Envelope and the Centroid each have four values for every sound sample. The MFCCs on the other hand have four principal component values for each of the 15 coefficients. This gives 60 data values for each sound sample for this feature alone. Because of this, the experiment was

Table 5.4: *Classification results from training on the Temporal Envelope, Centroid Envelope and MFCC combined with Inharmonicity, Spectral Irregularity and Number of Peaks*

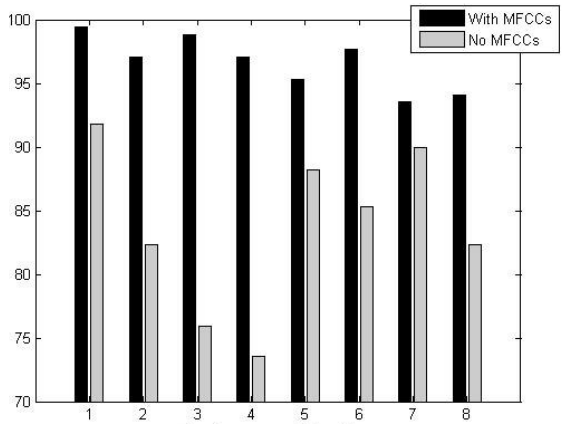
Feature							
Envelope	X	X	X	X	X	X	X
Centroid	X	X	X	X	X	X	X
MFCC	X	X	X	X	X	X	X
Inharmonicity	X			X	X		X
Spec. Ir		X		X		X	X
No. Peaks			X		X	X	X
% Correct:	97.06	98.82	97.06	95.29	97.65	93.53	94.12

run again without the computational expense of the MFCCs, to determine how accurate the classifier could be without them. The results are shown in Table 5.5. It is clear from these results that the MFCCs are very important in instrument identification. None of the combinations come close to the accuracy of those achieved with the MFCCs present. A comparison of the results achieved from the various feature combinations both with and without the MFCCs present is illustrated in the bar chart in Figure 5.17.

Table 5.5: *Classification results from training on just the Temporal Envelope and the Evolution of the Centroid combined with Inharmonicity, Spectral Irregularity and Number of Peaks*

Feature							
Envelope	X	X	X	X	X	X	X
Centroid	X	X	X	X	X	X	X
Inharmonicity	X			X	X		X
Spec. Ir		X		X		X	X
No. Peaks			X		X	X	X
% Correct:	82.35	75.88	73.53	88.24	85.29	90	82.35

Classification of Specific Instruments The above classification results are averaged across the three instrument sets. This section details the classification accuracy for the individual instruments. The training and test sets were run on all features as above. The classification results of the individual instrument are illustrated in the bar chart in Figure 5.18. This indicates that regardless of which feature was used, the violin is consistently the least accurately classified instrument. This lack of clarity in discerning the violin would indicate that the tone or timbre of the violin is somewhat 'in between'



(a) Bar Chart

X-label	Features Included
1	Env, Cent
2	Env, Cent, Ih
3	Env, Cent, SIR
4	Env, Cent, #P
5	Env, Cent, Ih, SIR
6	Env, Cent, Ih, #P
7	Env, Cent, SIR, #P
8	Env, Cent, Ih, SIR, #P

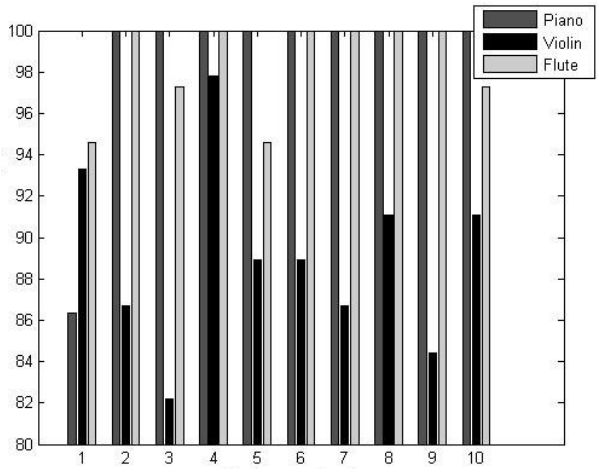
(b) Feature Set-up

Figure 5.17: *Comparison of classification accuracy of feature combinations with and without MFCCs*

the timbre of the other two instruments. This conclusion is reflected from the PCA plots of the Temporal and Centroid envelopes in Figure 5.7 and Figure 5.10. In both of these plots, the red violin samples overlap the areas of the piano and flute samples more than the violin and flute overlap each other. The results show that the piano is the most accurately classified instrument. It can be seen from Figure 5.18 that the piano is recognised with 100% accuracy in every feature combination apart from those that do not include the MFCCs. The flute is the next most accurate with it again achieving a 100% correct recognition rate for several of the feature combinations. Classification of the violin, however, rarely exceeds a recognition accuracy of 95%. This result encourages the inclusion of more instruments in further experiments to determine which other instruments are difficult to classify.

Conclusions

The results from this experiment indicate that of all the features examined here, the MFCCs are the most important for accurate musical instrument identification. On the other hand the Inharmonicity, Spectral Irregularity and Number of Peaks were not found to be of significant benefit in a classifier such as this. Finally it was determined that of the three instruments examined, the piano was the easiest to classify, followed by the flute with the violin being the most difficult to classify. The results from this experiment were presented in Loughran et al. (2008b). Many features other than those looked at here have been included in numerous studies undertaken in the development of an automatic instrument identifier. As can be seen from this



(a) Bar Chart

X-label	Features Included
1	Env, Cent
2	Cent, MFCC
3	Env, MFCC
4	Env, Cent, MFCC
5	Env, Cent, MFCC, Ih
6	Env, Cent, MFCC, Ih, SIR
7	Env, Cent, MFCC, Ih, #P
8	Env, Cent, MFCC, Ih, SIR, #P
9	Env, Cent, MFCC, #P
10	Env, Cent, MFCC, SIR, #P

(b) Feature Set-up

Figure 5.18: *Comparison of individual instrument classification using all features*

experiment, all of these features may not be necessary for such classification tasks. Incorporating more and more features into a system without justifying their inclusion is not necessarily guaranteed to provide an improvement in any classification results. As can be seen from the literature, the list of features that may be included is quite extensive, which means that the number of different possible combinations of features which one may use is extremely large. It would be impractical to attempt to implement each such combination. It would instead be beneficial to develop a system to find an optimum or best selection of features to include in such tasks. Much of the further work in this thesis is intended towards finding such a system.

5.4 Conclusions

This chapter described a data reduction technique (PCA) and a classification technique (MLP), both of which have been used extensively in musical instrument recognition experiments. The calculations of the PCA algorithm were examined and details were given as to how it is implemented in this thesis. Artificial Neural Networks were examined as a type of classifier. After discussing one of the simpler types of ANN, the perceptron, we looked at an MLP, a more complex network, capable of representing and hence learning more complicated datasets. A summary of the Backpropagation algorithm was presented, which was used to train the MLP. Finally three experiments described in Loughran et al. (2008a,b,c) undertaken with these methods were discussed. These experiments described training an MLP with a variety of

features reduced using PCA. Overall, it was found that the MFCCs were the most effective feature to use for such classification experiments. The Temporal Envelope and Centroid Envelope were also found to be useful. However not all features incorporated into these experiments proved beneficial to the performance of the system.

The classification results reported in this chapter were very high — over 99% in some cases. This is higher than the accuracy of previous similar instrument classification studies as reported in Chapter 4. The main difference between this study and those from the literature is in the large number of samples used over just three instruments. It was proposed at the end of Chapter 4 that to train a robust, accurate classifier, many samples at different pitches and playing styles were required. The high accuracy achieved in this chapter demonstrates this point. In particular it was shown that broadening the range of pitches examined by the classifier may increase classification accuracy. The main limitation of these experiments, however, was in the choice of instruments. Limiting the number of instruments this severely limits the potential of the classifier. Further experiments expand the potential of the classifier by introducing a further two instruments. It is expected that the accuracy of subsequent classifications would thus drop — as the timbre of further instruments would be 'in between' those of the current instruments, much as the timbre of the violin was found to be in between that of the piano and flute.

The main objective in this thesis is to analyse and understand timbre in an effort to create a robust musical instrument identifier. The methods used in this chapter have been shown to have the necessary reductional and classification power to be used in such a development. Where such a system falters however, is in that it has no method of selecting the features best incorporated within itself. Any amount of features may be used as inputs to a MLP; more and more features may be added, but are they of use? The experiment reported above and in Loughran et al. (2008b) has shown that adding extra features does not necessarily help and can be detrimental to the performance of the system. In addition to this, the inclusion of an increasingly large amount of data to a classifier such as this will ultimately necessitate a more complex network incorporating more neurons and hence more computations leading to longer learning times. While longer learning times may be acceptable for a marked improvement in performance of the network, they are far from ideal considering no improvement is guaranteed to be achieved. This chapter only examined a small number of features, but as

described in Chapter 4 many more features have been included in previous studies. Implementing the experiments in this chapter with all combinations of features is impractical as it would be too time consuming and may not lead to more accurate results. As such, we look to methods of improving the selection of features chosen as input to such a network. A prominent area within data mining used for optimisation of parameters is the field of Evolutionary Learning. The following chapters looks at two methods from this field, namely Genetic Algorithms and Genetic Programming, and their application in optimising features for a system such as this one.

Chapter 6

Genetic Algorithm Experiments

6.1 Introduction

In the previous chapter, MLPs were used to classify musical samples based on a number of different timbral features. These MLP classifications are similar to previous attempts at classifying musical samples such as those described in Chapter 4. These previous studies varied widely in their choice and in the number of features used in such classification systems. However, it was also seen from experiments in Chapter 5 that increasing the amount of data available to such systems does not necessarily improve their classification accuracy. Increasing the amount of data provided to any classification system increases the number of computations such systems must make, making classification increasingly complicated. Although increasing numbers of features have been included in classification studies, no basis has been given for including this large number of features to distinguish a musical instrument. We have discussed in Chapter 2 and Chapter 3 how the recognition of a musical instrument is largely based on its timbre. Studies in describing timbre have used multi-dimensional scaling techniques to *reduce* the number of dimensions or descriptors for timbre and to assign physical or psychological attributes to these dimensions. This attempt to maintain a low number of descriptors in empirical work is in direct contrast to the increase in features used in machine learning classification techniques. It is proposed in this chapter to bridge the gap between these two areas by searching for the optimum selection of timbral features before incorporating them into a classifier. This approach has three distinct aims:

- to create an improved musical instrument identifier
- to reduce the complexity of a musical instrument identifier by reducing

the amount of data used by it

- to gain a deeper understanding of timbre by determining which timbral features are most important in musical instrument identification

In this chapter, the optimum feature selection is made using a Genetic Algorithm (GA). Section 6.2 introduces the data used in experiments in this chapter and decribes the way in which this data is organised. GAs are a specific type of Evolutionary Computation technique, introduced in Chapter 4. A more detailed introduction to the workings of a GA and how it is incorporated into the experiments here is given in Section 6.3. Section 6.4 describes the fitness function used by the GA in these experiments. The fitness function is an integral part of the system, which dictates how the GA performs. This section details the meaning of *good* fitness and the development of the function using a scaled-down version of the problem. Three fitness functions are derived, each of which is applied to the full set of data in Section 6.5. The results are discussed in terms of the genomes evolved by the GA and the use of these genomes in classification experiments using an MLP. One of these fitness functions is then used on a scaled up set of experiments in Section 6.7. Finally, Section 6.8 offers some conclusions on the results obtained.

6.2 Data

As in experiments in the previous chapter, the samples used here were taken from the RWC Music Database (Musical Instrument Sound) (Goto, 2004) and the MUMS database (Opolko and Wapnick, 1987). However, unlike the previous experiments that classified three instruments, these experiments classify five instruments: the piano, violin, flute, trumpet and guitar. The piano, violin and flute samples included were as described in Chapter 5[1]. The RWC samples included two models of trumpets, Vincent Bach and Schilke, sampled at dynamic level *f*, *mf* and *p* both with and without vibrato. This database also included three models of classical guiter, Stafford, Kohno Masaru and Imai Yuichi sampled at the same three dynamic levels. The MUMS database provided a further set of trumpet samples. This gave 616 piano samples, 813 violin samples, 481 flute samples, 394 trumpet samples and 702 guitar samples. Thus the complete data set consists of 3006 sample values.

[1]with the exception that the Bosendorfer piano samples were excluded as, on close aural inspection, a large proportion of the high pitched samples were found to be of low quality.

Each of the features described in Chapter 3 was calculated on each of these 3006 samples to create the data set used in these experiments. These features are listed below in Table 6.1. A full list of the feature values in the order that they are used in the evolutionary experiments in this and the following chapter is given in Appendix B. Although a number of these features result in one data point per feature, all features based on time-varying envelopes contain multiple data points. As before in the experiments described in Chapter 5 the data from these features was reduced using PCA. The experiments in Chapter 5 found, in general, that the first four principal component values were the most useful in musical instrument classification. Hence the first four principal components for the Temporal, Residual and Spectral Envelopes, Centroid Envelope and the envelopes of each of the first 16 MFCCs were calculated and included as data points. This resulted in a total of 95 data points to describe all features for each of the 3006 instrument samples.

Table 6.1: *List of temporal and spectral features included in the experiments in this chapter*

Temporal	**Spectral**
Temporal Envelope	Spectral Envelope
Residual Envelope	Number Spectral Peaks
Attack Time	Irregularity
Attack Slope	MFCCs (1-16)
Centroid Envelope	Inharmonicity
Zero-Cross Rate	Centroid
Number Onsets	Spread
Onset Distance	Skew
	Kurtosis
	Regularity
	Rolloff
	Brightness

6.2.1 Cross-Validation Sets

This 3006 by 95 data-set is quite large in size. Evolutionary algorithms can be computationally expensive when evolving a solution based on a large amount of data. To reduce the computation time and to provide a method of validating the results, the data was split into 10 sub-sets. The original set was ordered as per instruments (piano followed by violin etc.). To ensure an even distribution of the instruments among the 10 sets, every set was iteratively filled with data from the tenth consecutive sample until all sets contained the

data for 300 samples[2]. The remaining six samples were added to Set1. In parallel to this, a target set for each validation set was created so that the order (instrument) of samples within each set is known. In these target sets, 1 represents a piano, 2 a violin, 3 a flute, 4 a trumpet and 5 a guitar. The GA evolves a solution, or *genome*, for each of these data sets. These genomes may then be compared to test the validity of the results.

6.3 Genetic Algorithm

GAs, like all evolutionary techniques, are based on the principal of *natural selection*. Such techniques are modelled on the process of biological evolution, whereby a solution is modified and ideally improved over many generations from a population of solutions. GAs 'evolve' an optimum solution for a given problem by searching a population of possible solutions. The initial population contains a number of randomly created solutions to the problem under investigation. In binary GAs these solutions consist of binary strings whereas in floating-point GAs (as used here) each solution consists of a vector or string of floating-point values. A measure of how successful each individual is at solving the given problem is measured as its *fitness* according to a *fitness function*. Successive generations are produced from this initial population using a combination of three operators (Goldberg, 1989):

- **Selection or Reproduction** — individual strings succeed into the following generation according to their fitness. Thus individuals with better fitness are more likely to contribute to one or more offspring in the next generation.
- **Crossover or Recombination** — two (parent) strings are combined to produce two new (children) strings in the next generation, both of which contain traits of the parent solutions.
- **Mutation** — one or more aspects of an individual is changed to create a new individual in the next generation.

There are a number of methods used for the selection process. One of the original methods is *roulette-wheel* or *proportional* selection. Using this method, the population is represented by a roulette wheel where each individual occupies a space on the wheel proportional to its fitness. With multiple spins of the wheel, the probability of an individual being selected is given by the ratio

[2] *sample* in this sense refers to an individual instrument sample or note

of its fitness to the fitness of the rest of the population. *Tournament* selection is another method of selection in which a set of k individuals are selected from the population and the fittest individual among this set is considered for selection. The selection pressure may be controlled with an appropriate selection of k. In *Rank* selection the population is ordered in terms of their fitness and individuals are selected according to their rank on this list rather than their absolute fitness (Affenzeller et al., 2009).

The combination of selection with both crossover and mutation are essential for any GA. Crossover combines the best elements from individuals to create potentially superior children and mutation adds new elements into the population, preventing the population from converging prematurely. The combination of these operators ensures that the best elements of individuals survive from one generation to the next, while maintaining diversity within the population. To ensure that the best individuals are not destroyed using these operators however, in each generation the individuals with the best fitness are passed directly to the next generation through the process of *elitism*. The behaviour of the GA over a number of generations is controlled by the user by setting parameters at the beginning of each evolutionary run. Such parameters may dictate the size of the population, number of generations or probability of each operator. The selection of these parameters is discussed later in this chapter.

6.3.1 Optimisation of Features Using GA

GAs have been shown to be powerful when used for large-scale feature selection (Kudo and Sklansky, 2000; Siedlecki and Sklansky, 1993a). A GA is used here to optimise the selection of features used in a neural network classifier. Each individual or *genome* is a string containing 95 floating point numbers. Each element of the genome, or *gene*, represents one of the 95 features included in this experiment. This genome is multiplied by the 95 feature data values of each musical sample in a *genotype to phenotype* matching. Thus the genome acts as a weighting vector dictating how much of each feature is included in the data. This ranges from 0 which means this feature is *not* included, to 1 whereby the maximum amount of the feature is included in the data. The fitness of each genome is then calculated according to the clustering of this multiplied feature data set — as described in the next section. The GA experiments in this chapter were implemented using the Genetic Algorithm and Direct Search Toolbox in Matlab (MATLAB7, 2006).

6.4 Fitness Function

The GA selects an individual based on its fitness, thus the fitness function defines how the GA will behave. In this case we wish to choose the best genome to optimise a set of features for training an MLP. All of the fitness functions developed in this chapter use minimising fitness functions: the smaller fitness values correspond to more successful genomes. It was seen in Chapter 5 that data which separates distinctly is more easily classified by an MLP. Thus we measure how well each set of multiplied data or phenotype forms distinct clusters according to instrument, as a measure of how successful each genome is as a solution: the more distinctly the data clusters, the better the fitness of the corresponding genome.

To measure the clustering of the data, it is first reduced in dimension using PCA. It was already discussed that using PCA can reduce the dimensionality of a data set by concentrating the variance of the data in the lower dimensions. This process has been implemented on a number of the features used in this experiment such as the Temporal Envelope and the Centroid Envelope, as explained in the previous chapter. It is implemented here to measure the clustering of the data in a lower number of dimensions. PCA is applied to this multiplied dataset and the obtained *scores* are noted. The fitness of the given genome is calculated in relation to the clustering of the data from these scores: the more distinct and separated the clusters are, the lower (better) the fitness will be. For distinct clustering or separation of the data, we require that the data forms individual clusters whereby each element of the cluster is of the same instrument type, and also that these clusters are well separated from each other. The fitness function returns a single numerical value as a measure of how well the genome solves this problem. Thus a fitness function must be designed that results in a numerical value that is indicative of this clustering. A measure of how well the data clusters is calculated along with a measure of how far apart each cluster is from each other instrument cluster. The fitness is calculated from the relationship between these two measures.

To determine the optimum relationship between clustering and separation, the size of the problem was reduced, as in Goldberg's 'little problems' (Goldberg, 2002). The number of samples and number of features were reduced to observe the relationship between these measures.

6.4.1 'Toy' Feature Selection

To develop the details of the fitness function used with the GA, a smaller data set was created on which small-scale experiments could be conducted. This 'Toy Set' consisted of one octave of notes, C4 to C5 played on each of the instruments, resulting in 65 data samples. For these experiments the features calculated on these samples were limited to the following 11 features: Zero-Crossing Rate, Rolloff, Brightness, Number of Onsets, Onset Distance, Attack Time, Attack Slope and the four principal components from the Temporal Envelope. These features were arbitrarily chosen for this set of experiments. Thus the following experiments on developing the fitness function were conducted on a data set of size 65 by 11, rather than the original 3006 by 95 data set. These experiments were run on this data-set with a population of 100 for 200 generations. This set-up ensured that each GA run took less than 10 minutes, allowing ample time for adjustments to be made to the fitness function. As a comparison Figure 6.1 displays the original Toy data set clustered with all features equally present. Clearly this data was already clustering well, unsurprising as the data was limited in scope. The objective of these experiments was to analyse the behaviour of the fitness funtion, rather than find the optimum genome at this point.

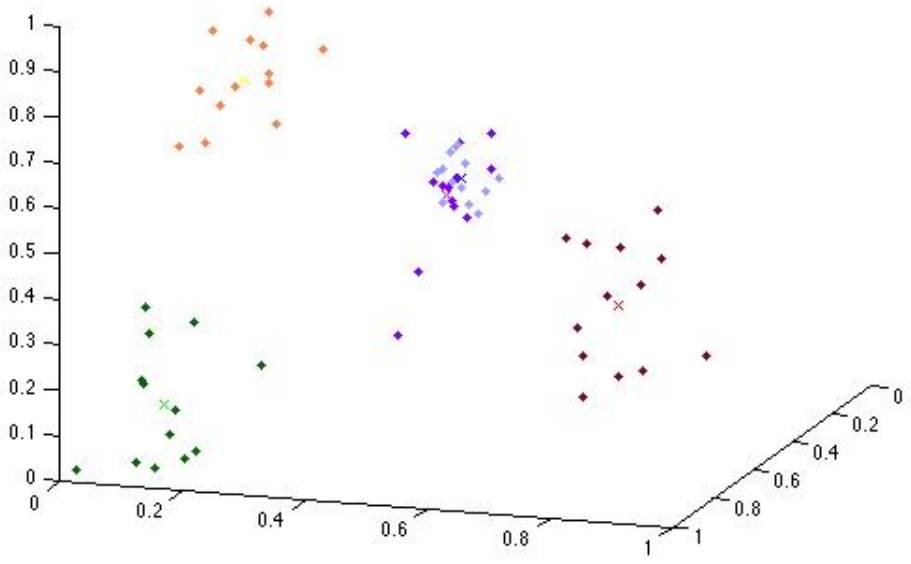


Figure 6.1: *Principal Component Clustering for Original Toy data*

This fitness function was developed and examined according to two distinctions:

- The way in which the clustering and separation measures are calculated, and
- The relationship between the clustering and the separation.

As we are considering multi-dimensional distances, the Euclidean distance is one obvious measure for such distances. The following sections consider this measure along with another *Dimensional* measure of calculating the fitness of each genome. For ease of illustration, the clustering of the scores of the data in this Toy Problem is limited to three dimensions.

6.4.2 Calculation of Fitness: Method 1, Euclidean

This Euclidean Fitness is a measure of the relationship between the average Euclidean distance of each sample to the mean of the cluster to the Euclidean distance from the mean of the cluster to the mean of all the other clusters. The scores calculated are grouped into clusters according to a target vector indicating which instrument they correspond to. Thus the piano, violin, flute, trumpet and guitar clusters were formed from the first three values of the scores calculated for these instruments. The mean of each cluster was calculated. The clustering value for each instrument can then be calculated from its *spread* measure and *separation* measure. The spread measure (Δ) was calculated as the sum of the 3-dimensional Euclidean distance from each instrument point to the mean of the cluster, divided by the number of instrument samples. Thus for the piano cluster:

$$\Delta_p = \frac{\sum_{i=1}^{i=p} \sqrt{(C_i(x) - M_p(x))^2 + (C_i(y) - M_p(y))^2 + (C_i(z) - M_p(z))^2}}{p} \quad (6.1)$$

where C is the piano cluster, $C_i(x)$ is the x-component of the i^{th} piano sample, M_p is the piano mean and p is the number of piano samples.

The separation measure (Γ) is similarly calculated as the sum of the Euclidean distances from the mean of a cluster to the mean of each other instrument cluster, divided by the number of other clusters. Thus the separation

for the piano cluster is calculated as:

$$PVDist = \sqrt{(M_p(x) - M_v(x))^2 + (M_p(y) - M_v(y))^2 + (M_p(z) - M_v(z))^2}$$
$$PFDist = \sqrt{(M_p(x) - M_f(x))^2 + (M_p(y) - M_f(y))^2 + (M_p(z) - M_f(z))^2}$$
$$PTDist = \sqrt{(M_p(x) - M_t(x))^2 + (M_p(y) - M_t(y))^2 + (M_p(z) - M_t(z))^2}$$
$$PGDist = \sqrt{(M_p(x) - M_g(x))^2 + (M_p(y) - M_g(y))^2 + (M_p(z) - M_g(z))^2}$$
$$\Gamma_p = \frac{PVDist + PFDist + PTDist + PGDist}{4}$$

where M_p is the mean of the piano cluster, M_v is the mean of the violin cluster, M_f is the mean of the flute cluster, M_t is the mean of the trumpet cluster and M_g is the mean of the guitar cluster. The Euclidean distance from the mean of the piano cluster to each other cluster is calculated and averaged over 4. Similar measures of Δ and Γ were calculated for the other instrument clusters.

For good separation of the instrument clusters we require the spread measure of each cluster to be small and the separation between each cluster to be high. Figure 6.2 illustrates this point in two dimensions. In this figure the circles represent the average spread of each colour cluster. Unless the data is well clustered, there will be some amount of overlap between the different colours. Thus for distinct clustering of the blue samples, for example, Δ_{blue} must be small in relation to both $\Gamma_{bluetored}$ and $\Gamma_{bluetogreen}$. The function of the GA is to evolve a genome that optimises this clustering relationship when multiplied by the feature data. In each experiment an optimum genome is evolved according to a numeric measure between Δ and Γ for each instrument. The merits of the current fitness function may then be analysed by observing the measurement details and plots of the clusters obtained by multiplying the evolved genome by the data.

Relationship between Δ and Γ

Initially, the fitness was calculated from the relationship of the average instrument spread to the average separation between instruments. Thus a common Δ and Γ were calculated over all instruments and compared. However, this approach did not yield good results. Such a fitness function tended to merely optimise one or two instruments at the expense of the others, yielding a 'good' numerical result which actually did not produce good clustering. This ten-

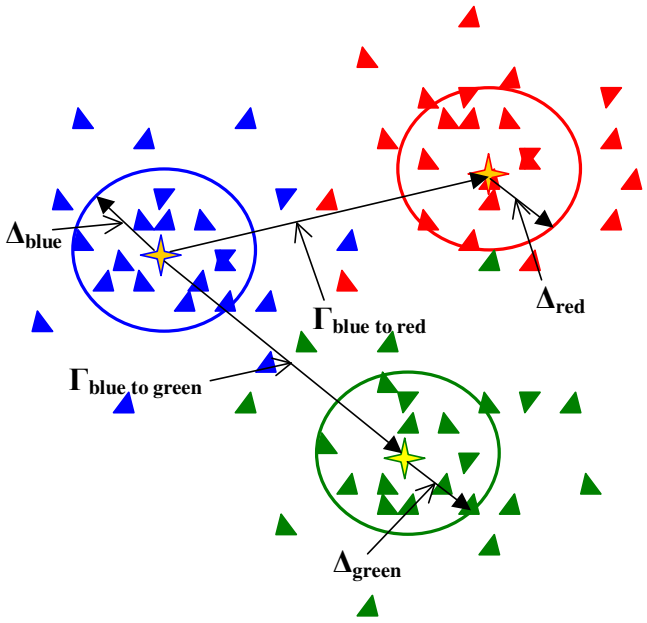


Figure 6.2: *Illustration of clustering of 3 colour groups in 2 dimensions*

dency implied that local minima with very low fitness were causing the GA to converge on a sub-optimum result. It was found, instead, that comparing the spread with the separation for each instrument (Δ_i and Γ_i for all instruments i) produced better results. Thus the fitness function was based on the fitness of each individual cluster.

For each instrument the relationship between the spread measure and the separation measure was examined. As an initial experiment, the genome was optimised by minimising

$$fitness_i = |\Delta_i - \Gamma_i| \tag{6.2}$$

for each instrument i. The plot of the clusters achieved are shown in Figure 6.3. Although this plot does show some clustering, there is much overlap between the various clusters.

For good clustering all samples should to be close to their own mean while at the same time the means of each instrument cluster group should be well separated. Thus ideally Γ should be large *in relation to* Δ. This relationship is enforced by multiplying Δ by a *clustering coefficient*, δ. This forces the value of Γ to be several times (as specified by δ) the magnitude of Δ i.e. the distance between the clusters is forced to be δ times the average distance of

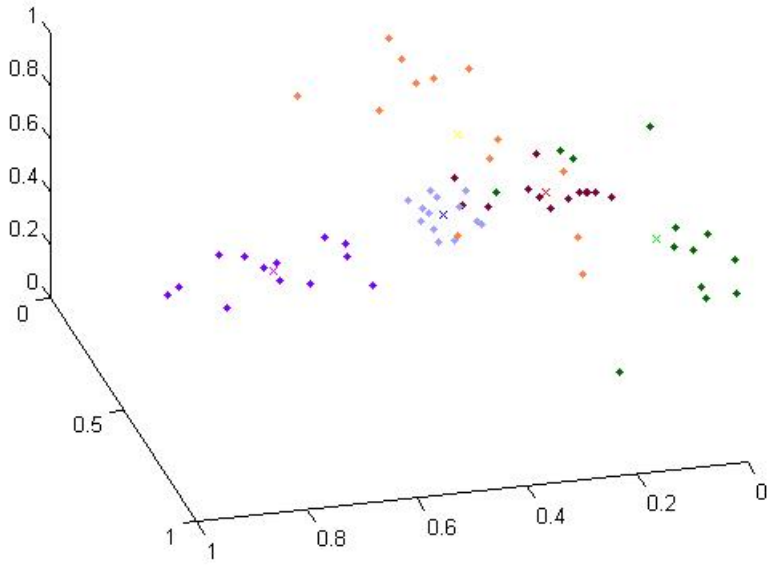


Figure 6.3: *Principal Component Clustering for Fitness function based on fit* $= |\Delta_i - \Gamma_i|$

each sample to the mean of the instrument cluster. Δ is thus decreased in relation to Γ by minimising the fitness according to:

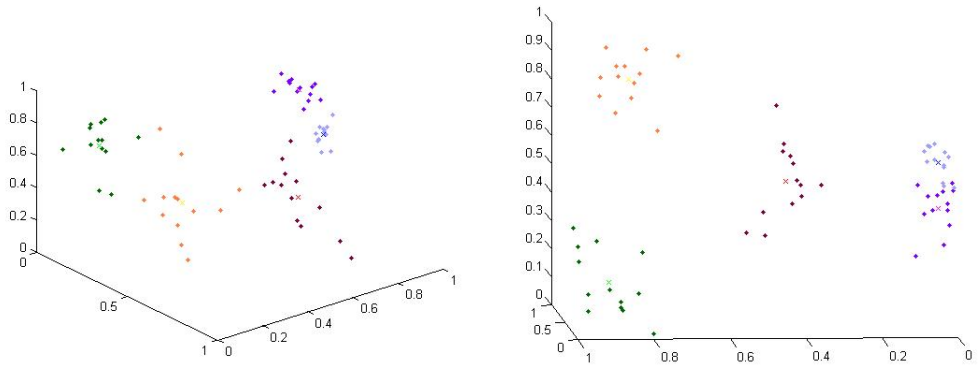
$$fitness_i = |\delta\Delta_i - \Gamma_i| \tag{6.3}$$

This means that in the ideal situation (fitness = 0), the separation would be δ times the spread. Clusters obtained from genomes evolved with δ equal to 5 and 10 are displayed in Figure 6.4. These plots indicate that using δ equal to five offers an improvement in clustering compared to Equation 6.2 which may be improved further by setting δ equal to 10.

Absolute vs. Offset

The absolute value used in the calculation of fitness prevents negative fitness values from occurring. This does not take into account, however, that a negative value may actually indicate a very good fitness. For example if Genome A resulted in $\Delta_A = 1$ and $\Gamma_A = 8$. Then according to the fitness equation:

$$\begin{aligned} fitness_A &= |10\Delta_A - \Gamma_A| \\ &= |10 - 8| \\ &= 2 \end{aligned}$$



(a) δ=5 (b) δ=10

Figure 6.4: *Clustering of instrument groups with varying δ*

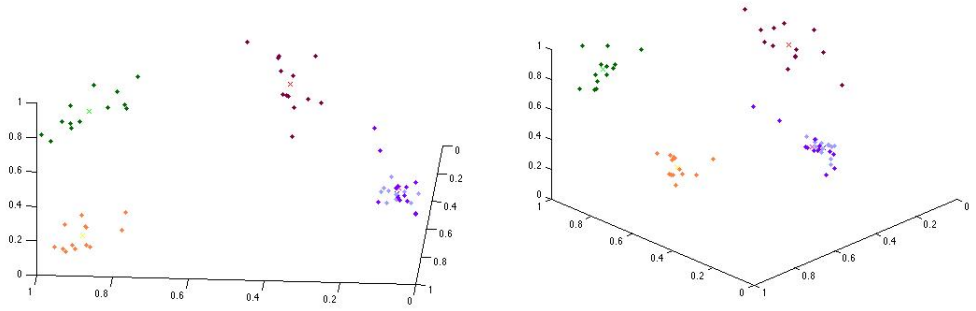
If on the other hand Genome B resulted in $\Delta_B = 0.5$ and $\Gamma_B = 8$. Then the fitness equation gives:

$$\begin{aligned} fitness_B &= |10\Delta_B - \Gamma_B| \\ &= |5 - 8| \\ &= 3 \end{aligned}$$

Hence Genome A is given the lower (better) fitness even though Genome B results in tighter clustering with the same separation. This may be rectified by replacing the absolute measure with the addition of a constant *offset*. In practice, a negative value appeared very unlikely to occur as it would require very tight clustering — particularly in later experiments which contain many more samples per cluster[3]. Nevertheless, an addition of a positive offset value would ensure that should a negative result occur, it would keep its 'good' fitness. The data used in these experiments are all normalised — all values lie between 0 and 1. In taking a Euclidean measure in three dimensions, the furthest two samples may be from each other in this space is $\sqrt{3}$. Hence the square root of the number of dimensions examined was chosen as an offset. As we are dealing with the sum of the distances between five clusters, it is mathematically possible — although practically very unlikely, for a negative value still to occur. The offset was maintained at the square root of the number of dimensions examined, and not a multiple of this, to prevent unnecessarily high fitness values. Plots of the clusters obtained from

[3]It may be noted here that the GA Toolbox can manage negative fitness values as it minimises to negative infinity. It is for ease of analysis that we attempt to keep the values obtained positive

using this offset instead of the absolute measure for δ equal to 5 and 10 are shown in Figure 6.5. It is evident from these plots that the use of an offset value improves clustering.



(a) δ=5 (b) δ=10

Figure 6.5: *Clustering of instrument groups with varying δ with offset value*

From Figure 6.5 it would appear the exact choice of δ does not appear to have a very significant effect on the clustering. To examine these clusters further, we look at the numerical results obtained from each evolved genome. Table 6.2 displays the values of the fitness Δ and Γ for each of the instrument clusters obtained from the genome evolved with δ equal to 1. Similar results for the clusters obtained from the evolved genomes with δ equal to 5 and 10 are shown in Table 6.3 and Table 6.4. Within these results, the most important data is the relationship between Δ and Γ for each instrument. Although the fitness increases with an increase in δ, this is directly due to the multiplication of δ by the positive value Δ. As already stated, good clustering ideally requires Δ to be small and Γ to be large. We can see from these tables that although the numbers vary between instruments, in general Δ reduces somewhat with an increase in δ. This is accompanied however with a slight decrease in Γ. Thus the clustering is getting tighter but the distance between the clusters is also getting smaller. This is due to the fitness equation currently used:

$$fitness = \delta\Delta - \Gamma + \text{offset} \tag{6.4}$$

In these calculations, Γ tends to be significantly higher than Δ i.e. the separation was initially higher than the spread as displayed in Figure 6.1 of the initial data. As the offset is a constant ($\sqrt{3}$ or approximately 1.77) the simplest way to minimise Equation 6.4 would be to maximise Γ. Although this works on this set of data, this equation must be applicable to more general data sets. Hence the equation must be modified to ensure that both Δ and

Γ play a significant role in the fitness calculation. This is ensured with the inclusion of thes clustering coefficient as described above — this clustering coefficient is multiplied by Δ to ensure that it is taken into account along with the larger value of Γ. Thus it is important to multiply Δ by the clustering constant to ensure it is considered in the fitness calculation. The decrease in both Δ and Γ in Table 6.4 indicate that with δ equal to 10, the spread Δ is now playing a role in the calculation of fitness. An increase of δ up to 1000 was implemented but was not found to improve the results significantly. Thus the fitness across all 5 instruments according to the Euclidean distances of the spread and separation of the clusters was calculated according to:

$$Fitness = \sum_{i=1}^{5}(10\Delta_i - \Gamma_i + \sqrt{3}) \tag{6.5}$$

Table 6.2: *Fitness,* Δ *and* Γ *results from genome evolved from: fitness =* Δ - Γ *+ offset*

	Piano	**Violin**	**Flute**	**Trumpet**	**Guitar**
Fitness	1.3187	0.9953	1.0245	0.9394	1.1347
Δ	0.0699	0.1488	0.1619	0.0987	0.1079
Γ	0.4833	0.8856	0.8695	0.8913	0.7053

Table 6.3: *Fitness,* Δ *and* Γ *results from genome evolved from: fitness = 5*Δ - Γ *+ offset*

	Piano	**Violin**	**Flute**	**Trumpet**	**Guitar**
Fitness	1.5547	1.5660	1.6314	1.3835	1.5490
Δ	0.0594	0.1412	0.1509	0.1026	0.1016
Γ	0.4741	0.8720	0.8552	0.8615	0.6908

Table 6.4: *Fitness,* Δ *and* Γ *results from genome evolved from: fitness = 10*Δ - Γ *+ offset*

	Piano	**Violin**	**Flute**	**Trumpet**	**Guitar**
Fitness	1.8014	2.2722	2.1854	2.0032	1.9844
Δ	0.0524	0.1327	0.1259	0.1019	0.089
Γ	0.4551	0.7869	0.8052	0.7476	0.6372

6.4.3 Calculation of Fitness: Method 2, Dimensional

As a comparison to the Euclidean measures discussed above, an alternative *Dimensional* measure was developed for calculating the spread Δ and separation Γ for each instrument cluster. These measures were taken from the spread and separation distances projected onto one dimension and then summed over all dimensions. Thus the spread measure Δ was calculated for the piano as:

$$\Delta_p = \frac{\sum_{i=1}^{p}[\sqrt{(C_i(x) - M_p(x))^2} + \sqrt{(C_i(y) - M_p(y))^2} + \sqrt{(C_i(z) - M_p(z))^2}]}{p} \quad (6.6)$$

where p is the number of piano samples, C is the piano cluster, $C_i(x)$ is the x-component of the i^{th} piano sample and M_p is the mean of the piano cluster.

The separation measure Γ is similarly calculated from the projectional dimensional distances between the means of each cluster. Thus for the piano cluster, Γ is calculated as:

$$\Gamma_p = \sum_{d=1}^{3} \frac{\sqrt{(M_p^d - M_v^d)^2 + (M_p^d - M_f^d)^2 + (M_p^d - M_t^d)^2 + (M_p^d - M_g^d)^2}}{4} \quad (6.7)$$

where again M_p is the mean of the piano cluster, M_v is the mean of the violin cluster, M_f is the mean of the flute cluster, M_t is the mean of the trumpet cluster and M_g is the mean of the guitar cluster and d is the dimension (x, y and z).

The relationship between Δ and Γ was investigated in a similar manner as with the Euclidean distances. A plot of the clusters obtained from using δ again equal to 1, 5 and 10 are shown in Figure 6.6. The corresponding results are displayed in Table 6.5, Table 6.6 and Table 6.7. Again it can be seen that the increase in the value of δ from 1 to 10 causes the Δ value for each instrument to become more significant in the calculation of the fitness. This in turn causes a slight decrease in the value of Δ along with a slight decrease in the value of Γ (as with the Euclidean fitness). Again the value of δ was chosen to be 10. Thus the relationship between Δ and Γ for the Dimensional distance is calculated the same as that of the Euclidean distances:

$$Fitness_{dim} = \sum_{i=1}^{5}(10\Delta_i - \Gamma_i + \sqrt{3}) \quad (6.8)$$

Table 6.5: *Fitness, Δ and Γ results from genome evolved from Dimensional Fitness: fitness = Δ - Γ + offset*

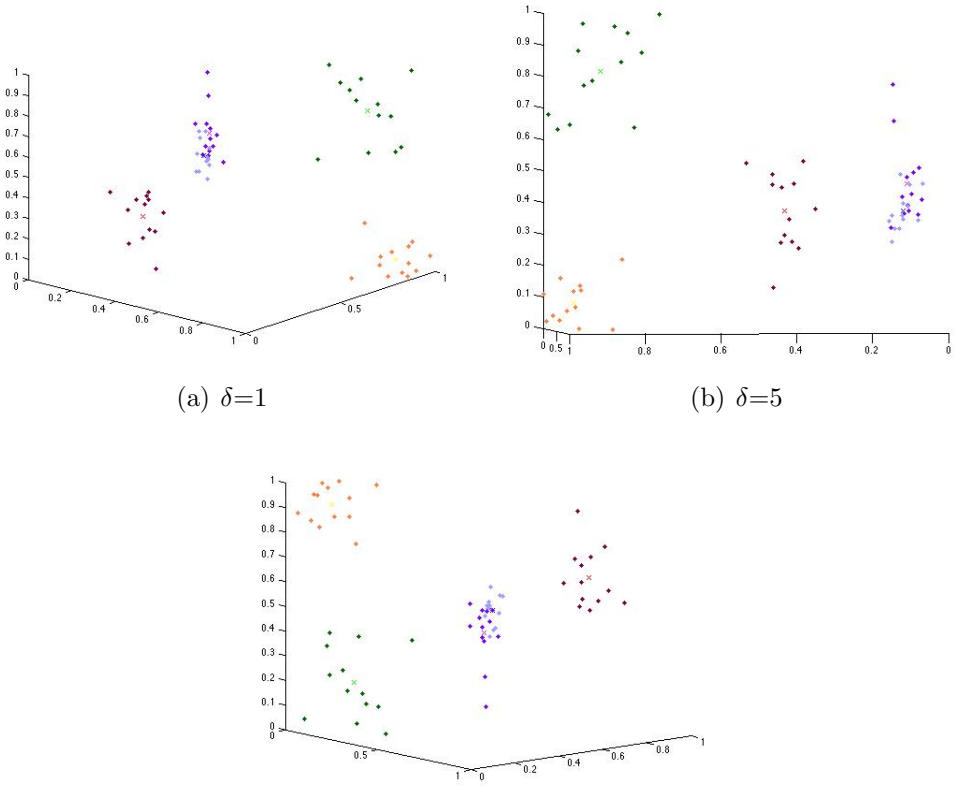
	Piano	**Violin**	**Flute**	**Trumpet**	**Guitar**
Fitness	1.1382	1.0877	1.0796	1.0293	1.1423
Δ	0.0337	0.0696	0.0837	0.0522	0.0548
Γ	0.6276	0.7140	0.7361	0.7549	0.6446

Table 6.6: *Fitness, Δ and Γ results from genome evolved from Dimensional Fitness: fitness = 5Δ - Γ + offset*

	Piano	**Violin**	**Flute**	**Trumpet**	**Guitar**
Fitness	1.2754	1.3672	1.4131	1.2364	1.3538
Δ	0.0339	0.0694	0.0833	0.0518	0.0530
Γ	0.6260	0.7116	0.7353	0.7545	0.6434

Table 6.7: *Fitness, Δ and Γ results from genome evolved from Dimensional Fitness: fitness = 10Δ - Γ + offset*

	Piano	**Violin**	**Flute**	**Trumpet**	**Guitar**
Fitness	1.4265	1.7037	1.8185	1.5186	1.6139
Δ	0.0316	0.0679	0.0816	0.0532	0.0521
Γ	0.6213	0.7072	0.7293	0.7454	0.6388



(a) δ=1 (b) δ=5

(c) δ=10

Figure 6.6: *Clustering of instrument groups with varying δ with offset value*

6.4.4 Calculation of Fitness: Method 3, Euclidean-Dimensional

As a further measure of fitness, a combination of the above two measures were considered as a *Euclidean-Dimensional* fitness function. Using this method the spread measure, Δ, is calculated from the Euclidean measures of the cluster as in Equation 6.1. In contrast to this the separation measure, Γ, is calculated from the projectional dimensional distances between the means of each cluster, according to the Dimensional separation in Equation 6.7. Although the concept behind the measuring of these two values is different, they each result in a numerical distance that can be easily compared. The Euclidean-Dimensional fitness is then calculated as the relationship between between this Euclidean spread and Dimensional separation according to Equation 6.5 derived earlier.

The above experiments were conducted to determine the details of the fitness function to use for evolving the desired feature weight-vector or genome. Limiting the scope of the data and the dimensions observed allowed an ex-

amination of the workings of the genome evolution on a little problem. Now that the ideal relationship between the spread and separation of the clusters has been decided on, the problem may be scaled up to include the full set of features on the full data set. The following section implements the GA on the full data set using the Euclidean, Dimensional and Euclidean-Dimensional methods derived in this section.

6.5 Full Dataset

The full set of features described in detail in Chapter 3 and listed above in Section 6.2 are used in the remainder of these experiments. The fitness function derived above is dependent on reducing the data using PCA. It has already been noted that PCA concentrates the variance of the data into the lower dimensions. This dimensional reduction reduces the number of calculations that must be performed by the fitness function for every individual in every generation. In the case of the Toy data set above, it was found that over 99% of the variance of the data was included in the first three principal components. On the larger data-set from each cross-validation set, of size 300 by 95, the first three dimensions only cover a small percentage of the variance of the data. Hence, to cover a representative range of the data, the number of dimensions examined must be increased. Approximately 60% of the variance of the data was found to be contained in the first eight dimensions of the data. As this represents a considerable reduction in computation while maintaining the majority of the variance, the following experiments were conducted using the first eight principal components of the reduced data. Thus the fitness function now computes the spread and separation distances on eight rather than three dimensions.

6.5.1 Selecting the GA Parameters

As discussed at the beginning of this chapter, the GA evolves individuals from generation to generation by processing them using the operators reproduction, crossover and mutation. The operators create the children that make up the next generation from the individuals in the current generation. In Matlab, the number of children created using crossover and mutation is controlled by the *Crossover Fraction* (or Crossover Probability) parameter. This Crossover Probability controls the number of children, apart from elite children which are passed directly into the next generation, that are created using crossover,

the remainder of the population being created using mutation. The following sections examine the changes in fitness obtained from varying the operator parameters.

Crossover Probability

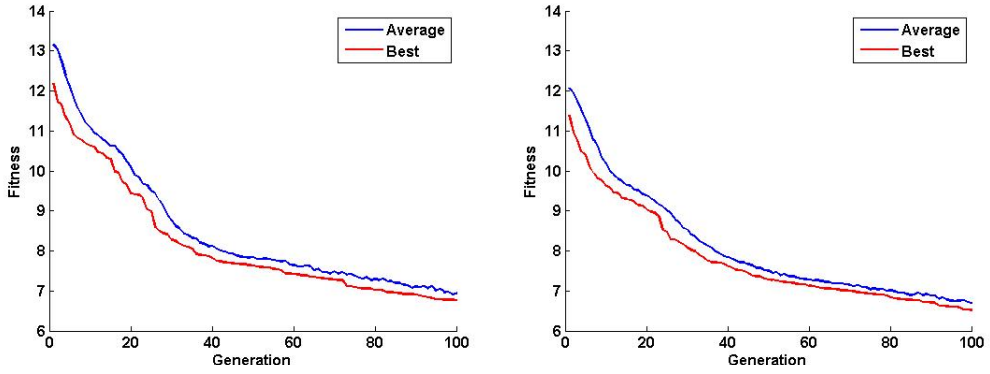
The default value for the Crossover Probability in Matlab is 0.8. To determine if this is the best value to use, the changes in the fitness were observed by varying the Crossover Probability from 0.7 to 0.95 in steps of 0.05. Each of these experiments was run for 100 generations on a population of size 50 with each of the 10 data-sets. The average and best fitness achieved throughout each of the 10 runs were noted and averaged over each set. The changes in average and best fitness over all generations for each of the values of the Crossover Probability are displayed in Figure 6.7. The best fitness achieved from each of the values of the Crossover Probability were similarly noted and averaged over each set. A boxplot of the results of this best fitness for each value of the Crossover Probability is shown in Figure 6.8. These boxplots show the median for each value (red line) surrounded by a box notating the *interquartile range*. The *whiskers* of the plot define the upper limit and lower limit. Values outside of these limits are considered outliers (Martinez and Martinez, 2002).

The fitness plots in Figure 6.7 are all relatively similar: regardless of the choice of the Crossover Probability, the average and best fitness decrease and converge over many generations. Figure 6.8 indicates that once the Crossover Probability is kept below 0.9 the fitness remains low, with an optimal value obtained at a Crossover Probability value of 0.75. Figure 6.7(b) shows that this value gives a steady decrease in both the best and average fitness over many generations. Hence the value of the Crossover Probability for the remainder of these experiments was chosen to be 0.75.

Mutation Shrink

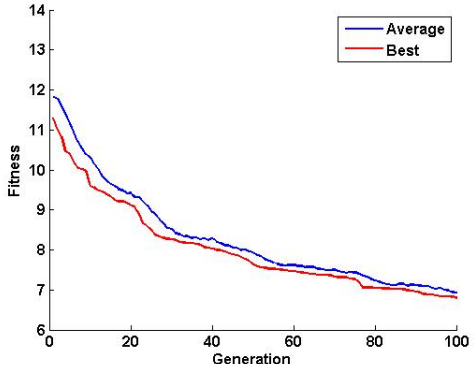
With the Crossover Probability set to 0.75, the amount of mutation is automatically set to 0.25 (after elite children). Mutation is implemented in Matlab by adding a random number from a Gaussian distribution to each of the parent vectors. Ideally, the GA should converge to a good solution over a number of generation, hence it would require more mutation at the beginning of a run to prevent premature convergence, but less towards the end of a run to prevent 'jumping' away from highly evolved solutions. The amount

of mutation applied to an individual may be decreased at each generation in Matlab using the *Shrink* parameter. The amount of mutation is decreased so that it is *(1 - Shrink)* times its initial value at the first generation. The default value of Shrink is 1, leading to no mutation at the end of the run. Again to determine the use of Shrink in our data we observed the changes in fitness with the value of Shrink set from 0 to 1 in steps of 0.25. The experiment was again run for 100 generations on a population of 50 with the Crossover Probability set to 0.75. The changes in the average and best fitness are shown in Figure 6.9 with the corresponding best fitness results displayed in Figure 6.10.

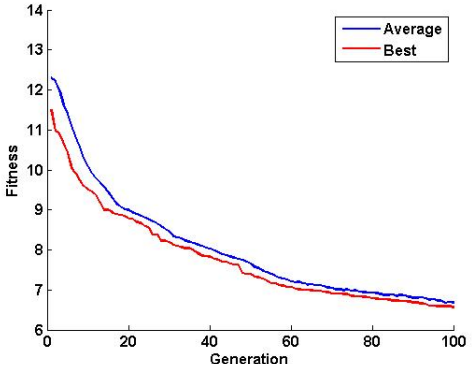


(a) Crossover Fraction=0.7

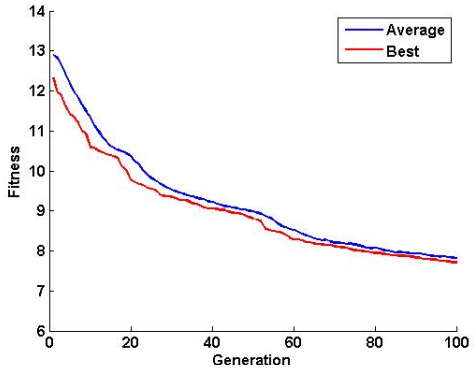
(b) Crossover Fraction=0.75



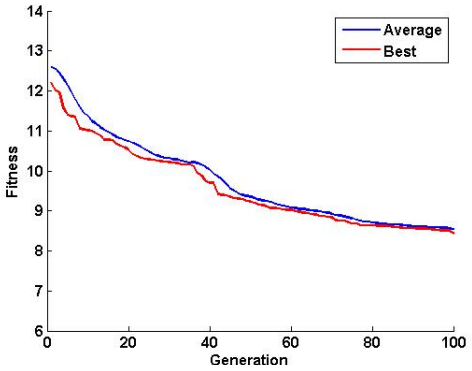
(c) Crossover Fraction=0.8



(d) Crossover Fraction=0.85



(e) Crossover Fraction=0.9



(f) Crossover Fraction=0.95

Figure 6.7: *Average vs. Best fit over 100 generations for varying values of Crossover Fraction*

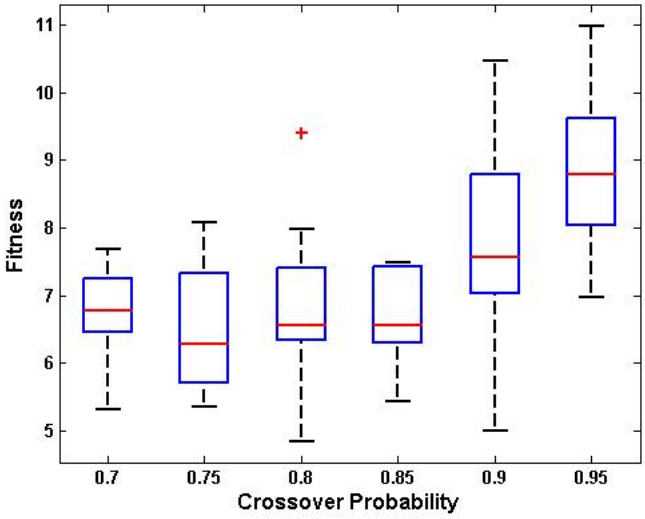
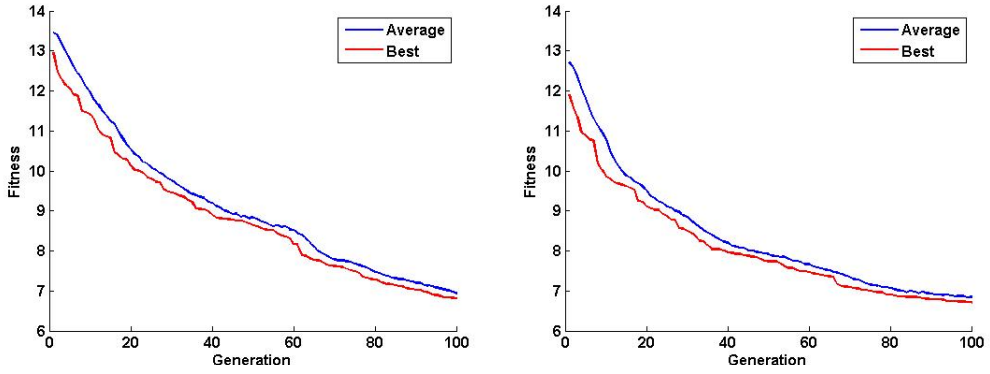


Figure 6.8: *Best fit obtained using different values of Crossover Fraction, averaged over 10 sets*

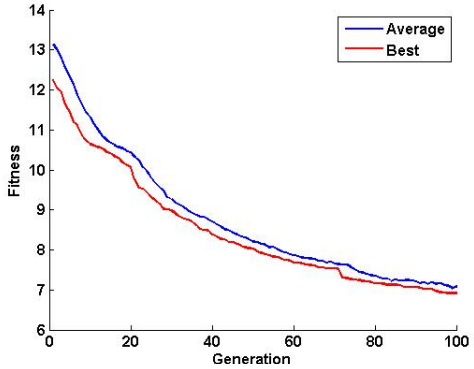
As before with the Crossover Probability, there is not a significant difference in the relationship between the best and average fitness over the number of generations with a change in Shrink. The best fitness is obtained from having Shrink equal to 0.25. It is worth noting that even when the value of Shrink is equal to 1 (ie no mutation at the end of the run) the average and best fitness do not completely converge. This implies that this data is not easily converged so some mutation should be kept in the algorithm up until the last generation, and that the population size and the number of generations may need to be increased in future runs.

The results of these experiments have led to the implementation of the remainder of the experiments on a population of 500 over 500 generations with an Elite Count of 50, a Crossover Probability of 0.75 and a Shrink value of 0.25. Although it was seen that a change in the Crossover Probability or Shrink value did not have a significant effect on the results, these particular values did perform marginally better. The selection function was set to Stochastic Uniform — the default setting in the Genetic Algorithm and Direct Search Toolbox in Matlab. This method allocates a parent by moving in equal steps along a line formed by the all potential parents. Each potential parent occupies a length of this line proportional to its scaled fitness value. The following sections detail the results of experiments using these parameters on the complete (3006 samples with 95 features) data set with the fitness functions derived earlier in the chapter.

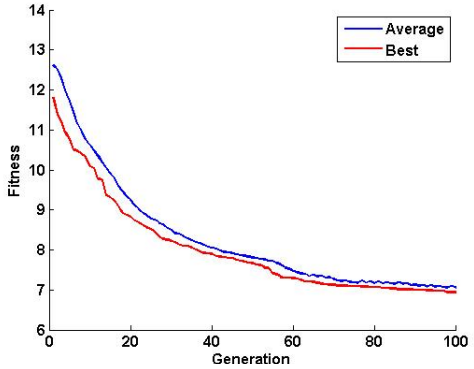


(a) Shrink=0

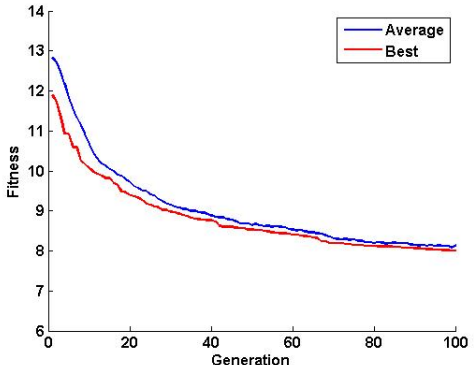
(b) Shrink=0.25



(c) Shrink=0.5



(d) Shrink=0.75



(e) Shrink=1

Figure 6.9: *Average vs. Best fit over 100 generations for varying values of Shrink*

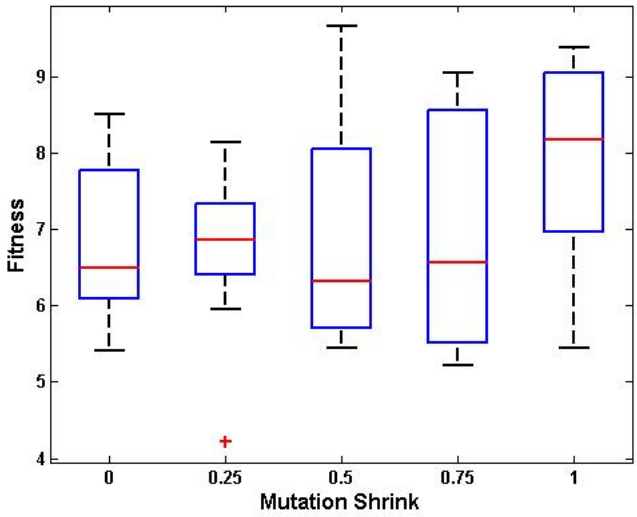


Figure 6.10: *Best fit obtained using different values of Shrink, averaged over 10 sets*

6.6 Fitness Methods

This section describes experiments with the various fitness functions developed earlier applied using eight dimensions. These experiments are carried out using the entire data set split into 10 cross validation sets as described in Section 6.2.1. A GA is computationally expensive to run — implementing one full run on the full data set for an appropriate number of generations is unfeasible. Instead the GA was implemented to evolve one genome for each of the 10 sets, which could then be compared against each other. Using the parameters described later in this section, each genome took approximately six hours to evolve in Matlab. These genomes were then multiplied by the original data and used to train a Multi-Layered Perceptron (MLP). The trained MLP was then tested using the smaller 65 note set described in Section 6.4.1. The results in this section are given in terms of the genomes evolved by the GA for each set and the classification accuracy of the MLP when these genomes are used to modify the training and testing data.

As a benchmark for the following classification results, an MLP was trained and tested with the original unaltered data to determine the accuracy of the classifier. The network used here was trained using the backpropagation algorithm with three layers of hidden neurons. As with previous experiments in Chapter 5 it was implemented in Matlab. It was set up with a learning rate of 0.1 and a momentum constant of 0.95. It was batch trained with a

goal of 0.0001 to a maximum epochs of 500. The network was set up with 57 neurons in the first layer, with three hidden layers containing 28, 20 and 15 neurons respectively. The classification result of the unaltered data is given in Table 6.8. It is evident from this table that the network does not recognise all instruments equally well; the accuracy of recognition of the piano and violin is much higher than that of the trumpet and guitar. This classification result is compared to those obtained by training a network of the same structure with data modified by the evolved genomes.

Table 6.8: *Classification results from full unaltered data*

Total	Piano	Violin	Flute	Trumpet	Guitar
68.31%	90.77%	92.31%	56.15%	55.38%	46.92%

6.6.1 Euclidean Fitness

For a fitness based on the Euclidean method, the spread and separation were measured as per Equations 6.1 and 6.2 above, except that they were now calculated across eight dimensions instead of three. The fitness was then calculated according to Equation 6.5 with an offset of $\sqrt{8}$ instead of $\sqrt{3}$ as there were eight dimensions. The GA was run on each of the 10 cross-validation sets and a best genome (that with the lowest fitness value) was noted for each set. Again, each gene in this genome represents the amount (0 to 1) of the corresponding feature that will be included in the weighted dataset for classification by the MLP. The GA was run on a population of 500 individuals, 50 of which were elite and passed on to the next generation. As per the experiments in the preceding section the crossover fraction was set to 0.75 and the shrink value was set to 0.25. The fitness limit was set to zero and each experiment was run for 500 generations.

Genome Results

The GA resulted in an individual best genome for the 10 validation sets. Ideally, each of these would be similar to one another — representing a consistent choice of each of the individual timbral features. To determine how consistent these results actually were, the diversity between these 10 genomes was examined. The average of the genomes is shown in Figure 6.11, with the standard deviation displayed in Figure 6.12.

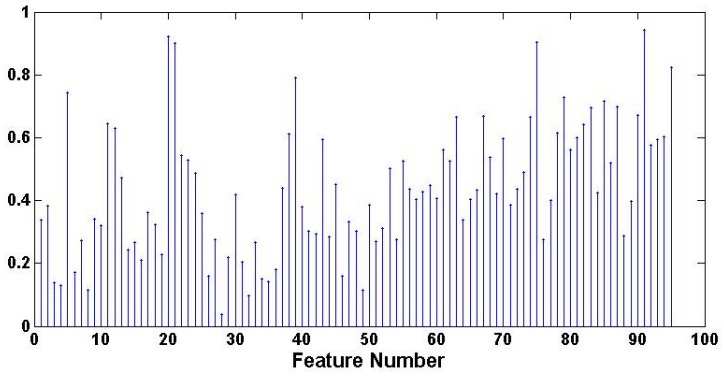


Figure 6.11: *Average of the 10 best genomes*

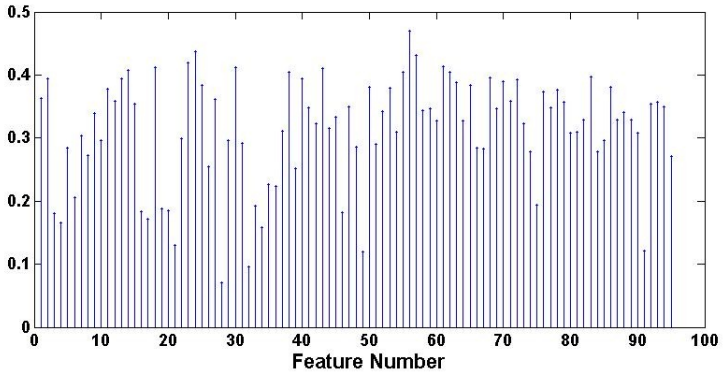


Figure 6.12: *Standard Deviation of the 10 best genomes*

While the diversity in the averages of the genomes indicates that certain features were included to a greater extent than others, the high standard deviations indicate that these average results are not consistent across the 10 validation sets. For 61 of the 95 features, the standard deviation is above 0.3 indicating that the genome value for most of the features varies considerably across the 10 sets. This does not indicate however if certain features were consistently picked or dismissed in relation to other features. To investigate this, we consider the features most frequently emphasised over the 10 runs.

Each genome was sorted into ascending and descending order. The first 10 genes of the descending genome correspond to the 10 features most emphasised for that validation set. Likewise, the first 10 genes of the ascending genome correspond to the 10 features least emphasised or ignored for that validation set. In some sets there were more than 10 values at zero; in these instances all zero-value genes were considered. The common genes observed in multiple genomes as the 10 strongest and weakest values were noted. Table

6.9 displays the feature number and feature whose corresponding gene was in the 10 strongest for four or more instances out of the possible 10 evolved genomes. Table 6.10 displays a similar result for the weakest common genes among the genomes. These tables show that the first principal component of the the Centroid Envelope was the most often selected feature. Similarly, the fourth principal component of a number of the higher MFCCs were often chosen. In contrast to this the first principal component of the Spectral Envelope was the least chosen feature. These results are compared to those obtained from the other fitness methods further on in the chapter.

Table 6.9: *Strongest Features as chosen by the Euclidean fitness GA*

Instances (/10)	**Feature No.**	**Feature**
8	20	Cent1
7	91	MFCC15-4
5	79	MFCC12-4
5	83	MFCC13-4
5	95	MFCC16-4
4	11	MIRSkew
4	63	MFCC8-4

Table 6.10: *Weakest Features as chosen by the Euclidean fitness GA*

Instances (/10)	**Feature No.**	**Feature**
7	28	Spec1
6	35	MFCC1-4
5	8	Onset Dist
5	14	No. Peaks
5	31	Spec4
4	18	Env3
4	26	Res3
4	27	Res4
4	40	MFCC3-1
4	46	MFCC4-3

Classification Results

The classification results are given in terms of the test accuracy of an MLP trained on a given set of data. In each case the training set was comprised of all 3006 samples and the test set consisted of the smaller 65 sample set. The accuracy of the trained network was measured as the percentage of times the network correctly identified the test samples. The GA evolved 10 different

genomes — one for each validation set —- which was multiplied by the data prior to being input to the network. Every genome contains 95 genes, each of which corresponds to a specific feature value. Each genome was multiplied by the training data and the test data. The MLP was trained and tested 10 times using data calculated from each genome and the average of these 10 runs was noted. The results of these training and testing runs are shown in Table 6.11. These results show that the multiplication by the evolved Euclidean genome is beneficial to the system, with the average accuracy increasing from 68.1% to 70.74%, although the average identification accuracy still varies widely between the instruments. The average classification accuracy also varies widely between the different genomes: in this case Genome 2 performed the worst at 63.23% whereas Genome 5 performed best at 80.15%. As we have seen in the previous section, each evolved genome is different and hence emphasises different features. This, combined with the 'black box' nature of training MLPs, may explain the inconsistencies within the results.

Table 6.11: *Average classification results for each Euclidean genome multiplied by the full dataset*

Genome	Total	Piano	Violin	Flute	Trumpet	Guitar
1	73.74%	97.69%	98.46%	59.23%	55.38%	54.62%
2	63.23%	94.62%	94.62%	76.15%	22.31%	28.46%
3	64.62%	95.38%	94.62%	53.08%	60.77%	19.23%
4	73.38%	93.08%	93.85%	76.15%	55.38%	48.46%
5	80.15%	96.92%	99.23%	85.38%	65.38%	53.85%
6	67.38%	99.23%	95.38%	49.23%	64.62%	28.46%
7	65.85%	92.31%	96.15%	57.69%	51.54%	31.54%
8	71.69%	93.08%	99.23%	71.54%	56.15%	38.46%
9	71.08%	90.77%	97.69%	66.15%	43.85%	56.92%
10	76.92%	90.77%	91.54%	80%	61.54%	60.77%
Avr	**70.74%**	**94.39%**	**96.08%**	**67.46%**	**53.69%**	**42.08%**

The aim of evolving these genomes was not only an attempt to increase the accuracy of the MLP, but also to determine if certain features were superfluous to such a system and could be removed from the data set while maintaining the same level of identification accuracy. In determining this, the experiment was repeated with reduced sets of data according to a *cut-off* threshold value in the genomes. The network was trained and tested only using feature values whose corresponding gene had a value of 0.3 or higher. This was repeated for values higher than 0.7, 0.85 and 0.95. A summary of the average results obtained across all 10 genomes is shown in Table 6.12. These results indicate that the accuracy of the classifier reduces as the number of features included

is reduced. The increase in the genomes cut-off value resulted in a much smaller data set being used to train and test the MLP. Table 6.13 shows the number of genes within each genome that are above each cut-off value. This indicates that any cut-off significantly reduces the amount of data used with the network. The reduction of classification accuracy for the cut-off equal to 0.3 is understandable considering the number of features used was reduced to as low as 40 — in the case of Genome 9. Further reductions in the number of features used (by increasing the cut-off value) led to a corresponding decrease in classification accuracy. Thus we may say that in this case, the network is not improved with a reduction in features. In the remainder of this section we examine further classification results for the alternative fitness functions developed earlier.

Table 6.12: *Classification results from Euclid weighting all data with corresponding gene values greater than 0.3, 0.7, 0.85 and 0.95*

Gene	**Total**	**Piano**	**Violin**	**Flute**	**Trumpet**	**Guitar**
>0.3	59.85%	80.23%	91.69%	51%	43.15%	33.15%
>0.7	50.4%	70.62%	75.54%	44.77%	37.54%	23.54%
>0.85	46.48%	61.08%	79.46%	37.31%	29.54%	25%
>0.95	42.8%	46.62%	69.69%	37.92%	34.62%	25.15%

Table 6.13: *Number of genes within each Euclidean genome that are above 0.3, 0.7, 0.85 and 0.95*

Gene	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
>0.3	50	45	48	56	51	49	61	55	40	52
>0.7	22	28	30	33	36	29	39	30	25	28
>0.85	19	22	25	25	30	23	32	23	19	23
>0.95	10	16	18	15	21	15	20	13	12	18

6.6.2 Dimensional Fitness

The Dimensional fitness was calculated as in Section 6.4.3, from the relationship between the dimensional spread and separation of the clusters formed by the PCA. The Dimensional spread and separation were calculated according the Equations 6.6 and 6.7 respectively. As above with the Euclidean fitness, the Dimensional fitness was then calculated according to Equation 6.5 with an offset of $\sqrt{8}$ instead of $\sqrt{3}$. The GA was again run on all 10 cross validation sets with the same parameters used for the Euclidean fitness above.

Genome Results

An examination of the average and standard deviation of the 10 evolved genomes yielded plots similar to those for the Euclidean fitness in Figures 6.11 and 6.12 indicating that again the evolved genomes are different from one another. The strongest features most often chosen by the genomes are shown in Table 6.14. From this it is evident that the fourth principal component of the twelfth MFCC is the most often chosen strong feature and again, as with the Euclidean fitness, the fourth principal components of the high MFCCs were strong in general. Similarly, the first component of the Centroid Envelope (the strongest feature chosen by the Euclidean fitness) again featured highly with this Dimensional fitness. The weakest features chosen by all the genomes is shown in Table 6.15. Several of the lower MFCC values are among the lowest chosen features, as are two of the Spectral Envelope values — similar to the previous Euclidean result.

Table 6.14: *Strongest Features as chosen by the Dimensional fitness GA*

Instances (/10)	Feature No.	Feature
8	79	MFCC12-4
7	43	MFCC3-4
5	20	Cent1
4	24	Res1
4	75	MFCC11-4
4	83	MFCC13-4
4	91	MFCC15-4

Table 6.15: *Weakest Features as chosen by the dimensional fitness GA*

Instances (/10)	Feature No.	Feature
6	34	MFCC1-3
5	7	No. Onsets
5	28	Spec1
5	30	Spec3
5	36	MFCC2-1
5	44	MFCC4-1
4	6	First Att Slope
4	40	MFCC3-1
4	51	MFCC5-4
4	73	MFCC11-2

Classification Results

The classification results for the MLP trained and tested by data multiplied by the genomes evolved using the Dimensional fitness function are shown in Table 6.16. These results display a slight reduction in classification accuracy in comparison to the Euclidean fitness, although the results are similar. The classification of the data multiplied by the amended genomes (reduced according to the cut-off values) is shown in Table 6.17. Again this shows a reduction in accuracy as more features are removed according to the cut-off threshold.

Table 6.16: *Classification results from multiplying data by evolved Dimensional genomes*

Genome	Total	Piano	Violin	Flute	Trumpet	Guitar
1	67.69%	93.85%	96.15%	33.08%	53.08%	62.31%
2	67.23%	81.54%	92.31%	46.15%	58.46%	57.69%
3	72.46%	82.31%	92.31%	60%	66.15%	61.54%
4	74.92%	98.46%	95.38%	67.69%	60%	53.08%
5	71.69%	96.15%	95.38%	40.77%	61.54%	64.62%
6	68.46%	93.85%	99.23%	72.31%	66.92%	10%
7	51.85%	71.54%	96.92%	54.62%	23.85%	12.31%
8	78.46%	93.08%	98.46%	88.46%	70%	42.31%
9	62.92%	94.62%	93.08%	60.77%	53.08%	13.08%
10	59.23%	88.46%	90%	23.85%	65.38%	28.46%
Avr	**67.49%**	**89.39%**	**94.92%**	**54.77%**	**57.85%**	**40.54%**

Table 6.17: *Classification results from Dimensional weighting all data with corresponding gene values greater than cut-off thresholds of 0.3, 0.7, 0.85 and 0.95*

Gt	Total	Piano	Violin	Flute	Trumpet	Guitar
>0.3	58.89%	82.85%	85.08%	57.62%	36.85%	32.08%
>0.7	55.62%	72.23%	75.92%	49.92%	48.54%	31.46%
>0.85	52.4%	67.69%	73.46%	50%	44.31%	26.54%
>0.95	43.23%	52.85%	60.31%	44.08%	41.15%	17.77%

6.6.3 Euclidean-Dimensional Fitness

Finally these experiments were repeated using the combination Euclidean-Dimensional fitness measure. As described earlier, this method uses the spread value calculated according to the Euclidean fitness as in Equation 6.1 and the separation value according to the Distance fitness measure in Equation 6.7. The Euclidean-Dimensional fitness was then calculated according

to Equation 6.5 as before. The GA was again run to evolve 10 genomes, one from each of the cross validation sets with the same parameters used above.

Genome Results

As with the previous fitness methods, the genomes were found to be different from one another. The strongest and weakest features chosen across all the genomes are shown in Table 6.18 and Table 6.19. The Centroid Envelope and the fourth principal components of the higher valued MFCCs feature strongly again in agreement with the genomes from the previous fitness methods. The Spectral Envelope and certain low value MFCCs are included in the weaker features, although this is not as prominent as it was with the other fitness methods, with the Number of Spectral Peaks emerging as the weakest feature overall.

Table 6.18: *Strongest Features as chosen by the Euclidean-Dimensional fitness GA*

Instances (/10)	Feature No.	Feature
8	20	Cent1
8	83	MFCC13-4
6	21	Cent2
6	95	MFCC16-4
5	75	MFCC11-4
5	91	MFCC15-4
4	13	Inharmonicity
4	18	Env3
4	43	MFCC3-4

Table 6.19: *Weakest Features as chosen by the Euclidean-Dimensional fitness GA*

Instances (/10)	Feature No.	Feature
6	14	No. Peaks
5	32	MFCC1-1
4	3	Brightness
4	6	First Att Slope
4	27	Res4
4	28	Spec1
4	29	Spec2
4	48	MFCC5-1

Classification Results

The classification results for an MLP trained and tested with data multiplied by the genomes evolved using the Euclid-Dimensional fitness function are shown in Table 6.20. These results are comparable to that of the Euclidean fitness measure: they show a slight improvement on the classification of the original data, but there is still a large variation between the instruments. The classification results obtained when the data set was reduced according to the cut-off threshold on the genomes is given in Table 6.21. Although the reduction in data does again lead to a decrease in classification accuracy, this fitness method does appear to be somewhat more resilient to this data reduction than the other fitness methods. The classification accuracy remains up above 60% until the cut-off threshold is increased above 0.7. As noted in Table 6.22, this cut-off value can correspond to less than 30 features being input to the system — a significant decrease from the original 95.

Table 6.20: *Classification results from Euclid-Dimensional weighting all data with corresponding gene values*

Genome	Total	Piano	Violin	Flute	Trumpet	Guitar
1	70.77%	95.38%	98.46%	72.31%	40.77%	46.92%
2	75.54%	97.69%	99.23%	61.54%	50%	69.23%
3	70.77%	88.46%	99.23%	63.85%	46.92%	55.38%
4	66.15%	95.38%	93.08%	60.77%	39.23%	42.31%
5	65.08%	93.08%	97.69%	59.23%	65.38%	10%
6	69.85%	87.69%	99.23%	79.23%	41.54%	41.54%
7	75.69%	85.38%	91.54%	63.08%	68.46%	70%
8	68.92%	89.23%	99.23%	46.92%	58.46%	50.77%
9	61.54%	93.85%	97.69%	44.62%	61.54%	10%
10	71.38%	93.08%	100%	84.62%	45.38%	33.85%
Avr	**69.57%**	**91.92%**	**97.54%**	**63.62%**	**51.77%**	**43%**

Table 6.21: *Classification results from Euclidean-Dimensional weighting all data with corresponding gene values greater than thresholds*

Genome	Total	Piano	Violin	Flute	Trumpet	Guitar
>0.3	63.82%	84.92%	96.46%	56.31%	48.69%	32.69%
>0.7	61.43%	73.77%	90.62%	41.77%	59.69%	41.31%
>0.85	54.45%	63%	85.38%	45.62%	50.85%	27.39%
>0.95	43.93%	50%	71.15%	38.77%	40.31%	19.39%

Table 6.22: *Number of Dimensional-Euclidean genes within each genome that are above 0.3, 0.7, 0.85 and 0.95*

Gene	1	2	3	4	5	6	7	8	9	10
>0.3	52	48	54	48	54	61	59	51	56	56
>0.7	31	33	35	28	32	31	23	30	32	36
>0.85	25	28	27	22	25	21	16	24	23	23
>0.95	12	15	12	15	18	11	9	21	13	14

6.6.4 Combining Classification Results

The above training and testing runs were simulated with each individual genome and the average of the results were given. In this section the experiment was run again, but in this case classification of a sample was calculated according to the majority of classifications by the genomes. Each genome gives a classification of a particular sample. These classification were combined in a type of voting system, whereby the instrument picked by the majority of the genomes was considered to be the correct classification. The results are shown in Table 6.23. Clearly this shows a marked decrease in the accuracy of recognition of the instruments, particularly for the trumpet and guitar. The violin, on the other hand, was correctly classified 100% of the time. On closer inspection of the results it was found that the trumpet was most often mistaken for a violin and the guitar was misclassified as a piano or violin. As these two instrument choices dominated the results, combining the classifications in this manner caused the individual correct choices of trumpet and guitar to be ignored. Hence, combining the results in this manner was not found to be useful.

Table 6.23: *Classification results from combining the classification results of each genome*

Total	Piano	Violin	Flute	Trumpet	Guitar
57.08%	94.62%	100%	66.15%	8.46%	16.15%

6.6.5 Discussion of 8-Dimensional results

The results obtained in this section on 8-dimensional fitness methods offer a number of insights into the methods and the features used in this experiment. From the classification accuracy of each of the fitness methods, it is clear that there is not a significant difference between them. Overall, when the full data set is multiplied by the evolved genomes, those evolved using the Euclidean and Euclidean-Dimensional fitness methods perform slightly better than the Dimensional method. It was proposed at the beginning of these experiments that a number of the data features were superfluous to the study, and as such the system may perform better if these 'weaker' features were removed. We have seen from the results above that this is not necessarily so, as the accuracy of the MLP tended to decrease with the removal of the less important features. As noted above however, the Euclidean-Dimensional fitness method performed better than the other methods when features were removed. On average, across all the instruments, the accuracy of the classifier remained as high as 61.43% when the cut-off was reduced to 0.7. At this cut-off value, the number of features used ranged between 23 and 36, a significant reduction on the original 95. Using this method, similar results to this were presented in (Loughran et al., 2009). This Euclidean-Dimensional method was chosen as the fitness measure for further GA experiments.

As noted above, in each of the three methods, the 10 evolved genomes were different from one another. Unfortunately this implies that each genome has found a local minimum in the search space rather than the global minimum. It is in the nature of GAs to find an *acceptable* solution to the given problem and it may not necessarily be able to find the global solution i.e. one specific weighting for all 95 features that consistently gives the best classification results. Each of the 10 evolved genomes has its own strong and weak points. We must point out here, that the results above are all *averaged* over a number of runs. The sets of classification results are averaged over 10 MLP training and testing runs. Further to this, the classification results given using the cut-off threshold were averaged over each of the 10 genomes. The genomes did

not perform equally well however — certain genomes performed quite strongly even when the amount of data used was significantly reduced. For example, using the Euclidean-Dimensional methods with a cut-off of 0.95, genome 10 produces an average accuracy rate of 73.54% across all instruments, even though this cut-off value results in just 14 feature values being used. In comparison to this, genome 7 with the same method and cut-off value only produces an average classification accuracy of 22%.

These average accuracy values do not indicate the differences in classification accuracy between the instruments. It can be seen from the results above that the piano and violin are much more successfully recognised than the flute, trumpet and guitar. The guitar in particular appears to be very difficult for the MLP to classify correctly. This difficulty in classifying a specific instrument was found to be exaggerated by combining the results from the individual genomes in Section 6.6.4.

The three methods above were examined not just for their success at classifying samples, but also in the strongest and weakest features chosen by the GA for each method. Even though as noted above, each of the genomes were different, there were still some common features that appeared regularly as particularly 'strong' or 'weak'. Although there were some changes in the strongest features chosen, it must be noted that the Centroid Envelope — in particular its first principal component, and the fourth principal component of several of the higher MFCCs were among the strongest picked features for each of the three measures. This agrees with previous work, both in the current thesis (as in Chapter 5) and work done by other researchers who have found the centroid to be one of the most important features in music recognition. It is of particular interest that it is values from the Centroid Envelope rather than the static Centroid that have emerged as the more prominent feature. The Centroid is recognised to be important, but these results indicate that it may be the changes in the Centroid throughout the duration of a note that is more useful. The strong presence of the higher MFCCs is also an interesting finding. MFCCs have been regularly used in speech analysis, whereby the standard number to use is 12 (O'Shaughnessy, 1987). Experiments in Chapter 5 found that using up to 16 MFCCs was of benefit to a system such as this for instrument recognition. The strong placement of the higher MFCCs here reinforces that finding. It is unclear, however, why the fourth principal component of these are most often chosen. As stated before, PCA transforms the data, so it is not clear as to what physical or spectral attribute (if any) the resulting principal components represent. Although there

are a few repetitions among the weakest features — the Spectral Envelope for example, in general these are not as pronounced as the strongest features. It is clear however that those features that are strongest according to one method do not appear among the weakest by another method and vice-versa. This consistency among the different (albeit similar) fitness function calculation methods in the choosing of specific features is encouraging for this type of experiment: even though the genomes may be different, they share common strengths that may offer some insight into the problem they address — in this case the problem of timbre description.

All of the fitness functions implemented here only used 8 dimensions from the PCA of the data in forming and analysing the clusters. This only accounts for approximately 60% of the variance of the original data. With such a significant amount of the variance of the data unaccounted for, it is unlikely that a global best solution may be found. If, however, the number of dimensions were increased to 40, the variance explained by the PCA would comprise 98% of the original data. The following section examines the implementation of the Euclidean-Dimensional fitness method calculated from 40 dimensional clusters.

6.7 40 Dimension Experiments

The above experiments were repeated on a larger scale by including 40 dimensions in the fitness function, using only the Euclidean-Dimensional method of calculation. The GA was again run on a population of 500 with an elite count of 50. The crossover value was set to 0.75 and the shrink value was set to 0.25. These experiments were run for 1000 generations. With these parameters, the GA took approximately 12 hours to evolve each genome. The MLP used in the classification experiments had the same architecture and parameters as those described above.

6.7.1 Genomes Evolved

By increasing the computational dimensions to 40 it was hoped to evolve genomes that were more consistent than those evolved over 8 dimensions. A plot of the average genome values and standard deviations of the genome values is shown in Figure 6.13 and Figure 6.14. These plots indicate that the newly evolved genomes are not significantly more consistent than those evolved before. In particular, the plot of the standard deviation is high for

most values, with almost all features having a value greater than 0.2. This high standard deviation indicates that increasing the search dimensions to 40 does not result in the same features being chosen consistently by each genome. The average of these genomes in Figure 6.13 appears to have fewer distinct high values in comparison to the average of the 8-dimensional genomes displayed in Figure 6.11. Likewise the standard deviation of the 40-dimensional genomes appears to be more consistently high than that of the 8-dimensional genomes in Figure 6.12. This implies that the 40-dimensional genomes are actually more diverse than those evolved using the 8-dimensional fitness function. This means that running the GA for more generations (1000 as opposed to 500) and using more dimensions in the fitness function *over-fits* the data and did not ensure a more consistent genome was evolved.

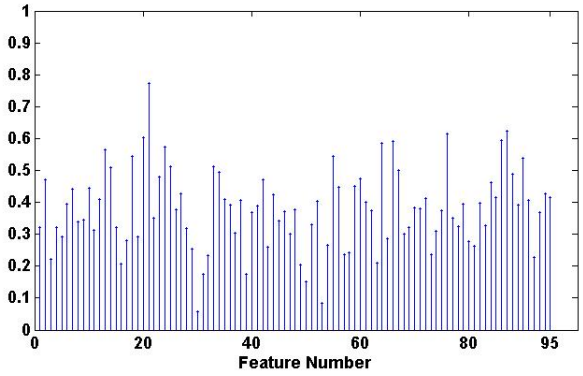


Figure 6.13: *Average of the 10 Genomes evolved using 40 dimensional fitness*

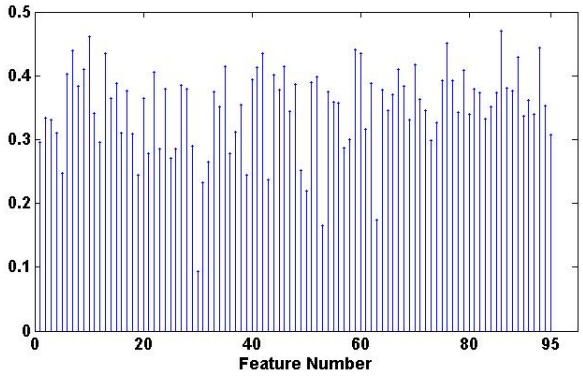


Figure 6.14: *Standard Deviation of the 10 Genomes evolved using 40 dimensional fitness*

Most Common Features

To determine the most commonly chosen features, we again look at those that are most often one of the strongest 10 features in each evolved genome. The strongest 10 and weakest 10 features are shown in Table 6.24 and Table 6.25. The number of instances of these features is again indicative of less consistent genomes: whereas in the 8-dimensional results, the most common features were present in 6, 7 or 8 of the genomes, here the most common features only appear 4 times. Furthermore, only three features were chosen as one of the 10 strongest four or more times, and only one feature was among the 10 weakest four or more times. The strongest features selected do, however, agree with the results from the earlier experiments. The first and second principal components of the Centroid Envelope and the fourth component of a high MFCC were the strongest features selected; the same features were included in the strongest 8-dimensional genomes. The weakest features selected however was the first principal component of the Temporal Envelope which was not among the weakest features chosen by the 8-dimensional genomes

Table 6.24: *Strongest Features as chosen by the 40 Dimensional Euclidean-Dimensional fitness GA*

Instances (/10)	**Feature No.**	**Feature**
4	20	Cent1
4	21	Cent2
4	87	MFCC14-4
3	10	MIRSpread
3	13	Inharmonicity
3	44	MFCC4-11
3	64	MFCC9-1
3	66	MFCC9-3
3	67	MFCC9-4
3	86	MFCC14-3
3	93	MFCC16-2

Table 6.25: *Weakest Features as chosen by the 40 Dimensional Euclidean-Dimensional fitness GA*

Instances (/10)	Feature No.	Feature
4	16	Env1
3	22	Cent3
3	30	Spec3
3	39	MFCC2-4
3	40	MFCC3-1
3	45	MFCC4-2
3	47	MFCC4-4
3	50	MFCC5-3
3	53	MFCC6-2
3	59	MFCC7-4
3	78	MFCC12-3
3	83	MFCC13-4
3	92	MFCC16-1

The above measure of which are the 'strongest' chosen features is somewhat ambiguous. There may be more than 10 strong features within any one genome — certain genomes have more strong genes than others. As an illustration, Figure 6.15 displays Genome 10. Evidently, there are more than 10 strong values within this one genome. Many of these values are above 0.9 and thus play a strong role within this genome. Hence it may be more informative to look at the strongest range of values within the genomes and note which occur most frequently. Table 6.26 displays the most common strong features, whereby a strong feature is defined as one with a corresponding gene greater than 0.95. Table 6.27 display a similar result for the weakest features with value less than 0.1. By comparing the results in these tables to those of the strongest and weakest features in Tables 6.24 and 6.25, it is clear that there is a strong similarity between them. The same features have emerged using both criteria as either strong or weak, but there is some change in the order of which are deemed the strongest or weakest. For example, in the second approach which considers all features above 0.95 to be strong, the third component of MFCC14 is the strongest feature, appearing five times. There are only three instances of this being the strongest feature in Table 6.24 whereby only the strongest 10 features in each genomes were considered 'strong'. Likewise Table 6.27 below displays that the second component of MFCC6 is considered weak in 9 of the 10 genomes when weak means less

than 0.1, but it only appears as one of the weakest features in Table 6.25 3 times. Further to this, of the 8 weakest features chosen using the less than 0.1 criterion, only 4 of these were considered weak using the weakest 10 criterion. Thus, although certain features are common regardless of the selection criterion used, what defines certain particularly strong or weak features may be dependent on the way in which the genomes are examined.

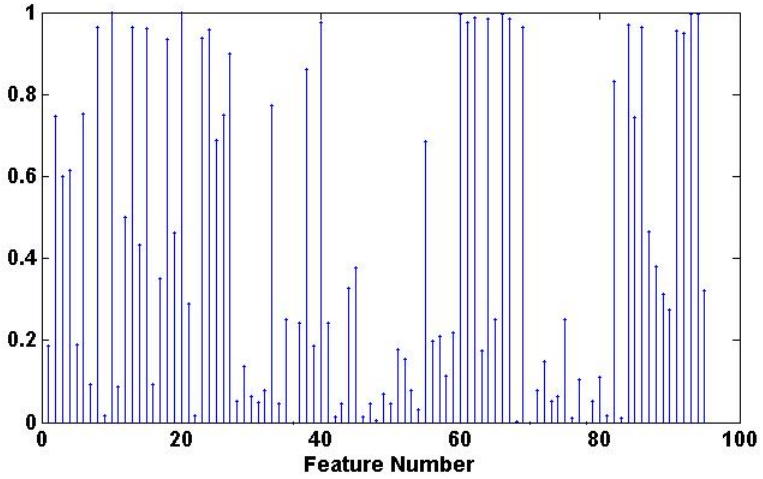


Figure 6.15: *Genome 10, evolved using 40-dimensional fitness function*

Table 6.26: *Strongest (> 0.95) Features as chosen by the 40 Dimensional Euclidean-Dimensional fitness GA*

Instances (/10)	Feature No.	Feature
5	86	MFCC14-3
4	13	Inharmonicity
4	20	Cent1
4	21	Cent2
4	87	MFCC14-4
3	10	MIRSpread
3	24	Res1
3	44	MFCC4-1
3	60	MFCC8-1
3	64	MFCC9-1
3	66	MFCC9-3
3	76	MFCC12-1
3	93	MFCC16-2

Table 6.27: *Weakest (< 0.1) Features as chosen by the 40 Dimensional Euclidean-Dimensional fitness GA*

Instances (/10)	Feature No.	Feature
9	53	MFCC6-2
8	30	Spec3
6	3	Brightness
6	16	Env1
6	17	Env2
6	31	Spec4
6	17	MFCC5-3
6	31	MFCC13-2

6.7.2 Analysis of Genomes per Set

The lack of consistency among the strong features within the genomes may indicate that *over-fitting* has occurred; as the GA has been run for a longer number of generations, it may have evolved genomes that are specific to the given cross validation data set and not general to the full set. Throughout this chapter four genomes have been evolved for each validation set: one with each of the three 8-dimensional fitness methods and a further one with the 40-dimensional methods. If the genomes are being evolved according to the specific set (i.e. being over-fit to an individual set rather than representing the more general full data-set), there should be similarities between the four evolved genomes for each set. This is tested by examining the strongest 10 features of each of the four evolved genomes for one specific set — Set 5. The results of this are shown in Table 6.28. It is evident from this set that although there are a few common features — Cent1, Inharmonicity and MFCC13-4 appear in three of the genomes — there is not one single feature that is present in all four of the genomes. If over-fitting to Set 5 had occurred, we would expect to see a much similar set of features emerging for each of the methods used to evolve a genome for this set. The strongest 10 features for each of the four fitness methods listed in Table 6.28 show that this is not the case. Similar results were found for the other genomes, indicating that the genomes are not over-fitted to their specific validation sets.

Although no strong common features were found among the genomes, the genomes were again tested as an aid to classification with the MLP.

Table 6.28: *Strongest 10 features in the genomes evolved using each of the fitness methods on Set 5*

Euclidean	Dimensional	Euclidean-Dim	40 Dim Euclid-Dim
First Att Time	Res1	Kurtosis	Brightness
Skew	MFCC12-1	Inharmonicity	Inharmonicity
Kurtosis	MFCC12-4	Cent1	MFCC9-3
Inharmonicity	MFCC13-4	MFCC11-4	MFCC12-1
Cent1	MFCC16-2	MFCC12-3	MFCC15-1
MFCC2-3	MFCC16-3	MFCC14-2	Spread
MFCC10-2	Cent1	MFCC15-4	MFCC11-2
MFCC12-4	MFCC5-1	MFCC13-4	MFCC14-1
MFCC13-3	MFCC12-3	MFCC1-2	MFCC15-2
MFCC13-4	First Att Slope	MFCC2-4	Cent2

6.7.3 Classification One Octave Test Set

The MLP was trained on the full 3006 samples and tested on the smaller set of 65 samples as before. The classification results of the training and test data multiplied by each of the genomes is shown in Table 6.29. This shows the average results of 10 training and testing runs for each genome. The overall accuracy does not show a marked improvement on the accuracy obtained from the genomes evolved using the 8-dimensional fitness method. Again it can be noted from this table that the classification accuracy between the instruments varied greatly: the piano and violin were again recognised much more often than the guitar. The average classification results across all genomes, when the number of features used was decreased according to the gene threshold cut-off value, is shown in Table 6.30. Again this shows a marked decrease in the accuracy of the system as the number of features used is decreased.

The one-octave data set used here and in earlier experiments, is very limited in its representation of the samples database. A common pitch range of C4-C5 played at dynamic level f is all that is included. In contrast, the training samples include samples covering the entire natural pitch range of each instrument — up to eight octaves in the case of the piano. In addition to this, the training samples are played at three dynamic levels, both with and without vibrato. The dynamic of a note can have a very strong effect on the perception of that note. It was noted in Chapter 3, that the strong attack of a struck instrument is very pronounced in the Temporal Envelope, but that this attack may be less pronounced if the same note is played more softly. This could lead to ambiguities between struck and plucked instruments, such

Table 6.29: *Classification results from 40-dimensional fitness method, tested on the one-octave sample set, weighting all data with corresponding gene values*

Genome	Total	Piano	Violin	Flute	Trumpet	Guitar
1	63.69%	90.77%	93.08%	42.31%	57.69%	34.62%
2	62.62%	100%	92.31%	70.77%	30.77%	19.23%
3	74%	90.77%	90%	78.46%	56.92%	53.85%
4	57.38%	90.77%	90.77%	66.15%	34.62%	4.62%
5	79.85%	83.03%	98.46%	86.92%	64.62%	66.15%
6	74.46 %	84.62%	97.69%	85.38%	40.77%	63.85%
7	68.92%	100%	88.46%	83.85%	43.08%	29.23%
8	71.23%	98.46%	100%	73.08%	45.38%	39.23%
9	66.31 %	98.46%	94.62%	81.54%	35.38%	21.54%
10	77.08%	93.08%	93.85%	63.08%	71.54%	63.85%
Avr	**69.55%**	**93%**	**93.92%**	**73.15%**	**48.08%**	**39.62%**

Table 6.30: *Classification results from 40-dimensional fitness method, weighting all data with corresponding gene values greater than thresholds*

Genome	Total	Piano	Violin	Flute	Trumpet	Guitar
>0.3	55.11%	75.31%	81.46%	56.31%	37.15%	25.31%
>0.7	50.02%	61.31%	73.85%	55.46%	32.77%	26.69%
>0.85	44.34%	58%	65.92%	52.38%	27.23%	18.16%
>0.95	44.18%	53.92%	61.54%	56.08%	25.69%	23.7%

as in the piano and guitar above. For example, if the training data contained many loud piano samples and soft guitar samples, it may end up associating the strong attack of a loud guitar test sample with a piano. This may lead to the network being capable of recognising piano sounds while not being able to determine guitar samples, such as for example the results of Genome 4 in Table 6.29. A bias in the testing samples, such as this, may cause inaccurate or misleading test results. To prevent this, the network was trained and tested again using nine of the cross-validation sets to train the network and the remaining set to test it. This resulted in 2706 training samples and 300 test samples [4].

6.7.4 Cross-Validation Test Sets

Using one of the cross-validation sets as a test set increases the number of samples being used to test the network, from 65 to 300, resulting in more reliable analysis of the test samples. In addition to this, each cross-validation test set contains a better representation of the entire samples database than the single octave set. As the sets are iteratively filled with every tenth consecutive sample, each set contains an even distribution of samples in regard to pitch and dynamic. The MLP was trained using the data from Sets 1-9 and tested with Set 10. The classification accuracy of the MLP trained and tested on this data unaltered by the genomes is given in Table 6.31. These results are averaged over 10 training and testing runs. It is evident from these results that there is a sharp increase in the accuracy of the network when using this data as opposed to the one octave test data. The instruments are all equally well recognised — each at almost 100% accuracy. This is surprising given the low accuracy of recognition using the one octave set. To confirm this result, the network was trained again with the data from Sets 2-10 and trained with the data from Set 1, again unaltered by the genomes. The results of the average of 10 training and testing runs using this data is given in Table 6.32. Although slightly lower than those results found using test Set 10, these results are still significantly higher than those obtained using the one octave test set. The drop in accuracy of recognition of the trumpet samples may be explained if we are reminded that there are extra trumpet samples in Set 1 — the extra 6 samples added into this set were high pitched trumpet samples. These results indicate that the choice of test samples is evidently very important in testing the accuracy of such classifiers.

[4] unless Set 1 is the test set in which case there were 2700 training samples and 306 test samples

Table 6.31: *Classification accuracy of network trained on Sets 1-9 and tested on Set 10 with original data unaltered by the genomes*

Total	Piano	Violin	Flute	Trumpet	Guitar
99.63%	100%	98.76%	100%	100%	99.85%

Table 6.32: *Classification accuracy of network trained on Sets 2-10 and tested on Set 1 with original data unaltered by the genomes*

Total	Piano	Violin	Flute	Trumpet	Guitar
95.59%	97.7%	98.65%	93.95%	86.88%	96.62%

The data was then modified using the evolved genomes, to determine if features can be removed while maintaining the accuracy of the classifier. As the results in Table 6.31 were so accurate, the MLP was again trained using Sets 1-9 and tested using Set 10, but this time with the data modified by the genomes. As before, the data was reduced according to a number of different cut-off values for the individual genes within the genome. The results in Table 6.33 display the accuracy of the network when trained on the full modified data, features with non-zero corresponding genes, and features with corresponding gene values that are > 0.3, > 0.7, > 0.85, > 0.95, > 0.99. In this case we could not expect the classification accuracy of the MLP to be improved by modifying the data by the genomes, but it is interesting to note that the accuracy is not decreased as noticeably as before with the removal of features using Set 10 as the test set. The average classification accuracy across all five instruments is maintained above 99% until the cut-off is raised to 0.85. To illustrate the reduction in data when the cut-off thresholds are used, Table 6.34 displays the number of genes within each genome that are above each cut-off value. These show that a cut-off value of 0.7 may mean that as few as 22 instead of the original 95 features would be used. This is a large decrease in data for a very small corresponding average decrease in classification accuracy. This high level of accuracy with a similar decrease in recognition with increase in cut-off was also seen in experiments using Set 1 and Set 5 as the (unseen) test set.

Table 6.33: *Classification results testing with Set 10, weighting all data with corresponding gene values greater than thresholds*

Genome	Total	Piano	Violin	Flute	Trumpet	Guitar
All	99.63%	100%	98.76%	99.83%	100%	99.93%
>0	99.63%	99.97%	97.75%	99.9%	100%	99.97%
>0.3	99.57%	99.9%	98.72%	99.61%	100%	100%
>0.7	99.34%	99.79%	98.47%	99.06%	99.69%	99.93%
>0.85	98.96%	99.22%	98.2%	98.53%	99.26%	99.76%
>0.95	96.69%	96.39%	96.87%	96.43%	96.97%	96.77%
>0.99	89.79%	90.43%	92.12%	83.84%	88.49%	91.42%

Table 6.34: *Number of 40-dimensional genes within each genome that are above the cut-off thresholds of 0, 0.3, 0.7, 0.85, 0.95 and 0.99*

Genome	1	2	3	4	5	6	7	8	9	10
>0	93	85	93	86	89	92	85	89	90	92
>0.3	38	35	41	40	39	54	34	42	43	44
>0.7	25	22	29	25	22	30	23	22	24	30
>0.85	20	13	20	17	18	22	20	16	18	24
>0.95	11	9	8	14	12	15	10	11	9	19
>0.99	4	6	7	8	5	7	7	7	7	6

As before, these results were all averaged over the 10 genomes, each of which was run 10 times. For a closer inspection of one of these results, Table 6.35 displays the average results of each of the 10 evolved genomes with the cut-off threshold value equal to 0.95. It is clear from this table that the classification accuracy varies more between the genomes than between the instruments. Genome 3 results in the lowest overall classification, whereas Genome 9 results in the best classification of all 10 genomes. From Table 6.34 it may be seen that Genome 3 is the smallest genome at this value of cut-off, with the data from only eight features used to train and test the network. Although this implies that fewer features result in lower classification accuracy, it may also be noted that Genome 9 contains data from only nine features and yet this gives the best classification at this cut-off threshold. The eight features used by Genome 3 consist of:

- No Onsets, Res4, MFCC4-1, MFCC9-2, MFCC10-4, MFCC11-3, MFCC14-4 and MFCC16-2

whereas the nine features used by Genome 9 consist of:

- ZeroCross, Rolloff, MIRSkew, Cent2, MFCC1-2, MFCC4-1, MFCC5-1, MFCC11-1 and MFCC12-1

Clearly, there is no overlap between these features: each selection contains a mixture of temporal, spectral and MFCC features. This illustrates that the GA does not select the same set of features or even a common subset of features each time. This agrees with the genome results above, where it was found that many common strong features were not found among the genomes.

Table 6.35: *Classification results tested with Set 10 with the cut-off threshold set to 0.95*

Genome	Total	Piano	Violin	Flute	Trumpet	Guitar
1	97.57%	99.18%	96.30%	96.73%	97.96%	98.14%
2	94.80%	97.87%	96.91%	92.24%	85.64%	96.57%
3	87.43%	82.95%	93.95%	88.57%	97.18%	77.57%
4	98.97%	98.85%	97.65%	99.39%	100%	99.71%
5	98.63%	98.69%	97.41%	98.78%	98.46%	100%
6	97.93%	98.89%	96.30%	98.78%	99.74%	99.14%
7	97.30%	95.57%	97.78%	97.55%	97.18%	98.14%
8	95.90%	94.26%	95.80%	95.31%	94.36%	98.71%
9	99.23%	99.84%	98.15%	98.98%	100%	99.71%
10	99.13%	99.84%	98.40%	97.96%	99.49%	100%
Avr	**96.69%**	**96.39%**	**96.87%**	**96.43%**	**96.97%**	**96.77%**

Although a high number of common features were not strong among all the genomes, a number of them were found to occur more often than others as depicted in Table 6.26. This table lists 13 features, each of which had a corresponding gene value greater than 0.95 in three or more of the genomes. Thus this may be considered as the 'optimal' selection by the GA. To investigate this, the MLP was again trained with 9 sets and tested with Set 10, this time with just the data from the features listed in Table 6.26. The average results are shown in Table 6.36. Although these results are high, they do show a reduction in average accuracy compared to the results obtained using more features. The recognition results for the guitar and piano however were particularly high, but a lower recognition rate for the flute and to a lesser extent the violin and trumpet meant that the average result was quite low. The average recognition rate did not vary much between the different genomes, ranging from 98.03% for Genome 1 to 99.23% for Genome 10. Hence, choosing the most common strong features within the genomes alone is not enough to maintain the accuracy of the full data set.

Table 6.36: *Classification results tested with Set 10 with data from the top 13 common features*

Total	Piano	Violin	Flute	Trumpet	Guitar
98.80%	99.12%	98.63%	97.18%	98.97%	99.76%

6.7.5 Discussion of 40 Dimensional Results

It was hoped that increasing the number of dimensions used to calculate the fitness from 8 to 40 would create more stable genomes. It was already noted that the first 40 principal components of the original data contain 98% of the variance of the data, thus these 40 dimensions may be used to accurately represent the data. Based on this it was anticipated that using a fitness function that incorporated these 40 dimensions would yield more consistent genomes. It is evident from a comparison of the 40 dimensional genomes to the 8 dimensional genomes, that this was not the case. The 40 dimensional evolved genomes appear to be more diverse than those evolved using only 8 dimensions, with fewer strong common features emerging among the 10 genomes. With the Euclidean-Dimensional fitness method over 8 dimensions, nine strong features were found among four or more of the genomes with the two strongest features occurring in eight genomes. By comparison, using the same *top ten* method, only 3 of the features were found among the 40 dimensional genomes 4 or more times. This is not necessarily detrimental to the system however. Each genome was evolved using a different sample set. The 40 dimensional genomes were evolved using more data over twice the number of generations than the 8 dimensional genomes. Thus it is possible that the 40 dimensional genomes are merely more highly evolved: that they have been given the opportunity to traverse the search space and evolve a solution that is specific to the current sample set. Such an idea would indicate that there may be no one particular set of features that is commonly best used for classifying *any* group of samples or instruments — that each classification problem may be unique.

It was seen that in using the one-octave test set, the use of the genomes evolved over 40 dimensions does not significantly improve the classification results with the MLP. Most notably though, it was found that increasing the test set to include a larger group of samples more representative of the whole data set produced much more accurate results. This emphasises the importance of using an appropriate set of test samples when conducting such classification experiments. Using one of the cross-validation sets as a test set

resulted in equally high classification results across all five instruments. The accuracy of classification varied somewhat between genomes — particularly as the weaker features were removed.

Limitations of the Method

In the above method, each gene represents the amount of the corresponding feature to be used in the training and testing of the MLP. As discussed earlier, this is represented by a vector of 95 values between 0 and 1 that is multiplied by the data. We determined the 'strongest' or most important features in the experiments above by looking at which had the highest corresponding gene. However, each feature data point already has a value between 0 and 1. Thus the resultant multiplied data is dependent on how strong the original data point was *and* the strength of the weighting gene. Thus if a feature with a very small original value was emphasised with a strong corresponding gene value, it may still only be as significant in the resultant weighted set as a strong original value that was multiplied by a medium strength gene. This may explain why the classification results drop when only the very strong gene values are included — such as when the cut-off threshold was set $>$ 0.85.

Another issue that may arise is in the fundamental workings of the GA. It was seen that each evolved genome is different. Although we hypothesise above that this may be due to the different samples used — that each genome may be unique to the samples it is evolved from — it is also possible that the GA is simply unable to find a global best solution. As already discussed, GAs are very good at finding a good solution to a problem, but they may not necessarily find the best possible solution. Thus the diversity in the evolved genomes — and hence the lack of clear choice on which are the best features to use in musical instrument classification problems may be due to one of two factors:

- An exact ideal set of features to use for accurate instrument classification exists, but this GA was unable to determine this set, or
- No such ideal set of features exist — each classification problem is unique and dependent on the samples and instruments involved in the experiment.

6.8 Conclusion

This chapter examined the use of GAs in determining the best set of timbral features to use for musical instrument identification. Section 6.1 introduced the aims of the chapter. Section 6.2 described the data used in this chapter and detailed the way in which it is organised. Section 6.3 introduced the basic concept of a GA and the method in which it was implemented in the experiments in the chapter. Section 6.4 described the development of the fitness function used in these experiments. In particular this section reduced the given problem into a 'little problem' to develop the methods used. This little problem was then scaled up to include the full data set with the fitness calculated across 8 dimensions in Section 6.5. These experiments were further scaled up to 40 dimensions and tested using a more general test set of samples in Section 6.7.

This chapter started with three distinct aims. First, to use a GA to create an improved musical instrument identifier. It was shown in Section 6.5 that using an evolved genome to modify the input data to an MLP can improve the recognition accuracy of the MLP while using the one octave test set. These experiments were then repeated using a fitness function based on 40 dimensions of the data. It was shown that using these genomes improved the classification accuracy over the unmodified data, although it did not result in a significant improvement on the results obtained using 8 dimensions. When the trained MLP was tested with one of the cross-validation sets, the recognition accuracy was significantly higher than that obtained with the one-octave set, with the accuracy close to 100%. This was not improved with the introduction of the evolved genomes, but using the genomes maintained this high level of accuracy. These results emphasised the importance of the selection of samples in testing a classifier such as this one.

The second aim was to reduce the complexity of an instrument identifier without decreasing the level of classification accuracy. It was hoped that the genome could be used to eliminate certain superfluous timbral features that might be unnecessary for the system. This was investigated by removing features whose corresponding genes were below a certain cut-off threshold. It was found in both the 8 dimensional and 40 dimensional experiments using the one octave test set, that reducing the number of features incorporated into the system reduced the classification accuracy. This reduction in accuracy became more significant the more features were removed. When the larger cross-validation test set was used however, the reduction in classification ac-

curacy was not as significant. It was seen that the classification accuracy was maintained above 99% even when up to 73 of the 95 original features were removed. Thus it may be concluded that a reduction in features may make a bad systems worse, but will not cause significant problems to a system trained and tested on a well balanced set of samples.

The final aim was to use the evolved genomes to determine which timbral features are most important for instrument classification. It was seen above however, that regardless of the method used, the genomes evolved were very diverse with few common features emerging between them. One notable exception to this is that at least one principal component of the Centroid Envelope featured very strongly in each method used. The Centroid has featured very strongly in previous literature and in experiments described earlier in this thesis. Most notably here however, is that it is always the Centroid Envelope and not the static Centroid (calculated from the MIRToolbox) that was chosen. This means that it is the *changes* within the centroid across the duration of the sound that was found to be most important for instrument classification. Along with the Centroid Envelope, the fourth principal component of a number of the higher MFCCs were chosen, supporting results given in Chapter 5 that found the higher MFCCs to be very important in instrument recognition. The weakest features found by the system were not as strong or consistent as the strongest found although some that featured included the Spectral and Residual Envelopes and values from some of the lower MFCCs. Most interestingly it was found that increasing the number of dimensions that the genomes were calculated over, and increasing the number of generations as in Section 6.7 apparently made the genomes more diverse. This result may support the argument that there is no one ideal set of features to classify any group of instrument sample — that each experiment is unique and would therefore require a unique solution.

Although the evolutionary method described in this chapter has raised some interesting questions, it remains a computationally expensive model. The fitness function of the GA is dependent on PCA and the testing of each genome requires the training and testing of a MLP. The following chapter looks at a different evolutionary algorithm that can evolve a solution to a problem such as this, without the need for these extra processes — Genetic Programming.

Chapter 7

Genetic Programming

The previous chapter used GAs to optimise features for instrument identification using an MLP. This method addressed the issue of over-burdening a classifier with too much information by limiting the number of features used as inputs to the classifier. Although the use of the GA offered insight into the best timbral features to use, the method is cumbersome and computationally expensive using both PCA and linear algebraic calculations to determine the fitness of each individual. In addition to this an MLP was still needed to make use of this evolved genome in classifying a sound, as the genome evolved by the GA cannot represent a solution in this classification problem domain. An evolved genome may represent the relative *weight* of each feature to use, but it does not indicate how these features are best implemented as a classifier. This chapter proposes to evolve not only which features to include but also the method by which these features should be combined to classify musical instruments.

Chapter 4 discussed many classification methods that have been implemented to identify musical instruments. These studies all used machine learning techniques to classify sounds. Although some studies compared the merits of a selection of techniques, few justified the use of specific classifiers. The previous chapter simplified the number of features to use in a classification study by evolving a representation of the set of features with a GA. This chapter furthers this idea by evolving the features used *and* the classification method. A new program for classifying musical instruments, based on the features used before, may be evolved using Genetic Programming (GP). Unlike GAs, GP evolves a program that can represent the solution to instrument classification without the need for a further external classification method such as the MLP. GP is described in more detail in Section 7.1. The implementation of GP on the data used in this thesis is described in Section

7.2. Section 7.3 describes the results obtained from these GP experiments. In both this and the previous chapter, a number of features emerged as more important than others. Section 7.4 discusses the variations between samples for these particular features. The visualisation of the program trees evolved and the issue of bloat is discussed in Section 7.5. Finally Section 7.6 offers a discussion on the results and some conclusions for the chapter.

7.1 Genetic Programming

The term Genetic Programming was coined by Koza, whose seminal 1992 book (Koza, 1992b) remains the definitive introduction to the topic. As with GAs, GP works on a population of solutions rather than a single solution. The main difference between GA and GP is in the representation of the solution. Although GAs were developed first, a GA may actually be seen as a specific type of GP where the solution is limited to a single fixed-length vector (Langdon and Poli, 2002). GP, on the other hand, evolves solutions that consist of executable programs. These programs are represented by structures, the most commonly used being the *tree structure*. Other commonly used structures are *linear structure* and *graph structure*. The GP implemented in this thesis operates on a tree structure. A program tree consists of several *branches*, each of which contain internal *nodes* and ends with a *leaf*. The number and type of branches in a program form the *architecture* of the program (Langdon et al., 2008). These tree structures may easily be written out as strings. A simple example of a tree representing the string $(+a(*b2))$ or $a + 2b$ is shown in Figure 7.1.

7.1.1 GP Design

As with GAs, a number of parameters must be set at the beginning of a GP run. These parameters will dictate how the GP behaves over successive generations.

Terminals and Functions

The terminals and functions are the 'pieces' available in creating the programs. Each leaf on the tree consists of a terminal of the program. A terminal node returns a numeric value and does not require any inputs — that is it has an *arity* of 0. The set of possible terminals for any given problem consists of the variables of the problem along with any user defined constants.

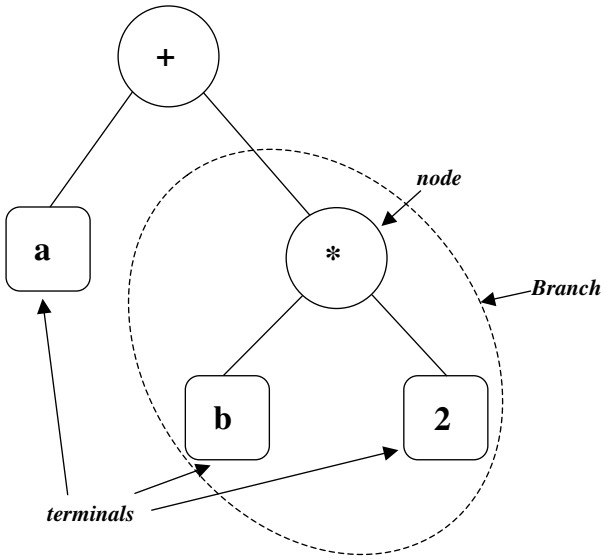


Figure 7.1: *A basic program tree displaying a node, branch and terminals*

Internal nodes within the tree are comprised of operators or functions. The range of functions that may be used by a GP is varied and includes operators of types such as boolean functions, arithmetic functions, conditional statements as well as more complicated functions such as variable assignment functions, subroutines or automatically defined functions (ADFs) (Koza, 1994). The choice of functions is dependent on what the evolved program is required to do. Although many functions may be used, it is advised to start with a set of simple program functions as GP has shown to be very creative at taking simple functions and combining them for its own needs (Banzhaf et al., 1998). The set of allowed functions and terminals for any given run form the *primitive set* of the GP. This primitive set must satisfy the conditions of *closure* and *sufficiency*. Closure requires that each of the functions in the set must be able to accept, as its input arguments, the output returned by any of the functions or the values from any of the terminals. Sufficiency requires that the primitive set is capable of expressing a solution to the problem (Koza, 1992b).

Initialisation

A GP run starts by creating the initial population of solutions. Each tree is randomly created from the defined terminals and functions. There are a number of different ways of generating this initial population. Two of the simplest and earliest methods used are the *full* and *grow* methods. Both of these methods create trees up to a specified maximum depth. In the full method, functions are selected as nodes along each branch until the maximum tree depth is reached, at which point terminals are chosen as leaves. Thus the full method creates full, symmetrical trees of a specific size and shape. The grow method on the other hand allows either a terminal or function to be chosen at each node up until the maximum tree length is reached. Thus trees created with the grow method may be of varying size and shape. As neither of these methods would create a particularly varied or diverse initial population, a combination of the two methods known as *ramped half-and-half* was proposed. In this method half of the population is created using the grow methods and half using the full method. In addition to this, the population is divided up so that equal numbers of individuals have tree depths *up to* the specified maximum depth (Koza, 1992b). Thus the population will consist of symmetrical and asymmetrical trees of varying lengths.

Operators

The operators of GP are similar to those used for GA, namely crossover, mutation and selection. Selection is based on the fitness level of each individual: individuals with good fitness are more likely to be used in the next generation. As with GA a number of selection techniques are available such as roulette and tournament. The most common type of crossover used in GP is *subtree crossover*. In subtree crossover a random point is selected on two parent trees A and B. The children are created by replacing the subtree at the crossover point on tree A with the subtree at the crossover point of tree B and vice-versa. Other types of GP crossover include one-point crossover, size-fair crossover and uniform crossover (Langdon et al., 2008). As with GAs, mutation in GP operates on one individual. Subtree mutation randomly selects a point in the tree and replaces it with another randomly generated subtree. Point mutation, on the other hand, replaces a single random node with a different terminal or function of the same arity.

Fitness Function

The probability of an individual being selected for the next generation is dependent on the fitness of that individual. The function used in this chapter is a minimising fitness function — the lower an individual's fitness, the higher it's probability of selection will be. The way in which fitness is measured is dependent on the fitness function. The fitness function must be defined or developed by the user before the beginning of a GP run. The fitness function specifies the goals of the search process. The details of each fitness function are specific to the type of program being evolved by the GP run. Each individual program is evaluated and assigned a fitness in accordance with how well it performs compared to the optimum program being evolved.

7.1.2 Advantages of Using GP for Instrument Classification

Since its conception in 1992, GP has been used for many applications from game theory (Koza, 1992a) to finance (Brabazon and O'Neill, 2009). However, it was seen in Chapter 4 that GP was not included as one of the many different machine learning techniques used previously for instrument recognition. GP is implemented here for this purpose as it has potentially many advantages over other machine learning techniques:

- As with GAs, certain features may be emphasised or excluded from the solution, without specification from the user. These chosen features may be seen by the user — offering an insight into timbre.
- Unlike GAs, GP can evolve the entire classification solution.
- Unlike ANNs which offer a black-box solution to the problem, the program evolved by the GP is accessible to the user.
- A GP only uses functions defined or specified by the user for a specific problem.
- The size of the program may be controlled by the user, thus avoiding overly large programs or bloat.

7.2 Implementation of GP

GP as described in this thesis was implemented using the GPLAB Toolbox for Matlab (Silva, 2004). This toolbox includes most of the traditional fea-

tures used with GP and has the benefit of having a highly modular structure — making it easily modified and specified for any given purpose. As it is generalist yet extendable, it is suitable for all levels of users. It offers the user as much or as little control over the parameters and primitives as desired. This section details the parameters used for the GP implemented and described in this thesis.

7.2.1 Functions and Terminals

The data used in these experiments is the same data used in the GA experiments in the previous chapter. Hence 95 timbral features were used from 3006 audio samples split into 10 sets. Each of the 95 features may be an input to the system, contributing 95 variable terminals to the system. In addition to this a number of constants were also implemented as terminals. The number and values of these constants were individual to specific runs and are detailed in the discussion of these runs later in the chapter.

GPLAB offers a large number of possible arithmetic and logical functions to the user. Many of these functions, particularly the mathematical functions such as trigonometric functions may be overly complicated for the purpose of this classifier. The algorithm is designing a program to determine the best combination of the available features to use in recognising a musical instrument. The ideal relationship between the features is unlikely to rely on complicated mathematical functions. A number of different runs included different functions, as detailed later in the chapter, but in general these were limited to simple logical and mathematical functions.

While it may be easy to satisfy the closure condition in these experiments, it is not so certain that the condition of sufficiency is satisfied. Sufficiency states that it must be possible for the primitive set to express a solution to the problem, but this is the point under investigation in this chapter. The question raised in this chapter is: is it possible to create an instrument classifier using just a combination of the features and a small set of functions? We know from the classification success of the GA experiments in the previous chapter that this set of features is sufficient for training and testing an MLP. What needs to be be confirmed here is if the same features may be combined with just a few simple functions to accurately classify the data. The confirmation of sufficiency must therefore wait until the results are obtained.

7.2.2 Initialisation

GPLAB allows the user to set the values of all the parameters of the GP at the beginning of a run. There are three possible methods for tree initialisation — full, grow and ramped half-and-half, as described above. The trees in these experiments were initialised using the ramped half-and-half method as it results in the most diverse initial population. The initial maximum depth of the tree was set to 6. The trees developed using GPLAB may be subjected to a number of size constraints throughout the run to avoid the development of overly large programs or bloat. The maximum permissible size of a tree is set using the parameter `realmaxlevel`. In addition to this a smaller dynamic maximum level may be specified using `inicdynlevel`. If an operator results in a tree whose size is greater than inicdynlevel but less than realmaxlevel its fitness is measured. If this individual's fitness is lower than that of the current best individual, the maximum dynamic level is increased and the individual is passed to the next generation. If the individual is found to be not as good as the current best individual it is rejected. This technique of dynamic fitness has been shown to be efficient in preventing bloat (Silva and Almeida, 2003). In these experiments realmaxlevel was set to 15 and inicdynlevel was set to 6.

To ensure the best individuals survive to successive generations, GPLAB may select certain individuals for the next generation according to the `elitism` parameter. The elitist method implemented in these experiments is `keepbest`. Using this method, the best individual from both parents and children is kept for the new population, the remaining places in the population being filled by children only.

Once the elite children have been passed to the new generation, the remainder of the new population is created using a combination of the operators selection, crossover and mutation. GPLAB offers four sampling (or selection) methods for selecting parent individuals: `roulette`, `sus`, `tournament` and `lexictour`. Roulette and sus are both methods based on a roulette wheel whereby each individual 'owns' a proportion of the roulette wheel, the difference being that in sus the pointers are equally spaced. Tournament and lexictour choose parents by randomly drawing a number of individuals from the population and selecting the best from this group. In lexictour, if two individuals are equally fit, the shorter individual will be favoured for the next generation. The sampling method implemented in these experiments is lexictour.

Any number of operators may be created by the user and used with

GPLAB. The standard operators of mutation and crossover are provided by the toolbox and implemented in these experiments. GPLAB offers an implementation of an automatic adaptation procedure for the operator probabilities based on the method introduced in Davis (1989). This method adjusts the probability of an operator according to the fitness of the individuals that are created by it; operators that create fit individuals will have their probabilities increased and those that create poor individuals will have their probability decreased. These operator probabilities were set to be variable from the first generation through to the end of the run. The initial values of the crossover and mutation probability values were both set to the default GPLAB values of 0.5.

The GP will run until the specified maximum number of generations has been reached or one of the stop conditions is satisfied. The stop condition may be used to stop the run when a specified percentage of the population is within a specified tolerance of the ideal fitness value. In these experiments, the default value of stopping when 100% of the population is at the exact fitness value is implemented.

7.2.3 Fitness Function

GPLAB offers three distinct fitness functions: one suitable for symbolic regression and parity problems and two suitable for artificial ant problems. Due to the modular framework of the toolbox however, it is relatively simple to create a new fitness function specific to the given problem and implement it with GPLAB. This user-defined fitness function may calculate a 'good' fitness according to the problem under investigation. The fitness functions implemented in this chapter are similar to those used for regression problems in that they calculate the difference between the expected output value, or target, and the values returned by the current individual. The result was calculated as a numerical output which could then be assigned to a pre-defined instrument class, according to the various experiments described later in the chapter. Similar numerical fitness classifications have been implemented successfully in previous GP experiments (Landry et al., 2006; Loveard and Ciesielski, 2001).

The fitness was based on the sum of the difference between the target vector of the data and the output of each individual rounded to the nearest whole number. Initially, the target was set so that 1 corresponded to piano, 2 to violin, 3 to flute, 4 to trumpet and 5 to guitar. The fitness function calculated the output of each individual when implemented using the training data

and rounded each output to a whole number. The output of this individual could then be compared to the target vector to determine how many training samples this individual misclassified. The fitness of this individual is the number of samples misclassified during training. Hence this is a minimising fitness function, which may be specified in GPLAB by setting the parameter `lowerisbetter` equal to 1. The best individuals are those programs which output the greatest number of correct identity numbers for all samples. The algorithm was trained on 2706 samples leaving 300 for testing at a later stage. Thus the optimal fitness is 0 — where no samples are misclassified and the highest (worst) fitness value that could be obtained while training was 2706 whereby all samples were mis-identified.

7.3 Experimental Runs

As discussed above, GP has not yet been employed for musical instrument identification. As an initial experiment, simple programs were developed to analyse the way in which they combined the features available to them. This section describes the various implementations of GP on the features data and the results obtained.

7.3.1 Feature Analysis

The GA experiments in the previous chapter examined the evolved genomes to determine the most commonly used features for instrument identification. The first GP experiments perform a similar analysis in determining which features are most often chosen by the program trees evolved using GP. The full 95 by 3006 data set employed by the GA experiments was again used to create the programs. As the depth of the tree was confined to prevent bloat, evolving the trees would most likely involve selecting the features that give the best training fitness value[1]. Each GP was run 30 times with a population of 100 individuals over 500 generations. The functions allowed in the programs were limited to the arithmetic functions `times`, `plus`, `minus` and `mydivide`. The fitness function calculated the number of incorrect identifications as described above.

The initial training fitness values from these programs were not as low as anticipated, as the lowest fitness obtained from any of the 30 programs was equal to 608. Nevertheless, these evolved programs were analysed to ascertain

[1]technically a full tree at depth 15 could have 2^{15} terminals, but this is unlikely to occur

if any features emerged as more likely to be selected for these programs. The number of times each of the 95 features appeared in any of the strings of each of the 30 best individuals was totalled. It emerged that the first principal component of the Centroid Envelope was the most commonly selected feature — being selected 80 times among the 30 strings. The next most often chosen features were the first principal component of the Temporal Envelope (57), the first principal component of MFCC4 (51), the first principal component of the Spectral Envelope (41) and the second principal component of the Centroid Envelope (40). The Centroid Envelope and the Temporal Envelope were both also found to be important in both the GA experiments in the previous chapter and the neural network experiments in Chapter 5 agreeing with these initial results.

To examine this method of analysis further, the data set was split up into two groups — the MFCCs and the data without the MFCCs. The MFCCs, although only representing one type of feature, contain 64 individual feature points. We may recall that experiments in Chapter 5 ran ANN classification experiments both with and without the MFCCs and determined that they were very important for instrument identification. These experiments, however, only had a choice of 11 other feature points, whereas the GP used in these experiments may use any combination of 35 timbral feature points in addition to the MFCCs to create the classification program. Splitting up the data may also lend itself to the idea of a *hierarchical* GP, whereby a combination of programs built of two subsets of the data may be used for classification (Koza, 1992c).

The GP experiment was run with the same parameters as before but without the 64 MFCC data points included. The results were found to be similar to those achieved with the whole data set as the first principal component of the Centroid Envelope was chosen most often (96 times) followed by the first principal component of the Temporal Envelope (77), the second principal component of the Temporal Envelope (56), the second principal component of the Centroid Envelope (51) and the Zero-Crossing Rate (48). When the GP was repeated for just the MFCC values, it was found that the lower MFCCs — in particular their first principal components — emerged to be the strongest: first principal component of MFCC4 (87), first principal component of MFCC2 (65), second principal component of MFCC1 (47), first principal component of MFCC1 (28), first principal component of MFCC3 (23). This result appears contrary to those achieved in the previous GA experiments, where the higher MFCC values emerged to be the more dominant.

Table 7.1: *No. of times each feature was selected for the four data sets*

Feature	All	No MFCCs	MFCCs	Reverse MFCCs
Zero-Cross	9	48	-	-
Rolloff	22	25	-	-
Brightness	14	21	-	-
Regularity	3	23	-	-
First Att. Time	6	16	-	-
First Att. Slope	8	23	-	-
No. Onsets	9	25	-	-
Onset Distance	12	20	-	-
MIRCent	23	9	-	-
MIRSpread	20	26	-	-
MIRSkew	15	20	-	-
MIRkur	6	20	-	-
Inharmonicity	5	5	-	-
No Peaks	4	5	-	-
Spec. Irreg	8	30	-	-
Envelope(1)	57	77	-	-
Envelope(2)	27	56	-	-
Envelope(3)	35	37	-	-
Envelope(4)	16	37	-	-
Centroid(1)	80	96	-	-
Centroid(2)	40	51	-	-
Centroid(3)	3	6	-	-
Centroid(4)	15	15	-	-
Residual(1)	12	13	-	-
Residual(2)	1	12	-	-
Residual(3)	3	18	-	-
Residual(4)	2	19	-	-
Spec Env(1)	41	41	-	-
Spec Env(2)	2	26	-	-
Spec Env(3)	1	9	-	-
Spec Env(4)	3	14	-	-
MFCC1(1)	**16**	-	**28**	**39**
MFCC1(2)	29	-	47	49
MFCC1(3)	13	-	17	24
MFCC1(4)	10	-	11	17
MFCC2(1)	**19**	-	**65**	**77**
MFCC2(2)	23	-	12	2
MFCC2(3)	7	-	8	7
MFCC2(4)	6	-	10	14
MFCC3(1)	**20**	-	**23**	**30**
MFCC3(2)	13	-	21	50
MFCC3(3)	22	-	9	13
MFCC3(4)	4	-	18	18
MFCC4(1)	**51**	-	**87**	**116**
MFCC4(2)	17	-	21	20
MFCC4(3)	15	-	3	5
MFCC4(4)	9	-	11	10

Table 7.2: *No. of times each feature was selected for the four data sets (cont.)*

Feature	All	No MFCCs	MFCCs	Reverse MFCCs
MFCC5(1)	**7**	-	**17**	**12**
MFCC5(2)	3	-	7	9
MFCC5(3)	4	-	3	14
MFCC5(4)	14	-	17	12
MFCC6(1)	**19**	-	**5**	**14**
MFCC6(2)	3	-	9	10
MFCC6(3)	8	-	6	6
MFCC6(4)	14	-	2	6
MFCC7(1)	**10**	-	**7**	**14**
MFCC7(2)	6	-	4	11
MFCC7(3)	1	-	6	7
MFCC7(4)	8	-	3	23
MFCC8(1)	**5**	-	**18**	**20**
MFCC8(2)	0	-	7	3
MFCC8(3)	3	-	3	2
MFCC8(4)	5	-	1	11
MFCC9(1)	**3**	-	**3**	**21**
MFCC9(2)	2	-	10	22
MFCC9(3)	0	-	2	3
MFCC9(4)	3	-	7	9
MFCC10(1)	**12**	-	**1**	**4**
MFCC10(2)	1	-	7	6
MFCC10(3)	3	-	7	6
MFCC10(4)	18	-	0	14
MFCC11(1)	**2**	-	**3**	**8**
MFCC11(2)	0	-	6	2
MFCC11(3)	1	-	11	3
MFCC11(4)	5	-	6	1
MFCC12(1)	**3**	-	**0**	**7**
MFCC12(2)	1	-	8	10
MFCC12(3)	2	-	3	6
MFCC12(4)	2	-	13	11
MFCC13(1)	**0**	-	**0**	**9**
MFCC13(2)	4	-	3	18
MFCC13(3)	12	-	10	11
MFCC13(4)	1	-	3	3
MFCC14(1)	**0**	-	**1**	**9**
MFCC14(2)	0	-	3	13
MFCC14(3)	0	-	10	10
MFCC14(4)	0	-	6	5
MFCC15(1)	**0**	-	**2**	**2**
MFCC15(2)	0	-	1	6
MFCC15(3)	0	-	4	7
MFCC15(4)	0	-	8	13
MFCC16(1)	**0**	-	**2**	**8**
MFCC16(2)	0	-	1	8
MFCC16(3)	0	-	17	3
MFCC16(4)	0	-	11	1

To confirm this result, the experiment with only the MFCCs was repeated with the data presented in inverse order to the GP. The results appear consistent with the previous forward run, with the lower MFCCs emerging as the stronger features: first principal component of MFCC4 (116), first principal component of MFCC2 (77), second principal component of MFCC3 (50), second principal component of MFCC1 (49), first principal component of MFCC1 (39). A full list of how many times each of the features was chosen for each of the data sets is shown in Table 7.1 and continued in Table 7.2[2]. It is clear from this table that there are many consistencies among the results.

The above GP runs were only examined in terms of common emerging features. As mentioned above, the training fitness obtained using this GP was not particularly encouraging. The next section describes modifications made to the GP run in an attempt to create a more accurate classifier. The results are discussed in terms of training fitness, test classification accuracy and features present.

7.3.2 Modifying the Fitness Function

GP was run again a number of times with various set-ups. The parameters for creating the first generation and for the operators were kept as before. In each case the maximum level of the tree was again set to 15 and the dynamic level was set to 6. For each of these experiments, the number of individuals in each population was 100 and each experiment was run for 500 generations. These experiments were repeated 30 times.

As discussed earlier, GP may accept any number of functions in creating the individual trees. The above experiment limited the selection to just four arithmetic operators. Boolean logical operators may also be useful in the evolved programs. The combination of arithmetic and boolean operators is not always a sensible idea. Boolean operators return a logical value — a 0 or 1 whereas the mathematical operators manipulate the numbers given by each feature (valued between 0 and 1 as the feature values are all normalised). Preliminary experiments indicated that the inclusion of too many boolean operators, such as AND and OR tended to create over-simplified programs and so were not included. However, operators that compare two feature values — such as less than (lt) and greater than (gt) may be of more interest to such programs. As in the GA experiments in the previous chapter, the relationships between the features may be of interest. A strong 0 output,

[2]As a visual aid, the first principal component of each MFCC is highlited in bold font

should *Residual*(2) be greater than *Centroid*(1) (an 'unexpected' relationship), for example, could be useful for determining an unusual relationship between features — thus possibly identifying a particularly difficult timbre. These two operators ($<$ and $>$) were combined with the mathematical operators $+$, $-$ and $*$, the constants 2, 3 and 4 and a random constant generator which can give any floating point number between 0 and 1[3]. As the targets range from 1 to 5, these constants should enable the programs to reach these targets numerically. It was found in early experiments that the use of constants and boolean operators may cause the GP to fall into a local minimum with a very small tree giving a mediocre result that is not dependent on any features. This can easily happen if a tree is developed early on that gives a constant output such as a 1. If this happens the fitness abruptly drops (as all pianos are now correctly identified but no other instrument will be) and the program may fall into a local minimum that it cannot escape. This was prevented in the fitness function by heavily penalising any tree that did not give a different answer for each sample value. This penalty was implemented by forcing the fitness of that tree to 3000 — a fitness higher than if all samples are incorrectly identified. Likewise, program strings that contained less than 50 characters were penalised to prevent very small trees being included.

In addition to a change in function selection, an alternative fitness method was considered. Up to this point, the fitness was compared against a target of values 1 to 5, each representing a different instrument. This is a very specific for GP to match. As discussed in Koza (1992b), judgement must always be exercised in creating fitness cases for the given problem. It is possible that it may be easier for GP to evaluate values to a range of target values rather than an individual number. To allow more scope in the evaluation of the trees, the fitness values were transformed to a range of fitness values. Thus in this case the following fitness ranges were implemented:

- [1 - 10] = Piano
- [11 - 20] = Violin
- [21 - 30] = Flute
- [31 - 40] = Trumpet
- [41 - 50] = Guitar

[3]mydivide was not included in these runs as it was causing problems with the logical operators, plus it may be effectively implemented by combing times with a constant between 0 and 1

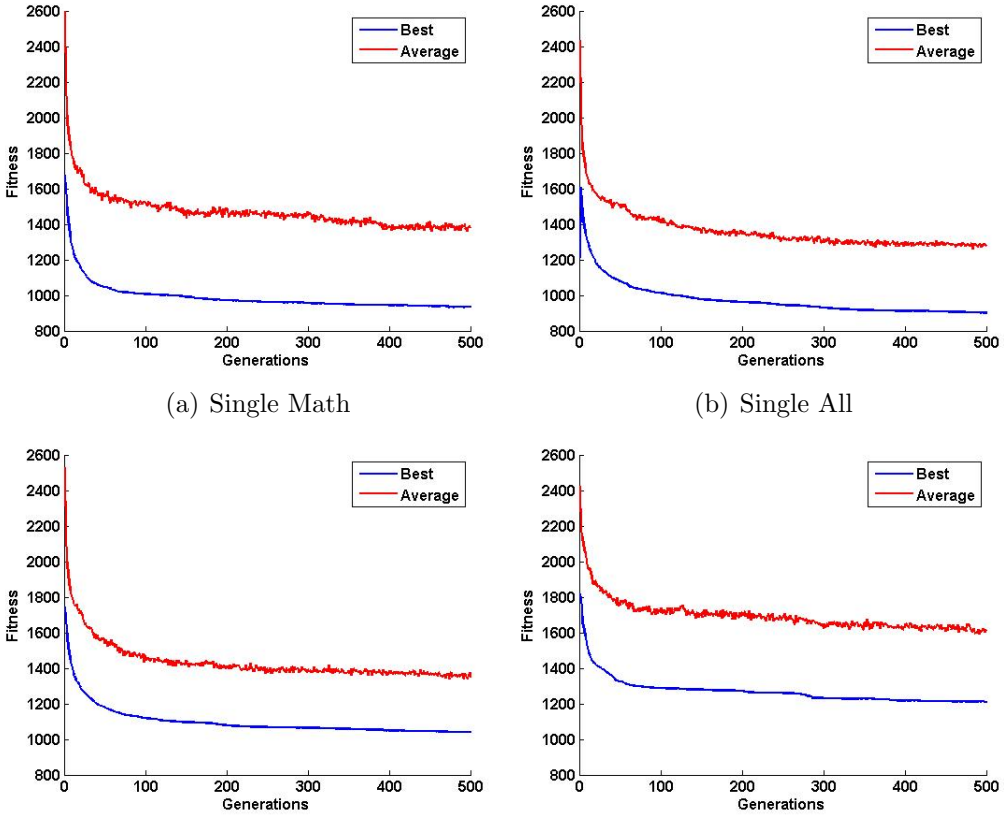
To ensure that the program trees can generate values that high, further constants — namely, 5, 10, 20 and 40 were included in each run. This *Range* fitness method may expand the search space, enabling the GP run to find a solution more easily.

These methods were all implemented using the GP and run 30 times. The four separate GP runs consisted of:

- Single Math — original fitness method using just arithmetic operators
- Single All — original fitness method using arithmetic and boolean operators *greater than* and *less than*
- Range Math — range fitness method using just arithmetic operators
- Range All — range fitness method using arithmetic and boolean operators *greater than* and *less than*

The training fitness of each individual was noted and a measure of the average and best fitness for each generation was stored. As when examining the GAs in the previous chapter, the changes in the average and best fitness over the 500 generations may be examined as an indication of how the GP run is behaving. A plot of the average and best training fitness for each of the fitness methods is shown in Figure 7.2. It is clear from these plots that the average fitness does not converge towards the best fitness substantially over the 500 generations. This implies that the population is still diverse, with both good and bad individuals remaining in the population. It may be noted however, that in each case the best fitness appears to be still decreasing — indicating that the search may be continued for more generations as a global minimum does not appear to have been found. These plots also indicate that a lower fitness is obtained from using the Single fitness methods rather than the Range fitness methods. After 500 generations, the lowest training fitness observed appears to be from the Single All fitness method. This fitness remains over 800 which may be still a poor result, although it must be remembered that this is *averaged* over the 30 runs, therefore the best individual training fitness is likely to have achieved a much lower value than this.

To test the classification accuracy of the evolved trees, the corresponding programs were used to classify a set of test samples. The trees evolved by GP above were trained using 2706 instrument samples. These were taken from the Cross Validation sets 1-9 as described in the previous chapter. To test the evolved trees, initially the one-octave test set was used. This set (as described in the previous chapter) consisted of one octave of notes from each of the five



(a) Single Math

(b) Single All

(c) Range Math

(d) Range All

Figure 7.2: *Average vs. Best fitness over 500 generations for the four fitness methods*

instruments. The 30 best program trees from each of the four fitness methods were used to evaluate each of the notes. The classification results are given as the percentage of times each program correctly identifies an instrument sample. A boxplot of the results obtained for each of the four methods is shown below in Figure 7.3. It can be seen from this plot that the results are somewhat disappointing, as the mean accuracy of classification for each of the methods is around 40%. The Single All method gives the highest mean accuracy of 43.49%. The range of accuracy achieved by each of the methods is quite large however. The highest accuracy achieved using any of the programs evolved is 73.85% from a program using the Range All methods. This was closely followed by an accuracy of 69.23% using the Single All method. These are clearly much higher than the average classification results, demonstrating that when GP is run a number of times, a good classifier will emerge. Thus it appears that using a combination of arithmetic and boolean operators as functions may result in the most accurate classifications.

To examine this point further, the above classifications were repeated

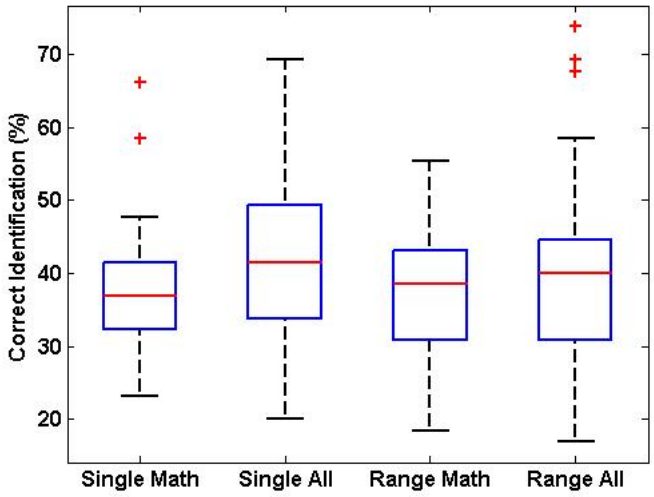


Figure 7.3: *Classification results for the one-octave test set by each of the four fitness methods*

using a larger unseen test set. As Sets 1-9 were used to evolve the programs, Set 10 consists of unseen samples that may be used to test the resultant strings. Set 10 contains 300 samples of various pitch and dynamic ranges from each of the five instruments. As the target or correct classification of each sample is known, the classification result may be calculated as the percentage of times each string correctly identifies one of the 300 unseen samples. The classification results of each of the four fitness methods may be seen in Figure 7.4. It is clear from this plot that a higher classification accuracy may be achieved by using these more general test samples rather than the samples that are specific to one octave. This result is similar to that achieved in the previous chapter using the GA and MLPs. This demonstrates again that constraining the test samples according to pitch and dynamic in this way does not test the generalisation of the classifier. The highest classification accuracy achieved in this experiment was 88.3%, obtained using the Single All method. Again the Range All method also achieved a high accuracy of 80.67%. Notably, the mean accuracy of classification is higher for this larger test set with the Single All method displaying the highest mean accuracy of 64%. The Range All method actually achieved the lowest mean classification accuracy of just 49.63%. These results combined with those for the one octave test and the best and average fitness imply that of the methods considered, the most consistent results were obtained using the Single fitness method with a combination of arithmetic and boolean operators.

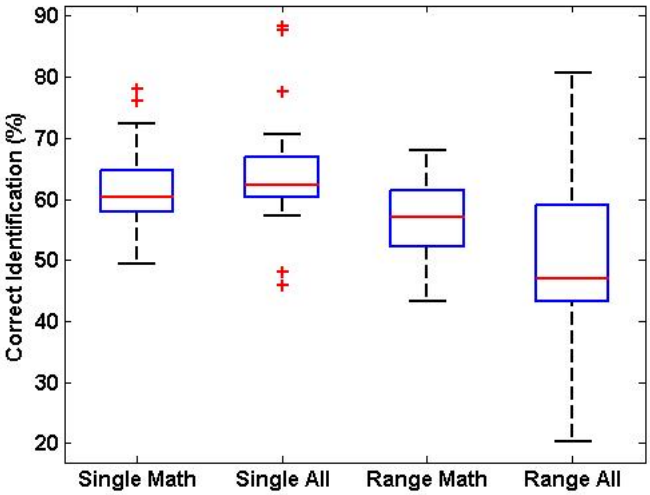


Figure 7.4: *Classification results for the 300 element test set by each of the four fitness methods*

Feature Analysis of Evolved Trees

The above four methods evolved 30 program trees each. As seen from the classification results above, these programs obtained varying degrees of success in classifying new sample sounds. This section examines the features chosen most often for the 30 most successful programs chosen by each fitness method. Table 7.3 displays the features most often chosen for the Single Math fitness method. As can be seen from this table, the first principal component of the Centroid Envelope was found to be the most often chosen feature followed the first principal component of MFCC4 and the first principal component of the Temporal Envelope. This is very similar to the result for the Single All fitness method shown in Table 7.4. For this method the first component of the Centroid Envelope was again the most often chosen followed by the first component of the Temporal Envelope and the first component of MFCC4.

The most common chosen features for the Range Math and Range All fitness methods are shown in Table 7.5 and Table 7.6 respectively. Again, these show a strong correlation with the results from the Single fitness methods. Although there may be a difference in the order of prominence, the first principal component of the Centroid Envelope, the Temporal Envelope and MFCC4 appear in the top five features chosen by this method. There are also similarities in that further principal components of the Temporal Envelope were chosen, along with first components of a number of other MFCCs — notably the early MFCCs. These results re-inforce the values obtained

Table 7.3: *Number of times the top 10 features were selected across 30 runs using the Single Math Fitness method*

Instances	Feature No.	Feature
49	20	Cent1
37	44	MFCC4-1
28	16	Env1
20	21	Cent2
19	17	Env2
18	28	Spec2
18	41	MFCC3-2
18	45	MFCC4-2
17	40	MFCC3-1
16	33	MFCC1-2

Table 7.4: *Number of times the top 10 features were selected across 30 runs using the Single All Fitness method*

Instances	Feature No.	Feature
88	20	Cent1
77	16	Env1
64	44	MFCC4-1
56	36	MFCC2-1
36	17	Env2
32	21	Cent2
28	2	Rolloff
25	28	Spec1
19	3	Brightness
18	60	MFCC8-1

from GP experiments shown earlier in the chapter in Table 7.1 and Table 7.2 which also found the first principal component of the Centroid Envelope and Temporal Envelope along with the low valued MFCCs to emerge as the most important features.

Table 7.5: *Number of times the top 10 features were selected across 30 runs using the Range Math Fitness method*

Instances	Feature No.	Feature
84	44	MFCC4-1
83	36	MFCC2-1
60	16	Env1
60	20	Cent1
28	45	MFCC4-2
26	17	Env2
22	28	Spec1
21	33	MFCC1-2
20	19	Env4
19	18	Env3

Table 7.6: *Number of times the top 10 features were selected across 30 runs using the Range All Fitness method*

Instances	Feature No.	Feature
42	16	Env1
36	3	Brightness
36	20	Cent1
35	44	MFCC4-1
29	36	MFCC2-1
27	17	Env2
17	14	No Peaks
15	28	Spec1
15	40	MFCC3-1
12	5	First Attack Time

Although there appears to be significant consistencies in the features being chosen for the evolved programs, the training and classification fitness performances remain quite poor. As we have seen, the training fitness was still decreasing after 500 generations. In addition to this, it may be noted that a number of the best evolved program trees had reached the maximum allowed depth of 15. To attempt to increase the performance of the evolved program trees, the values of these parameters were relaxed.

7.3.3 Extended GP Run

The method that achieved the best fitness, Single All, was run again 30 times with a population of 100 for 2000 generations. In this run, the dynamic level of the trees was set to 15 and the realmaxlevel was set to 25. This would allow much larger trees to be created if their fitness was better than those in the current generation. A plot of the averages of the best versus average training fitness for all 30 runs is shown in Figure 7.5. This plot shows that the average and best fitness do continue to decrease after 500 generations and appear to level out before reaching 2000 generations. The average and best fitness still do not converge however, indicating that the populations are still diverse. The average of the best training fitness indicates that there is not a great improvement from the individuals evolved over 500 generations. However, in examining the strings individually, it was found that three individuals had a best training fitness of under 200 with the best achieving a low fitness of 149.

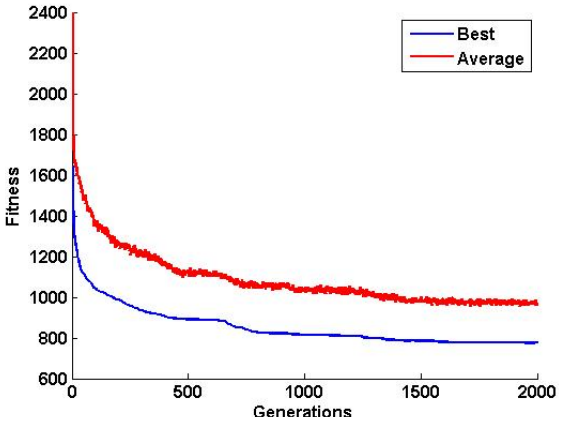


Figure 7.5: *Average vs. Best fitness across 2000 generations*

The evolved programs were again tested by using them to classify a test set of unseen samples, which again consisted of the 300 samples from Set10. A boxplot of the classification results is shown in Figure 7.6. This again shows a wide range of classification accuracy between the programs. The best classification accuracy of 94% was achieved by the program from individual nine. Further analysis of the individual programs and the features chosen is given in the next section.

Analysis of Evolved Individuals

The 30 programs represent the best trees or individuals evolved from 30 separate runs. As seen above, these individuals vary in terms of their fitness.

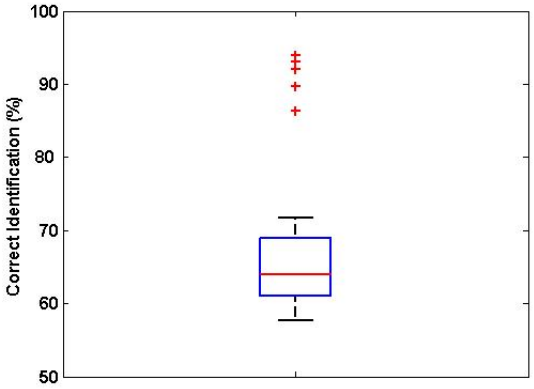


Figure 7.6: *Classification results for the 300 element test set for extended run (over 2000 generations)*

Table 7.7 lists the size of these individuals in terms of depth and number of nodes. This table also displays the training fitness both in terms of absolute fitness (how many out of 2706 were misclassified) and the percentage of correct training identifications, along with the test classification fitness as a percentage of correct identification by each of these individuals on the test set. It is clear from this table that both the training and testing accuracy vary between the different individuals. It must also be noted, however, that each individual only experiences a slight decrease in accuracy from training to test fitness — and in fact the best individual, number nine, experiences a slight increase in accuracy from training to test data. This implies that while the results of the individuals may vary, each individual is approximately as good (or as weak) at recognising a new sample as it was at recognising the training samples. This shows that the individuals are generalised over the data set, as opposed to being over-fitted to the training data.

It may be seen from this table that the three most successful programs, both in terms of low training fitness and high classification accuracy were from individuals 2, 9 and 15. It may also be noted from this table that these are three of the largest evolved trees — with depths of 19, 23 and 20, and 245, 241 and 173 nodes consecutively. This demonstrates that increasing the allowed depth size of the trees was necessary to create more accurate program trees. Although a test accuracy of 94% is quite high, it may be noted that it is still not as high as the classification accuracy of the GA-MLP classifiers described in the previous chapter.

Table 7.7: *The depth, number of nodes, training fitness and classification accuracy for each of the 30 best evolved strings*

Ind.	**Depth**	**#Nodes**	**Train(/2706)**	**Train(%)**	**Test Class(%)**
1	4	9	979	63.8	60.33
2	19	245	149	94.5	93
3	17	79	953	64.7	61
4	17	59	874	67.7	64
5	11	33	1014	62.5	58.33
6	20	81	923	65.9	63
7	16	69	1051	61.2	57.67
8	19	75	814	69.9	66.67
9	23	241	174	93.6	94
10	15	87	771	71.5	67.33
11	14	67	316	88.3	86.33
12	18	35	948	65	61
13	18	79	973	64	60
14	23	95	979	63.8	59
15	20	173	155	94.3	92
16	23	199	773	71.4	68.67
17	17	85	775	71.4	68.67
18	19	89	999	63.1	58.33
19	22	121	704	74	71.67
20	20	121	860	68.2	64
21	21	125	970	64.2	61.33
22	7	21	1007	62.8	57.67
23	20	77	890	67.1	64.67
24	16	147	762	71.8	69
25	22	123	766	71.6	70
26	10	25	882	67.4	62.33
27	21	147	779	71.2	67.67
28	19	47	894	66.9	62.67
29	23	105	956	64.7	61
30	23	225	193	92.8	89.67

Feature Analysis

The 30 programs evolved over 2000 generations were again analysed to determine which features were most often used among the trees. The ten features most often chosen are displayed below in Table 7.8. This shows that again the first principal component of the Centroid Envelope and the Temporal Envelope and the lower valued MFCCs — in particular the first principal component of MFCC4 were the most common features among the best programs. The more successful programs were found to use these features several times within the one program tree. These particular features have been found to be the most common among all the GP runs described in this chapter. The consistency of selection of these features implies that these features are very important for accurate instrument classification.

Table 7.8: *Number of selections of the top 10 features among the 30 strings evolved over 2000 generations*

Instances	Feature No.	Feature
137	20	Cent1
98	16	Env1
86	44	MFCC4-1
81	36	MFCC2-1
71	17	Env2
58	28	Spec1
51	21	Cent2
39	18	Env3
37	35	MFCC1-4
35	33	MFCC1-2

7.3.4 Extended Size of Population

The previous section described the improvement obtained in the classification of the individuals by increasing the length of the run — or increasing the number of generations. Although this did improve individual results, it was clear that there was still much variation within the population. From Figure 7.5 it may be seen that little improvement is made on the fitness after 100 generations. This section again attempts to improve the performance of the entire population, this time by using a larger population evolved over fewer generations. Thus the experiments in this section run GP 30 times with a population of 500 over 100 generations. Having more different individuals in any given generation would make it more likely that a better individual may be found in the next generation.

The best and average fitness is plotted in Figure 7.7. Again this shows that most of the improvement in the fitness occurred within the first 40 generations. Again the best and the average fitness do not converge, indicating that there is still much diversity among the 30 populations. The training and testing fitness of the best three evolved programs are shown in Table 7.9. This shows that the best classification accuracy by any program in this experiment was 88.3%. This is not quite as accurate as the program evolved over 2000 generations that achieved 94% accuracy. Nevertheless, this is a generalised program that has a test accuracy that is almost as high as its training accuracy (88.7%). Although this is a slight decrease in accuracy, it may be noted that this experiment took less than three days to evolve all 30 GP programs (in comparison to over 10 days for 2000 generations).

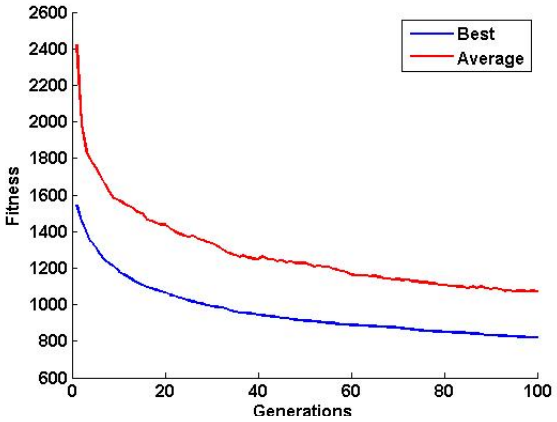


Figure 7.7: *Average vs. Best Fitness across 100 generations with population 500*

Table 7.9: *The depth, number of nodes, training fitness and classification accuracy for the 3 best evolved strings with population 500 over 100 generations*

Ind.	Depth	#Nodes	Train(/2706)	Train(%)	Test Class(%)
25	14	93	305	88.7	87.3
22	17	141	579	78.6	76
23	13	47	606	77.6	76

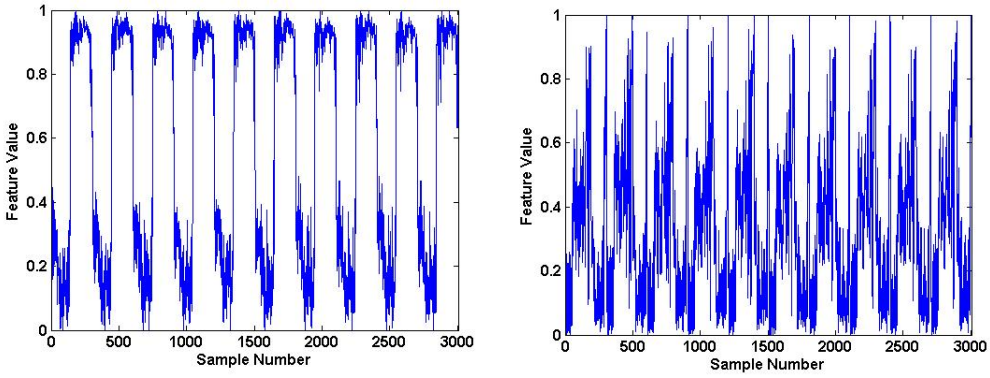
The next section examines in more detail the most commonly chosen features by both GP and the GA in the previous chapter.

7.4 Comparison of Features Chosen

Experiments in both the previous and the current chapter have examined which of the 95 features emerged as most important for instrument identification. This section examines the variation between samples for some of these features to determine what may make them so useful to the classifers. The GP experiments in this chapter found that the first principal component of the Centroid Envelope, the Temporal Envelope and MFCC4 were the most commonly chosen features. Analysis of the genomes from the GAs in the previous chapter also found the first PC of the Centroid Envelope and to a lesser extent that of the Temporal Envelope to be important. A plot of the first component of the Centroid Envelope and the Temporal Envelope of all 3006 samples ordered according to sets (Set1 followed by Set2 etc.) is shown in Figure 7.8. Clearly there is a periodicity emerging in these plots. This may be explained by the cyclical order in which the sets were formed. Before filling the sets, the RWC samples were arranged in the order of piano, violin, flute, trumpet, guitar and then followed by the MUMS samples. Filling each set with the 10th sample (ie. note sample) from this list meant that each set contains approximately 54 piano samples followed by approximately 75 violin, 44 flute, 35 trumpet and 70 guitar samples, and finally approximately 20 MUMS samples (again in the order piano, violin, flute and trumpet). Thus for features that are indicative of the instrument, this periodicity should be evident when the samples are lined up like this.

The cycles in the Centroid Envelope are particularly evident. To inspect this further Figure 7.9 displays the values for the first 300 values of Set1. Although the five instrument regions are not all distinct, there is a very clear distinction between the piano and violin section compared to the other instruments. A similar divide is seen in each set, explaining the periodic plot of this value across all 3006 samples. In contrast to this, Figure 7.10 displays the values for Inharmonicity across all 3006 values. The lack of periodicity in this figure indicates that this feature is not consistent for a particular instrument, thus explaining why it was not chosen very often by either evolutionary technique.

Similar plots for the MFCCs produced varied results. Low valued MFCCs such as MFCC4-1 were the most often chosen feature using GP whereas high valued MFCCs such as MFCC13-4 were favoured by the GA. A plot of these two values across all samples is given in Figure 7.11 with accompanying plots of the first 300 samples in Set1 displayed in Figure 7.12. It is clear from these



(a) Centroid Evolution (b) Temporal Envelope

Figure 7.8: *Plot of the variation across all 3006 samples for the first principal component of the Evolution of the Centroid and the Temporal Envelope*

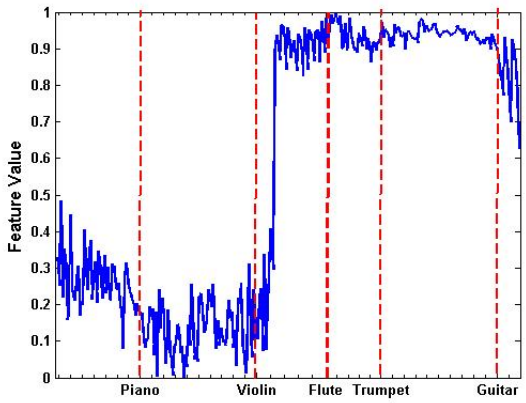


Figure 7.9: *Plot of the variation across 300 samples for the first principal component of the Evolution of the Centroid*

plots that the lower MFCC value gave the more consistent periodic plots — the five distinct intervals may be associated with each of the instruments. Although there is some distinction between the instrument for MFCC13-4, it is not as pronounced as that in MFCC4-1. This implies that while GP strongly favours features with this 'instrument periodicity', GAs may not.

7.4.1 Limiting GP to 'Popular' Features

As seen in the experiments in this chapter, the individuals created using GP do not, in general, use all 95 features due to limitations on their size. Thus along with deciding how to combine the features used, GP must also decide which features to use. It may be seen from the experiments in this chapter that there is a strong tendency for GP to choose certain features over others.

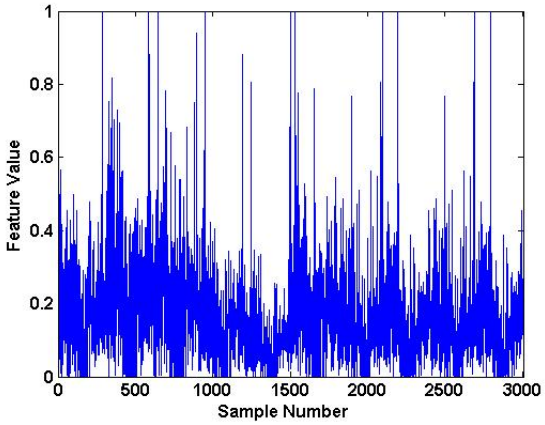
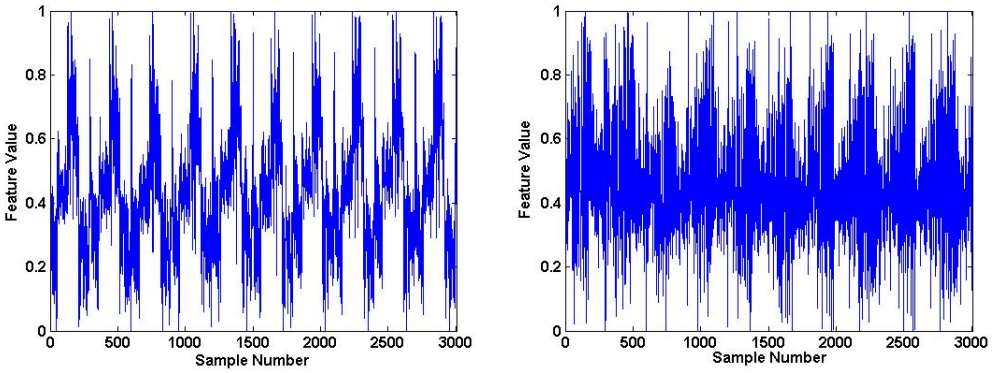


Figure 7.10: *Plot of the variation across all 3006 samples for Inharmonicity*

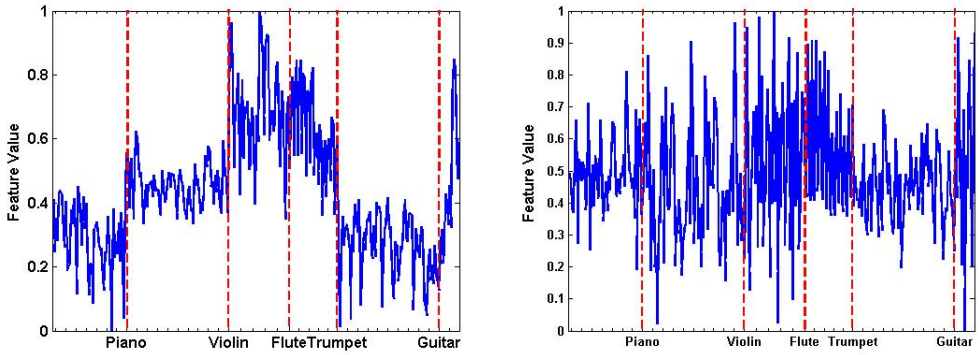


(a) MFCC4-1 (b) MFCC13-4

Figure 7.11: *Plot of the variation across all 3006 samples for the first principal component of MFCC4 and the fourth principal component of MFCC13*

Therefore as an attempt to reduce the variation within the individuals, GP was run again but with a limited number of features. The features were limited to the most often chosen features according to the 'All' category in Tables 7.1 and 7.2. Only features that appeared 20 or more times were included in this experiment. This limited the features to the 14 values listed in Table 7.10.

The best and average training fitness averaged over the 30 runs are shown in Figure 7.13. Again this shows that the average and best fitness do not converge, indicating that there is still a lot of diversity in the population. The best individual result was found from individual number 20 which only made mistakes on 281 of the 2706 samples, giving a training accuracy of 89.6%. When this was tested with the 300 unseen samples of Set10, it was found to have a classification accuracy of 91.3%. The training and testing of the



(a) MFCC4-1 (b) MFCC13-4

Figure 7.12: *Plot of the variation across the first 300 samples for the first principal component of MFCC4 and the fourth principal component of MFCC13*

top three evolved programs is shown in Table 7.11 indicating that these three programs achieved test accuracy rates of almost 90%. This indicates when evolving classification programs with GP such as these, a large number of features may be removed with only a slight decrease in classification accuracy. This implies that a large number of features used in previous classification studies may add unnecessary calculations and actually be superfluous to the identification of musical instruments.

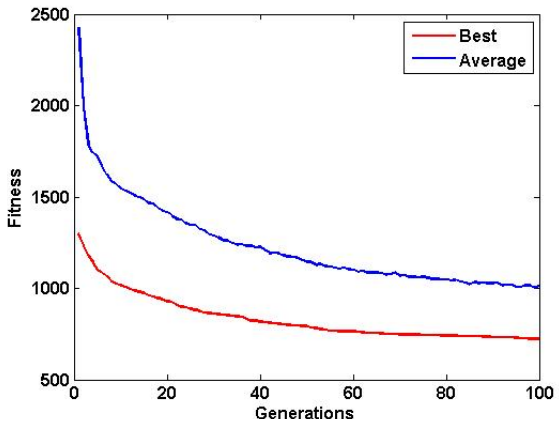


Figure 7.13: *Average vs. Best Fitness across 100 generations with population 500 for limited feature data*

Limited Features Extended Run

It is clear from the results in Table 7.11 that the test classification accuracy of the best strings evolved using a population of 500 over 100 generations with limited features is higher than the classification results using these same

Table 7.10: *The top 14 features as found from the Feature Analysis experiment displayed in Tables 7.1 and 7.2*

Feature No.	#Instances	Feature
2	22	Rolloff
9	23	MIRCent
10	20	MIRSpread
16	57	Env1
17	27	Env2
18	35	Env3
20	80	Cent1
21	40	Cent2
28	41	Spec1
33	29	MFCC1-2
37	23	MFCC2-2
40	20	MFCC3-1
42	22	MFCC3-3
44	51	MFCC4-1

Table 7.11: *The depth, number of nodes, training fitness and classification accuracy for the 3 best evolved strings using limited features with population 500 over 100 generations*

Ind.	Depth	#Nodes	Train(/2706)	Train(%)	Test Class(%)
20	15	57	281	89.6	91.3
26	19	221	273	89.9	87.6
29	12	173	283	89.5	88.66

parameters with the full set of features as shown in Table 7.9. However it was also shown that the best program was created using all features over a longer GP run (2000 generations). Thus as a final experiment, GP was run again with a population of 100 over 2000 generations, this time only using the 14 features listed in Table 7.10. The training and classification results are shown in Table 7.12

Table 7.12: *The depth, number of nodes, training fitness and classification accuracy for each of the 30 best evolved strings using only the top 14 features*

Ind.	Depth	#Nodes	Train(/2706)	Train(%)	Test Class(%)
1	19	189	669	75.3	73
2	17	151	170	93.7	93.3
3	24	145	441	83.7	82.3
4	25	125	758	72	68.7
5	22	195	249	90.1	89.7
6	24	199	678	74.9	71.7
7	20	161	246	90.9	89.3
8	25	125	771	71.5	69.7
9	21	263	151	94.4	94.3
10	16	99	965	64.3	61.3
11	22	223	707	73.9	69.7
12	18	181	659	75.6	72.7
13	17	115	266	90.2	90.3
14	21	101	1040	61.6	56.4
15	25	241	726	73.2	68.7
16	25	253	662	75.5	74.3
17	18	155	747	72.4	68.7
18	24	259	710	73.8	70.3
19	23	193	744	72.5	68.7
20	24	149	901	66.7	64.7
21	24	327	329	87.8	86.7
22	20	57	777	71.3	67.7
23	22	293	174	93.6	93.7
24	22	295	693	74.4	71
25	23	169	632	76.7	75
26	21	203	703	74	71.3
27	24	263	789	70.8	69.3
28	23	231	238	91.2	89
29	20	101	887	67.2	65.3
30	22	229	460	83	83.7

The best program evolved in this run was string number 9 which had a training fitness of 94.4% and a test classification accuracy of 94.3%. This is a very slight improvement on the best classification accuracy using all features. The average of the test classification accuracy has risen to 75.67% as opposed

to 68.2% for the strings incorporating all 95 features. Six individual programs achieved a test classification accuracy above 89%. Hence, limiting the features to those most often chosen has increased the classification accuracy of the resultant evolved programs. This again demonstrates that accurate instrument identification is dependent on a careful selection of features, rather than merely incorporating as many features as one can in a given classifier.

From the depth of the programs shown in Table 7.12 and Table 7.7 it appears that limiting the features used by GP has created larger rather than smaller programs. Unfortunately, this makes the resultant program trees more difficult to visualise. The following section discusses this issue of tree visualisation and the problem of ever-expanding trees or *bloat*.

7.5 Tree Visualisation and Bloat

One of the most significant advantages of GP is that the resultant program is accessible to the user. It is this accessibility that has allowed the programs evolved throughout this chapter to be analysed in terms of the features used. These features may be analysed as each program is described by a program string. This string in turn may be drawn as a program tree. GPLAB includes the function `drawtree` for drawing individual program trees. An example of a program tree is shown in Figure 7.14. This is the tree that corresponds to one of the best training fitness for the GP run using all feature data with a population of 500 over 100 generations, or string number 23 in Table 7.9. The string itself is :

minus(minus(minus(times(plus(X10, 4), plus(X20, X21)), X43), X21), minus(X40, minus(X45, minus(X40, gt(minus(X7, minus(X36, X19)), minus(minus(minus(minus(times(4, plus(X20, X21)), X17), X44), gt(X39, minus(minus(minus(minus(2, X44), X44), X16), X3))), X44))))))

This program is easily drawn, as shown in Figure 7.14, as it has only a depth of 13 and contains just 47 nodes. By comparison, the best testing tree of that experiment was string number 25, which has a depth of 14 and contains 93 nodes. The program tree of this individual is shown in Figure 7.15. Its full string is:

plus(minus(times(X20, minus(plus(X20, X21), X44)), times(minus(X36, plus(gt(X86, plus(X2, plus(lt(minus(plus(X20, minus(plus(X20, X21), X44)), X28), X91), plus(X3, X16)))), plus(X1, X33))), X40)), plus(times(gt(minus(plus(X20, X46), X40), plus(X2, plus(lt(minus(plus(X20, X90), plus(lt(minus(plus(X20, 2), X13), X35), X6)), X10), plus(X45, X16)))), minus(times(X20, minus(plus(X20, 2), X44)), times(X17, plus(lt(minus (plus(X20, X21), X28), X10), plus(X16, X16))))), plus(lt(X42, 2), plus(X16, X44))))

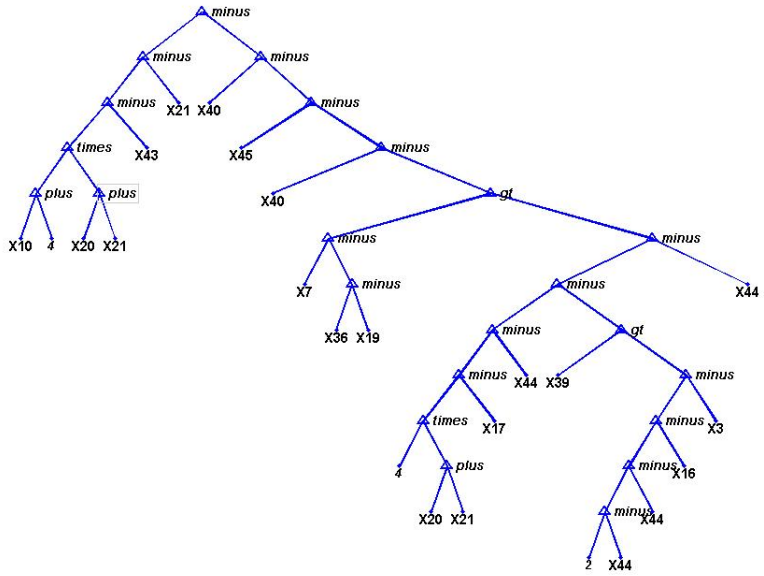


Figure 7.14: *Best Training Program Tree (program 23 in Table 7.9) using all data with a population of 500 over 100 generations*

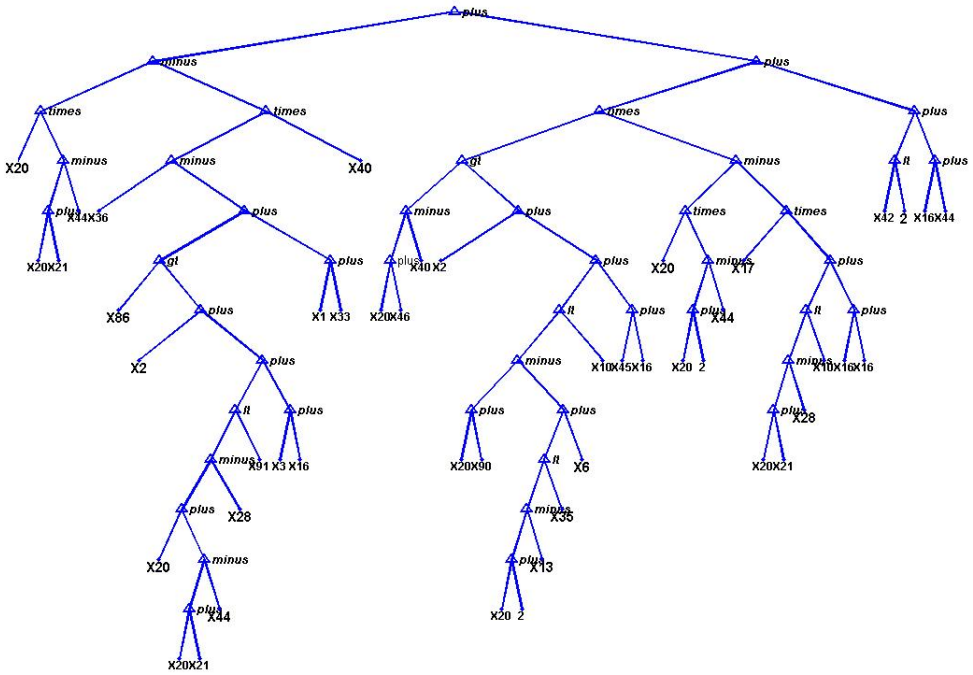


Figure 7.15: *Best Testing Program Tree (program 25 in Table 7.9) using all data with a population of 500 over 100 generations*

It is clear that although the tree is Figure 7.15 is only one level deeper than that in Figure 7.14, it is already a much more complicated tree. In creating larger trees in the experiments throughout this chapter it was evident that it was not practical to draw these larger trees as they were not clearly readable. For example, the best tree evolved using all features over 2000 generations, with 94% testing accuracy was found to have a depth of 23 with 241 nodes. The string of this program is:

plus(plus(X17, plus(plus(X20, plus(lt(plus(plus(plus(plus(lt(X36, X17), X16), X28), X16), X15), plus(X20, lt(X18, X20))), lt(X50, plus(X40, minus(X17, lt(plus(X23, minus(X16, lt(X76, lt(plus(lt(X36, X17), X42), plus(plus(X36, lt(X23, plus(plus(X8, lt(X18, plus(X39, plus(X5, lt(X18, X20))))), X11))), X32))))), lt(plus(lt(X56, X17), X5), plus(plus(X5, lt(X18, X20)), X5)))))))), X21)), plus(X20, lt(X17, plus(lt(plus(plus(plus(X20, X17), X16), X40), plus(lt(X85, plus(plus(lt(plus(plus(X36, X20), X40), plus(lt(X20, plus(X17, X23)), lt(X17, plus(X5, lt(X84, plus(lt(X36, X34), X20)))))), lt(X56, X12)), lt(lt(X36, X44), lt(X17, X20)))), lt(X17, plus(X21, lt(X36, plus(lt(X36, plus(X20, X21)), X5)))))), lt(X77, plus(plus(lt(X20, plus(X93, lt(X16, plus(X75, minus(minus(X50, X1), lt(X21, plus(lt(X36, X64), X16)))))), minus(lt(X87, lt(X18, plus(lt(plus(plus(0.57809, X45), X44), plus(X36, X20)), lt(X18, X8)))), lt(X21, X20))), minus(lt(X8, lt(X14, plus(lt(X11, X41), minus(lt(X7, plus(lt(plus(plus(plus(X20, X17), plus(X20, X21)), plus(X11, X36)), plus(lt(X87, plus(X44, X11)), lt(X40, plus(X21, lt(X36, X20))))), lt(0.57809, plus(lt(X11, X41), minus(lt(X44, lt(X64, X20)), 0.57809))))), lt(X11,X35))))), lt(X11, 0.40868))))))))

This tree was too complicated to draw legibly using the `drawtree` function in GPLAB. Similarly, the best tree found using the limited set of 14 features across 2000 generations was found to have a depth of 21 with 263 nodes which was again too large to draw.

The growth in length or size of the individual programs evolved using GP is a well documented problem commonly know as *bloat*. Bloat can become a problem if trees grow much larger in size without any significant improvement in fitness. It contradicts the principle of *'Occam's Razor'* or the *'law of parsimony'*, which states that 'entities should not be multiplied beyond necessity' (Affenzeller et al., 2009). If there are no size restrictions in GP, programs may become unnecessarily large over many generations. This can be due to the inclusion of *introns* — redundant code that is of no benefit to the solution yet adds to its size. Excessive bloat is prevented in the experiments in this chapter by restricting the depth of the programs. Smaller programs are also encouraged as described in Section 7.2 using the selection method `lexitour` which favours shorter programs.

Although bloat may be a problem with GP in general, it must be noted from the results in this chapter that the two best evolved programs with 94.3% and 94% accuracy had depths of 21 and 23 (quite large programs). When the maximum program depth was limited to 15 the classification was not this accurate, it only managed this accuracy when the maximum depth was set to

25. Also, it was noted that the average classification accuracy was increased when only the top 14 features were used. By reducing the number of features used from 95 to 14, it was thought that the programs might not tend to grow this large. In contrast to this, the reduction in features coincided with an increase in average depth from 18 to 22 and an average number of nodes from 103 to 188. From Table 7.12 it may be seen that 10 of the 30 best evolved strings had a depth of either 24 or 25, whereas none of the strings evolved using the full feature set were this deep. Thus it appears that for creating a musical instrument classifier, the programs benefit from growing in size. This demonstrates that although, unnecessary bloat must be discouraged, programs should not be forced to be too small as this may prevent some of the best solutions from being found.

7.6 Conclusion

This chapter used GP to develop a musical instrument classifier. Section 7.1 described the basic operation of GP. Section 7.2 described the implementation of GP as used in experiments throughout this chapter, the results of which were discussed in Section 7.3. The most prominent features that emerged in both this and the previous chapters were examined in Section 7.4. Finally, Section 7.5 took a brief look at some of the program trees evolved throughout the chapter and discussed them in terms of tree size.

The training fitness from experiments run over 500 generations was found to be still decreasing which prompted an increase in the number of generations to 2000. In addition to this, it was found that from the 500 generation experiments a number of the best individuals were at the maximum depth of 15, hence the maximum allowable depth was increased up to 25. The difference between the average and best training fitness values even after 2000 generations indicated that the populations were still diverse and so a global minimum may not have been reached. A further run with an increase in the population up to 500 over 100 generations found that this caused no reduction in the diversity of the population. As with GAs, GP has the tendency to evolve a general acceptable solution to a problem rather than a global optimal solution.

The highest classification accuracy of the best evolved tree program using all features was 94%. This was slightly outperformed by the best evolved program using just the top 14 features at 94.4%. The average of the classification of the top 30 programs was also higher for those evolved using just 14

features than those evolved from all 95 features. Thus better classifiers were created using a small carefully selected set of features, rather than the whole set. This demonstrates that a classifier such as this one may be improved by reducing the number of features implemented with it. This reduction of timbral features was one of the main aims of this thesis — it has been shown here that for a GP classifier, the classification accuracy may be increased by decreasing the number of features used. This result implies that further experiment in musical instrument identification should carefully consider which features should be incorporated into any classifier used.

This 94.4% correct classification is not as accurate as the GA-MLP classifier developed in the previous chapter. However, as discussed at the beginning of this chapter, GP has the advantage over GA in that it can evolve the entire solution to the problem. The musical instrument classifier developed in this chapter that can classify a musical note with 94% accuracy is the direct result of GP; this solution did not need any PCA in the fitness function to help it evolve, nor any MLP to help it to classify new samples. In addition to this, once evolved, the resultant program may be simpler to understand than the MLP classifier used previously. An MLP is an unintuitive 'black box' classifier, whereas by specifying the primitive set, it was ensured that the evolved GP program is limited to the feature values and a small set of simple functions. Section 7.2.1 described how the concept of sufficiency may not be guaranteed from this primitive set until results have been found. Although not a perfect score, a classification accuracy of 94% implies that this primitive set of features and function is sufficient for representing a solution to the problem of musical instrument recognition.

The high diversity both within the final populations and between the results of the best evolved genomes would imply that further experiments in this area could improve on the results discussed here. The main issue in implementing these experiments is in the amount of computation required for an individual run. Although the resultant classifier may consist of a simple program, the evolution of it may be very computationally expensive. The experiment of evolving 30 program trees, using the full set of data, with 100 individuals over 2000 generations took approximately 10 days to compute. The same experiment using less data (14 as opposed to 95 features) took just under 8 days to compute. As discussed in the previous chapter, evolving the genomes with the GA took an even longer amount of time — approximately 1 day for each genome, each evolved on one tenth of the data. These experiments were all run on Matlab on a Windows PC with a 3.2GHz Pentium

Processor. While Matlab is an excellent language for signal processing, further GA and GP experiments may be quicker and hence easier to run using a more powerful language on a different platform.

A number of common features were found to be chosen consistently from all the experiments in this chapter. Values from the Centroid Envelope and the Temporal Envelope, along with a number of the lower MFCCs were found to be consistently chosen more often than other features. In the case of the best evolved programs, these features were often chosen numerous times within the one program tree. This level of consistency in selection of features implies strongly that these features are important for instrument recognition and hence musical timbre. Although the classification results were not as high in this chapter as they were in the previous chapter, the selection of specific features is more consistent, implying that creating classifiers using GP may offer more insight into the timbre of specific instruments.

The success of a classifier may be determined by comparing its accuracy with that of the original musical instrument classifier — a human mind. The following chapter describes listening tests carried out on human subjects, the results of which may be compared to those of the classifiers developed over the last two chapters.

Chapter 8

Listening Tests

8.1 Introduction

The preceding chapters of this thesis have discussed the automatic identification of musical instrument samples. To test the success of these developed classifiers, they are compared in this chapter to a series of human listening tests. The human ability to aurally identify musical instruments may vary from person to person for a number of reasons. Before comparing the classifier results to those of the human participants a number of factors were examined that may affect our ability to recognise instruments from single notes. Are certain instruments more recognisable than others? Does musical exposure affect our ability to identify sounds? Does the presence of vibrato or the loudness of a note make it easier to identify? Are instruments more difficult to recognise at their pitch extremities? Initially, it was expected that single note samples may be difficult to identify at dynamic or particularly pitch extremities — that a very high pitched violin or flute may merely sound like an unidentifiable 'squawk'. A wide range of samples were presented to a varied participant group to examine the effect of varying such attributes on the participant's ability to accurately recognise an instrument.

This chapter describes the set-up, results and implications of these listening tests. Section 8.2 looks at some relevant tests undertaken by previous researchers and discusses the differences in motivation and method between those and the tests carried out here. Section 8.3 describes the data and choice of samples played to each participant. Section 8.4 describes the experimental set-up for the listening tests. Section 8.5 analyses the results obtained from the participants. These results are compared to those obtained from the automatic classifiers in 8.6. Finally Section 8.7 offers some conclusions to the experiments.

8.2 Previous Tests

The method of testing used in this experiment is a straightforward 'Name that Instrument' type of classification test. Similar tests have been carried out for numerous reasons with varying degrees of success in the past. In 1947 Eagleson and Eaglson (1947) compared the ability of musicians and non-musicians in identifying musical tones both directly and over a loudspeaker. No list of instrument options was given to the participants and overall their results were quite low. Surprisingly they found that although the average accuracy of the musician group was higher than that of the non-musicians, this was not particularly significant and several non-musicians gave some of the most accurate identifications. Of the instruments examined, they found the piccolo and the alto horn were the least accurately identified whereas the cymbals, violin, trumpet and bells were often correctly identified. It is possible, however, that the lack of specific instruments to choose from meant that more obscure instruments would be less likely to be chosen — particularly among non-musicians.

In the 1960's and 70's a number of tests were carried out to investigate the effect of removing temporal or spectral information on identifying instruments. Berger investigated the recognition of a number of brass and wind instruments both in their natural state and with information removed (Berger, 1964). He found that the lowest recognition rate was found after removing the upper partials, but also that recognition rates deteriorated when the attack and decay portions of the sound were removed or when the sound was played backwards. In his experiments he found the oboe to be the most easily recognised instrument and the flute and trumpet to be the most difficult. Elliott also looked at the effect of removing the attack and decay portions of the sounds (Elliott, 1975). He found that with these sections removed only the $B\flat$ clarinet, oboe and trumpet were recognised a significant number of times. He also found that the cello was the least recognised instrument overall. Saldanha and Corso performed similar processes on musical sounds in examining the recognition of 10 instruments (Saldanha and Corso, 1964). Unlike the previous studies that examined only one pitch, they looked at 3 different pitches — C4, F4 and A4 on each instrument as they expected the tonal quality to become more simplified as the fundamental frequency increases due to the removal of formants that are indicative of the instrument. They found that notes at pitch F4 were better classified than notes at the other pitches however, which does not agree with this hypothesis. Their re-

sults on the importance of the transients agree with the other works showing them to be of high importance for instrument recognition. Of the 10 instruments examined they found the clarinet, oboe and flute to be the most easily recognised and the violin, cello and bassoon to be the most difficult.

More recent studies in instrument recognition examined more than the one pitch in experiment set-ups. Kendall compared the recognition accuracy of single notes and whole phrases of music (Kendell, 1986). He analysed 3 folk songs played on 2 different clarinets, violin and trumpets and examined the effect of removing the transients from these recordings. It was found that instruments were more accurately recognised from whole phrases than from individual notes. From comparing the results he concluded that the transients were neither necessary nor sufficient for instrument recognition from whole phrases and that they were sufficient but not necessary in single note recognition. Brown performed tests to identify either the saxophone or oboe by playing sections of solo performance (Brown, 1999). This experiment was forced choice and achieved very accurate recognition rates. Probably the most comprehensive study on identifying instruments through listening tests to date was conducted by Martin in his doctoral thesis (Martin, 1999). He looked at the recognition of tones at 10 different pitches covering 27 instruments both with isolated tones and from segments of solo performances. He used 14 subjects with substantial musical experience and found that the musical segments were more easily recognised than the individual tones. The results for individual tones were quite low in general but when instruments were grouped by their instrument families the results dramatically increased. His results found the trumpet and flute to be the most accurately classified. Srinivasan et al. (2002) repeated a number of the above experiments with conservatory students and found their scores to be higher than those from previous studies. He also found that subjects who played orchestral instruments performed better than those who were pianists, guitarist or singers indicating that prolonged exposure to such instruments does indeed improve one's ability to distinguish between them aurally.

The main difference between the current experiment and those discussed above is in the number of instruments chosen and the number of presentations of each instrument to the participants. Participants are given the choice of six options in identifying a tone: flute, guitar, piano, trumpet, violin or none. Thus the number of options is kept small but we are not enforcing a forced-choice decision. Each choice of instrument represents a different instrument family so there will be no intra-family confusion, but the 'none' option allows

the participant to still have the freedom to decide if a sound is not exactly as they would expect it to be. This approach allows the participants to decide with confidence if they believe the sound to be produced by the instrument chosen, as opposed to picking the best of a bad selection of options. The instruments included were purposely picked to be of the most commonly known instruments within their respective instrument families. As many of the subjects chosen are not musicians, offering obscure choices would cause unnecessary confusion and lead to inaccurate results. A large number of samples from each instrument were included in this experiment and particular attention was paid to those within the extreme ranges of pitch as discussed in the next section.

8.3 Data

This experiment involved presenting a large number of samples to each participant. The goal of the experiment was not to gauge the ability of the candidates in recognising instruments, but to determine how identifiable the instruments are across their ranges. Thus the selection of samples presented during the experiment was carefully chosen.

8.3.1 Samples

The set of samples used during this experiment were obtained from the same set of samples described in Chapter 5 and Chapter 6 from the RWC (Goto, 2004) and MUMS (Opolko and Wapnick, 1987) databases. With such a large set of samples available it was important to select the most interesting samples. While it was necessary to bear in mind the length of time it would take each participant to complete the test, it was important to select a significant and relevant selection of samples for the experiment. Samples at pitch and dynamic extremes are more likely to pose a problem resulting in more interesting analysis. The pitch range of each instrument was divided into regions from which appropriate proportions of samples were chosen. Details of the samples chosen for each instrument are given in the following sections.

Flute

The RWC flute samples ranged from C4 to C7 and were recorded both with and without vibrato at dynamic levels *p*, *m* and *f*. These were divided into ranges low (C4 to F4) mid (F4 to F6) and high (F6 to C7). For each dynamic

level, one sample from ranges low and mid were chosen for each make either with or without vibrato. More samples were chosen from the high range, however, where two samples were chosen for each dynamic both with and without vibrato. The MUMS offered samples again from C4 to C7 from which one was taken from the low and mid ranges, and two were chosen from the high range. This gave a total of 41 flute samples.

Guitar

The RWC Database includes samples from three guitar manufacturers at the three dynamic levels. The instrument range is from E2 to E5 but a complete octave of samples is given for each string on each instrument. Playing a note whose pitch is out of the more typical range of each string (ie. a pitch that could be played more comfortably on a higher string) will significantly alter the timbre of that note. Hence, the range of the instrument was split into low (E2 to E3), mid (A3 to B5) and high (E4 to E5), but in conjunction with this, the upper 5 semitones on each string were considered 'high string' for that string. For each dynamic level one sample was selected from each of the low and mid range and two from the high range, with particular attention paid to include a selection of high string samples. No MUMS were included for the guitar. This gave a total of 36 guitar sample.

Piano

The RWC piano samples range from A0 to C8 at the three dynamic levels. The range was split into low (A0 to C1) mid (C1 to C6), high (C6 to C7) and very high (C7 to C8). At each dynamic level, one sample was chosen from each make in ranges low, mid and high and two were chosen from very high. The MUMS offered a similar range of samples, of which five were chosen in the same manner. This gave a total of 44 piano samples.

Trumpet

Trumpet samples played both with and without vibrato at three dynamic levels are included with the RWC Database. The range of these recordings differ somewhat, from E3 to F6, depending on the make and dynamic level. These were split into ranges low (E3 to E4), mid (E4 to C6) and high (C6 and upwards). For each dynamic level, with and without vibrato, one sample was chosen from the low and mid ranges whereas two were chosen from the high range. Again the MUMS offers samples from a similar range of pitches. From

these a further five samples were included. This gave a total of 53 trumpet samples.

Violin

The RWC Database includes violin samples from a range of G3 to E7. The tones with vibrato are sampled at the three dynamic levels p, m and f but the non-vibrato tone are only sampled at level m. As with the guitar, each string is sampled for one complete octave and so samples were chosen depending not only on the range within the instrument but the range within each string also. Thus for each dynamic level with vibrato and for the set without vibrato one note was chosen for the low string range and one for the high string range while two were chosen from the overall high range (E6 to E7). A further six violin samples were taken from similar ranges from the MUMS. This gave a total of 55 violin samples.

The above resulted in a total of 229 samples for the listening test. It was important to include as many samples as possible for accurate results while not allowing the test to run too long as fatigue may set in with the participants causing inaccuracies in judgement. Presuming that each participant spends a maximum of ten seconds dealing with each sample, the test would take under 40 minutes. The samples were played in a random but consistent order to each participant. A list of the samples in the order they were presented to the candidates is given in Appendix C of this thesis.

8.3.2 Subjects

This test was taken by a total of 31 participants. This consisted of 21 males and 10 females, ranging in age from 21 to 59. As musical training was not a requirement to participate, these subjects had varying degrees of musical experience. They were recruited through word of mouth and most were eager to participate. On completion of the test, each subject was given a short questionnaire asking them to detail any musical experience they might have. They were also asked for any comments on the experiment, some details of which are discussed in Section 8.5.5.

8.4 Experimental Set-Up

The experiment was conducted on a PowerBook G4 Apple Mac laptop over a set of Technics headphones in a quiet room. The user had control of the volume at all times during the experiment. The test was written as a GUI in Matlab (MATLAB7, 2006). A screenshot of the GUI used is shown in Figure 8.1. The user was instructed to play the current sample and then choose whether they thought it was played on

A. A flute

B. A guitar

C. A piano

D. A trumpet

E. A violin

F. None of the above

The user was able to play the sample as many times as they pleased and then choose their selection from a pop-down menu. Once happy with their selection they can push Next and their answer is stored in an excel file. The counter in the top left hand corner is to clarify that they have moved on to the next sample.

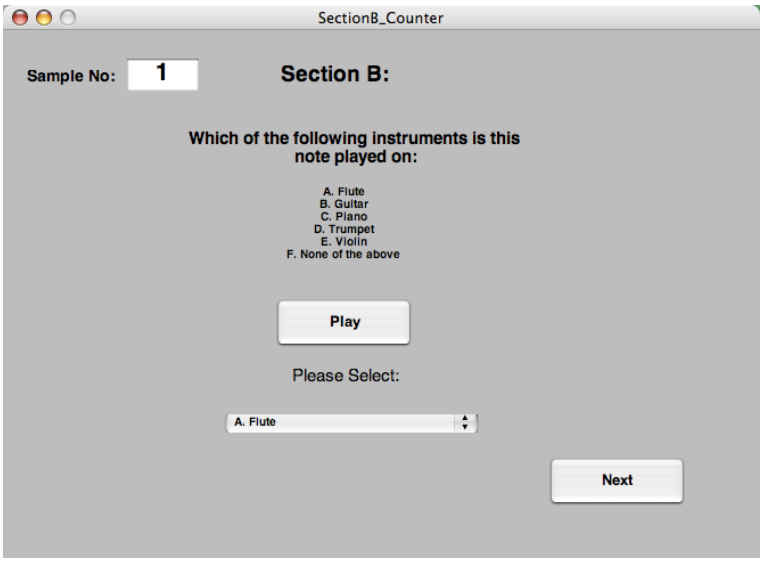


Figure 8.1: *Graphical User interface used for the Listening Test*

Before starting the experiment each participant was asked to read a Subject Information Sheet, detailing what they are being asked to do and the

relevance of the work. After completing the experiment, each subject was asked to complete the short questionnaire. Copies of these documents are included in Appendix D of this thesis. The experiment was conducted in two sections, A and B. Section A was included merely as a training session before the real experiment. It allowed the user to become familiar with the experimental set-up and to have a few practice runs before they undertook the full experiment. Two samples for each instrument were included in Section A. They were purposely picked to be very typical samples of each instrument with none in the extremities. This left 219 samples for Section B.

8.5 Results

As discussed above, Section A of the experiment was intended to allow each subject to become familiar with the environment. The samples included in Section A posed no difficulty to most subjects, although mistakes were made by subjects unfamiliar with such a set-up or particularly by over-zealous subjects who clicked many buttons to get started. The inclusion of this Section eliminated the possibility that such mistakes would be made early in the actual experiment. Thus the results in this section only refer to the results obtained from Section B.

A comparison of the correct identification (as %) for each instrument by the whole subject group is shown in the bar chart in Figure 8.2. From this chart it is clear that the piano was the most easily recognised instrument with 95.4% correct identification overall, followed by the guitar at 90.7 %, violin at 86.54 %, flute at 83.3% and finally the trumpet was the most difficult instrument to recognise with only 77.6% correct identification by all subjects. The confusion matrix in Table 8.1 shows the mis-identification between instruments. This matrix indicates the number of instances (not %) each instrument was identified as each of the other instruments. It can be seen from this table that the flute was most often not identified — or misidentified as none (77 instances), followed by confusions with the violin (60) and trumpet (54) but it was rarely mistaken for a piano or guitar. The guitar was most often mistaken for the piano (44 instances) followed by the violin (25) and not identified (25). As the piano was overall the most recognised instrument, its confusions were not high although it was most often not identified (24) or confused with a guitar (21). The trumpet was most often confused with the flute (191) or not identified (135). Finally, the violin was most often mistaken for a flute (127) followed by not being identified (70).

From the above results it is clear that the candidates did not automatically pick the none category if they were unsure of a sample. The flute and the guitar were the only instruments to be most not identified, although a 'none' categorisation did feature strongly among the other instruments. The confusion between the piano and the guitar is not surprising as they both comprise of a strong attack whereas the flute, violin and trumpet are more sustained sounds. In the confusion between trumpets and flutes it is shown that the trumpet was mistaken for the a flute much more often than a flute was confused as a trumpet as the flute samples were more often not identified or confused with a violin. The vast majority of mistakes made on the violin samples however incorrectly identified them as a flute. As hypothesised at the beginning of this experiment, we expect that many of these mis-identified sample sounds are from the extreme ranges within the instrument. The following section examines the individual samples that were misclassified the most to check the validity of this.

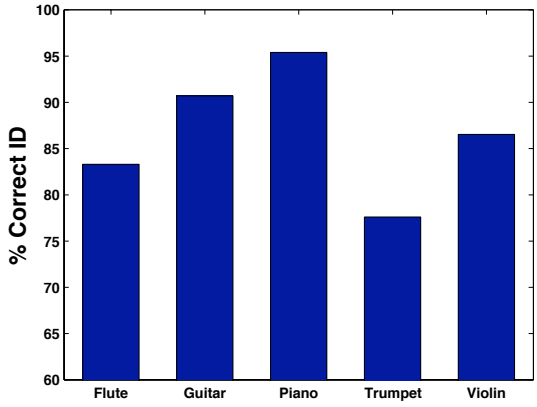


Figure 8.2: *Percentage of correct identification of each instrument for all subjects*

Table 8.1: *Confusion matrix for all subjects*

	Fl	Gu	Pi	Tr	Vln	None
Flute	**1007**	4	7	54	60	77
Guitar	2	**956**	44	2	25	25
Piano	3	21	**1242**	5	7	24
Trumpet	191	2	7	**1227**	19	135
Violin	127	6	2	16	**1442**	70

8.5.1 Samples

Pitch Range

It is clear from the results obtained that certain samples posed more difficulty for identification than others. Of the 219 samples played to the 31 subjects, 29 samples (7.6 %) were misclassified 10 or more times. A list of these most problematic samples is shown in Table 8.2. These misclassified samples are examined here in terms of instrument, pitch, vibrato and dynamic. Of these worst identified samples, 15 were trumpet samples, 8 were violin samples and 6 were flute samples. The least recognised sample was D6 played at dynamic level *f* without vibrato on the Vincent Bach trumpet. This is the highest pitch that is played on this particular instrument, and it was only correctly identified by 2 subjects. A number of other trumpet samples caused serious confusion as 7 trumpet sounds were misclassified by at least 21 of the 31 subjects. Each of these samples were in the highest range of pitch for this instrument. Furthermore of the 15 trumpet sounds that were included as the worst classified, 14 were in the highest range of pitch[1]. This difficulty in the identification of particularly high pitches is as expected. Similarly all violin samples included in this list were in the high pitch region. The 6 flute samples however were not all included in the high pitch region — 2 were low pitches, 3 were mid pitches and only one was a high pitch. This would imply that high pitches may cause the most confusion in the recognition of the trumpet and the violin but that this is not necessarily true for all instruments.

Vibrato

Of the five instruments included in this experiment only the violin, flute and trumpet can be played with or without vibrato. Of the 15 problematic trumpet samples, 9 are without vibrato and 6 are played with it. Of the 6 flute samples, 2 were without vibrato and 4 were with it and of the 8 violin samples 5, were without vibrato and 3 were played with it. Let us be reminded from Section 8.3 however, that although there are equal numbers of vibrato and non-vibrato samples for the flute and the trumpet, there are approximately 4 times as many vibrato violin samples (at dynamic level *f, m, p* and the MUMS) than non-vibrato samples. This implies that vibrato does help in the recognition of trumpet and violin samples but is less important for the recognition of flutes.

[1]The 15th misclassified trumpet sample G5 *p* on Schilke is in the mid region but is only 1 tone below the high region and so is also quite a high pitch

Dynamic

Most of the samples (apart from the non-vibrato violin and MUMS) include instances of each sample played at three dynamic levels. The 15 misclassified trumpet samples consist of 4 at levels *f* and *m*, 6 at level *p* and 1 with no dynamic. The violin samples consist of 1 at level *f*, none at level *m*, 2 at *p* and 5 with no dynamic. The flute samples consist of none at level *f*, 2 at level *m*, 3 at *p* and 1 with no dynamic. These results show that a softer dynamic does tend to lead to more difficult identification although this relationship is not as pronounced as might have been expected.

Table 8.2: *Samples most often misidentified*

Number	Instrument, note, dynamic	# mistakes	Pitch Range
1	Trumpet Bach NV D6 *f*	29	H
2	Trumpet Bach NV B5 *f*	26	H
3	Trumpet Schilke V A♯5 *p*	25	H
4	Trumpet Bach V A♯5 *p*	25	H
5	Trumpet Schilke NV D6 *m*	22	H
6	Trumpet Bach NV A5 *p*	22	H
7	Trumpet MUMS D♯6	21	H
8	Flute Sankyo NV D5 *p*	17	M
9	Violin Fium NV E7E	16	H
10	Trumpet Schilke NV C♯6 *p*	16	H
11	Trumpet Bach V A5 *m*	15	H
12	Flute Sankyo NV A6 *p*	14	H
13	Trumpet Bach NV C♯6 *m*	14	H
14	Violin Carcassi NV C♯7E	14	H
15	Trumpet Schilke NV E6 *f*	14	H
16	Trumpet Bach NV C6 *m*	13	H
17	Violin JFPres V E7E *p*	13	H
18	Trumpet Bach NV D6 *p*	13	H
19	Flute LouisLot V G4 *m*	12	M
20	Violin JFPres V D♯7E *f*	11	H
21	Trumpet Bach V A♯5 *f*	10	H
22	Trumpet Schilke V G5 *p*	10	M
23	Violin JFPres NV C♯7E *p*	10	H
24	Violin Carcassi NV D♯7E	10	H
25	Flute LouisLot V C♯4 *p*	10	L
26	Violin JFPres NV E7E	10	H
27	Violin JFPres NV B6E	10	H
28	Flute MUMS C♯5	10	M
29	Flute Sankyo V E4 *m*	10	L

8.5.2 Subject Groups

As discussed previously, candidates were not chosen on grounds of musical experience although a number of them did have musical experience to some degree. A number of subjects had received some level of formal musical training whereas a number of others had achieved postgraduate qualifications in music technology. During the analysis of the results, the subjects were therefore split into the following three categories:

A. Non-musicians — those that have had no (or minimal) formal or academic training in music

B. Musicians — those who have studied music to Leaving Cert standard or studied an instrument for a number of years

C. Music Technologists — those who have obtained a third level qualification in Music Technology

There are some subjects (three in total) that belong to both categories B and C above. In these cases the results were included in both categories for analysis.

A. Non-musicians

There were a total of 18 subjects in the Non-musician category. The identification of each instrument for this category is shown in the bar chart in Figure 8.3. This shows that again the piano was the most easily identified instrument with 96.8%correct identification, followed by the guitar at 92.6%, the violin at 86.0%, the flute at 83.2% and again the trumpet was the least identified instrument at 80.2%. Surprisingly this represents an increase in accuracy compared with the whole subject group.

The confusion matrix in Table 8.3 displays the number of times this non-musican group identified an instrument as one of the other instruments. The confusion between instruments is not unlike that in Table 8.1, apart from the number of instances whereby the subject picked none. For each instrument the proportion of none choices was significantly lower by comparison to the other mistaken choices than from the results obtained from all subjects. This is particularly noticeable in the case of the guitar which was never identified as a none. The implications of this result are discussed further in Section 8.5.3.

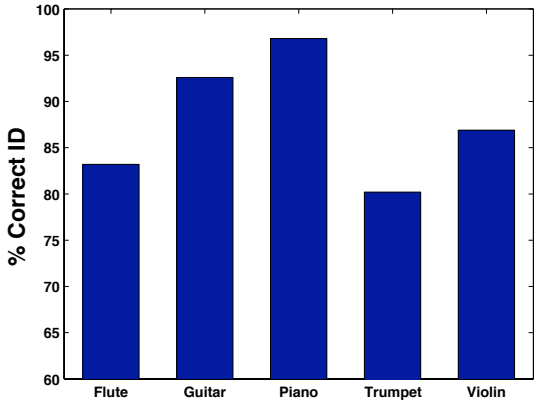


Figure 8.3: *Percentage of correct identification of each instrument for Non-musicians*

Table 8.3: *Confusion matrix for Non-musicians*

	Fl	Gu	Pi	Tr	Vln	None
Flute	**584**	2	6	39	37	34
Guitar	0	**567**	29	1	15	0
Piano	2	11	**732**	2	5	4
Trumpet	122	1	5	**736**	13	41
Violin	79	3	1	7	**829**	35

B. Musicians

There were 9 subjects categorised as Musicians in this experiment. The musical experience of these subjects ranged from those who had studied a musical instrument throughout their childhood to those who played numerous instruments and have third level qualifications in music. Comparisons of the identification of each instrument for this category is shown in the bar chart in Figure 8.4. This shows the piano as the most recognised at 95.5%, followed by the violin at 94.1%, the guitar at 90.9%, the flute at 83.5% and finally the trumpet at 80.8%. The most notable difference between these results and the previous categories is that the violin was better recognised than the guitar. It is possible that this result may be due to those with classical training being more familiar with a classical instrument such as the violin than those with no formal training. Apart from this one result, the identification results for the Musician group appear comparable with those of the Non-musician group.

The confusion matrix in Table 8.4 indicates that the instrument confusion is comparable with that of the whole subject group. The piano was again most often not identified, or confused with a violin or trumpet, the guitar mistaken as a piano or violin, the piano as a guitar or none, the trumpet as none or a

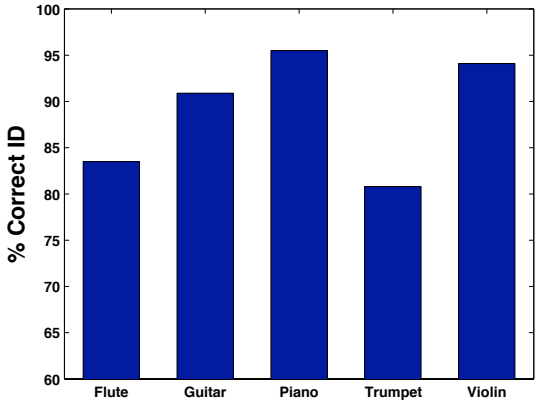


Figure 8.4: *Percentage of correct identification of each instrument for Musicians*

flute and the violin as a flute or none.

Table 8.4: *Confusion matrix for Musicians*

	Fl	Gu	Pi	Tr	Vln	None
Flute	**293**	1	1	11	21	24
Guitar	2	**278**	15	1	9	1
Piano	0	8	**361**	2	1	6
Trumpet	36	0	0	**371**	5	47
Violin	13	2	0	5	**449**	8

C. Music Technologists

There were 7 subjects with a qualification in Music Technology that took part in this experiment. Most of these subjects had obtained a Masters degree in Music Technology in recent years. The identification of each instrument for this category of subjects is shown in the bar chart in Figure 8.5. Again this shows that the piano was most easily identified at 90.8%, followed closely by the guitar at 89.1%, the flute at 86.1%, the violin at 80.6% and finally the trumpet at 68.3%. These results appear to be on average the lowest between all categories of subjects. The only instrument that was recognised more by this Music Technologist group than any other was the flute.

A confusion matrix detailing what each instrument was confused with is shown in 8.5. It is evident from this table that this group chose the none option more often than the other groups. The most common error for the flute, guitar, piano and trumpet was to be misidentified as none. The violin was more often confused with being a flute — although the none option was still very high for misidentified violin samples.

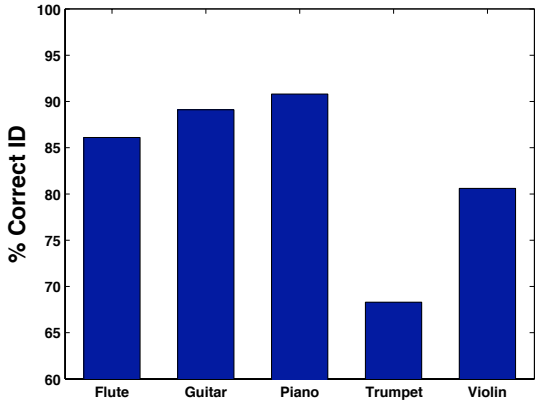


Figure 8.5: *Percentage of correct identification of each instrument for Music Technologists*

Table 8.5: *Confusion matrix for Music Technologists*

	Fl	Gu	Pi	Tr	Vln	None
Flute	**235**	1	0	6	4	27
Guitar	0	**212**	0	0	2	24
Piano	1	5	**267**	1	1	19
Trumpet	39	1	2	**244**	2	69
Violin	35	1	1	5	**299**	30

To compare the performances of the different groups Table 8.5.2 displays the average and standard deviation of correct identification for each group. It can be seen that the Musician group on average scored the highest although they were closely followed by the Non-musican group. The Music Technologists scored significantly lower than these two groups. The standard deviation is also significantly higher for the Music Technologists than for the other categories, indicating a large variation in the accuracy of the subjects. The Music Technologists chose the none option a significantly higher proportion of the time than the other groups. From speaking to these subjects it became clear that as they were used to dealing with synthesised sounds they expected that some of the sounds in the experiment were synthesised — and indeed a number of them thought that this was what was being looked for. As such they considered the none option a more likely possibility than the other candidate groups. To account for this anomaly between the groups we now examine the results without the none category.

Measure	Total	Non-musicians	Musicians	MusTech
Mean	182.94	191.56	194.67	179.57
Mean(%)	86.19%	87.47%	87.74%	84.47%
Std. Dev.	39.98	17.23	19.05	33.76
Std. Dev.(%)	10.05%	7.87%	9.4%	12.81%

Table 8.6: *Average and standard deviation of identification results for each participant group*

8.5.3 Removing the 'none'

In total, the Non-musicians chose the none category 114 times. From a total of 3,942 choices that gives a none choice rate of 2.89%. In contrast the Musicians chose none 86 times or 4.36% of all choices, whereas the Music Technologists chose none 169 times or just over 11% of all their choices. Thus the Music Technologists were almost 4 times more likely to pick a none category than the Non-musicians. To examine the effect of this on the overall result, the mean and standard deviation of correct identifications of all subject categories without including the none choices were evaluated. These results are shown in Table 8.5.4. This increases the average correct identification in each category. Most notably, the average correct response in the Music Technologists group has risen from 84.47% to 91.03% and the standard deviation has halved. There is now a much smaller difference in the overall correct identification percentage for each group.

8.5.4 Outliers

The majority of subjects that undertook the experiment made identification errors on less than 40 samples. There were, however, a few subjects who for some reason made mistakes on a significantly larger number. Subject 25 erred on 68 samples. It is known that this subject has a hearing impairment and so his results may be somewhat misleading [2]. Subject 26 mis-identified 84 samples. Being the oldest subject to take part it is possible that her hearing may have deteriorated, although she gave no indication of this. It has been shown (TestMyBrain, 2009) that the human ability to recognise faces drops

[2]Although we were purposely recruiting subjects with healthy hearing, this subject was still included as his hearing has not prevented him from studying music throughout his life and achieving honours at grade 8 level in both piano and violin

after the age of 30 — thus it is possible that this subjects hearing deteriorated over time in a similar manner. She was also the most uncomfortable with the experiment set up as she had no computer experience. Subjects 28 and 29 erred on 81 and 93 samples respectively. It is not known why these subjects scores were so poor: they are both young men with experience in music technology [3]. As these four subjects' errors are in the region of twice that of the rest of the subject group, they are here considered as outliers to the experiment. To gain some perspective of the results without these subjects the averages for each category are again calculated but with these subjects removed. This leaves 27 subjects overall, 17 of whom are Non-musicians, 8 are Musicians and 5 are Music Technologists. The averages of these categories of subjects both with and without the none classification are shown in Table 8.5.4.

Measure	Total	Non-musicians	Musicians	MusTech
Without 'none'				
Mean	90.69%	90%	91.59%	91.03%
Std. Dev.	5.13%	4.8%	5.32%	5.89%
Without 'outliers'				
Mean	87.76%	87.81%	88.38%	88.5%
Std. Dev.	8.36%	8.2%	8.53%	8.52%
Without 'none' or 'outliers'				
Mean	90.84%	90.74%	91.7%	92.02%
Std. Dev.	4.7%	4.46%	4.8%	4.87%

Table 8.7: *Identification results without 'none' identifications and outliers for each subject category*

As predicted, removal of the outliers improves the results somewhat. This is most notable again in the Music Technologists group as 2 out of the 7 members of this group were considered outliers. Naturally, removing these outliers decreased the standard deviation in all categories also. From looking at all the groups when the results are adjusted to exclude outliers and to

[3] One could hypothesise that their hearing may have been damaged due to over-exposure to loud sounds in the music technology field, but more tests would need to be carried out to make such a claim

exclude the none identifications, the Music Technology has the most successful average recognition followed by the Musicians and then the Non-musicians. These results are illustrated in Figure 8.6.

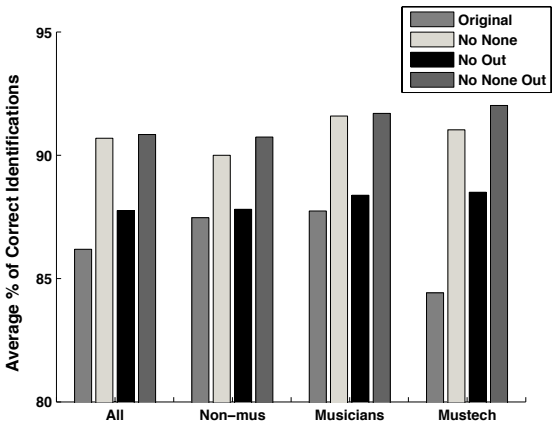


Figure 8.6: *Average correct identification of each subject category of the original data, the data without none results, the data without outliers and the data without outliers or none results*

8.5.5 Discussion

This listening experiment was conducted to evaluate how recognisable the samples used in previous tests in this thesis actually are to human listeners. Out of 219 samples, the number of samples misidentified by each subject varied from 6 to 93 with an average of 36. Overall the trumpet was found to be the most difficult instrument to identify and the piano the simplest. This agrees with many of the comments made by the subjects after taking the experiment. A number of the subjects said that they found the piano and guitar the easiest to recognise but they found the trumpet and the flute the most difficult. This would explain the large confusion rate between the trumpet and the flute. As mentioned earlier the piano and guitar (being struck and plucked instruments) both have a strong attack and quick release whereas the other instruments are more sustained. The results here indicate that such a prominent feature may influence the recognition of instruments but more focussed research would be needed to confirm this. It is true that the transients are dominant in such sounds and as discussed in Section 8.2 transients have been found to be very important features in instrument recognition. Hence it follows that those with sharp transients should be easier to recognise as we have found here. The results here do not however agree with previous test

by Elliott (1975) and Martin (1999) who found the trumpet to be one of the most easily recognised instruments.

It was found that particularly high pitched trumpet samples were very difficult to identify. Difficulty in recognition was somewhat more evident in quieter sounds than in louder ones although this was not pronounced. Quieter sounds tend to have a smoother amplitude envelope with a less pronounced attack due to a softer play by the performer. However, it was observed that the pitch of the note played a more important role in making certain sounds unrecognisable. This was not the case for the flute however, as only one high pitched flute sample was found in the most problematic list. The presence of vibrato did not greatly influence the identification of the flute or the trumpet, but a large proportion of non-vibrato sounds among the problematic samples would indicate that it plays some importance in the recognition of violin tones.

The subjects were divided into groups to determine the effect (if any) of musical training on their ability to identify sounds. It was interesting to note that 2 subjects achieved the most accurate score of misidentifying only 6 instrument sounds. Of these two subjects one was an accomplished musician (who is also a Music Technologist) whereas the other had no formal music training at all. This result is similar to that noted by Eagleson and Eaglson (1947) who found that the best result overall was by a non-musican. Of the three groups it was initially found that the average of the Musicians group was the highest, followed by the Non-musicians and finally by the Music Technologists. After removing outliers and the none category however it was found that the Music Technologists performed best followed by the Musicians and finally the Non-musicians. Although this may initially appear to encourage the removal of the none category altogether it is worth considering the implications of this result. These subjects have expertise in dealing with synthesised sounds and so their high rate of selection of the none category is meaningful and indicative of the quality of these sounds. Thus the none classifications by the Music Technologists are possibly more important than those by the Non-musicians who were most likely trying to discern between instruments rather than really judging the quality of the sounds. It was the quality of the sounds being scrutinised here — we were not looking for the highest correct scores of the subjects. This result shows the importance of knowing the abilities of your test subjects in analysing such results.

In conclusion, we are left with a real set of human results on our sample sounds, details of which we may now compare with the instrument classifiers

developed in the preceding chapters.

8.6 Comparison with Automatic Instrument Classifiers

This section discusses the accuracy of the GA-MLP classifier developed in Chapter 6 and the GP strings developed in Chapter 7 at classifying the set of sounds tested in these listening tests. Both of these classifiers have already been tested with a 65 sample one-octave and a more general 300 sample set with encouraging results. Rather than representing 'typical' samples, as discussed above this set of listening samples includes a higher proportion of problematic samples. Hence it is anticipated that these samples may be difficult for the classifiers to identify correctly.

8.6.1 GA-MLP Classifier

Chapter 6 described the development of an automatic instrument classifier using an MLP with features selected by a GA. The accuracy of this classifier was discussed in classifying a typical one-octave set and a larger set that was more representative of the whole data set. Using the genomes derived from the 40-dimensional fitness function, the average classification of the one-octave set was just under 70% whereas that of the larger set was above 99%. In this section, we examine the accuracy of this classifier at identifying the set of samples used in the above listening tests.

The GA was trained and tested 10 times for each of the 10 genomes. The training set consisted of the whole set of data and the testing set comprised of 219 listening tests samples. The results are discussed in terms of the classification of the system and the confusion between the tested instruments. The average classification results of the MLP trained using the 10 genomes is given in Table 8.8. Clearly this shows a dramatic reduction in classification accuracy compared to the results of this classifier on the other test sets. Notably, the piano results have decreased considerably, whereas the piano was one of the best classified instruments for both the one-octave test set and the larger test sets. The only instrument to be consistently recognised from this test set is the violin. Identification accuracy for the trumpet and the guitar are both particularly poor. Of the 10 genomes, Genome 3 resulted in the best overall accuracy at 43.97%. The variation in accuracy was not large between the 10 runs for each genome. From inspecting the results of the

individual runs it was noted that the total accuracy across all five instruments did not exceed 50% for any one run.

Table 8.8: *Results for the GA-MLP classifier tested on the Listening Set samples for each of the 10 genomes*

Genome	Total	Piano	Violin	Flute	Trumpet	Guitar
1	35.57%	17.38%	90.38%	44.36%	3.92%	10%
2	33.20%	5.95%	88.87%	51.54%	5.88%	0%
3	43.97%	46.9%	87.92%	60%	4.9%	12.06%
4	30.68%	3.81%	91.13%	35.38%	6.08%	1.18%
5	39.18%	24.29%	68.49%	52.82%	13.73%	34.41%
6	32.33%	5.95%	85.28%	50.51%	4.51%	3.24%
7	41.19%	47.62%	73.58%	65.64%	9.61%	2.06%
8	38.68%	31.9%	81.89%	58.72%	6.08%	5.59%
9	37.08%	30.71%	65.47%	68.72%	6.67%	10%
10	37.58%	18.81%	90.94%	32.31%	19.61%	10.59%
Average	**36.95%**	**23.33%**	**82.4%**	**52%**	**8.1%**	**8.91%**

This classifier performs very poorly in comparison to the human results discussed earlier in this chapter. As before with the human subjects, certain samples posed more of a problem than others to the classifier. Each sample was classified 100 times (10 runs by 10 genomes). Of the 219 samples, 9 were misidentified all 100 times. All 9 of these were trumpet samples, five of which were in the highest range. A total of 45 samples were misidentified 95 or more times. Of these samples, 29 were trumpet samples, 15 were guitar samples and one was a piano sample[4]. The high misclassification of trumpet sounds agrees with the human test results, but for the human subjects the guitar was one of the best classified instruments. The misidentifications between instruments is shown in the confusion matrix in Table 8.9.

This confusion matrix shows the number of times the classifier misidentified an instrument sample as on of the other instruments. Note that there is no *none* classification in this table as the MLP was trained to select one of five options, making it a forced choice decision. This table shows that while the violin and the flute were more often correctly identified than confused with another instrument, this is not true for the other instruments. The piano was more often confused with a violin rather than be correctly identified. The guitar was more often confused with the piano, violin or flute rather than correctly identified and the trumpet was more often confused with the violin or flute, rather than correctly identified. Across all instruments, the most common confusion was with the violin — indicating that the timbre of the

[4]B0 P — a particularly low, soft piano sample.

violin may have been found by the classifier to be somehow 'in-between' the timbres of the other instruments. These results, and those of the human tests may now be compared with those obtained from the GP classifier.

Table 8.9: *Confusion matrix for GA-MLP classifier*

	Fl	Gu	Pi	Tr	Vln
Flute	**2028**	21	64	94	1693
Guitar	831	**303**	1197	104	965
Piano	970	70	**980**	179	2001
Trumpet	1781	34	179	**413**	2693
Violin	786	11	98	38	**4367**

8.6.2 GP Classifier

The GP classifier developed in Chapter 7 combined the 95 features using only simple arithmetic and logical operators. The best results were found by running a GP using both mathematical and logical operators over 2000 generations. This run created 30 individual programs. Although the accuracy of classifications varied considerably between individuals, a high accuracy of 94% was achieved with the best strings using Set10 as the test sample set. A further set of 30 individual programs were evolved using only the top 14 features. These were found to have higher test accuracy, on average, than the programs using all 95 features. The highest test accuracy found was 94.3% by programs number 3. The accuracy of these evolved strings at classifying the set of listening test samples is examined in this section.

As with the GA, when the 30 GP programs were used to test the listening test samples, the results were again found to be very poor. The highest accuracy achieved by any one program using 95 features was 46.6%, which was obtained using string number 29. The three programs found to be the most accurate when tested using Set10 — 9, 2 and 30 were found to have poor classification accuracy (26.9%, 32.9% and 31.9% respectively) when tested with these listening test samples. The average classification accuracy of all 30 programs was 28.34%. The programs evolved using just the top 14 features achieved slightly higher accuracy when tested with these listening samples. Although the best program (number 9) when tested using Set10 again gave a poor classification of 29.7%, the highest test accuracy achieved by any program was 56.16% by program number 25. The average of the classification results of the 30 programs evolved using the top 14 features was 31.5% — slightly higher than that of the programs with all features. Hence, the re-

mainder of this section considers the classification results of the 30 programs evolved using just the top 14 features.

A total of 48 samples were incorrectly identified by all 30 programs. Of these, 31 were guitar samples, 16 were trumpet samples and 1 was a flute sample. Again, the high proportion of problematic trumpet sounds agrees with the GA and the human results. The notably higher misclassification of guitar sounds, however, again disagrees with the human results.

Table 8.10 displays the misidentifications between the musical instruments. Note that the *none* classification was possible again in these results. The identification is given by the numerical output of the current string — 1 for piano, 2 for violin etc. If however the output was not one of the five numbers accounted for (a 0 or a 6 for example) the classification was considered a 'none'. It is evident from this table that the piano and the violin were the only instruments most often correctly identified. Each of the other three instruments were more often confused with a different instrument than correctly identified. The flute was most often confused with a violin or a piano, the guitar with the piano or the violin and the trumpet was most often confused with the violin or the flute. Evidently, most confusions were with the violin, piano and flute with few instances of any instrument being confused with a trumpet or guitar. It was first considered that this might be due to some bias in the GP coding — as the piano, violin and flute are identified with the low numbers 1,2 and 3. However, this is unlikely to be the case as the trumpet and guitar were recognised quite accurately when tested with Set10. Furthermore, it may be seen from the results of the GA-MLP in Table 8.9 that a similar tendency to favour misidentifications as the piano, violin and flute were also noted, where this classifier had no such numerical bias.

Table 8.10: *Confusion matrix for the 30 GP programs evolved using the top 14 features*

	Fl	Gu	Pi	Tr	Vln	None
Flute	**281**	12	329	34	467	51
Guitar	39	**23**	716	3	200	79
Piano	61	5	**828**	8	307	47
Trumpet	474	16	330	**100**	578	32
Violin	238	9	417	10	**837**	39

These results show that the strings evolved by the GP and the GA-MLP were both disappointing as classifiers in comparison to humans. The generality of both the GA and GP methods were shown in the experiments in

Chapter 6 and 7 as they demonstrated recognition rates of unseen samples at over 99% and 94% respectively. As stated at the beginning of this section, it was anticipated that the accuracy of the classifier would drop as these samples were picked to contain a high proportion of problem samples to test the limits of the human ear. It is evident from these results, that the human ear is more successful as an instrument classifier than the GA and GP methods employed in earlier experiments. Regardless of musical experience or ability, the humans outperformed both the GA-MLP and GP classifiers at recognising this set of samples.

8.7 Conclusion

This chapter described a series of human listening tests on individual note samples undertaken for comparison with the automatic instrument classifiers developed in this thesis. Section 8.2 reviewed listening tests previously undertaken by other researchers. Section 8.3 detailed the samples used in the experiments which were discussed in Section 8.4. The results from the full participant set and groups of participants selected according to musical experience were discussed in Section 8.5. Finally, Section 8.6 described the results of the previously evolved GA and GP classifiers when tested with these listening test samples.

The human listening tests found that, in general, people are very accurate at recognising instrument sample sounds. The accuracy of the full set of participants ranged from 58% to 96% with an average classification accuracy by all participants of 84%. In comparison to this, the best genome used with the GA-MLP had a classification accuracy of just under 44% and the best accuracy obtained from the strings evolved using GP was just over 56%. This shows that although results from tests in Chapter 6 and Chapter 7 may show very high classification results for these classifiers, they do not perform at the standard of the human ear when tested with a particularly difficult test-set. The high accuracy of the human participants are a testament to the adaptability of the human aural recognition system. Even though these samples were at the extreme ranges of the instruments, the majority of participants had little difficulty in identifying most of them.

A notable difference between the human tests and that of both automatic classifiers is in the success of identifying the guitar. In the human tests the guitar was the second best identified instrument, and many candidates commented on it being the easiest to identify. In contrary to this, the guitar

was the least recognised instrument for the GA-MLP and GP classifiers. We may hypothesise that this might be due to the familiarity of the guitar to humans, whereas no such familiarity could exist for the classifiers. Arguably, the guitar is one of the most common instruments — which most people would have had opportunity to play at some point throughout their lives. Thus humans have had much opportunity to *learn* the sound of a guitar. Evidently there are aspects of the guitar sound that humans recognise that the features used do not convey to the GA and GP. As humans we learn sounds through exposure, thus the familiarity of the guitar could have given humans more chance to learn the distinguishing features of this instrument.

The results of this chapter show that although successful musical instrument classifiers have been developed for classifying 'typical' instrument sounds, they are not a replacement for the human ear. Our human hearing and perception must still depend on attributes not included in this thesis for identifying more tricky sounds. Although many timbral features were incorporated in these classifiers a perfect recipe for identifying any given musical sound has not been established.

Chapter 9

Conclusion

This thesis examined musical sound recognition and the features that are most important for accurate musical instrument identification. An introduction to what was meant by a musical sound was given in Chapter 2. Chapter 3 offered an introduction to timbre — what it is and the difficulties in measuring it. The timbral features used throughout the thesis were introduced in this chapter. The majority of these features have been used already in previous experiments in sound and instrument recognition, although some of them such as the *Centroid Envelope* had not been measured in this particular way before. A review of previous work in musical instrument recognition was given in Chapter 4. It is evident from these studies that previous methods of musical instrument classification varied widely in terms of samples used, features examined and the type of classifier used to perform the classification. This chapter also introduced the field of evolutionary computation and discussed ways in which it had been applied to the field of sound and music production.

Chapter 5 described a number of experiments that used PCA and MLPs for musical instrument recognition. By including many samples for classifying just three instruments, a robust classifier was developed that was highly accurate at classifying previously unseen samples. This chapter showed however that not all features were necessarily beneficial to a classifier — some features appeared more important than others. As testing each combination of features is unfeasible, an optimisation technique was required. This optimisation was performed using Evolutionary techniques as although such techniques have been shown to be powerful optimisation tools, they have not yet been extensively applied to the area of musical sound recognition. Thus, the following two chapters selected the best timbral features for classification using EC methods.

Chapter 6 developed a GA with a clustering fitness function to determine the best set of features to use with a MLP. The GA created a genome with 95 gene values, each of which corresponded to a weighting of one of the 95 timbral features. It was possible to reduce the number of features used by excluding those that had a corresponding gene value below a certain threshold. Thus the GA was used both to examine the most relevant features and to improve the performance and reduce the complexity of the classifier. As this evolutionary method was not able to classify the sample itself, it was compared to another evolutionary method — GP. The experiments in Chapter 7 used GP to evolve a program that can classify an unseen instrument sample. By limiting the available functions to just simple arithmetic and boolean functions GP was able to evolve a simple classifier, the structure of which is available to the user. The best evolved classifier was capable of achieving up to 94% accuracy in classifying instrument samples. While not as accurate as the GA-MLP method, these results are still encouraging for future work using GP. Chapter 8 described a series of listening tests undertaken by a group of human participants. These tests had a high percentage of problematic samples and showed that samples, in particular trumpet samples, in the higher pitch regions prove more difficult to identify than those in the lower or mid regions. When compared with the classifiers developed by the GA and GP the human subjects were shown to outperform both methods.

The remainder of this chapter discusses the conclusions and implications that may be drawn from the main results found in this thesis.

9.1 Classification Results

The MLP used in the experiments in Chapter 5 is a well established classification method. Other studies described in Chapter 4, have used this method in the past to successfully identify musical instruments. Hence, the MLP results may be used as a benchmark for comparison with the results obtained from both the GA and the GP. The motivation for including these evolutionary techniques in the instrument classifier was threefold:

- To attempt to create a more accurate musical instrument identifier.
- To reduce the number of features used with such classifiers, which may reduce computational costs while maintaining a high rate of identification.

- To determine the most important features, which in turn may lead to some further insights into timbre measurement and the identification of musical instruments.

In the experiments carried out throughout this thesis, each of the classification methods were tested using three sets of samples: a *one-octave* set containing an octave of pitches common to each instrument played at dynamic *f*, a *general* set containing 300 samples with equal proportions of samples played at all pitches and dynamic and a *listening* set containing a high proportion of difficult or problematic samples (as described in Chapter 8). A summary of the best classification results is given in Table 9.1.

Table 9.1: *Comparison of the GA and GP developed classifiers against the benchmark MLP classifier tested with the one-octave samples, the general samples and the listening samples.*

Set	MLP	GA-MLP	GA Reduced	GP	GP Reduced
1 Octave	70.31%	69.55%	50.02%	53.85%	70.77%
General	99.63%	99.63%	99.3%	94%	94.3%
Listening	45.21%	43.97%	44.34%	46.6%	56.2%

It is evident from these results that regardless of the method used, the highest accuracy was reported by testing with the general set. It is not surprising that the general test set is more easily classified than the listening samples — these were purposely chosen to be difficult. The fact that the general set is recognised more easily than the one-octave set may be slightly more surprising as these samples may be considered 'easy' for a human listener. This result demonstrates that what may be easy for a human is not necessarily easy for an artificial classifier. The common *f* dynamic in the one-octave set may have had an affect on the accuracy rates. Transients — particularly the attack — may be more pronounced in loud sounds than in quiet sounds. This may cause confusions with instruments which have particularly strong attacks, such as the piano. Also the pitch range, C4-C5, while common to each instrument, may not be the most comfortable range for each instrument to be played in. The smaller size of this test set may also affect the accuracy of classification, as each individual mistake made on this set causes a decrease of 1.5% in accuracy as opposed to each mistake on the general set which causes a drop in accuracy of just 0.33%. These sample set results demonstrate that to get the most general accuracy of the classifier, it is important to use a test set of samples that represent the entire set, rather than 'pigeon-holing' the test set into a specific range of pitch, dynamic or

other quality. Restraining the test samples to those of a specific pitch and dynamic caused a bias in the test set that skewed the results.

Overall, when all features were used, the MLP performs the strongest with the majority of test sets. The GA-MLP performs almost as well as the original MLP in each set (or exactly the same when tested with the general set). The GP classifier performs slightly worse on the one-octave and general test set but it slightly outperforms both other methods on the more difficult listening test set. This implies that the classifier evolved with GP may be more robust than the other methods at recognising particularly difficult samples. In general, however, there was no marked improvement on the results of the benchmark MLP classifier. While the GA-MLP and GP classifiers developed in this thesis did not outperform the MLP, they also did not perform badly in comparison to it. In addition to this, both methods indicate which are the most important features — thus allowing the reduction of the data incorporated to reduce computation costs.

The reduced GA-MLP quoted in Table 9.1 limits the data to those features whose corresponding genomes are 0.7 or higher. This corresponded to 25.2 features on average being used for classification with the MLP. From the results it is evident that this reduction in features caused a very small reduction in classification accuracy of the general samples, and resulted in a slight increase in the best accuracy on the listening set. However, the classification accuracy on the one-octave has been significantly reduced. It is worth noting that this result is averaged over the 10 genomes and that the results between the genomes varied quite widely from 27% for Genome 2 to 68% for Genome 6. Thus the effect of removing features was more detrimental for some genomes than others. The success of the classification using the general set, however, implies that this GA successfully reduced the number of features from 95 to an average of 25.2 with a minimal reduction in classification accuracy.

The reduced GP classifier used just 14 of the 95 features in classifications. It is evident from the results in Table 9.1 that this reduction in features was beneficial to the system for each of the test sets. Most notably, the classification accuracy of the more difficult listening samples was increased significantly. Thus the program evolved by GP with reduced features is the most generalised of all the classifiers developed in this thesis. It must be noted, however, that the program that gave the best results for the one-octave and listening test sets was from individual number 25, whereas that which gave the best classification results for the general set was from individual number 9. The program from number 25 gave a classification accuracy of

76.7% when training and 75% when tested with the general set. Thus, this may not be the most accurate program on the general set, but it does give the most consistent training and testing results regardless of the samples used.

9.1.1 Human Classification Results

The results of the experiments in Chapter 8 demonstrated that humans are very accurate at identifying familiar musical instruments. Even though the selection of samples presented to the candidates had a high proportion of problematic samples, the average recognition rate was high at 84%. This is considerably higher than the best classification accuracy of the MLP (45.21%), GA-MLP (44.34%) and the GP (56.2%) classifiers when tested with the same set of samples. This shows that although high recognition accuracy was reported for each of the developed classifiers, when tested with problematic samples they cannot match the human ear. As discussed in Chapter 4 many previous studies have developed musical instrument classifiers quoting results that may lead one to believe that the problem is effectively solved. However, the results presented in Chapter 8 show that when tested with difficult realistic samples, such developed classifiers may remain wanting. Of all of the classifiers, the one evolved with GP using limited features performed the best at recognising the listening test samples, but the results were still not as accurate as those from the human tests. It is demonstrated in this thesis that although a classifier may be tested and shown to be accurate, it may still not be an accurate replacement for the human ear.

9.2 Features

Along with classification results, one of the objectives of this thesis was to determine which features were most important for musical instrument selection. As discussed in Chapter 3, musical timbre remains an ill-defined property that is difficult to measure. As aurally identifying a particular instrument is largely dependent on its timbre, it was hoped that determining the best features to use for classification may offer some insight into the nature of timbre. Throughout the thesis, a number of features have emerged to be more important than others. This section discusses the details of these features.

9.2.1 Centroid Envelope

Experiment in Chapters 5, 6 and 7 all found the Centroid Envelope to be a very important feature for accurate instrument classification. In the evolutionary experiments in both Chapter 6 and particularly in Chapter 7, the first principal component of the Centroid Envelope emerged as the most prominent feature from all 95 features examined. The consistency of this result among the variety of experiments implies that this particular feature is very important in timbre analysis. Although the first principal component was the strongest, the other components — particularly the second, also emerged as being relatively strong features in a number of experiments. Interestingly, the static centroid did not emerge as a strong feature in any of the experiments. Many studies in the literature have found the centroid to be important in timbre analysis (Grey, 1977; Jensen, 1999; McAdams et al., 1995). It was shown in this thesis that it is not just the value of the centroid, but the changes of the centroid over the duration of the note that is the most revealing quality of a musical sound.

9.2.2 MFCCs

Along with the Centroid Envelope, throughout this thesis the MFCCs have emerged as a very prominent feature for instrument classification. However, there were some differences in the choice of MFCC values from the different methods of feature selection. Experiments in Chapter 6 using the GA found higher valued MFCCs to be the most important. In particular the fourth principal component of higher value MFCCs were found to be of particular benefit. This agrees with earlier results in Chapter 5 which found that a higher number of MFCCs (12 or above) using four principal components gave the best classification across three instruments. The meaning of these results differ though, in that the results from Chapter 5 could only indicate that higher MFCCs are best used *along with* the lower valued MFCCS, whereas the GA experiments selected the higher valued MFCCs as being the most important overall. Thus the results of the GA may indicate that it is solely the higher valued MFCCs that are important for accurate musical instrument identification.

The MFCC results from the GP experiments in Chapter 7 do not agree with the above results however. Over every experiment it was the first or second component of the lower valued MFCCs that were consistently chosen for the evolved programs. In particular the first component of MFCC4 was

one of the top three features selected in almost every experiment. The reasons for the priority of the different MFCC values in the two methods is not clear. A close inspection of the values indicated that certain lower valued MFCCs showed more distinction between instruments, thus explaining their prominence as a useful feature. Higher valued MFCCs showed less prominent distinctions between the instruments. The higher classification accuracy of the GA-MLP classifiers may indicate that the GA chose the 'right' selection of MFCC values. However, further GP runs may also result in more accurate programs. If these programs were to include more high MFCC values they would agree with the GA-MLP results, proving that higher MFCCs values are more important. Alternatively, if these more accurate GP programs continued to stress the lower value MFCCs, they would consolidate the GP runs in this thesis that found these to be the most important values.

9.2.3 Other Features

Although the Centroid Envelope and the MFCCs produced the most consistent results as described above, a few other similarities emerged among some other features to a lesser extent. After the Centroid Envelope, the Temporal Envelope was found to be particularly dominant in the GP experiments. On the other hand while it was chosen somewhat often by the GA, Temporal Envelope values appeared among the weakest features chosen by the GA in a small number of experiments. The Temporal Envelope was found in early MLP experiments in Chapter 5 to be a very useful feature — obtaining over 84% classification when used alone to classify between three instruments. It is also regarded as a very important feature in previous instrument classification studies. Thus it appears that the GP made the correct decision in including this feature in the majority of its evolved programs.

From Table 7.1 it appears that the Inharmonicity and the Number of Peaks were two of the least often chosen features using GP. This again agrees with MLP results in Chapter 5 which found the addition of these features to be detrimental to the system. Likewise the values from Residual Envelope were not often included by either the GA or GP reinforcing the decision made in early MLP experiments in Chapter 5 to not include values from the Residual in further experiments.

In general it appears that for a number of features, the selection by GP is consistent with experiments conducted earlier in the thesis. Although the selections by the GA and GP vary, GP was in general more consistent with its selections over many experiments than the GA. The selection criteria by

each of these algorithms was different. GP could only use a selection of the features — constraint on the size of the programs evolved meant that it could not select them all. On the other hand, the GA could select all 95 features, it was only the relative amount of each feature used with the classifier that it was controlling. A feature that was 'strong' according to the GA meant that there was more of this feature incorporated in relation to the other features. On the other hand a feature was only 'strong' according to GP if it was included in the classifier at all — weak features were completely dismissed. This may imply that the selections by GP are a better means of judging which features are important, as weaker features were completely ignored rather than merely reduced in magnitude.

9.3 Limitations of Method

Although this thesis does put forward some interesting results, there are areas within the work which may have been susceptible to problems or limitations within the methods employed. These possible problems are discussed in this section.

The most important drawback of evolutionary methods is in the time it takes to evolve a solution. As discussed in earlier chapters, each of the final 10 GA genomes took approximately one day to evolve, and the longest GP run (30 runs with population of 100 over 2000 generations) took over 10 days to compute. Along with the experiments described in Chapter 6 and Chapter 7, many other evolutionary runs were performed that did not yield results deemed good enough to persue. Even smaller runs at earlier points in the experimentation were time-consuming. From an experimental point of view such long times can be tedious, particularly when waiting for results to decide on the best methods to continue with. These long run times may limit the potential of the evolutionary runs.

The lack of consistency in the evolved results was discussed in both Chapter 6 and Chapter 7. This lack of consistency is apparent in both the differences in the best evolved genomes or program trees for each run and in the variety of individuals among the last population of each run. The diversity among the final population implies that the algorithm is still searching for the best solution. Thus longer runs may be a step towards a solution in this case. However, the differences between the best evolved solutions imply that the algorithm is finding a different best solution every time — it may be finding a local minimum rather than the global minimum. The implications of this

are different for the GA and the GP. The 10 genomes evolved by the GA are found using 10 different data sets. These sets were chosen to be equally representative of the whole sample set, but ultimately do contain different samples. It is therefore possible that the GA is creating genomes that are specific to the samples used, and so all 10 genomes will be different from one another. Such a problem is known as *over-fitting*. However, this in turn may raise another problem: what if the best set of features to use for each group of samples are different because no one best set exists for all samples? If one best set of features does exist, then the GA in Chapter 6 did not find it. On the other hand if no best set exists, then the GA could not be able to find it regardless of what set of samples was used to evolve the genomes. It is possible that the best set of features to use is dependent on which instruments are being recognised.

The GP programs on the other hand were evolved using the entire sample set. It was noted in Chapter 7 that although the best evolved genomes were still diverse, there was more continuous selection of specific features using GP than using GAs. This implies that using the full set of samples with the GA might have produced more consistent results. However, the GA took longer to evolve than the GP which is why the data was split into 10 sets for the GA experiments. Conducting the GA experiments using the whole set of data over more generations would require too much time using Matlab on the platform used in this thesis. Thus the best solution for this problem would be to implement the same tests using a different language on a more powerful platform.

The variations in the results may also be due to the randomness involved in the initialisation of the evolutionary runs. This random initialisation means that it is impossible to exactly reproduce the results from any one run. An exact result cannot be taken from just one run — it must be compared to a number of independent runs to get a true indication of what the results mean. Thus as in the analysis in the evolutionary chapters, trends emerging from numerous evolutionary runs must be examined to determine what the most prevalent features and results are. When the GA or GP is performing properly, this randomness at initialisation should not matter, as the algorithm will converge towards the correct result. The fact that there is still so much variation implies that either over-fitting has occurred, or the algorithm has not been given enough time to converge.

The previous experiments on musical sound classification, described in Chapter 4, contained a wide variety of different timbral features. The list

of features used throughout this thesis and described in Chapter 3 cover the majority of features used before, but it may be noted that every possible timbral feature was not incorporated in this thesis. It is therefore possible that one or more alternative features may have emerged as important, had they been considered. The problem with this theory is that there is no upper limit to the amount of features that may be included in an experiment. New features may always be created and added, therefore making it impossible to confidently state that an adequate number of features are included. Nevertheless, the evolutionary methods employed as feature selectors in this thesis should be able to determine which features, of those included, are the most important for timbral analysis of musical notes.

Another practical limitation of the work carried out in this thesis is that the experiments are only conducted using five instruments. It was stated that this limitation was purposely implemented so that numerous instances of each instrument sound could be incorporated into the experiments. Nevertheless, this does limit the developed classifier to choosing between a piano, violin, flute, trumpet and guitar only.

9.4 Future Work

Although there are limitations to the work, there are many prospects for the expansion of the type of experimentation described in this thesis. Even though the variations among the evolved results imply that a global set of optimal solutions have not been found, the classification results given by the EC methods are still very encouraging. For both the GA and GP methods, testing on unseen samples produced accurate results, indicating that the classifiers developed were general enough to handle unseen sounds. The GA-MLP results were particularly accurate, and remained so even with the removal of a number of features. Thus further evolved genomes might be more successful at removing superfluous features for sound classification. Future work using the GA would require a faster computer set-up that would be able to run for many days or weeks. Only then could it be determined if a global best selection of features could be found.

From the discussion on previous work in Chapter 4, it was seen that GP had not been used previously for musical instrument identification. The results from Chapter 7 indicate that it is well suited for this purpose. Although the classification results were not as accurate as with the GA, GP did create programs limited to a small set of functions that displayed relatively consis-

tent and high recognition rates. The program trees evolved using GP were created using a quite limiting set of primitive functions. It is likely that the classification results of the evolved programs may be improved by further searching through the possible functions. In particular, the inclusion of more boolean logical operators may be of benefit to the system as they may be able to exploit relationships between the features. Boolean operators may be employed to evolve a different type of classifier that is purely built on the relationships between the features, rather than a mathematical combination of the feature values for a given sample. Such programs may contain boolean operators only, and output a classification 'label' that is assigned to a specific instrument, rather than a numerical calculation. Although the results from Chapter 7 show that the methods employed here do work, there is no 'mathematical' relationship between the instruments: a '1' does not literally *mean* a piano anymore than a '4' can *mean* a trumpet — they are merely defined as such for these experiments. Likewise, another program could be developed that analyses the relationship between features and outputs a specified output according to certain conditions. The evolution of such a program would require some subroutines or Automatically Defined Functions (ADF) (Koza, 1994). Such functions may give the programmer some high level control over the expected relationships among the features, to ensure that only meaningful programs are evolved.

Other evolutionary methods may also be useful for evolving musical sound classifiers. In particular Grammatical Evolution (GE) (O'Neill and Ryan, 2001) may be used instead of GP to evolve a classifier. GE performs in a similar manner to GP in that it can create programs, but it evolves simpler strings and then maps them to programs using a specified *grammar* in a *genotype to phenotype* mapping. This may be beneficial to the system — in particular for building programs using more logical operators as described above. By defining the grammar, it may be possible to include and encourage certain sections of code such as those that may relate one feature to another and give a meaningful output.

Further work in this area may also involve creating a classifier that can classify between more instruments. The instruments used throughout this thesis were purposely chosen to be from different instrument families. Creating a more specific classifier that can classify between instruments of the same family or instruments from the entire orchestra would require a much larger set of samples and would therefore be much more computationally expensive. If the speed of calculation could be increased, however, evolving a classifier

that could recognise many more instruments would be possible.

A big advantage of the GP classifier is that, although it took a very long time to evolve, once it is evolved the solution program is very simple and quick to implement. Thus the actual classifier created may take a very short time to classify or test a new sample. This may be of interest for the evaluation of synthesised sounds. In particular, if evolutionary methods were used to evolve a synthesiser, as suggested in Loughran et al. (2007), the synthesised sounds must be evaluated in some way. Human evaluations are time consuming and costly, and other classifiers such as neural networks are computationally expensive if they must be used many times — such as in a fitness function in a evolutionary algorithm. One general point noted from the evolutionary experiments carried out throughout this thesis is that a computationally expensive fitness function (such as the function that included the PCA in Chapter 6) can slow down the evolution significantly. Thus if one wanted to evolve a synthesiser, it would be best to have a fitness measure that is quick to implement. It was shown that a GP may evolve a simple classifier with high recognition capability. Such an evolved program may be used as a classifier in a fitness function for an evolutionary synthesiser, as it may be quick to implement and gives an accurate decision on whether or not the given synthesised sound may be *like* a real instrument sound.

Although the machine learning classifiers have not proven to be as accurate as the human ear, it was shown that the use of evolutionary techniques for the selection of features —- particularly with GP — may increase the accuracy of an artificial classifier. Hence, the results from this thesis would warrant further studies in the use of evolutionary techniques in the field of musical instrument identification.

Appendix A

Publications

The following posters and papers have been published as part of this thesis.

Loughran, R., Walker, J., O'Neill, M., and O'Farrell, M. (2007) Instrument identification with artificial neural networks. Poster presentation in *Sound and Music Computing Summer School*, Stokholm, Sweden.

Loughran, R., Walker, J., O'Neill, M., and O'Farrell, M. (2008a). Musical instrument identification using principal component analysis and multi-layered perceptrons. In *International Conference on Auditory, Language and Image Proceedings*, pages 643—648, Shanghai, China.

Loughran, R., Walker, J., O'Neill, M., and O'Farrell, M. (2008b). Comparison of features for musical instrument identification using artificial neural networks. In *Computers in Music Modelling and Retrieval*, pages 19—33, Copenhagen, Denmark.

Loughran, R., Walker, J., O'Neill, M., and O'Farrell, M. (2008c). The use of mel-frequency cepstral coefficients in musical instrument identification. In *International Computer Music Conference*, Belfast, Northern Ireland.

Loughran, R., Walker, J., and O'Neill, M.(2009). An exploration of genetic algorithms for efficient musical instrument identification. In *Irish Signals and Systems Conference*, Dublin, Ireland.

Appendix B

List of Features

A total of 95 feature values were used in the experiments in Chapter 6 and Chapter 7. The following is a list of the 95 features included in the order they were presented in.

1. Zero-Crossing Rate
2. Rolloff
3. Brightness
4. Regularity
5. First Attack Time
6. First Attack Slope
7. No. Onsets
8. Onset Distance
9. Centroid (static)
10. Spread
11. Skew
12. Kurtosis
13. Inharmonicity
14. Number of Spectral Peaks
15. Spectral Irregularity
16. 1^{st} Principal Component of the Temporal Envelope
17. 2^{nd} Principal Component of the Temporal Envelope
18. 3^{rd} Principal Component of the Temporal Envelope
19. 4^{th} Principal Component of the Temporal Envelope
20. 1^{st} Principal Component of the Centroid Envelope
21. 2^{nd} Principal Component of the Centroid Envelope
22. 3^{rd} Principal Component of the Centroid Envelope
23. 4^{th} Principal Component of the Centroid Envelope
24. 1^{st} Principal Component of the Residual Envelope
25. 2^{nd} Principal Component of the Residual Envelope
26. 3^{rd} Principal Component of the Residual Envelope
27. 4^{th} Principal Component of the Residual Envelope
28. 1^{st} Principal Component of the Spectral Envelope
29. 2^{nd} Principal Component of the Spectral Envelope
30. 3^{rd} Principal Component of the Spectral Envelope
31. 4^{th} Principal Component of the Spectral Envelope
32. 1^{st} Principal Component of MFCC1
33. 2^{nd} Principal Component of MFCC1
34. 3^{rd} Principal Component of MFCC1

35. 4^{th} Principal Component of MFCC1
36. 1^{st} Principal Component of MFCC2
37. 2^{nd} Principal Component of MFCC2
38. 3^{rd} Principal Component of MFCC2
39. 4^{th} Principal Component of MFCC2
40. 1^{st} Principal Component of MFCC3
41. 2^{nd} Principal Component of MFCC3
42. 3^{rd} Principal Component of MFCC3
43. 4^{th} Principal Component of MFCC3
44. 1^{st} Principal Component of MFCC4
45. 2^{nd} Principal Component of MFCC4
46. 3^{rd} Principal Component of MFCC4
47. 4^{th} Principal Component of MFCC4
48. 1^{st} Principal Component of MFCC5
49. 2^{nd} Principal Component of MFCC5
50. 3^{rd} Principal Component of MFCC5
51. 4^{th} Principal Component of MFCC5
52. 1^{st} Principal Component of MFCC6
53. 2^{nd} Principal Component of MFCC6
54. 3^{rd} Principal Component of MFCC6
55. 4^{th} Principal Component of MFCC6
56. 1^{st} Principal Component of MFCC7
57. 2^{nd} Principal Component of MFCC7
58. 3^{rd} Principal Component of MFCC7
59. 4^{th} Principal Component of MFCC7
60. 1^{st} Principal Component of MFCC8
61. 2^{nd} Principal Component of MFCC8
62. 3^{rd} Principal Component of MFCC8
63. 4^{th} Principal Component of MFCC8
64. 1^{st} Principal Component of MFCC9
65. 2^{nd} Principal Component of MFCC9
66. 3^{rd} Principal Component of MFCC9
67. 4^{th} Principal Component of MFCC9
68. 1^{st} Principal Component of MFCC10
69. 2^{nd} Principal Component of MFCC10
70. 3^{rd} Principal Component of MFCC10
71. 4^{th} Principal Component of MFCC10
72. 1^{st} Principal Component of MFCC11
73. 2^{nd} Principal Component of MFCC11
74. 3^{rd} Principal Component of MFCC11
75. 4^{th} Principal Component of MFCC11
76. 1^{st} Principal Component of MFCC12
77. 2^{nd} Principal Component of MFCC12
78. 3^{rd} Principal Component of MFCC12
79. 4^{th} Principal Component of MFCC12
80. 1^{st} Principal Component of MFCC13
81. 2^{nd} Principal Component of MFCC13
82. 3^{rd} Principal Component of MFCC13
83. 4^{th} Principal Component of MFCC13
84. 1^{st} Principal Component of MFCC14
85. 2^{nd} Principal Component of MFCC14
86. 3^{rd} Principal Component of MFCC14
87. 4^{th} Principal Component of MFCC14
88. 1^{st} Principal Component of MFCC15
89. 2^{nd} Principal Component of MFCC15
90. 3^{rd} Principal Component of MFCC15
91. 4^{th} Principal Component of MFCC15

92. 1^{st} Principal Component of MFCC16
93. 2^{nd} Principal Component of MFCC16
94. 3^{rd} Principal Component of MFCC16
95. 4^{th} Principal Component of MFCC16

Appendix C

Listening Test Samples

The following lists detail the instrument samples presented to the subjects in Chapter 8. The samples are named in order of the properties:

[Instrument] [Model] [Vibrato] [Pitch] [String] [Dynamic]

Whether or not vibrato is present is only indicated for the the flute, trumpet and violin samples. Likewise the string is only specified for the violin and guitar samples. A dynamic is not specified for any of the MUMS samples or for the non-vibrato samples.

Section A

Violin Fiumebianca Vib F#G F
Flute Sankyo Vib C#4 F
Flute Sankyo Non-Vib F#6 M
Piano Yamaha F#5 M
Guitar Kohno Masaru F3 D F
Trumpet Bach Vib F4 F
Violin Carcassi Vib A3 G M
Guitar Kohno Masaru C#5 E M
Piano Steinway G#6 F
Trumpet Bach Vib B4 M

Section B

Piano Yamaha C7 P
Flute Louis Lot Non-Vib B6 M
Violin Fiumebianca Non-Vib E7 E
Trumpet Bach Vib E3 F
Flute Sankyo Vib C7 F
Violin JF Pressenda Vib F#4 D P
Trumpet Schilke Vib A#5 P
Flute Sankyo Non-Vib B6 M
Violin MUMS A3
Violin JF Pressenda Non-Vib G#3 G
Trumpet Bach Non-Vib B5 F
Piano MUMS A#0
Violin Fiumebianca Non-Vib A5 A
Violin Fiumebianca Vib A#6 E P
Guitar Stafford E5 E P
Flute Louis Lot Vib A#6 M
Trumpet Schilke Vib F#3 M
Piano Yamaha C7 M
Guitar Imai Yuichi D5 E P
Violin JF Pressenda Vib A6 E M
Guitar Kohno Masaru E5 E M
Violin Carcassi Vib B6 E F
Trumpet Bach Vib A #5 F
Guitar Stafford A#4 E P
Piano Yamaha E3 P
Guitar Stafford E5 E F
Trumpet Bach Vib G#3 M
Trumpet Schilke Vib A4 P
Flute Sankyo Non-Vib A#5 M
Violin Carcassi Vib G#3 G P
Violin Fiumebianca Vib D7 E P
Violin Carcassi Vib D#7 E M
Flute Louis Lot Vib G6 M
Trumpet Schilke Vib G5 P
Violin JF Pressenda Vib C#7 E P
Guitar Imai Yuichi C#5 E P
Piano Steinway G5 F
Flute Louis Lot Non-Vib B6 F
Piano Yamaha C1 F
Piano MUMS A7
Trumpet Bach Vib C5 F
Violin Fiumebianca Vib C5 A M
Trumpet Schilke Non-Vib B3 P
Violin Fiumebianca Vib A#6 E F
Guitar Kohno Masaru B4 B M
Piano Yamaha B6 F
Piano Yamaha F7 F
Piano Steinway D#1 F
Trumpet Schilke Non-Vib G#5 P
Trumpet Bach Non-Vib C4M
Violin JF Pressenda Vib E6 E M
Trumpet Bach Non-Vib D6 F
Flute Sankyo Non-Vib D5 P
Guitar Kohno Masaru F#2 E M
Flute Sankyo Vib G6 F
Trumpet Bach Non-Vib G4 M
Flute Sankyo Non-Vib G#6 P
Violin Carcassi Vib C7 E F
Piano Steinway D#7P
Trumpet Schilke Non-Vib C#6 P
Piano MUMS C1
Trumpet Schilke Vib F#5 F
Piano Steinway A#7 P
Trumpet Schilke Vib G#5 M
Guitar Stafford B4 E M
Guitar Stafford G#3 A F
Violin Carcassi Vib E7 E M
Violin Fiumebianca Non-Vib A6 E
Violin Carcassi Non-Vib D#7 E
Trumpet Schilke Non-Vib A#5 M
Flute Louis Lot Non-Vib F6 F
Flute Louis Lot Vib G4 M
Violin JF Presssenda Vib D6 E P
Piano Steinway B7 P
Piano Steinway C8 P
Trumpet Schilke Non-Vib D6 M
Trumpet Bach Non-Vib C6 M
Piano Yam F3 F
Violin Fiumebianca Vib G5 A P
Piano Yamaha A#7 F
Trumpet Schilke Non-Vib F5 P
Piano MUMS G#3
Piano Stafford D3 E P
Guitar Kohno Masaru G4 G F
Violin JF Pressenda Vib E7 E P
Guitar Imai Yuichi F#4 G P
Piano Yamaha B0 M
Flute Louis Lot Vib F6 P
Trumpet Schilke Vib D4 M
Trumpet Bach Vib A5 M
Violin Fiumebianca Vib E7 E F
Guitar Kohno Masaru B2 A P

Violin Carcassi Vib F4 G M
Violin Fiumebianca Vib C#7 E F
Trumpet Bach Non-Vib E5 F
Flute Sankyo Vib E4 M
Piano MUMS C7
Flute Louis Lot Non-Vib G6 F
Piano Yamaha A#6 M
Flute Sankyo Non-Vib A6 P
Piano Yamaha A7 M
Piano Steinway A#0 P
Piano Yamaha G#7 M
Guitar Imai Yuichi B4 E F
Piano Steinway G#4 P
Flute Sankyo Vib F6 M
Flute Sankyo Non-Vib F6 F
Guitar Imai Yuichi D#5 E F
Guitar Imai Yuichi C#4 D M
Piano MUMS B6
Trumpet Schilke Non-Vib D#4 M
Flute Sankyo Vib A#6 P
Trumpet Schilke Non-Vib G#3 M
Piano Yamaha B7 F
Piano Yamaha F#7 P
Flute Louis Lot Non-Vib A6 M
Flute Louis Lot Vib C#4 P
Violin Fiumebianca Vib F6 E M
Violin JF Pressenda Non-Vib D5 D
Violin Carcassi Vib D5 D P
Flute Louis Lot Non-Vib G#6 M
Guitar Imai Yuichi D#3 D M
Guitar Stafford D5 E F
Flute Sankyo Non-Vib F5 F
Flute Louis Lot Vib C7 F
Guitar Kohno Masaru C5 E F
Flute Sankyo Non-Vib A#6 F
Violin Carcassi Vib D7 E F
Flute Louis Lot Non-Vib B4 P
Piano Steinway F#7 F
Trumpet Bach Vib G#5 M
Guitar Imai Yuichi G#4 E P
Guitar Stafford F4 G M
Guitar Stafford F2 E F
Flute Sankyo Vib C7 M
Piano Steinway C8 F
Flute Louis Lot Vib E4 F
Guitar Kohno Masaru C#5 E F
Trumpet Bach Non-Vib C#6 M
Piano Yamaha C8 P
Violin JF Pressenda Vib D#4 D F
Piano Steinway C#1 M
Guitar Imai Yuichi D#5 E M
Violin Carcassi Non-Vib C5 A
Flute Louis Lot Non-Vib B6 P
Trumpet Bach Vib G#5 P
Trumpet Bach Vib A3 P
Guitar Imai Yuichi C5 E M
Violin Carcassi Non-Vib C#7 E
Trumpet Bach Non-Vib F#3 F
Piano Steinway G#7 M
Trumpet Schilke non-Vib E4 F
Guitar Imai Yuichi E3 E F
Violin Carcassi Vib D#7 E P
Flute Louis Lot non-Vib F#6 P
Piano Yamaha A0F
Flute Sankyo Vib C4 P
Violin Carcassi Vib G3 G F
Piano Steinway F2 M
Violin JF Pressenda Non-Vib E7 E
Trumpet Schilke Non-Vib A#3 F
Violin Fiumebianca Vib C7 E M
Guitar Kohno Masaru A3 A P
Violin Carcassi Vib C7 E P
Violin Fiumebianca Vib B6 E M
VIolin Fiumebianca Non-Vib G#6 E
Violin JF Pressenda Vib B4 D F
Trumpet Bach Vib C#5 P
Trumpet Schilke Vib F3 P
Piano Steinway A#6 F
Violin JF Pressenda Vib G#6 E F
Piano Steinway E6 M
Violin Carcassi Non-Vib A#4 A
Trumpet Schilke Non-Vib E6 F
Trumpet Schilke Vib A5 M
Guitar Stafford F4 B M
Trumpet Bach Vib A#5 P
Trumpet Schilke Vib A3 F
Flute Louis Lot Vib B6 F
Trumpet Schilke Vib D6 F
Trumpet Bach Non-Vib E3 P
Piano Steinway C8 M
Trumpet Schilke Non-Vib F6 F
Violin JF Pressenda Vib D#7 E F
Piano Steinway C3 M
Trumpet Bach Non-Vib A5 P

Guitar Kohno Masaru 5E P
Violin JF Pressenda Vib C4 G M
Trumpet Bach Non-Vib F5 P
Piano Steinway C#7 M
Guitar Kohno Masaru D#5 E P
Violin JF Pressenda Non-Vib B6 E
Flute Louis Lot Vib C7 P
Piano Yamaha A0 P
Trumpet Schilke Vib A5 F
Guitar Stafford D5 E M
Trumpet Bach Non-Vib D6 P
Flute Sankyo Vib G6 P
Violin Fiumebianca Vib F#5 E P
Guitar Stafford G2 E P
Guitar Imai Yuichi A4 E F
Violin JF Pressenda G5 A M
Piano Yamaha B0 P
Trumpet MUMS F#3
Violin MUMS B5
Flute MUMS A6
Violin MUMS G#6
Trumpet MUMS D#6
Violin MUMS B6
Trumpet MUMS A6
Trumpet MUMS B6
Flute MUMS C4
Flute MUMS C 7
Trumpet MUMS F4
Violin MUMS A4 D
Flute MUMS C#5
Flute MUMS F#6
Violin MUMS C6
Violin MUMS A4 A

Appendix D

Listening Test Documentation

Permission to conduct the listening experiments in Chapter 8 was applied for and obtained from the University of Limerick Research Ethics Committee (ULREC No: 08/33, *Audio Feature and the Classification of Musical Instruments Using Artificial Neural Networks*). In accordance with the committee guidelines, prior to the experiment each subject read an information sheet and signed an informed consent form. This information sheet and consent form is reproduced here, along with the questionnaire that each subject was asked to fill out after the experiment.

Audio Features and the Classification of Musical Instruments using Artificial Neural Networks

Subject Information Sheet

This study aims to identify how recognisable certain instrument sounds are. Most people can identify a specific instrument (eg a violin, piano, flute) from a single note sample. It is possible, however, that when instruments are played at the extremities of their pitch range (ie very low or very high) that they may not be as identifiable as they are within their more conventional range. We wish to determine from this study if notes played at a certain pitch or dynamic level (loud or soft) become unrecognisable to the human ear.

As a participant in this study you will be asked to listen to a number of audio samples over a set of headphones. You will be asked to determine whether you think this note was played on:

a. A flute

b. A guitar

c. A piano

d. A trumpet

e. A violin

f. None of the above

You may listen to the sample as many times as you like and choose the answer on a screen from a pop-down list. The notes will be played at different pitches and at different loudness levels. You may adjust the volume level at any point during the experiment to ensure you are comfortable. You will be asked to fill in a very short questionnaire indicating your age, sex, musical experience and whether or not you play a musical experience. The examiner will be present throughout the experiment, which should last no more than 40 minutes.

The benefit of this experiment is that we will be able to decide if there are note samples that are out of the range of recognition for human subjects. The results from these tests will then be compared against an automatic instrument classifier to determine how accurate it is. There are no anticipated

risks to subjects. There is no obligation to take part and you may stop the experiment at any time you may wish to without giving a reason.

We hope that about 20-30 participants will take part in this study. The results from each participant will be stored in an anonymous form on the computer. Results from all participants will then be analysed statistically and may be published in academic papers and in a PhD thesis.

If you have any questions regarding the study, or do not understand anything you may contact the investigator:

Róisín Loughran
CSIS
A2-012
Main Building
Email: roisin.loughran@ul.ie
Tel: (061) 233290

If you have any concerns about this study and wish to contact someone independent, you may contact:

The Chariman of the University of Limerick Research Ethics Committee
C/o Vice President Academic and Registrars Office
University of Limerick
Limerick
Tel: (061) 202022

Audio Features and the Classification of Musical Instruments using Artificial Neural Networks

Informed Consent Form

Please read the associated Subject information Sheet. If you wish to participate in this study, please sign and date this form.
I declare that:

- I have read and understood the subject information sheet.
- I understand what the project is about, and what the results will be used for.
- I am fully aware of all of the procedures involving myself, and of any risks and benefits associated with the study.
- I know that my participation is voluntary and that I can withdraw from the project at any stage without giving any reason.
- I am aware that my results will be kept confidential.

Subject: **Date:**

Witness: **Date:**

Investigator: **Date:**

Audio Features and the Classification of Musical Instruments using Artificial Neural Networks

Questionnaire

Age:

Sex:

Do you play a musical instrument(s)? If so, what instrument(s) and to what level (Highest grade or number of years studied)?

Please indicate below if you have any academic qualifications in music:

- Junior Cert Music
- Leaving Cert Music
- Degree (BMus)
- Postgraduate (please specify)
- Other (please specify)

Please give any comments on the experiment in general:

Appendix E

CD Listings

The CD accompanying this thesis contains a number of folders containing the main Matlab files and data files used in the experiments described throughout the thesis. A brief description of the content of the folders is given below.

- **ANN** This folder contains the Matlab files written for training and testing an MLP, along with code used for accessing and normalising stored data.
- **DataFiles** Due to the licensing of the samples used and space considerations, the original samples could not be included with this thesis. This folder contains some of the data extracted from the original samples in three subfolders. PCA data from the full set of samples is included along with the best genomes evolved from each of the methods described in Chapter 6 and the data in the cross validation sets used in both Chapter 6 and Chapter 7.
- **Feature Selection** This folder contains a number of files which, along with the MIRToolbox, were used to obtain timbral features from the audio sample files.
- **GA** This contains the main files written to conduct the GA experiments in Chapter 6 and to analyse the results obtained.
- **GP** This contains the main files written to conduct the GP experiments in Chapter 7 and to analyse the results obtained.
- **Listening Tests** This folder contains the files written to conduct the Listening Tests described in Chapter 8.

Bibliography

Affenzeller, M., Winkler, S., Wagner, S., and Behan, A. (2009). *Genetic Algorithms and Genetic Programming: Modern Concepts and Practical Applications*. CRC Press, Taylor and Francis Group.

Agostini, G., Longari, M., and Pollastri, E. (2001). Content-based classification of musical instrument timbres. In *International Workshop on Content-Based Multimedia indexing*.

Agostini, G., Longari, M., and Pollastri, E. (2003). Musical instrument timbres classification with spectral features. *EURASIP Journal on Applied Signal Processing*, 1.

Amatriain, X. (2007). CLAM: A Framework for Audio and Music Application Development. *IEEE Software*, 24.

ANSI (1973). American National Standard-Psychoacoustical Terminology. American National Standards Institute, New York.

Balaban, M., Ebcioglu, K., and Laske, O., editors (1992). *Understanding Music with AI: Perspectives on Music Cognition*. AAAI Press/MIT Press.

Bank, B. and Valimaki, V. (2003). Robust loss filter design for digital waveguide synthesis of string tones. *Signal Processing Letters*, 10:18–20.

Banzhaf, W., Nordin, P., Keller, R., and Francone, F. (1998). *Genetic Programming, An Introduction*. Morgan Kaufmann, San Francisco, California.

Beauchamp, J. (2007). *Analysis, Synthesis and Perception of Musical Sounds, The Sound of Music*. Springer Science and Business Media.

Bellman, R. (2003). *Dynamic Programming*. Mineola N.Y. : Dover Publications.

Benade, A. (1990). *Fundamentals of Musical Acoustics*. Dover Publications Inc., New York.

Berger, K. W. (1964). Some factors in the recognition of timbre. *Journal of Acoustical Society of America*, 36(10).

Biles, J. (1994). GenJam: A genetic algorithm for geverating jazz solos. In *International Computer Music Conference*.

Biles, J. (2001). GenJam: Evolution of a jazz improviser. In Bentley, P. and Corne, D., editors, *Creative Evolutionary Systems*. Morgan Kaufmann.

Blum, A. and Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intelligence*, pages 245–271.

Borg, I. and Groenen, P. (2005). *Modern Multidimensional Scaling: Theory and Application*. Springer-Verlag, New York, 2nd edition.

Brabazon, A. and O'Neill, M., editors (2009). *Natural Computing in Computational Finance (Volume 2)*, volume 185. Springer.

Bregman, A. (1990). *Auditory Scene Analysis: The Perceptual Organisation of Sound*. MIT Press.

Brookes, M. (2009). Voicebox toolbox. URL `www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html` Accessed 12 January 2009.

Brown, J. (1999). Computer identification of musical instruments using pattern recognition with cepstra coefficients as features. *Journal of Acoustical Society of America*, 105(3).

Brown, J., Houix, O., and McAdams, S. (2001). Feature dependence in the automatic identification of musical woodwind instruments. *Journal of Acoustical Society of America*, 109(3).

Burton, A. and Vladimirova, T. (1999). Generation of Musical Sequences with Genetic Techniques. *Computer Music Journal*, 23(4):59–73.

Caclin, A., McAdams, S., Smith, K., and Winsberg, S. (2005). Acoustic correlates of timbre space dimensions: A confirmatory study using synthetic tones. *Journal of Acoustical Society of America*, 118(1):471–482.

Carpenter, G., Grossberg, S., and Reynolds, J. (1991). Artmap: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. *Neural Networks*, 4.

Cemgil, A. and Gurgen, F. (1997). Classification of musical instrument sounds using neural networks. In *SIU*, Bodrum, Turkey.

Chang, S., Sikora, T., and Puri, A. (2001). Overview of the MPEG-7 standard. *IEEE Transactions on Circuits and Susyems for Video Technology*, 11(6).

Chetry, N. and Sandler, M. (2006). Linear predicative models for musical instrument identification. In *ICASSP*, volume 5, pages 225–228, Toulouse, France.

Chowning, J. (1973). The synthesis of complex audio spectra by means of frequency modulation. *Journal of Audio Engineering Society*, 21(7):526–534.

Conrads, M., Nordin, P., and Banzhaf, W. (1998). Speech sound discrimination with genetic programming. In *Lecture Notes in Computer Science*, volume 1391, pages 113–129.

Cooley, J. and Tukey, J. (1965). An Algorithm for the Machine Calculation of Complex Fourier Series. *Mathematics of Computation*, 19(90):297–301.

Cosi, P., De Poli, G., and Lauzzana, G. (1994). Auditory modelling and self-organizing neural networks for timbre classification. *Journal of New Music Research*.

Cover, T. and Van Campenhout, J. (1977). On the possible orderings in the measurement selection problem. *IEEE Transactions on Systems, Man and Cybernetics*, 7(9).

Cristianini, N. and Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

Cuadra, L., Alexandre, E., Alvarez, L., and Rosa-Zurera, M. (2008). Reducing the computational cost for sound classification in hearing aids by selecting features via genetic algorithms with restricted search. In *International Conference on Auditory, Language and Image Processing*, pages 1320–1327.

Darke, G. (2005). Assessment of timbre using verbal attributes. In *Conference on Interdisciplinary Musicology*.

Darwin, C. (1859). *On the Origin of the Species*. John Murray, London.

Dasarathy, B., editor (1990). *Nearest Neighbour: Pattern Classification Techniques*. IEEE Computer Society.

Dash, M. and Liu, H. (1997). Feature selection for classification. *Intelligent Data Analysis*, 1:131–156.

Davis, L. (1989). Adapting operator probabilities in genetic algorithms. In *Proceedings of the third international conference on Genetic Algorithms*, pages 61–69, San Francisco, California. Morgan Kaufmann Publishers Inc.

Day, P. and Nandi, A. (2007). Robust text-independent speaker verification using genetic programming. In *IEEE Transaction on Audio, Speech and Language Processing*, volume 15, pages 285–296.

de Cheveigne, A. (2005). Pitch Perception Models. In *Pitch: Neural Coding and Perception*, chapter 6. Birkhauser.

de Cheveigne, A. and Kawahara, H. (2002). YIN, A fundamental frequency estimator for speech and music. *Journal of Acoustical Society of America*, 111(4):1917–1930.

De Poli, G. and Prandoni, P. (1997). Sonological models for timbre characterization. *Journal of New Music Research*, 26(2):170–198.

Dennis, J. and Schnabel, R. (1996). *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Philadelphia: Society for Industrial and Applied Mathematics.

Donnnadieu, S. (2007). Mental representation of the timbre of complex sounds. In Beauchamp, J., editor, *Analysis, Synthesis and Perception of Musical Sounds, The Sound of Music*, chapter 8. Springer.

Dougherty, E. (1998). Random processes for image and signal processing. In *SPIE/IEEE series on Imaging Science and Engineering*, Bellingham, WA.

Dubnov, S., Tishby, N., and Cohen, D. (1997). Polyspectra as measure of sound texture and timbre. *Journal of New Music Research*.

Duda, R., Hart, P., and Stork, D. (2001). *Pattern Classification*. Wiley-Interscience, New York, second edition.

Eagleson, H. V. and Eaglson, O. W. (1947). Identification of musical instruments when heard directly and over a public-address system. *Journal of Acoustical Society of America*, 19(2).

Elliott, C. A. (1975). Attacks and releases as factors in instrument identification. *Journal of Research in Music Education*, 23(1).

Eronen, A. (2001). Comparison of features for musical instrument recognition. In *Workshop on Application of Signal Processing to Audio and Acoustics*, pages 19–22, New York.

Eronen, A. (2003). Musical instrument recogntion using ICA-based transform of features and discriminatively trained HMMs. In *Seventh International Symposium on Signal Processing and its Applications*.

Eronen, A. and Klapuri, A. (2000). Musical instrument recognition using cepstral coefficients and temporal features. In *ICASSP*.

Essid, S., Richard, G., and David, B. (2004). Efficient musical instrument recognition on solo performance music using basic features. In *AES 25th International Conference*.

Essid, S., Richard, G., and David, B. (2006). Musical instrument recognition by pairwise classification strategies. In *IEEE Transaction on Audio, Speech and Language Processing*, volume 14, pages 1401–1412.

Ferri, F., Kadirkamanathan, V., and Kittler, J. (1993). Feature subset search using genetic algorithms. In *IEEE Workshop on Natural Algorithms in Signal Processing*.

Fletcher, H., Blackham, E., and Stratton, R. (1962). Quality of Piano Tones. *Acoustical Society of America*, 34(6):749–761.

Fletcher, H. and Munson, W. (1933). Loudness, its definition, measurement and calculation. *Acoustical Society of America*, 5.

Fragoulis, D., Avaritsiotis, J., and Papaodysseus, C. (1999). Timbre recognition of single notes using an artmap neural network. In *ICECS*.

Fraser, A. and Fujinaga, I. (1999). Toward real-time recognition of acoustic musical instruments. In *ICMC*, pages 175–177, China.

Fritts, L. (1997). University of iowa musical instrument samplesowa musical instrument samples. URL `http://theremin.music.uiowa.edu/MIS.html`.

Fujinaga, I. (1998). Machine recognition of timbre using steady-state tone of acoustic musical instruments. In *ICMC*, pages 207–210, Ann Arbor, USA.

Fujinaga, I., Moore, S., and Sullivan, Jnr, D. (1998). Implementation of exemplar-based learning model for music cognition. In *International Conference on Music Perception*.

Garcia, R. (1996). Automatic generation of sound synthesis techniques. In *Program in Media Arts and Sciences*, MA:MIT Press. MIT.

Garcia, R. (2001). Automating the design of sound synthesis techniques using evolutionary methods. In *COST G-6 Conference on Digital Audio Effects*, Limerick, Ireland.

Gaver, W. (1993). How do we hear in the world - exploration in ecological acoustics. *Ecological Psychology*, 5:285–313.

Glasberg, B. and Moore, B. (1990). Derivation of auditory filter shapes from notched-noise data. *Hearing Research*, 47:103–138.

Goksu, H., Pigg, P., and Dixit, V. (2005). Music composition using genetic algorithms and multilayer perceptrons. In *Lecture Notes in Computer Science*, volume 3612/2005, pages 1242–1250.

Goldberg, D. (1989). *Genetic Algorithms in Search Optimization and Machine Learning*. Addison-Wesley Oublishing Company.

Goldberg, D. (2002). *The Design of Innovation (Genetic Algorithms and Evolutionary Computation)*. Springer.

Goto, M. (2004). Development of the RWC music database. In *Proceedings of the 18th Congress on Acoustics*.

Greenwood, D. (1961a). Auditory Masking and the Critical Band. *Journal of Acoustical Society of America*, 33(4):484–502.

Greenwood, D. (1961b). Critical bandwidth and the frequency coordinates of the basilar Membrane. *Journal of Acoustical Society of America*, 33(10):484–496.

Greenwood, D. (1997). The mel scale's disqualifying bias and a consistency of pitch-difference equisections in 1956 with equal cochlear distances and equal frequency ratios. *Hearing Research*, 103(1-2):199–224.

Grey, J. (1977). Multidimensional perceptual scaling of musical timbres. *Journal of Acoustical Society of America*, 61:1270–1277.

Grey, J. and Gordon, J. (1978). Percetual effects of spectral modifications on musical timbres. *Journal of Acoustical Society of America*, 63:1493–1500.

Guyon, I. and Elisseff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182.

Hajda, J., Kendell, R., Carterette, E., and Harshberger, M. (1997). Methodological issues in timbre research. In Deliege, I. and Sloboda, J., editors, *Perception and Cognition of Music*, chapter 12. Psychology Press Ltd.

Handel, S. (1989). *Listening, An introduction to the Perception of Auditory Events*. MIT Press.

Handel, S. (1995). Timbre perception and auditory object identification. In Moore, B., editor, *Hearing*, chapter 12. Academic Press Inc.

Haritaoglu, E. (1997). Human auditory masking and perception based coding. URL `www.umiacx.umd.edu/~desin/Speech1/node9.html`.

Haykin, S. (1999). *Neural Networks - A Comprehensive Foundation*. Prentice Hall.

Herrera, P., Amatriain, X., Batlle, E., and Serra, X. (2000). Towards instrument segmentation for music content description: A critical review of instrument classification techniques. In *ISMIR*, Plymouth, Massachusetts.

Herrera, P., Klapuri, A., and Davy, M. (2006). *Automatic Classification of Pitched Musical Instrument Sounds*, chapter 6. Springer US.

Herrera-Boyer, P., Peeters, G., and Dubnov, S. (2003). Automatic classification of musical instrument sounds. *Journal of New Music Research*, 32(1).

Holm, J. and Toiviainen, P. (2004). A Method for Modelling Finite Width Excitation in Physical Modelling Sound Synthesis of Bowed Strings. *Journal of New Music Research*, 33:345–354.

Horner, A. (1996). Double modulator FM matching of instrument tones. *Computer Music Journal*, 20(2):57–71.

Horner, A. (1998). Nested modulator and feedback FM matching of instrument tones. *IEEE Transactions on Speech and Audio Processing*, 6(4).

Horner, A. (2001). A simplified wavetable matching method using combinatorial basis spectra selection. *Journal of Audio Engineering Society*, 49(11):1060–1066.

Horner, A. (2003). Auto-programmable FM and wavetable synthesisers. *Contemporary Music Review*.

Horner, A. and Ayres, L. (1995). Harmonisation of musical progression with genetic algorithms. In *International Computer Music Conference*, pages 483–484. San Francisco: International Computer Music Association.

Horner, A. and Beauchamp, J. (1996). Piecewise linear approximation of additive synthesis envelopes: A comparison of various methods. *Computer Music Journal*, 20(2):72–95.

Horner, A., Beauchamp, J., and Haken, L. (1993a). Machine tongues XVI: Genetic algorithms and their application to FM matching synthesis. *Computer Music Journal*, 17(4):17–29.

Horner, A., Beauchamp, J., and Haken, L. (1993b). Methods for multiple wavetable synthesis of musical instrument tones. *Journal of Audio Engineering Society*, 41(5):336–356.

Horner, A. and Goldberg, D. (1991). Genetic algorithms and computer-assisted music composition. In *International Computer Music Conference*, pages 479–482. San Francisco: International Computer Music Association.

Horowitz, D. (1994). Generating rhythms with genetic algorithms. In *The Twelfth National Conference on Artificial Intelligence*, volume 2.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 498–520.

Hourdin, C., Charbonneau, G., and Moussa, T. (1997). A multidimensional scaling analysis of musical instruments' time varying spectra. *Computer Music Journal*, 21(2):40–55.

Howard, D. and Angus, J. (1996). *Acoustics and Psychoacoustics*. Focal Press.

Ihara, M., Maeda, S., and Ishii, S. (2007). Instrument identification in monophonic music using spectral information. In *IEEE Conference on Signal Processing and Information Technology*.

Ilmoniemi, M., Valimaki, V., and Huotilainen, M. (2004). Subjective evaluation of musical instrument timbre modifications. In *Joint Baltic-Nordic Acoustics Meeting*.

Information Society (2009). MUSCLE network of excellence. URL `www.ifs.tuwien.ac.at/mir/muscle/del/audio_tools.html` Accessed 12 January 2009.

Jensen, K. (1999). *Timbre Models of Musical Sounds*. PhD thesis, Department of Computer Science, University of Copenhagen.

Johnston, C. (1999). Exploring the sound-space of synthesis algorithms using interactive genetic algorithms. In *AISB Workshop on Artificial Intelligence*.

Johnston, I. (1994). *Measured Tones, The Interplay of Physics and Music*. Institute of Physics Publishing.

Jolliffe, I. (2002). *Principal Component Analysis*. Springer-Verlag.

Kaminskyj, I. and Materka, A. (1995). Automatic source identification of monophonc musical instrument sounds. In *IEEE International Conference on Neural Networks*, pages 189–194.

Karjailnen, M., Tolonen, T., Valimaki, V., Erkut, C., Laurson, M., and Hiipakka, J. (2001). An overview of new techniques and effects in model-based sound synthesis. *Journal of New Music Research*, 30:203–212.

Karplus, K. and Strong, A. (1983). Digital synthesis of plucked-string and drum timbres. *Computer Music Journal*, 7:43–55.

Keane, M. (2004). Understanding the complex nature of piano tones. Technical report, Acoustics Research Centre, University of Auckland.

Kendell, R. (1986). The role of acoustic signal partitions in listener categorization of musical phrases. *Music Perception*, 4(2).

Kendell, R. and Carterette, E. (1993). Verbal attributes of simultaneous wind instrument timbres. *Music Perception*, 10(4):445–468.

Kitahara, T., Goto, M., and Okuno, H. G. (2003). Musical instrument identification based on f0-dependent multivariate normal distribution. In *Multimedia and Expo, ICME*.

Klapuri, A. (2006). Multiple Fundamental Frequency Estimation by Summing Harmonic Amplitudes. In *International Conference on Music Information Retrieval (ISMIR)*, Victoria, Canada.

Koch, R. (2000). A tutorial in probability and statistics. URL `http://web.cecs.pdx.edu/~roy/tutorial/Stat_int.htm` Accessed 14 January 2009.

Kohonen, T. (2001). *Self-organizing Maps*. Springer.

Koza, J. (1992a). Genetic evolution and co-evolution of game strategies. In *The International Conference on Game Theory and Its Applications*, Stony Brook, New York.

Koza, J. (1992b). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press.

Koza, J. (1992c). Hierarchical automatic function definition in genetic programming. In Whitley, L., editor, *Foundations of Genetic Algorithms 2*, pages 297–318. Morgan Kaufmann.

Koza, J. (1994). *Genetic Programming II: Automatic Discovery of Reusable Programs*. MIT.

Krimphoff, J., McAdams, S., and Winsberg, S. (1994). Caracterisation du timbre des sons complexes. II. analyses acoustiques et quantification psychophysique. *Journal De Physique IV*, 4.

Kudo, M. and Sklansky, J. (2000). Comparison of algorithms that select features for pattern classifiers. *Pattern Recognition*, 33(1):25–41.

Landry, J., Da Costa, L., and Bernier, T. (2006). Disciminant feature selection by genetic programming: Towards a domain independent multi-class object detection system. *Journal of Systemics, Cybernetics and Informatics*, (3):76–81.

Langdon, W. and Poli, R. (2002). *Foundations of Genetic Programming*. Springer.

Langdon, W., Poli, R., McPhee, N., and Koza, J. (2008). Genetic programming: An introduction and tutorial, with a survey of techniques and applications. *Studies in Computational Intelligence*, 115:927–1028.

Lartillot, O. (2008). *MIRToolbox 1.1, Users Manual*. Finnish Center of Excellence in interdisciplinary Music Research, University of Jyvaskyla, Finland, 1.1 edition.

Lartillot, O. and Toiviainen, P. (2007). MIR in Matlab (ii): A Toolbox for Musical Feature Extraction from Audio. In *International Conference on Music Information Retrieval (ISMIR)*, Vienna, Austria.

Leman, M. (1996). *Music and Schema Theory: Cognitive Foundations of Systematic Musicology*. Springer.

Leman, M., Lesaffre, M., and Tanghe, K. (2005). IPEM toolbox. URL `www.ipem.ugent.be/Toolbox/` Accessed 12 January 2009.

Licklider, J. (1951). A Duplex Theory of Pitch Perception. *Journal of Acoustical Society of America*, 23(1).

Lim, S. and Tan, B. (1999). Performance of the genetic annealing algorithm in DFM synthesis of dynamic musical sound samples. *Journal of Audio Engineering Society*, 47(5):339–354.

Livshin, A. and Rodet, X. (2004). Musical instrument identification in continuous recordings. In *7th International Conference on Digital Audio Effects (DAFx '04)*.

Logan, B. (2000). Mel-frequency cepstral coefficients for music modelling. In *ISMIR*.

Loughran, R., Walker, J., and O'Neill, M. (2007). Instrument identification using artificial neural networks. Poster Presentation at Sound and Music Computing Summer School, Stockholm, Sweden.

Loughran, R., Walker, J., and O'Neill, M. (2009). An exploration of genetic algorithms for efficient musical instrument identification. In *Irish Signals and Sytems Conference*, Dublin, Ireland.

Loughran, R., Walker, J., O'Neill, M., and M., O. (2008a). Musical instrument identification using principal component analysis and multi-layered perceptrons. In *International Conference on Auditory, Language and Image Processing*, pages 643–648, Shanghai, China.

Loughran, R., Walker, J., O'Neill, M., and O'Farrell, M. (2008b). Comparison of features for musical instrument identification using artificial neural networks. In *CMMR*, pages 19–33, Copenhagen, Denmark.

Loughran, R., Walker, J., O'Neill, M., and O'Farrell, M. (2008c). The use of mel-frequency cepstral coefficients in musical instrument identification. In *International Computer Music Conference*, Belfast, Northern Ireland.

Loveard, T. and Ciesielski, V. (2001). Representing classification problems in genetic programming. In *Proceedings of the 2001 Congress on Evolutionary Computation*, volume 2, pages 1070–1077, Seoul, South Korea.

Maher, R. C. and Beauchamp, J. (1994). Fundamental Frequency Estimation of Musical Signals Using a Two-way Mismatch Procedure. *Journal of Acoustical Society of America*, 95(4):2254–2263.

Mandelis, J. (2001). Genophone: An evolutionary approach to sound synthesis and performance. In *ALMMA Workshop on Artificial Models for Musical Applications*, pages 108–119, Cosenza, Italy.

Manzoli, J., Maia, A. J., Fomari, J., and Damiani, F. (2001). The evolutionary sound synthesis method. In *Ninth ACM International Conference on Multimedia*, volume 9, pages 585–587.

Marques, J. and Moreno, P. (1999). A study of musical instrument classification using gaussian mixture models and support vector machines. Technical report, Cambridge Research Laboratory.

Martin, K. (1999). *Sound Source Recognition: A Theory and Computational Model*. PhD thesis, MIT.

Martin, K. and Kim, Y. (1998). Musical instrument: A pattern-recognition approach. In *136th Meeting of the Acoustical Society of America*, Norfolk, USA.

Martinez, W. and Martinez, A. (2002). *Computation Statistics Handbook with Matlab*. Chapman and Hall.

Mathews, M. (1999). The Ear and How it Works. In Cook, P., editor, *Music, Cognition and Computerised Sound: And Introduction to Psychoacoustics*, chapter 1. MIT Press.

MATLAB7 (2006). Version 7.2 (r2006a) matlab software. In *The Mathworks*.

McAdams, S. and Cunible, J. (1992). Perception of timbral analogies. *Philosophical Transactions of the Royal Society of London Series B*, 336:383–389.

McAdams, S., Winsberg, S., Donnnadieu, S., De Soete, G., and Krimphoff, J. (1995). Perceptual scaling of synthesised musical timbres: Common dimensions, specificities and latent subject classes. *Psychological Research*, 58:177–192.

McCullock, W. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133.

McDermott, J. (2008). *Evolutionary Computation Applied to the Control of Sound Synthesis*. PhD thesis, University of Limerick.

McDermott, J., Griffith, N., and O'Neill, M. (2005). Toward user-directed evolution of sound synthesis parameters. In *Lecture Notes in Computer Science*, volume 3449/2005. Springer.

McKinney, M. and Breebart, J. (2003). Features for audio and music classification. In *ISMIR*.

Merhav, N. and Lee, C. (1993). On the Asymptotic Statistical Behaviour of Empirical Cepstral Coefficients. *IEEE Transactions on Signal Processing*, 41(5):1990–1993.

Minsky, M. and Papert, S. (1969). *Perceptrons*. PhD thesis, Cambridge, MA:MIT Press.

Miranda, E. (1995). Granular synthesis of sounds by means of cellular automata. *Leonardo*, 28(4):297–300.

Miranda, E. (2000). The art of rendering sounds from emergent behaviour: Cellular automata granular synthesis. In Vajda, F., editor, *25th EUROMICRO Conference*, volume 2, pages 350–355, Los Alamitos, California. IEEE Computer Society.

Miranda, E. (2002). Mimetic model of intonation. In Anagnostopoulou, C., Ferrand, M., and Smaill, A., editors, *Lecture Notes on Artificial Intelligence*, pages 107–118, Berlin. Springer-Verlag.

Miranda, E. (2004). At the crossroads of evolutionary computation and music: Self-programming synthesisers, swarm orchestras and the origins of melody. *Evolutionary Computation*, 12:137–158.

Miranda, E., Kirby, S., and Todd, P. (2003). On computational models of the evolution of music: From the origins of musical taste to the emergence of grammars. *Contemporary Music Review*, 22(3):91–111.

Mitchell, M. (1996). *An Introduction to Genetic Algorithms*. MIT Press.

Moore, B. and Glasberg, B. (1983). Suggested Formulae for Calculating Auditory-filter Bandwidths and Excitation Patterns. *Journal of Acoustical Society of America*, 74(3):750–753.

Moroni, A., Manzolli, J., Von Zuben, F., and Gudwin, R. (2000). Vox Populi: An interactive evolutionary system for algorithmic music composition. *Leonardo Music Journal*, 10:49–54.

Nackaerts, A., Moor, B., and Lauwereins, R. (2003). A formant filtered physical model for wind instruments. *Transactions on Speech and Audio Processing*, 11:36–44.

Nielson, A., Sigurdsson, S., Hansen, L., and Arenas-Garcia, J. (2007). On the relevance of spectral features for instrument classification. In *ICASSP*, pages 485–488, Honolulu, Hawaii.

O'Neill, M. and Ryan, C. (2001). Grammatical evolution. *IEEE Transactions on Evolutionary Computation*, 5(4).

Opolko, F. and Wapnick, J. (1987). McGill university master samples (MUMS) (cds).

Oppenheim, A. and Schafer, R. (1989). *Discrete-time Signal Processing*. Prentice Hall, New Jersey.

Osgood, C., Suci, G. J., and Tannenbaum, P. (1957). *The Measurement of Meaning*. Univeristy of Illinois Press.

O'Shaughnessy, D. (1987). *Speech Communication Human and Machine*. Addison-Wesley Series in Electrical Engineering.

Peeters, G. (2003). Automatic classification of large musical instrument databases using hierarchical classifiers with inertia ratop maximization. In *AES 115th Convention*.

Peeters, G. (2006). Music pitch representation by periodicity measures based on combined temporal and spectral representations. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Toulouse, France.

Peeters, G. and Rodet, X. (2003). Hierarchical gaussian tree with inertia ratio maximization for the classification of large musical instrument databases. In *6th International Conference on Digital Audio Effects (DAFX)*.

Plomp, R. (1970). Timbre as multidimensional attribute of complex tones. *Frequency Analysis and Periodicity Detection in Hearing*.

Quinn, D. (2009). *Blind Source Separation of Music Streams*. PhD thesis, University of Limerick.

Raja, A., Azad, R., Flanagan, C., and Ryan, C. (2008). VoIP speech quality estimation in a mixed context with genetic programming. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, pages 1627–1634, Atlanta, GA, USA.

Rho, S., Hwang, E., and Kim, M. (2007). Music information retrieval using a ga-based relevence feedback. In *International Conference on Multimedia and Ubiquitous Engineering*, pages 739–744.

Riedmiller, M. and Braun, H. (1993). A direct adaptive method for faster backpropagation learning: The rprop algorithm. In *IEEE Conference on Neural Networks*. IEEE.

Riionheimo, J. and Valimaki, V. (2003). Parameter estimation of a plucked string synthesis model using a genetic algorithm with perceptual fitness calculation. *EURASIP Journal on Applied Signal Processing*, 8:791–805.

Roads, C. (1996). *The Computer Music Tutorial*. MIT Press, Cambridge, USA.

Rojas, R. (1996). *Neural Networks - A Systematic Introduction*. Springer.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for infomation storage and organization in the brain. *Psychological Review*, 65:386–408.

RWC (2001a). Rwc-mdb-i-2001-w01, instrument no.1 pianoforte.

RWC (2001b). Rwc-mdb-i-2001-w05, instrument no.15 violin.

RWC (2001c). Rwc-mdb-i-2001-w09, instrument no.33: Flute.

Saldanha, E. and Corso, J. (1964). Timbre cues and the identification of musical instruments. *Journal of Acoustical Society of America*.

Sandell, G. (1994). Sharc timbre database. URL `http://www.timbre.ws/sharc/files/README.txt`.

Scharf, B. (1970). Critical bands. In Tobias, J., editor, *Foundations of Modern Auditory Theory*, chapter 1. London:Academic Press.

Schottstaedt, B. (1977). The simulation of natural instrument tones using frequency modulation with a complex modulating wave. *Computer Music Journal*, 1(4):46–50.

Seneff, S. (1990). A joint synchrony/mean-rate model of auditory speech processing. In *Readings in Speech Recognition*.

Serra, X. (1990). Spectral Modelling Synthesis: A Sound Analysis/Synthesis System Based on a Deterministic Plus Stochastic Decomposition. *Computer Music Journal*.

Sethares, W. (1998). *Tuning, Timbre, Spectrum, Scale*. Springer.

Shephard, R. N. (1964). Circularity in judgements of relative pitch. *Journal of Acoustical Society of America*, 36(12):2346–2353.

Siedlecki, W. and Sklansky, J. (1993a). A note on genetic algorithms for large-scale feature selection. In Chen, C., Pau, L., and Wang, P., editors, *Handbook of Pattern and Computer Vision*. World Scientific.

Siedlecki, W. and Sklansky, J. (1993b). On automatic feature selection. In Chen, C., Pau, L., and Wang, P., editors, *Handbook of Pattern and Computer Vision*. World Scientific.

Silva, S. (2004). GPLAB - a genetic programming toolbox for MATLAB. URL `http://gplab.sourceforge.net/` Accessed 07 June 2009.

Silva, S. and Almeida, J. (2003). Dynamic maximum tree depth - a simple technique for avoiding bloat in tree-based gp. In *GECCO*, volume LNCS 2724, pages 1776–1787. Springer-Verlag.

Sima, C., Attoot, S., Brag-Neto, U., Lowey, J., Suh, E., and Dougherty, E. (2005). Impact of error estimation on feature selection. *Pattern Recognition*, 38(12):2005.

Simmermacher, C., Deng, D., and Cranefield, S. (2006). Feature analysis and classification of classical musical instruments: An empirical study. In *The Information Science Discussion Paper Series*.

Srinivasan, A., Sullivan, D., and Fujinaga, I. (2002). Recognition of isolated instrument tones by conservatory students. In *International Conference on Music Perception and Cognition*, Sydney, Australia.

Stevens, S. and Volkmann, J. (1940). The Relation of Pitch to Frequency: A Revised Scale. *The American Journal of Psychology*, 53(3):329–353.

Stevens, S., Volkmann, J., and Newman, E. (1937). A Scale for the Measurement of the Psychological Magnitude Pitch. *Journal of Acoustical Society of America*, 8(3):185–191.

Takala, T., Hahn, J., Gritz, L., Geigel, J., and Lee, J. (1993). Using physically-based models and genetic algorithms for functional composition of sound signal, synchronised to animated motion. In *International Computer Music Conference*.

Teolis, A. (1998). *Computational Signal Processing with Wavelets*. Birkhauser.

Terasawa, H., Slaney, M., and Berger, J. (2005). Perceptual distance in timbre space. In *ICAD*.

TestMyBrain (2009). Test my brain. URL `www.testmybrain.org`.

Thywissen, K. (1999). GeNotator: An environment for exploring the application of evolutionary techniques in computer-assisted composition. *Organised Sound*, 4(2):127–133.

Todd, P. and Loy, D., editors (1991). *Music and Connectionism*. MIT.

Todd, P. and Werner, G. (2000). Frankensteinian methods for evolutionary music composition. In Griffith, N. and Todd, P., editors, *Musical Networks: Parallel Distributed Perception and Performance*. MIT Press.

Tseng, L. and Yang, S. (1997). Genetic algorithms for clustering, feature selection and classification. In *Internation Conference on Neural Networks*.

Tzanetakis, G. and Cook, P. (2002). Music Genre Classification of Audio Signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302.

Van Noorden, L. (1975). *Temporal Coherence in the Perception of Tone Sequences*. PhD thesis, Eindhoven University of Technology, Eindhoven, Pays-Bas, Germany.

Verma, T. (2000). Extending Spectral Modelling Synthesis with Transient Modelling Synthesis. *Computer Music Journal*, 24:47–59.

Vinet, H., Herrera, P., and Pachet, F. (2002). The cuidado project. In *International Conference on Music Information Retrieval (ISMIR)*, Paris, France.

von Bismarck, G. (1974). Timbre of steady sounds: A factorial investigation of its verbal attributes. *Acustica*, 30.

Wasserman, P. (1989). *Neural Computing: Theory and Practice*. Van Nostrand Resinhold Co., New York.

Watkinson, J. (2004). *The MPEG Handbook*. Focal Press.

Wessel, D. (1979). Timbre space as a musical control structure. *Computer Music Journal*, 3(2):45–52.

Wiggins, G., Papadopoulos, G., Phon-Amnuaisuk, S., and Tuson, A. (1999). Evolutionary methods for musical composition. *International Journal of Computing Anticipatory Systems*, 1.

Wold, W., Blum, T., Keislar, D., and Wheaton, J. (1996). Content-based classification, search and retrieval of audio. *Multimedia*, 3(3).

Wolfram, S. (2005). Cellular automata. In Kier, L., Seybold, P., and Cheng, C., editors, *Modelling Chemical Sysytems Using Cellular Automata*. Springer Netherlands.

Zurada, J. (1995). *Introduction to Artificial Neural Systems*. West Publishing Company.

Zwicker, E. and Terhardt, E. (1980). Analytical expressions for critical-band rate and critical bandwidth as a function of frequency. *Journal of Acoustical Society of America*, 68(5):1523–1525.