A CONVOLUTIVE MODEL FOR POLYPHONIC INSTRUMENT IDENTIFICATION AND PITCH DETECTION USING COMBINED CLASSIFICATION

by

JOSHUA L. WEESE

B.S., Kansas State University, 2011

A THESIS

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

COMPUTING AND INFORMATION SCIENCES

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2013

Approved by:

Major Professor
WILLIAM H. HSU

Copyright

Abstract

Pitch detection and instrument identification can be achieved with relatively high accuracy when considering monophonic signals in music; however, accurately classifying polyphonic signals in music remains an unsolved research problem. Pitch and instrument classification is a subset of Music Information Retrieval (MIR) and automatic music transcription, both having numerous research and real-world applications. Several areas of research are covered in this thesis, including the fast Fourier transform, onset detection, convolution, and filtering. Basic music theory and terms are also presented in order to explain the context and structure of data used. The focus of this thesis is on the representation of musical signals in the frequency domain. Polyphonic signals with many different voices and frequencies can be exceptionally complex. This thesis presents a new model for representing the spectral structure of polyphonic signals: Uniform MAx Gaussian Envelope (UMAGE). The new spectral envelope precisely approximates the distribution of frequency parts in the spectrum while still being resilient to oscillating rapidly (noise) and is able to generalize well without losing the representation of the original spectrum. When subjectively compared to other spectral envelope methods, such as the linear predictive coding envelope method and the cepstrum envelope method, UMAGE is able to model high order polyphonic signals without dropping partials (frequencies present in the signal). In other words, UMAGE is able to model a signal independent of the signal's periodicity. The performance of UMAGE is evaluated both objectively and subjectively. It is shown that UMAGE is robust at modeling the distribution of frequencies in simple and complex polyphonic signals. Combined classification (combiners), a methodology for learning large concepts, is used to simplify the learning process and boost classification results. The output of each learner is then averaged to get the final result. UMAGE is less accurate when identifying pitches; however, it is able to achieve accuracy in identifying instrument groups on order-10 polyphonic signals (ten voices), which is competitive with the current state of the field.

Table of Contents

List of Figures

List of Tables

Acknowledgements

First I would like to thank my advisor, Dr. William Hsu for his support and guidance. His knowledge and experience in research helped push my progress along. I would also like to thank my family and my wonderful fiancé Chelsea Hanks for their continued support in furthering my education and experience. I would also like to thank Heath Yates for his assistance on the statistical methods implemented in this thesis.

Dedication

No one has said that losing a friend is easy. Through discussions about the facts of life, technology, science, cooking, literature, art, and education, we progressed not only as friends, but family. Your passion for educating the children you worked with, your dedication and love to those around you, your open-mindedness, and your freewill remains unsurpassed. Inspiration comes from many sources, especially from those close to your heart. Thank you for the drive, the encouragement, and inspiration. Gone, but not forgotten. In loving memory of Gail Turnbough.

> *"It is a curious thing, the death of a loved one. We all know that our time in this world is limited, and that eventually all of us will end up underneath some sheet, never to wake up. And yet it is always a surprise when it happens to someone we know. It is like walking up the stairs to your bedroom in the dark, and thinking there is one more stair than there is. Your foot falls down, through the air, and there is a sickly moment of dark surprise as you try and readjust the way you thought of things."*
>
> **—** *Lemony Snicket, Horseradish: Bitter Truths You Can't Avoid*

Chapter 1
Introduction

Since the rapid development of technology, music has been changed from "in-person" to digital thanks to radio, Internet, CDs, MP3 players, and alike. Due to improvements in technology, the science that is music is readily available for the general population. While technology has provided us with large amounts of music, ready at the click of a button, it also limits our ability on how we access it. As Marc Leman (2008) describes in "Embodied Music: Cognition and Mediation Technology," music is accessed merely by the title of the song, artist, and composer, but not by how it sounds or feels. Projects, such as the Music Genome Project by Pandora.com, aim to help expand how we listen to music by analyzing musical features. The Music Genome project (About The Music Genome Project, 2013) uses up to 450 distinct musical characteristics set by music analysts to provide a better experience for individuals so they may listen not only to specific genres of music, but music that they like; their own unique taste. However, Pandora, an online, customizable radio, does not use automated information retrieval (About The Music Genome Project, 2013).

Constructing identifiable features for music automatically remains a challenging problem. While some properties apply to particular instruments, styles, or genres, those properties may not apply to music globally. Firstly, we must understand the basis of Music Information Retrieval (MIR). A music signal in raw form (time domain) depicts a rather complex domain. Extended information can be extracted by transforming the signal from the time domain to the frequency or time/frequency domain by using the Fast Fourier Transform (FFT) or Short Time Fourier Transform (STFT). These are some of the most common algorithms to transform signals from one domain to the other and back. This thesis focuses on the FFT and frequency domain. Signals are generally transformed for a different level of analysis on data (*i.e.* going from studying the signal in the time domain to the frequency domain). Further data transformation is achieved by using convolution and filtering. Convolution in Digital Signal Processing (DSP) involves a machine which applies some function or impulse response to an input signal to produce an output signal. Also note that convolution in the time domain maps to multiplication in the frequency domain. Convolution directly relates to filtering which attempts to reduce or

eliminate specific frequencies or ranges of frequencies from the original signal. This reduces noise and complexity of the signal and simplifies the analysis of properties like timbre (harmonic structure or frequencies present).

Timbre can also be referred to as how music sounds or color. The definition can be subjective, as there is not a definite way on how timbre should be represented. One way to model timbre is by using the spectral envelope or the best fit line for all harmonic/inharmonic structure of a signal (spectral structure). A common approach is generating the power spectrum (squared magnitude), i.e. the strengths of frequencies present in a signal. Different ways of creating the spectral envelope can be seen in the following figure:

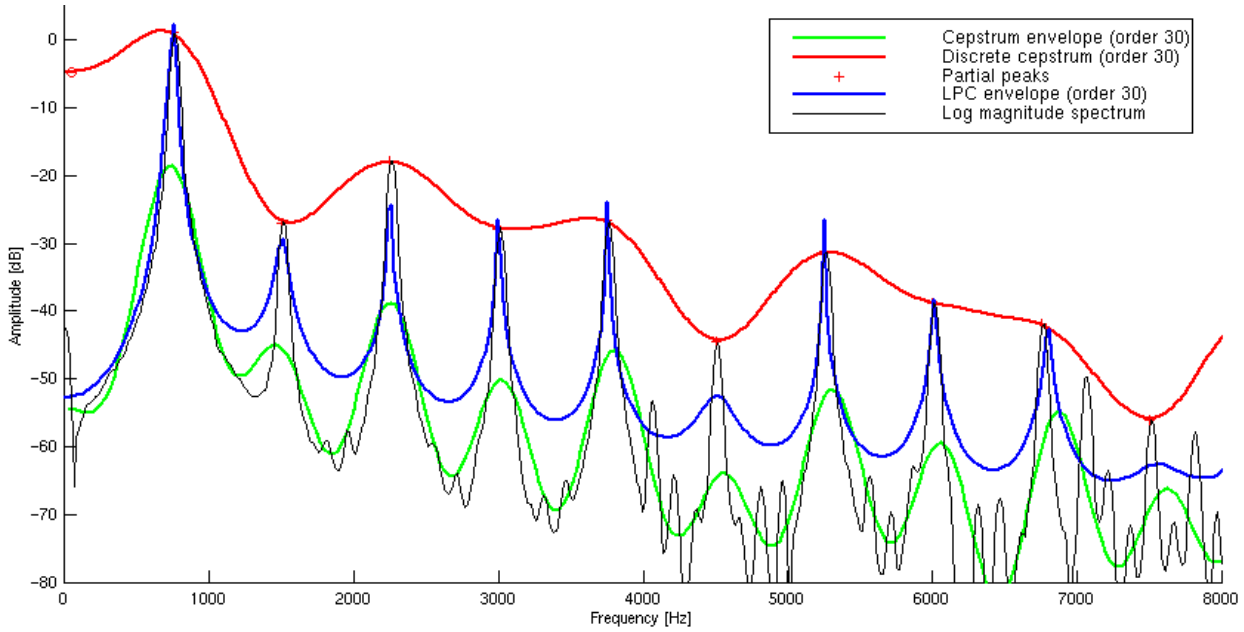

Figure 1.1 Common methods of creating spectral envelopes. Reused with written permission (Schwarz & Rodet, 1999).

Cepstrum (squared magnitude of the Fourier transform of the logarithm of the spectrum), discrete cepstrum, and LPC (Linear Predictive Coding) envelopes are graphed versus the original spectrum of an arbitrary signal in Figure 1.1. The major downfall of the discrete cepstrum envelope is that it is not resilient to noise. It correctly links all of the peaks of the partials together; however, it gives no notion of the residual noise between partials (Schwarz & Rodet, 1999). The cepstrum and LPC envelope apply well to signals with noise, although both do not

accurately link peaks of each partial together. The LPC envelope can also be too smooth if too low of an order is used (Schwarz & Rodet, 1999).

This thesis presents a new method for modeling the spectral structure of a musical signal. Uniform MAx Gaussian Envelope (UMAGE) uses a mixture of uniform (same width and variance) Gaussian distributions to tightly model the structure of signals in the frequency domain. This links all peaks of the spectrum together, but also includes frequencies between partials. This is beneficial if the signal contains inharmonic voices/instruments since the envelopes shown in Figure 1.1 rely heavily on periodicity (*i.e*. all partials are harmonic). Gaussian blurring can also be used as a post-processing step on UMAGE in order to reduce noise and rapidly oscillating frequencies. Simply, Gaussian blurring has a smoothing effect on the envelope. UMAGE also allows for high order (number of voices present) polyphonic signals, making it a robust model for representing timbre. UMAGE is shown to compete with the state of the field in instrument identification and, in some cases, pitch detection.

Chapter 2

Background

2.1 Sound of Music

The ability to construct and gather data from music is still an open problem. As for many machine problems, Music Information Retrieval (MIR) is achievable by humans but is a much more difficult task to automate and run with machines. MIR deals with feature selection and construction in the ever-increasing number of new songs in order to label and categorize music or in other words, to make music more 'browseable' by non-traditional means (Echeverri, 2011). The research field is studying the mediation of music to the end user in such a way that the technology seems transparent yet able to give the user higher capabilities in searching, playing, and experiencing music (Leman, 2008).

Instrument Family	Examples	Excitation Type	Method of Changing Pitch
String	Violin, viola, cello, double bass, guitar, banjo, mandolin	"String vibrations by plucking, hitting, or bowing strings"	"Different lengths, thickness, or tension of strings"
Woodwind (not always wooden)	Saxophone, clarinet, oboe, flute, piccolo, bassoon, English horn, bagpipes	"Blowing air across: an edge (flute); between a reed (saxophone); between two reeds (oboe)"	"Opening and closing holes along the instrument's length with fingers"
Brass	Trombone, trumpet, tuba, French horn, bugle, coronet	"The sound comes from a vibrating column of air inside a tube caused by vibrating lips of the player, who presses lips to the mouthpiece and forces air out"	Varying speed of air, vibration of lips, length of tubes, or valves
Percussion	Drums, vibraphone, marimba, tambourine, cymbals, gong, woodblock, triangle	"Sound source is a vibrating membrane or vibrating piece of solid material caused by hitting, shaking, or rubbing."	Some percussion is not pitched, but depends on the material, thickness, or tension
Keyboards	Piano, harpsichord, pipe organ, accordion	Strings or pipes.	Varying length, tension, diameter, and density

Table 2.1.1 There are many instrument types/families. Each family contains a variety of instruments, such that the method of producing sound and changing pitch differs not only by family, but by instrument as well. Adapted from (Vaseghi, 2013).

As difficult as data retrieval is from music, we may gain insight by studying properties of sound. So, what is sound? Vibrations caused by vocal chords, strings, reeds, and other means form sound waves of varying amplitude and frequency. A variety of musical instruments are described in Table 2.1.1; granted it does not cover all types of instruments, but it does represent the main instruments used in most small groups, bands, orchestras, and symphonies (Vaseghi, 2013). For further description and inner workings of the types of instruments presented in Figure 2.1.1, the interested reader is referred to (Vaseghi, 2013).

Amplitude of sound waves is measured by calculating the sound power or pressure level:

$$PL = s * \log_{10}\left(\sqrt{\frac{\sum_{n=i}^{N}\left(\frac{x_n}{\theta}\right)^2}{N}}\right) dB. \quad \textbf{2.1.1}$$

In Equation 2.1.1, s represents the segmentation size (commonly 20 ms), N represents the number of samples in the segment, and $\frac{x_n}{\theta}$ represents the n^{th} sample normalized by θ. The power level effectively measures the loudness of the sound wave. In music, loudness is represented by various levels. Loudness is commonly described as a dynamic level as seen in Table 2.1.2.

Dynamic	Relative Loudness
pp – pianissimo	Very soft
p – piano	Soft
mp – mezzo-piano	Moderately soft
mf – mezzo-forte	Moderately loud
f – forte	Loud
ff – fortissimo	Very loud

Table 2.1.2 A mapping from musical dynamics to relative loudness.

Amplitude leads to other properties of sound, particularly frequency of sound produced by musical instruments described in Table 2.1.1. The frequency of a sound can be defined as the number of oscillations from high to low pressure per second in units of Hertz (Hz) (Vaseghi, 2013). The human ear, illustrated in Figure 2.1.1, is capable of hearing frequencies ranging from roughly 20 Hz to 20 kHz (Vaseghi, 2013). Sound waves caught by the outer ear are funneled through the external auditory canal and eventually into a concentric spiral tube known as the cochlea (Thomas, 2012). The fluid in the cochlea channels vibrations to microscopic hair cells lined on the organ of Corti which resonate to particular frequencies (Thomas, 2012). This gives humans the ability to detect pitch. For further information on how the human ear processes sound, the interested reader is referred to (Fastl & Zwicker, 2007; Stuttle, 2003; Vaseghi, 2013; Chittka & Brockmann, 2005).

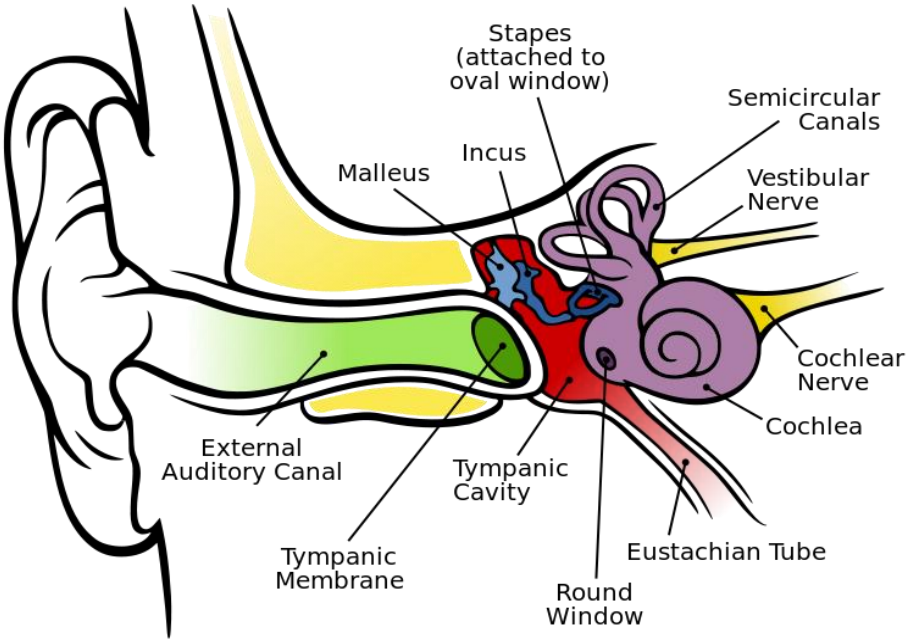

Figure 2.1.1 Inner workings of the human ear (Chittka & Brockmann, 2005) (Creative Commons Attribution).

A pitch is a series of partials (any single frequency present) perceived from a processed sound wave. Pitches are designated by their fundamental frequency ($F0$) in combination with a number of harmonic or inharmonic overtones. The fundamental frequency is generally noted to be the lowest frequency present. The structures of most pitches are periodic. This means, for most pitched sounds, harmonics of $F0$ are integer multiples of each other *i.e.* a pitch with a $F0$ of 440 Hz will have a first harmonic ($F1$) of 880 Hz, a second harmonic ($F2$) of 1320 Hz, and so on. This is false, however, for some pitched percussive and non-percussive instruments that contain inharmonics (partials that are not integer multiples of the fundamental), like the piano. An instrument that normally exhibits harmonic partials may also exhibit inharmonic partials if the instrument is out of tune. Tuning is a process where an instrument is adjusted to be in harmony with a target pitch (generally A4 or 440 Hz). Deviation from the target pitch is measured in cents by:

$$c = 1200\left(\log_2\left(\frac{f1}{f2}\right)\right), \qquad \textbf{2.1.2}$$

where $f1$ is the played pitch and $f2$ is the target pitch. This may deviate approximately plus or minus 50 cents; beyond that threshold, a pitch may not be considered out of tune, but to be the next corresponding pitch. A pitch produced by musical instruments (and even voice), as described in Table 2.1.1, is generally referred to not by fundamental frequency, but by note, accidental, and octave. Notes consist of labels from A through G with accidentals of either natural (♮), sharp (♯), or flat (♭) where the natural accidental represents a tone or whole step (A-G) and the sharp and flat accidentals represent a semi-tone or a half step above or below each whole step. An octave contains a total of twelve notes. Octaves begin on the note C♮ and contains the other following notes: C♯, D♮, D♯, E♮, F♮, F♯, G♮, G♯, A♮, A♯, and B♮. An alternative representation of a scale may include the flat accidental: C♮, D♭, D♮, E♭, E♮, F♮, G♭, G♮, A♭, A♮, B♭, B♮. Each pitch or $F0$ can be calculated by the following:

$$f = f_i * 2^{\left(\frac{i}{12}\right)}, \qquad \textbf{2.1.3}$$

where f_i is the frequency of reference (concert pitch) and 12 is the number of notes per octave. i is found by $p - p_i$ where p is the MIDI number of the pitch being calculated and p_i is the MIDI number of the concert pitch. The concert pitch is generally A4, with a frequency of 440Hz, and a MIDI number of 69. If the frequency of a pitch is known, the corresponding MIDI number can be calculated by the following:

$$p = p_i + 12 \log_2 \frac{1}{f_i}. \qquad \textbf{2.1.4}$$

The MIDI numbers for a piano with the corresponding note, octave, and frequency can be seen in Figure 2.1.2. For further information on the MIDI standard, the interested reader is referred to (Resources, 2013).

Note	Octave								
	0	1	2	3	4	5	6	7	8
C	-	24 : 32.7 Hz	36 : 65.4 Hz	48 : 130.8 Hz	60 : 261.6 Hz	72 : 523.3 Hz	84 : 1046.5 Hz	96 : 2093.0 Hz	108 : 4186.0 Hz
C#	-	25 : 34.6 Hz	37 : 69.3 Hz	49 : 138.6 Hz	61 : 277.2 Hz	73 : 554.4 Hz	85 : 1108.7 Hz	97 : 2217.5 Hz	-
D	-	26 : 36.7 Hz	38 : 73.4 Hz	50 : 146.8 Hz	62 : 293.7 Hz	74 : 5587.3 Hz	86 : 1174.7 Hz	98 : 2349.3 Hz	-
D#	-	27 : 38.9 Hz	39 : 77.8 Hz	51 : 155.6 Hz	63 : 311.1 Hz	75 : 622.3 Hz	87 : 1244.5 Hz	99 : 2489.0 Hz	-
E	-	28 : 41.2 Hz	40 : 82.4 Hz	52 : 164.8 Hz	64 : 329.6 Hz	76 : 659.3 Hz	88 : 1318.5 Hz	100 : 2637.0 Hz	-
F	-	29 : 43.7 Hz	41 : 87.3 Hz	53 : 174.6 Hz	65 : 349.2 Hz	77 : 698.5 Hz	89 : 1396.9 Hz	101 : 2793.0 Hz	-
F#	-	30 : 46.2 Hz	42 : 92.5 Hz	54 : 185.0 Hz	66 : 370.0 Hz	78 : 740.0 Hz	90 : 1480.0 Hz	102 : 2960.0 Hz	-
G	-	31 : 49.0 Hz	43 : 98.0 Hz	55 : 196.0 Hz	67 : 392.0 Hz	79 : 784.0 Hz	91 : 1568.0 Hz	103 : 3136.0 Hz	-
G#	-	32 : 51.9 Hz	44 : 103.8 Hz	56 : 207.7 Hz	68 : 415.3 Hz	80 : 830.6 Hz	92 : 1661.2 Hz	104 : 3322.4 Hz	-
A	21 : 27.5 Hz	33 : 55.0 Hz	45 : 110.0 Hz	57 : 220.0 Hz	69 : 440.0 Hz	81 : 880.0 Hz	93 : 1760.0 Hz	105 : 3520.0 Hz	-
A#	22 : 29.1 Hz	34 : 58.3 Hz	46 : 116.5 Hz	58 : 233.1 Hz	70 : 466.2 Hz	82 : 932.3 Hz	94 : 1864.7 Hz	106 : 3729.3 Hz	-
B	23 : 30.9 Hz	35 : 61.7 Hz	47 : 123.5 Hz	59 : 246.9 Hz	71 : 493.9 Hz	83 : 987.8 Hz	95 : 1975.5 Hz	107 : 3951.1 Hz	-

Figure 2.1.2 Musical notes on a piano with corresponding note name, note octave, MIDI number and frequency (separated by a :).

The representation and classification of sound can be very subjective, even when referring to a specific pitch, instrument, or genre. Psychoacoustics is the science of sound perception (Rossing, 1990). Given a set of instruments playing the same pitch, a trombone and a trumpet for example, we would be able to uniquely identify each instrument (Thomas, 2012). The representation of such characteristics is known as timbre, also known as color. Timbre is modeled using the spectra (partials) of the pitch as well as the spectral envelope which is a function that represents the collective partials. Timbre analysis is generally the key focus in most pitch detection and instrument identification.

2.2 Signal-Based Transforms

MIR systems rely heavily on the ability to transform music from the time domain to the frequency domain. Doing so allows timbre analysis and the calculation of spectral envelopes. The classical way of transforming signals is by utilizing the Discrete Fourier Transform:

$$X_k = \sum_{n=0}^{N-1} a_n * \ e^{-2\pi i n k/N}, \qquad \textbf{2.2.1}$$

where a_n are the time-domain samples of the audio waveform (Weisstein, 2013). One main problem with Equation 2.2.1 is that it runs in $O(N^2)$. A substantial speedup can be achieved by using the Fast Fourier Transform (FFT):

$$X_{k_0 j_0} = \sum_{n=0}^{N/2-1} a_{k_0} * e^{-2\pi i n k/(N/2)} + \sum_{n=0}^{N/2-1} a_{j_0} * e^{-\frac{2\pi i n k}{\left(\frac{N}{2}\right)}}, \quad \textbf{2.2.2}$$

where k_0 and j_0 represent the even and odd time-domain samples of the audio waveform respectively (Weisstein, 2013). A common notational shortcut for the complex principle root of unity may be represented as $W = e^{-2\pi i k/N}$.

The FFT breaks a Fourier series into two $\frac{N}{2}$ parts, where N is the number of the discrete Fourier coefficients. These two parts are separated into the even and odd points by a process called *decimation in frequency* (DIF) *radix-2* (meaning two groups). By splitting the DFT into parts, runtime can be decreased from $O(N^2)$ to $O\ (N \log N)$. For further derivation of the DFT and FFT algorithms, the interested reader is referred to (Cooley & Tukey, 1965; Oppenheim & Schafer, "Computation of the Discrete Fourier Transform", 2010; Oppenheim & Schafer, "The Discrete Fourier Transform", 2010; Smith S. W., "The Fast Fourier Transform", 1997).

So why are signal based transforms so important? As stated above, algorithms like the FFT allow the transformation from the time domain to the frequency domain, and given that the frequency domain is not construed by filters and other methods, the inverse can be applied to accurately reconstruct the signal back into the time domain. The transformation using an FFT can be seen in Figure 2.2.1.

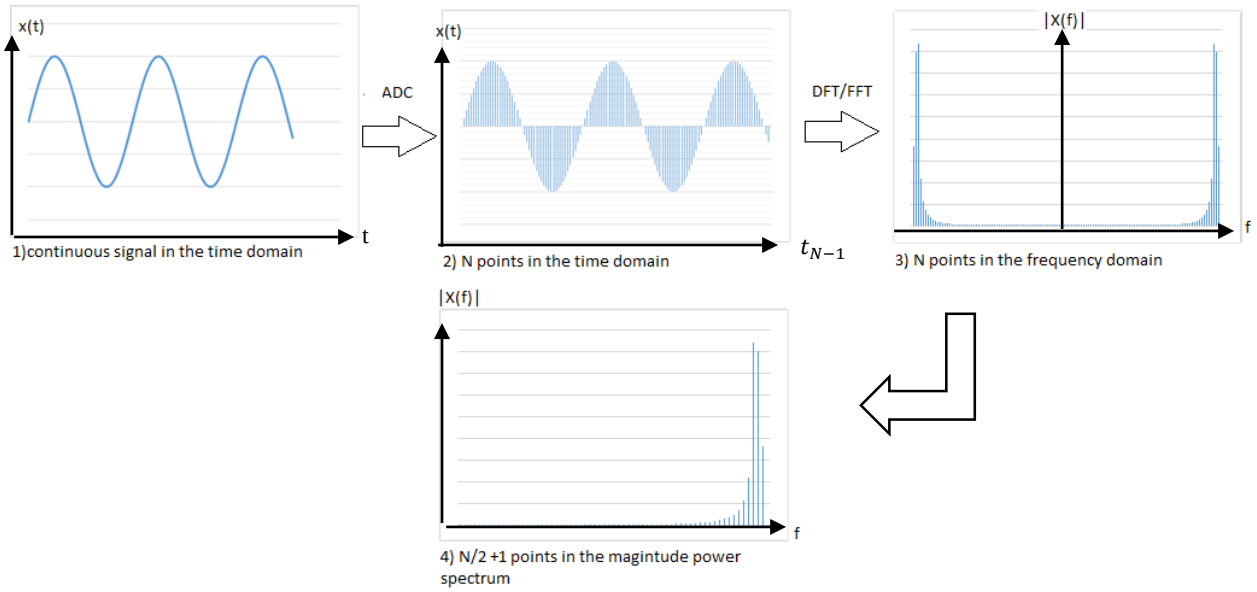

Figure 2.2.1 The process of transforming a continuous signal into a discrete signal; then into the frequency domain. Adapted from (Wang, 2009).

In order for a continuous signal to be processed digitally, an analog to digital converter (ADC) must be applied to sample the signal into *N* discrete points. This is shown in steps 1 and 2 in Figure 2.2.1. A common sample rate is 44.1 kHz or CD quality. The consequence of a set sample rate is that the highest detectable frequency is equal to $\frac{f_s}{2} - 1$ where f_s represents the sample rate in Hz; $\frac{f_s}{2}$ is commonly referred to as the *Nyquist frequency*. Conversely, the *Nyquist rate* is $2f_{max}$, where f_{max} is the highest frequency present (Oppenheim & Schafer, "Sampling of Continuous-Time Signals", 2010). Collectively, the Nyquist rate and Nyquist frequency are part of the *Nyquist-Shannon Sampling Theorem*, which states that if f_s is greater than twice that of the highest frequency present in the original signal (f_{max}), the analog signal can be perfectly reconstructed (Oppenheim & Schafer, "Sampling of Continuous-Time Signals", 2010; Wang, 2009). A Nyquist frequency less than the highest frequency present in a signal causes an error known as *aliasing*. Aliasing causes frequencies above half the sampling frequency to "fold over" and overlap with the lower frequencies (Wang, 2009). This problem does not have many solutions. It can be avoided by assuring that the sample rate is at least twice that of the highest frequency or the high frequencies above the Nyquist frequency can be filtered out by using anti-aliasing filters or low-pass filters which attenuate high frequency parts of the signal. Aliasing can mask a signal to appear as a different frequency. A frequency f that is above the Nyquist

frequency will be folded into existing lower frequencies and becomes $f^{\circ} = f_s - f$ (Wang, 2009).

2.3 Other Signal Processing Techniques

2.3.1 Onset Detection

A plethora of signal processing techniques exist. Some, like the Fourier transform, work to move signals across domains, others work as a pre and post processing step. *Onset detection* is the process of detecting the start of notes. This is particularly useful in pitch tracking. Pitch tracking can be exceptionally difficult when regarding polyphonic signals; it is not out of the norm to see accuracy average under 60% as seen in Music Information Retrieval Evaluation eXchange (MIREX) (Multiple Fundamental Frequency Estimation Tracking Results, 2010).

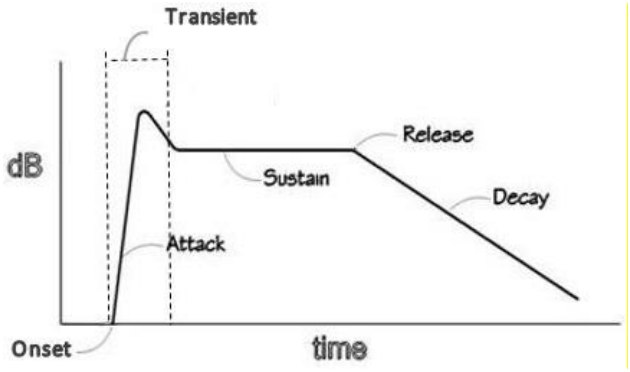

Figure 2.3.1 A simplified illustration of note structure, adapted from (Noxon, 2012).

Notes can be structurally be described by *onset, attack, transient, sustain, decay, and release* (Bello, et al., 2005):

- The onset of a note is the identified moment that a note begins.
- The attack of a note is the initial time interval during which amplitude increases.
- The transients of a note are periods where the signal changes in some "nontrivial or unpredictable way."
- The sustain of a note is defined as a time interval containing relatively constant amplitude.
- The release of a note can also be defined as a transient. The release marks the moment the excitation (playing) of the note is stopped. For example, this would mark the moment

a brass player would stop blowing air through the horn or a stringed player lifts their bow off the strings.

- The decay of a note refers to a relatively steady decrease of amplitude. The decay can also be considered a transient and usually is correlated with the reverberations after the release.

Figure 2.3.1 illustrates the relative amplitude envelope. The amplitude envelope is created by graphing decibel levels using a similar formula like Equation 2.1.1. Figure 2.3.1 labels the note structure elements of a single note. This simplistic representation gives an excellent visualization of note structure; however, in order to label note structures in real signals (especially polyphonic signals and complex musical pieces), many steps are required.

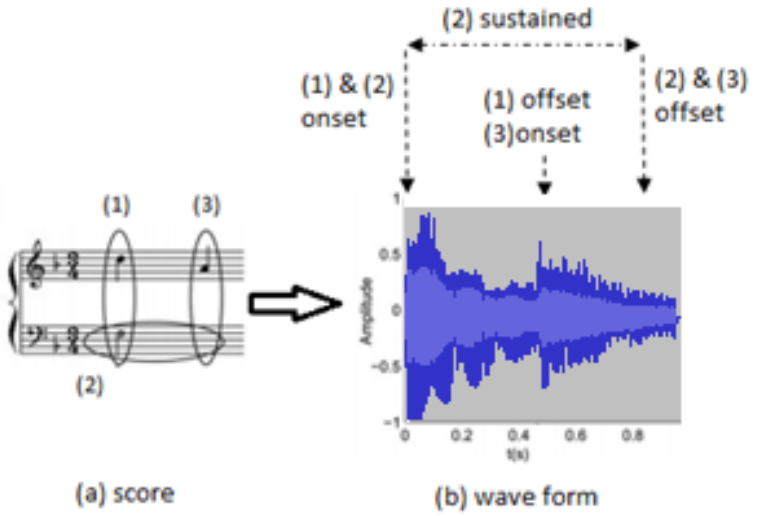

Figure 2.3.2 A small piano excerpt to show complexity of note tracking. (a): the score showing the notes present (1), (2), and (3). (b): the wave form of the score being played with labels denoting note structure.

As shown in Figure 2.3.2, the ability to label all elements of note structure becomes exceedingly difficult without additional processing. The approach above also encompasses pitch envelope tracking to assist in differentiating the overlap of a sustained note and the onset of a different note. By a visual inspection, it is impossible to deduce the structure of notes (1), (2), and (3)

from just the waveform (b). This is why most onset detection, note tracking, and pitch tracking algorithms utilize many pre and post processing steps before labeling note structure.

While the wave form (original signal) is relatively simple, it's both visually and especially algorithmically complicated to label note structure, even when the scope is limited to onset detection. The majority of onset detection algorithms deploy a similar framework including an optional pre-processing step, a reduction step, and a peak picking step (Bello, et al., 2005). The original signal may be preprocessed using various filter banks in order to reduce noise and to smooth each note. The pre-processed signal is then reduced by resampling at a lower rate (Bello, et al., 2005). This step usually generates an amplitude or relative loudness envelope to model the signal. Reduction minimizes signal complexity to increase algorithmic accuracy. The notes' structural elements become visible after the original signal is pre-processed and reduced. This allows for peak-picking algorithms to locally find each note onset. Note tracking is not only limited to onset detection, but also includes other note structure elements like attack, transients, release, and decay for music transcription. There are numerous onset detection methods, some of which perform best with certain types of sounds (*i.e*. pitched percussive, pitched non-percussive, and non-pitched percussive). Others are rather robust and work well for all types of instrumentation, such as *surprise onset detection*. Surprise onset detection is based on how an observer could build a model or become familiar with certain kinds of signals, such that the observer would be able to make predictions about the signal as it unfolds in time (Bello, et al., 2005). "The observer will be relatively surprised at the onset of a note because of its uncertainty about when and what type of event will occur next" (Bello, et al., 2005). On the other hand, if the observer has seen these onsets frequently, the surprise will be localized to the transients of the note (Bello, et al., 2005). The surprise onset can be described as the negative log likelihood of an event to happen given all previously observed data as shown in the following:

$$S(N) \equiv S\big(x(n)\big) := -\log p(x(n)|\,\{x(j) : j < n\}), \qquad \textbf{2.3.1}$$

where x is a discrete signal with N samples, $x(n)$ is a single sample, and $x(j) : j < n$ is all previously seen samples (Bello, et al., 2005). This model is also the basis for use in independent component analysis (ICA); for further information, the interested reader is referred to the original

paper that introduced the surprise model (Abdallah & Plumbley, 2003). For further overview of onset detection, including reduction, detection, and peak picking methods, the interested reader is referred to (Bello, et al., 2005).

2.3.2 Convolution and Filtering

Many major operations in digital signal processing (DSP) involve the mathematical term *convolution*. For simplicity, the scope of this section is limited to discrete convolution, although convolution may also be applied to continuous signals Convolution, in terms of DSP, is an operation in which an impulse response $h[n]$ is applied to input signal $x[n]$ to produce output signal $y[n]$. Convolution also makes use of *linear systems*, which are any processes that receive an input signal and produce an output signal (Smith S. W., "Linear Systems", 1997). Linear systems may add additional operations in addition to applying the impulse response. Any impulse response may be found from the originating delta function $\delta[n]$, where the value of $n = 0$ is 1 and all other n are 0. Many impulse responses can be described as a shifted and scaled delta function (Smith S. W., "Convolution", 1997). Once the impulse response is known, the output can be found by the sum of the scaled and shifted impulse responses from the input signal (Smith S. W., "Convolution", 1997). The input signal $x[n]$ is convolved with the linear system's impulse response $h[n]$ to produce output signal $y[n]$. Convolution can be viewed as the sum of weighted outputs. Please note that the convolution process is denoted by $*$; this should not be confused with the usage of $*$ in most programming languages, where it is used as the multiplication symbol. Likewise, the impulse response is also known by different names, depending on the application (Smith S. W., "Convolution", 1997). In regard to a *filter* system, the impulse response may be referred to as the *filter kernel*, *convolution kernel*, or *kernel* (Smith S. W., "Convolution", 1997). Filters are used in signal processing in order to mitigate noise caused by anomalies in recording, synthesizing, or just to reduce the presence of specific ranges of frequencies. Filters can also help reduce aliasing, as discussed in Section 2.2, or reduce the signal into a more simplified representation. A number of filters exists, each affecting signals differently. A few of the most commonly used filters are *low-pass filters, high-pass filters,* and *band-pass filters*. Low-pass filters attenuate (reduce amplitude to zero) high frequencies above a certain threshold, and "pass" frequencies below the threshold. This particular filter is useful when sampling frequency is low (prior to A/D conversion or digitization), as it can eliminate

frequencies above the Nyquist frequency. High-pass filters are just the opposite. This filter passes frequencies above a specific threshold and attenuates those frequencies below. Band-pass filters are actually a combination low and high-pass filters. Band-pass filters pass frequencies within a range and attenuate those outside of said range. Now, there are many variations of these filters, all with different filter kernels and different uses in DSP. One method that is used in this thesis is called *discrete Gaussian blurring*.

Figure 2.3.3 Effects of a Gaussian blur filter on a digital image (Patin, 2011).

Gaussian blurring is usually applied as an image filter; the effects of which can be seen in Figure 2.3.3. In image processing, a matrix of a specified size with weight coefficients blends the colors of each pixel by applying the matrix to the neighborhood of pixels and dividing by the sum of the coefficients. Coefficients may be assigned at the user's discretion to achieve desired effects.

Index N	Coefficients	Sum of coefficients = 2^N
0	1	1
1	1 1	2
2	1 2 1	4
3	1 3 3 1	8
4	1 4 6 4 1	16
5	1 5 10 10 5 1	32
6	1 6 15 20 15 6 1	64
7	1 7 21 35 35 21 7 1	128
8	1 8 28 56 70 56 35 8 1	256
9	1 9 36 84 126 126 84 36 9 1	512
10	1 10 45 120 210 252 210 120 45 10 1	1024
11	1 11 55 165 330 462 462 330 165 55 11 1	2048

Table 2.3.1 A table of Gaussian filter coefficients (Patin, 2011).

However, it is better practice to use the *binomial* coefficients shown in Table 2.3.1, which is Pascal's triangle enumerated to a depth of 11. In image processing, each vector shown in Table 2.3.1, is copied into two matrices (one of which is a single column and as many rows as coefficients and one of which is a single row with as many columns as coefficients) and multiplied to create the Gaussian filter kernel. The kernel matrix's final size is $(N + 1) \times (N + 1)$ and the sum of coefficients would be equal to $2\text{^}N$. The final effect of this filter (the blurriness) will be affected by the size of the kernel used as seen in Figure 2.3.4.

Figure 2.3.4 How different sized kernels affect the Gaussian blur filter, adapted from (Song, 2013) (Creative Commons Attribution).

Notice that in the original image, the colors of the hot air balloon are distinct and clear by each panel. After a Gaussian blur filter with a radius of 3 pixels is applied, the picture seems to be just out of focus. When the larger kernel is used, the third image resembles a tie-dye t-shirt. The panels of the balloon are no longer distinguishable, and the neighboring colors are beginning to blend together. Figure 2.3.4 perfectly demonstrates how filter kernel size can affect a filter's final output. *Filter banks* may also be used in order to increase the intensity of the blur. In other words, the filter may be applied several times to achieve the desired effect; however, there is a point of diminishing returns where each neighborhood of pixels becomes well blended and the blur effect is miniscule. Gaussian blur filters are also applicable to DSP with the same effect, though in a different medium. As noted in Chapter 1, if a signal is transformed from the time domain to the frequency domain via an FFT (or other transform), weights can be applied, and the inverse FFT can be used to transform the signal back to the time domain; however, some distortion may occur. In this case, the kernel is not a matrix, it is a single vector (the same sums in image filters are used here). The intermediate representation before the signal is transformed

back into the time domain is more useful. Blurring the frequency domain reduces and potentially eliminates artifacts, noise, and complexity (generalizes the frequency domain).

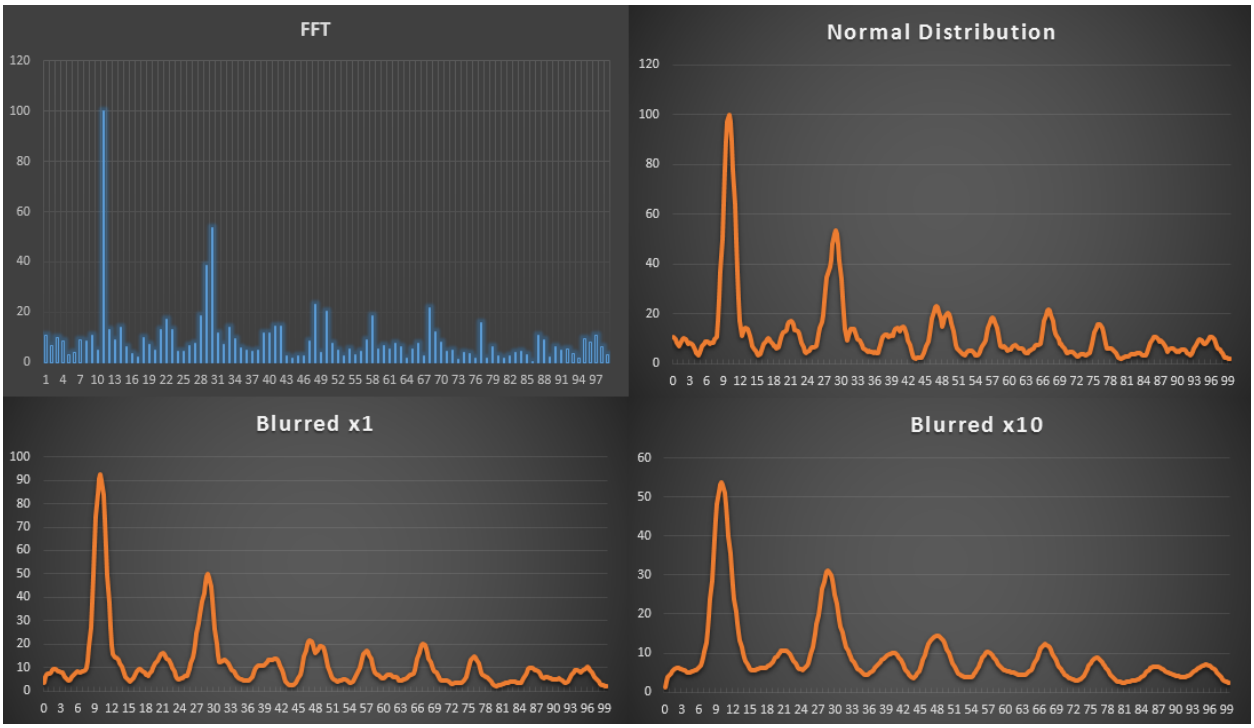

Figure 2.3.5 Gaussian blurring as seen in DSP, where each x-axis represents frequency bins of the FFT, and each y-axis represents normalized magnitude.

Figure 2.3.5 shows four graphs of a polyphonic signal containing two brass instruments, a trombone and tuba, and one string instrument, a cello, playing notes whose MIDI numbers are 45, 44, and 63 respectively. The top left graph shows the power spectrum of an FFT of length 4096 windowed to include 100 frequency bins (approximately 1076.66 Hz); the top right graph shows the normal distribution (calculation of which will be discussed in 0) of this histogram. A Gaussian blur filter, with the kernel $\{ 1, 6, 15, 20, 15, 6, 1 \}$ and sum of 64, is applied to the normal distribution a single time (bottom left graph) and ten times (bottom right graph). As seen in the normal distribution, the graph follows the original FFT very closely; however, when teaching a computer to learn a specific concept, like how to recognize musical instruments, a model that is too closely fit is not very robust and can perform poorly given a high number of examples. Even after a single pass, the distribution begins to smooth out, and after ten passes, a fairly uniform representation of the frequency components is left.

2.4 Related Work

Despite a few decades of research, polyphonic instrument identification and pitch detection are still in their infancy. Namely, a widely accepted model or solution does not exist. Even when datasets are small and contain five or less instruments, current research only achieves 60% accuracy or less on average in instrument identification and pitch detection (Donnelly, 2012). Common approaches include Gaussian Mixture Models (GMM). "A GMM estimates a probability density as the weighted sum of M simpler Gaussian densities, called components or states of the mixture

$$p(F_t) = \sum_{m=1}^{M} \pi_m N(F_t, \mu_m, \Gamma_m) \qquad \textbf{2.4.1}$$

where F_t is the feature vector observed at time t, N is a Gaussian PDF with mean μ_m, covariance matrix Γ_m, and π_m is a mixture coefficient" (Aucouturier, Pachet, & Sandler, 2005). Other models utilize:

- Principle Component Analysis (PCA): a method that does not use output information (unsupervised) and maximizes variance. "The sample, after projection on to principle component w_1, is most spread out so that the difference between the sample points becomes most apparent" (Alpaydin E. , 2010). For further information on PCA, the interested reader is referred to (Alpaydin E. , 2010; Smith L. I., 2002).
- Non-Negative Matrix Factorization (NMF): may be defined as the following problem: "given a non-negative matrix V, find non-negative matrix factors W and H such that $V \approx WH$" (Lee & Seung, 2001). Many different algorithms exist for solving this problem. One used in DSP can be found in (Cont, Dubnov, & Wessel, 2007).
- Spectral Clustering: a technique that makes use of the spectrum of the similarity matrix of data to cluster in fewer dimensions (Donnelly, 2012; Von Luxburg, 2007).
- Hidden Markov Models (HMM): a pattern recognition theory which models the probability density of a sequence of observations or states (Aucouturier & Sandler, 2001; Rabiner, 1989).

The scope of this thesis, however, is limited to the spectral envelope or the representation of timbre. Many of the above methods are complex feature extraction/construction and

classification techniques. Timbre is the basis of most DSP research. In Chapter 1, Figure 1.1 shows common spectral envelope approaches, and the strengths and weaknesses of the cepstrum envelope, discrete cepstrum envelope, and LPC envelope are discussed. An in-depth description of the LPC envelope may be found in (Aucouturier & Sandler, 2001). Briefly summarized, LPC uses autocorrelation to estimate $p + 1$ values then generates p filter coefficients from a set of linear equations (Aucouturier & Sandler, 2001). These filter coefficients make up an all-pole filter of order p, where the filter is one over a polynomial of order p. Applying the filter to the spectrum produces the LPC envelope. The cepstrum approach applies a low-pass filter based on to the power spectrum, where high order cepstrum envelopes follow the spectrum more accurately versus low order envelopes (*i.e*. the smoothing effect is reduced in higher order envelopes) (Schwarz & Rodet, 1999). The discrete cepstrum envelope uses a select number of points from the log scaled power spectrum (Schwarz & Rodet, 1999). The points are generally chosen using peak picking or additive synthesis which produce point representing the magnitude of each partial.

2.5 Machine Learning

Machine learning is applicable to many domains and problems typical algorithms cannot solve or solve well. For example, machine learning can be used to teach a computer system how to predict the weather outlook (rainy, sunny, hot, cold, etc.) or how to tell what emails are spam and which emails are relevant. One type of learning is supervised learning. Supervised learning problems aim to learn the mapping from the given input and output. Classifiers include Radial Basis Function (RBF) networks, logistic classifiers, support vector machines, and k-nearest neighbor (k-nn, Instance Based Learning (IBL, IBk)). k-nn classifier labels input based on the class having the most examples among the k neighbors of the input (Alpaydin E. , 2010). K-nn is a lazy learner such that the model is not generated when given training data. Instead, lazy learning algorithms postpone computations until they are given a test instance. RBF networks are similar to k-nn because it uses local representation to model the data *i.e*. normal Gaussian units are used to group data into local groups. Logistic classifiers are a form of regression and can be defined as a sigmoid function $P(C_1|x) = sigmoid(w^T x + w_0) = \frac{1}{1+\exp[-(w^T x+w_0)0}$, where C_1 is the class label. SVMs are also a form of regression analysis; however, it makes use of kernels to map a non-linearly separable space to a linearly separable space. The use of the soft margin

method and slack variables help reduce misclassification by maximizing the distance between separated examples.

Many measurements of performance for classifiers and other systems exist in machine learning. This thesis focuses on precision, accuracy, recall, and F-measure. Given *true positives TP* (number of correctly classified positive examples), true negatives *TN* (number of correctly classified negative examples), *false negatives FP* (number of positive examples classified as negative), and *false positives FP* (number of negative examples classified as positive), accuracy is defined as $\frac{TP+TN}{TP+TN+FP+FN}$ or the total number of correctly classified examples over the total number of classified examples. However, accuracy can be misleading if the number of true positives and true negatives are imbalanced. In order to better measure a classifier's performance, precision and recall are used to calculate F-measure. *Precision* is defined as $\frac{TP}{TP+FP}$ or the proportion of positively classified examples that are actually true positives. *Recall*, also known as sensitivity, is defined as $\frac{TP}{TP+FN}$ or the rate of truly positive examples. The *F-measure* can be defined as $2 * \frac{precision \times recall}{precision+recall}$ or the harmonic mean of precision and recall. F-measure gives a better measure of performance since there is often a tradeoff between recall and precision when data suffer from the class imbalance problem (high number of one class over another).

Chapter 3

Proposed Model

3.1 Data

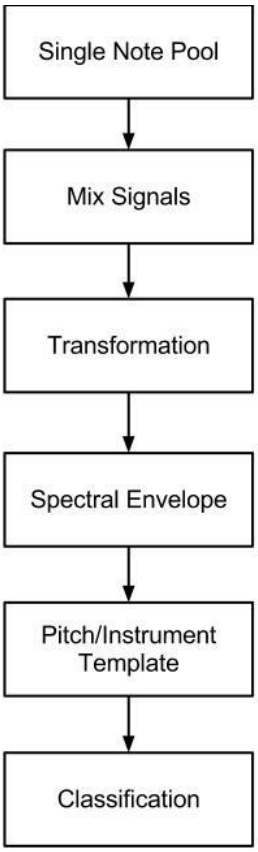

Figure 3.1.1 The overall algorithm presented in Chapter 1, from data collection and creation to transforming signals to the frequency domain and generating pitch and instrument templates.

The overall data flow of the proposed model can be seen in Figure 3.1.1. Instrument samples used in this thesis are from the *Philharmonia Orchestra's* "The Sound Exchange" (Philharmonia Orchestra, n.d.). This database includes samples from 19 instruments plus percussion with a large number of different dynamics, articulations, and length. Each sample may be downloaded in MP3 format. While this thesis uses audio in the WAV format, a simple conversion is done using the NAudio package, an open source digital signal processing library written in C# (Heath, 2013). Each sample contains a single pitch with minimal dead space (*i.e.* little to no silence). This made it a perfect dataset to use for mixing into polyphonic signals. All

samples were performed by human players; in other words, samples were not synthesized with equipment such as a Mellotron. Real samples are not the most ideal experimental setting since there may be variance between players and their styles; however, this provides a real world setting in which to test the model presented in this thesis. Another free to use, real music sample database was also considered from the University of Iowa Electronic Music Studios (UIEMS) (Fritts, 2011). The UIEMS database contains chromatic scales of over 20 instruments with different dynamic levels. A considerable amount of preprocessing was done in order to separate the chromatic scales into individual note samples using onset detection as described in 2.3.1. However, in the final stages of processing the database, inconsistencies were found in scale labels and scale performance. These errors were reported to the director of the Electronic Music Studios. Due to time constraints, the *Philharmonia* database was used in its place of UIEMS.

3.2 Signal Mixing and Domain Transformation

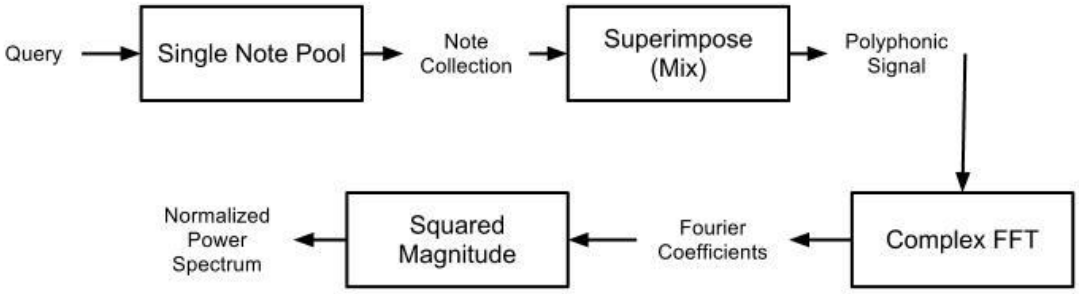

Figure 3.2.1 The process of creating polyphonic signals and their corresponding normalized power spectrum.

In this phase, which can be seen in Figure 3.2.1, polyphonic signals are created and transformed into the frequency domain. The collection of single notes include many of the normal instruments seen in symphonies from the *Philharmonia* database, excluding percussion. A query or settings file restricts the search space by limiting the pitches, dynamics, lengths, styles, voices (instruments), number of voices, and number of polyphonic signals to be created. The instance space must be reduced in interest of computational complexity. Without this

representation bias, the number of polyphonic signals possible is approximately $8.7e^{+10}$ (permutations of 14) when only considering just the instruments. This is increased even further by using all notes possible for each instrument. For this thesis, the dataset is limited to approximately 200 thousand polyphonic signals (80 batches of 2500 polyphonic signals), which is a mere fraction of the exhaustive dataset. Further information on how these are organized will be discussed in Chapter 4. For each set of single notes collected (via search query), a handful of notes are randomly selected to be mixed together, such that each note is the same length, and the number of notes is equal to the predefined number of voices in the polyphonic signal. Notes are mixed by superimposing each note on the other, in other words, adding the signals together, using the NAudio package. The polyphonic signal is then converted to an array of complex numbers, where the imaginary value is initialized to 0. The complex FFT is then calculated by using the DSP package from Exocortex (Houton, 2003). The power spectrum is generated from the FFT coefficients by taking the squared magnitude of each bin. In order to reduce large computations and data size, the range of the power spectrum is scaled to 100. This normalized power spectrum represents the strength of frequencies present in the signal.

3.3 Spectral Envelope

A spectral envelope is a function that best models all harmonic or inharmonic partials in a signal's frequency domain. As noted in Section 2.4, there are a variety of ways to generate the spectral envelope, not all of which are able to model the harmonic or inharmonic structure accurately yet still be resilient against noise and rapid changing amplitude in the frequency domain. Some measurements of goodness or properties of a spectral envelope are 1.) The envelope wraps tightly around harmonic/inharmonic structure, 2.) The envelope is relatively smooth, such that it does not oscillate erratically and still gives a good distribution of the harmonic/inharmonic structure, and 3.) The envelope adapts well to fast spectrum changes (Schwarz & Rodet, 1999).

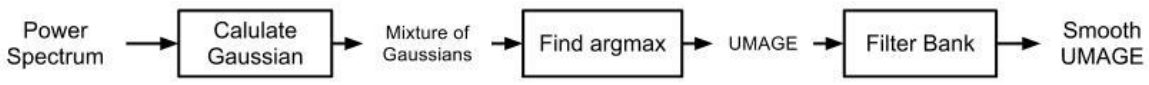

Figure 3.3.1 A flow chart showing the process of creating the smooth Uniform MAx Gaussian Envelope.

This thesis presents a new algorithm called Uniform MAx Gaussian Envelope (UMAGE). This phase in my methodology can be seen in Figure 3.3.1. The basis of this algorithm utilizes a mixture of normal Gaussian distributions of each frequency bin of the power spectrum created by taking the magnitude of a complex FFT. A single Gaussian distribution from the mixture is calculated using the following equation:

$$d\left(x_{i=\beta-W}^{\beta+W};\ \alpha,\beta,\sigma\right) = \alpha \times \left(e^{\frac{-(x_i-\beta)^2}{2\sigma^2}}\right), \quad \textbf{3.3.1}$$

where β is the frequency bin for the Gaussian to center over, α is the amplitude of the frequency bin β, σ is the given variance (affects the width of the distribution, values less than one result in a thin curve and values greater than one flatten the curve), W is the window size, and x is a vector where values of x are bounded by the window size W, such that the initial value of x equals $\beta - W$, and the upper bound of x equals $\beta + W$:
$\langle x_{i=\beta-W}, x_{i+step,}, x_{i+step\times 2,}\ \cdots, x_{i=\beta+W}\rangle$. This formula is adapted to apply to every bin in the spectrum by:

$$d_{spectrum}\left(x_i^{|spectrum|};\ \alpha,\sigma\right) \quad \textbf{3.3.2}$$
$$= \bigcup_{\beta_{b=0}}^{|spectrum|} d\left(x_{i=\beta_b-W}^{\beta_b+W};\ \alpha,\beta_b,\sigma\right) = \alpha \times \left(e^{\frac{-(x_i-\beta_b)^2}{2\sigma^2}}\right),$$

where $d_{spectrum}\left(x_{b=0}^{|spectrum|};\ \alpha,\sigma\right)$ is the union of the Gaussian distribution of each bin from the spectrum. Also note that x now has an upper bound that is the size of the spectrum, and a lower bound of 0.

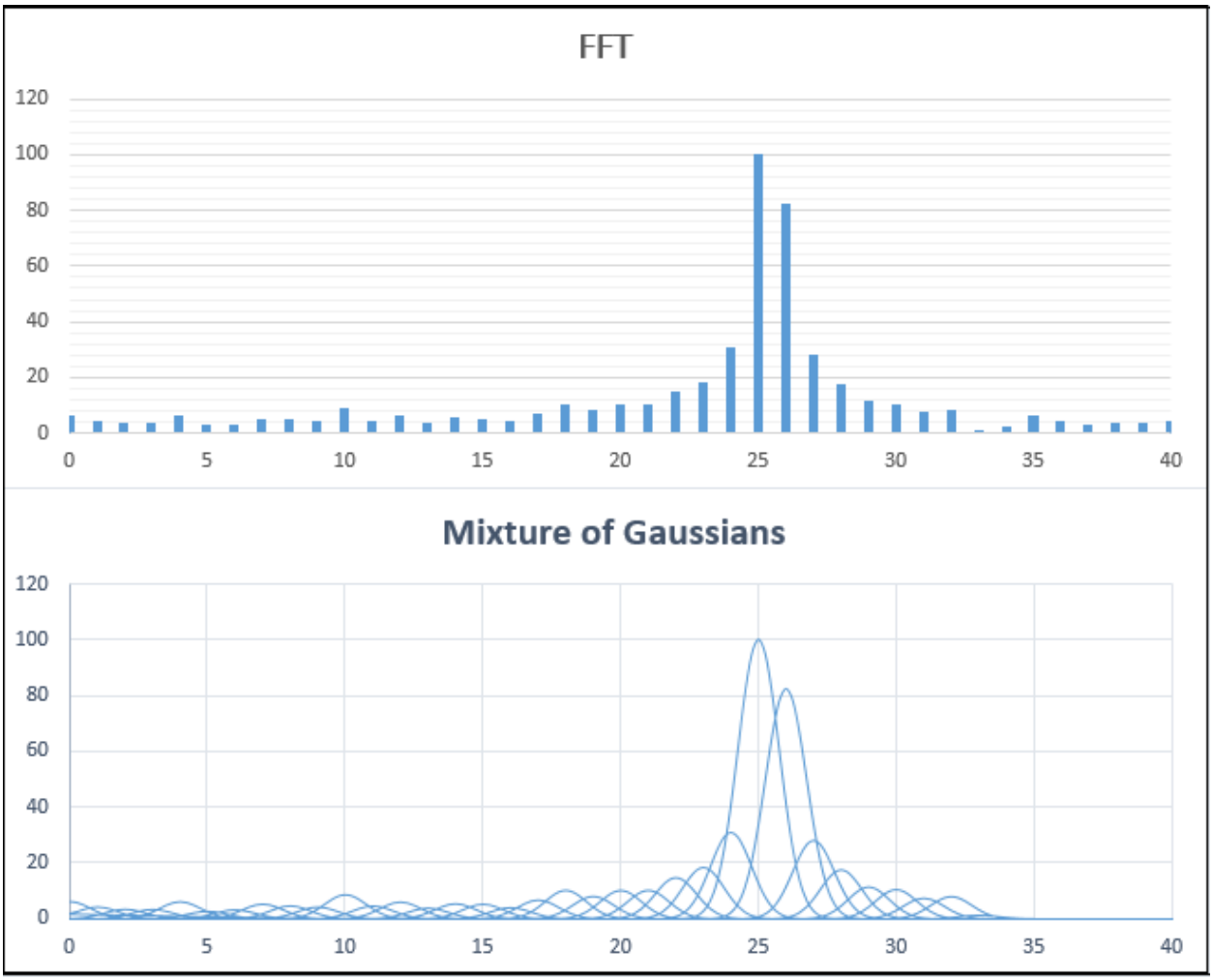

Figure 3.3.2 Power spectrum with normalized amplitude, restricted to 40 frequency bins (top) and the corresponding mixture of Gaussians as calculated in Equation 3.3.2 (bottom).

Figure 3.3.2 shows the power spectrum (restricted to 40 bins) of a trombone playing MIDI note number 41 and the corresponding result of applying Equation 3.3.2 to this spectrum up to 33 frequency bins. The mixture of Gaussians shows the intermediate representation of UMAGE, where $\sigma = 1.25$, $W = 10$, and a step of $.25$ used for values of x. Each Gaussian centered over individual bins allows uniform representation of all values in the spectrum. In its present condition, the mixture overlaps significantly; however, an envelope can be made out by following the tops of each curve. As such, the following equation is used to simplify the mixture:

$$UMAGE_{\ spectrum} = \underset{x}{\operatorname{argmax}}\, d_{spectrum}\left(x_{b=0}^{|spectrum|};\ \alpha, \sigma\right), \qquad \textbf{3.3.3}$$

This additional formulation develops the mixture of Gaussians into an envelope to represent the harmonic structure of the signal shown in Figure 3.3.3.

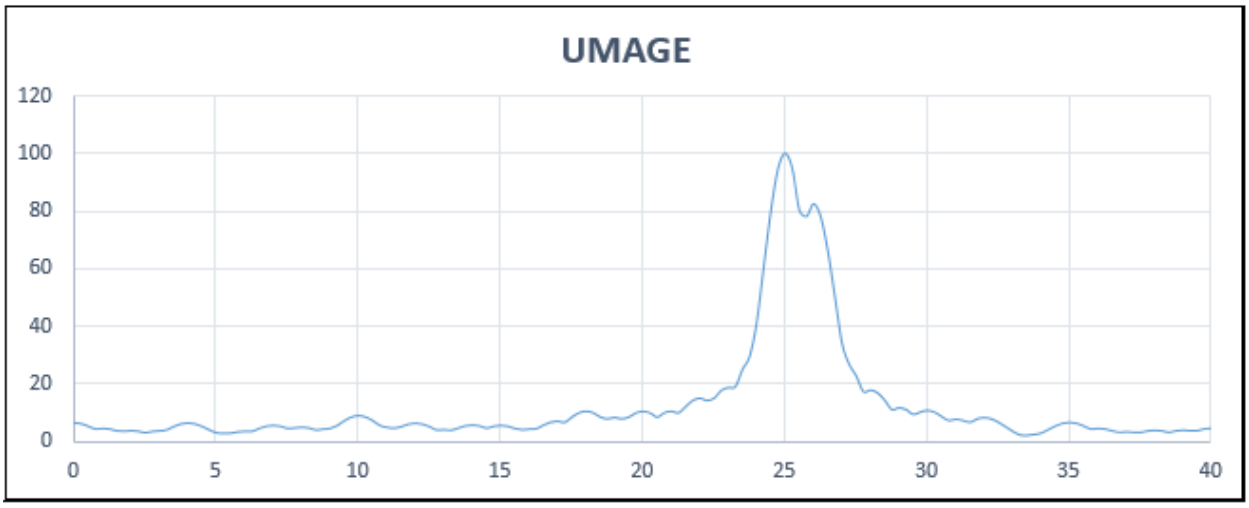

Figure 3.3.3 An example of a spectral envelope created my UMAGE.

The spectral envelope shown in Figure 3.3.3 exhibits the properties of goodness discussed earlier. The envelope is able to wrap tightly around the harmonic structure to capture the strengths of frequencies present and still be relatively smooth. This smoothness is furthered by applying Gaussian blur filters as shown in Figure 2.3.5. This form of a low-pass filter smooth's the rapid response to amplitude change in respect to neighboring frequencies so as to not lose the structure of partials entirely. The signal used in the above examples is relatively simple and does not exhibit this property entirely; however, sporadic frequency change may be seen in the following figure:

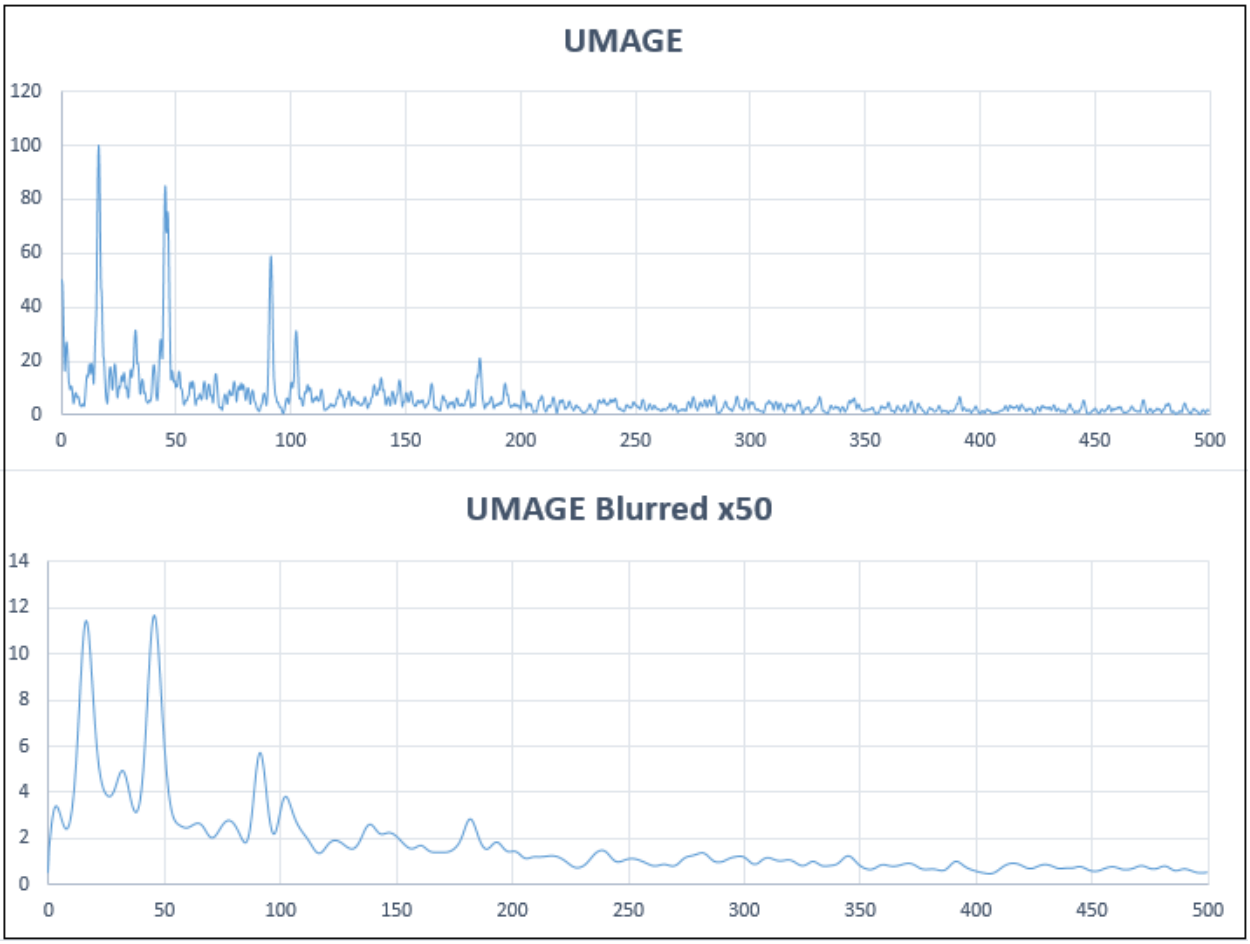

Figure 3.3.4 This is the UMAGE (top) and the UMAGE blurred 50 times with a Gaussian filter (top) of a polyphonic signal composed of seven voices.

Figure 3.3.4 shows the UMAGE of a seven voice polyphonic signal. Without blurring, the envelope fails one measure of goodness, namely it oscillates rapidly where amplitude of frequency bins change frequently. This is mitigated by utilizing the Gaussian blur filter. A filter, with the kernel $\{ 1, 6, 15, 20, 15, 6, 1 \}$ and sum of 64, was applied to the UMAGE of the complex signal to reduce the oscillation and improve the smoothness of the envelope while retaining the harmonic structure of the signal. As shown in the bottom graph of Figure 3.3.4, clarity of the UMAGE is increased greatly. Amplitude is reduced by almost 88%; however, the very little representational power is lost (*i.e.* the second tallest peak in the original UMAGE is just barely now the tallest peak) since it is reduced proportionally across the entire spectrum. UMAGE shows robustness in the ability to model simple and complex signals well when

measured using properties presented by (Schwarz & Rodet, 1999). Further performance measures are evaluated in later chapters.

3.4 Pitch and Instrument Template

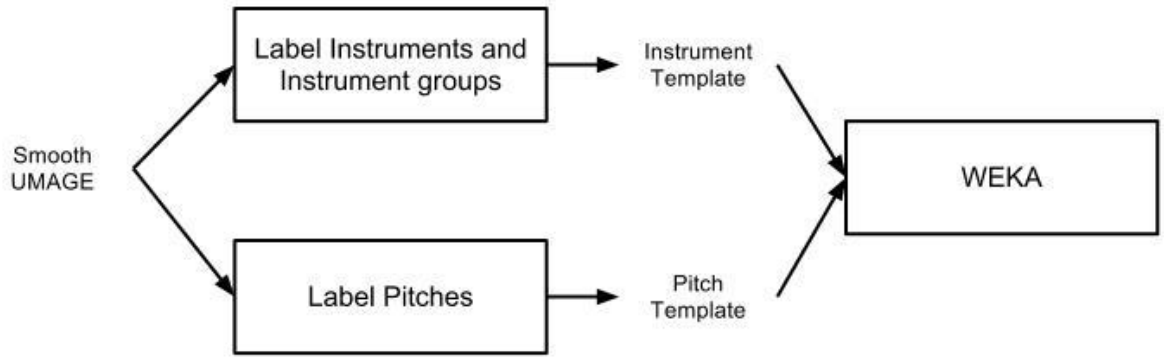

Figure 3.4.1 A flow chart depicting the creation of instrument and pitch templates used for classification.

Pitch and instrument templates are regularly used in DSP for modeling timbre as seen in (Bay & Beauchamp, 2006; Benetos & Dixon, 2010). In my algorithm, this phase, as seen in Figure 3.4.1, calculates the instrument and pitch templates separately from the same smooth UMAGE. This is done to reduce the complexity of the concepts to learn using Waikato Environment for Knowledge Analysis (WEKA). The size of the smooth UMAGE is equal to half of the FFT size minus one, multiplied by the step size used in the original calculation of the UMAGE. Using each value produces too many features and hinders classification performance. The curse of dimensionality is reduced by separating the UMAGE into windows of a predetermined size (8 in case of this thesis) and integrating to find the area under the curve for each window, which is also proposed in (Donnelly, 2012). These are then used as the features in the template. This processes is identical to both the pitch and the instrument template. The difference is how each instance is identified. For the instrument template, a set of 22 identifying features are added; one for each group of instruments present (strings, brass, and woodwinds), and one for each instrument present (19 total). For the pitch template, 88 identifying features are

added; one for each pitch present. Boolean values are used to indicate whether a particular pitch, instrument, or group of instruments is present. After each template is created, WEKA, an open source machine learning tool for Java, is used for classification, which is described in Chapter 4.

Chapter 4

Experiment

One of the greatest challenges is not only representing timbre of signals, but also the collection of signals themselves. The space of possible polyphonic signals grows exponentially as more voices (instruments or apparatuses used to create sound) are available, as well as possible pitches those instruments are able to create. The dataset is restricted the data to 80 batches, each with different limitations on the instruments possible, number of voices (instruments) in a signal, and the possible notes. Table 8.1 and Table 8.2 serve as a reference to the abbreviations, tags, and codes used in Table 8.3 which represents the contents of each batch. Originally, three sets of 5000 polyphonic signals were used for each batch; however, due to high runtime, the size of each set was reduced by half to 2500 polyphonic signals without any degradation in performance. Three sets of 2500 were randomly generated apriori as described in Section 3.2.

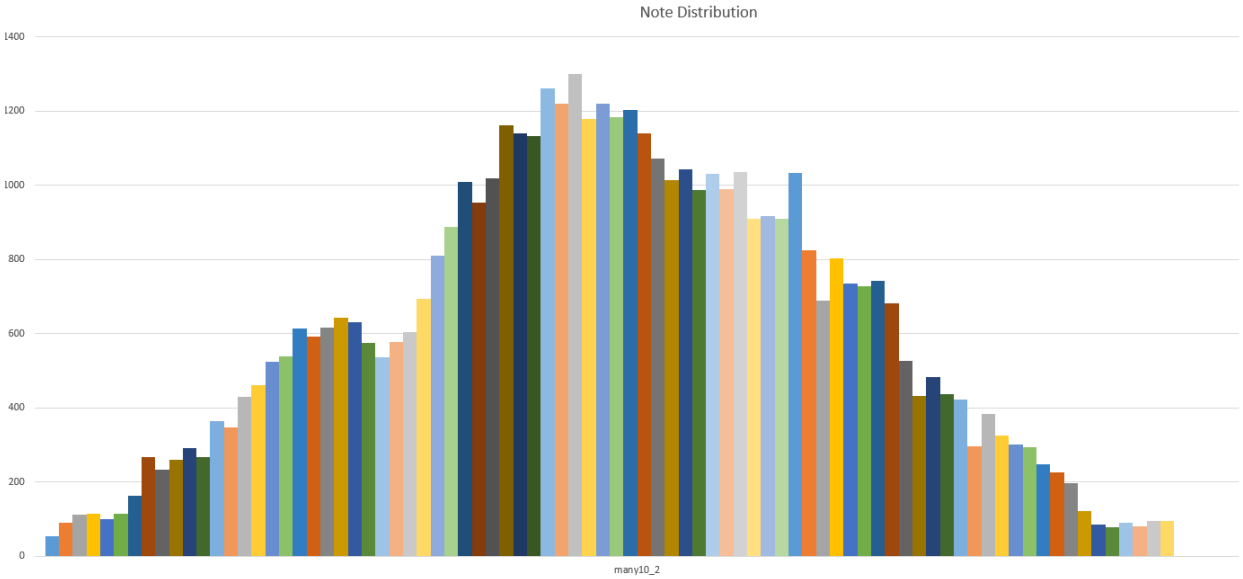

Figure 4.1 The distribution of MIDI note numbers. The x-axis denotes the MIDI numbers ranging from 21 up to 108, where each column represents the number of signals that contain that MIDI number.

Instruments contained in each batch were distributed evenly; however due to the nature and uniqueness of the instruments, some notes were represented more than others as shown by the graph in Figure 4.1. Notice that the shape of the distribution resembles a bell curve. It is tapered

towards the ends of the MIDI numbers represented (21-108) due to the fact that very few instruments can produce extremely high and/or extremely low pitches. The median is approximately center in the graph and represents the overlap of pitches that many instruments can produce. Figure 4.1 is representative of the note distributions for all batches (with respect to the notes contained in the batch).

Parameters used for UMAGE and feature extraction were set as constant throughout the experimentation process. Parameters were evaluated based on recall as to get a better true positive rate in final classification as shown in the following figures:

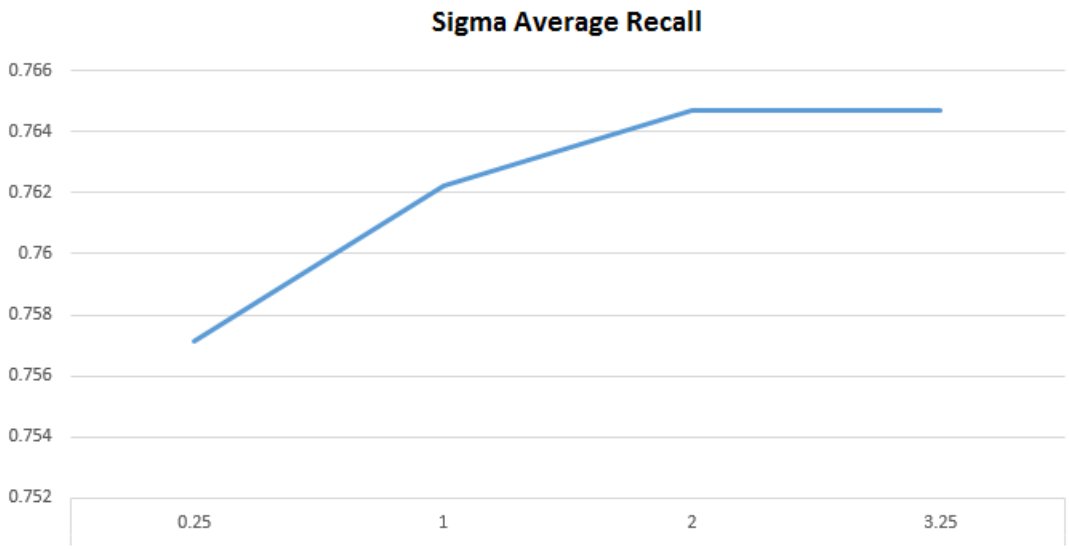

Figure 4.2 The average recall (y-axis) and the value of σ (x-axis) in preliminary experiments on batch 222 instrument template.

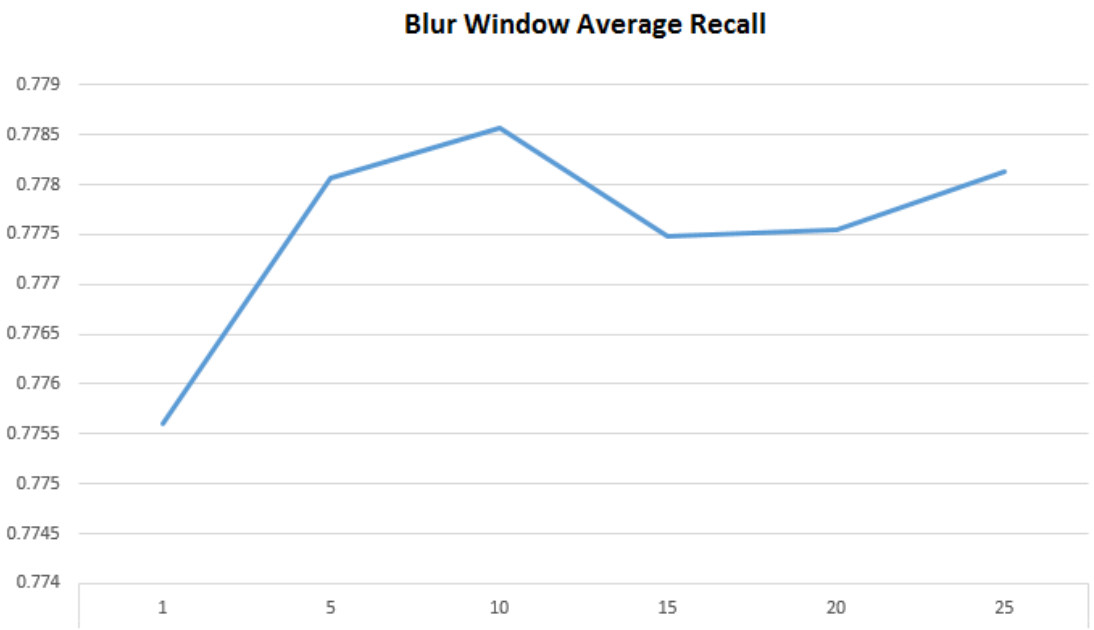

Figure 4.3 The average recall (y-axis) and the window size *W*(x-axis) in preliminary experiments on batch 222 instrument template.

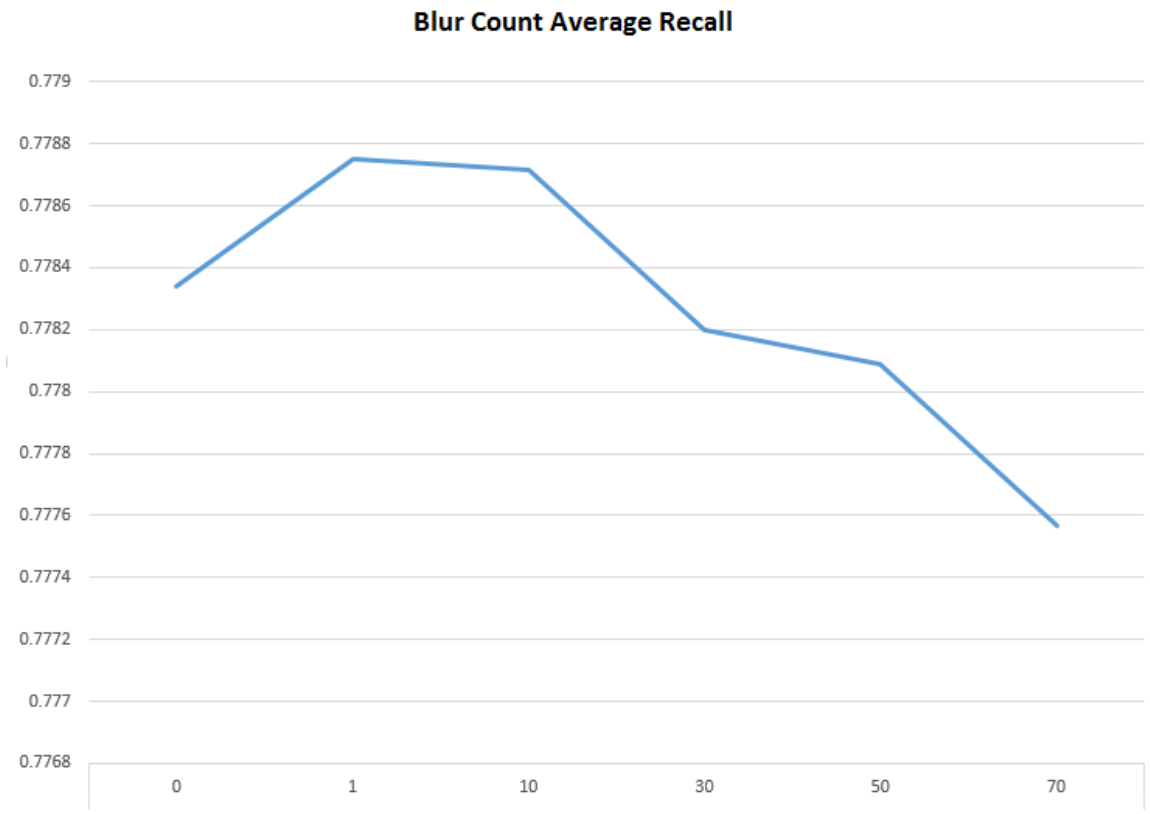

Figure 4.4 The average recall (y-axis) and the size of the filter bank (x-axis), *i.e.* the number of times the Gaussian blur filter is applied, in preliminary experiments on batch 222 instrument template.

In most cases, improvements were minuscule across all tests; however, small enhancements were considered as improvements based on runtime and average recall. The values of parameters for final experiments were the following: $\sigma = 2,\ W = 10, stepsize = .25, blurCoef \leq$ $1{,}6{,}15{,}20{,}15{,}6{,}1 >, blurCoefSum = 64, blurCount = 10, and\ fftLength = 4096.$ Increasing the size of the blur coefficients, blur coefficients sum, and FFT length produced a minor changes in recall as shown for other parameters, but the tradeoff with runtime performance significantly outweighed the gain in recall. Sigma (σ), window size (W), and blur count values were chosen based on Figure 4.2, Figure 4.3, and Figure 4.4 respectively. While the chosen blur count did not have the highest recall out of the other values, it was chosen in order to effectively smooth the envelope for more complex signals with more than two voices. On the other hand, W was chosen based on the highest recall; however, recall is shown to recover in Figure 4.3 as W increases to 25. As the window size increases, efficiency decreases

proportionately. Hence parameters were chosen based on both recall and runtime efficiency. Future experiments will involve a more exhaustive search for the best performing parameters.

Three different experiments were used in order to objectively gauge the performance of UMAGE. The original power spectrum is used as the control experiment. The UMAGE (no blurring) and smoothed UMAGE are used for experiments two and three respectively. WEKA was used for the classification task. Pitch detection and instrument identification are difficult to learn due to a large concept space and potentially large feature space. With using a single learner, classification accuracy was less than 5% (worse than random). The performance was boosted by using combined classification, where one learner was trained for each possible output. Then results were collected and averaged across output from all learners that produced any true positives or false negatives. For example, if an instrument or pitch was not present in the testing data, the results from the learner trained to for that instrument or pitch were considered to be void and were not used in the final collection of results. If these learners were used, final results would not be representative on the leaners ability to distinguish between positive (the pitch or instrument were present) and negative (the pitch or instrument were not present) examples.

Chapter 5

Results

The results collected include the average f-measure across all folds and sets (three sets of signals for each batch). This is mainly due to the fact that some instruments and pitches that have high precision but low recall (and vice versa), which produces misleading high accuracy. In other words, the learner may be exceptionally well at identifying true positives or true negatives, but not both. Results were taken from the many10, many6, brassQuartet, woodwindQuartet, stringQuartet, and 222 batches, including the corresponding limited note batches as listed in Table 8.3. Batches were selected to include those ranging all complexities (number of voices and notes possible). Reducing the number of batches to run in the final experiments gives a good representation of methodology used while also reducing runtime with classification. Representative results can be found in the following figures:

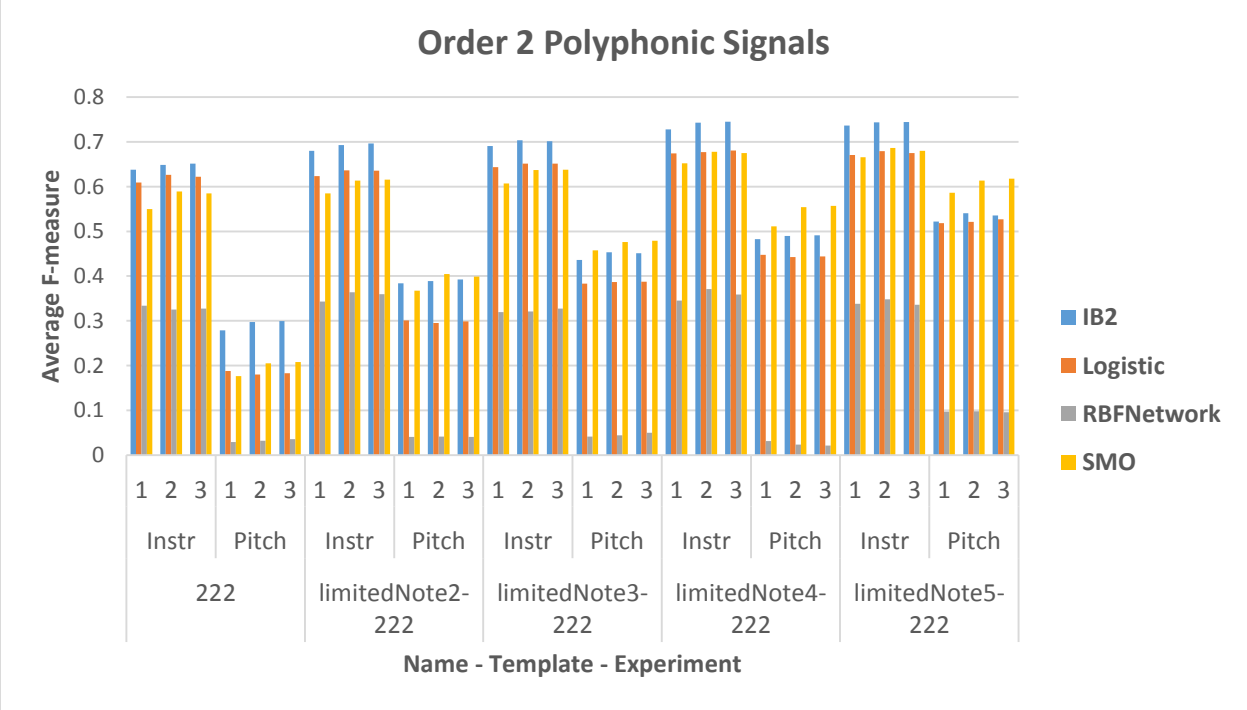

Figure 5.1 The average results for identifying groups of instruments (template Instr) and pitches (template Pitch) in the 222 batches. The y-axis represents the average f-value. The x-axis (from bottom up) represents the batch, template type, and experiment (power spectrum experiment 1, UMAGE experiment 2, and smooth UMAGE experiment 3). Each colored bar represents a different WEKA inducer.

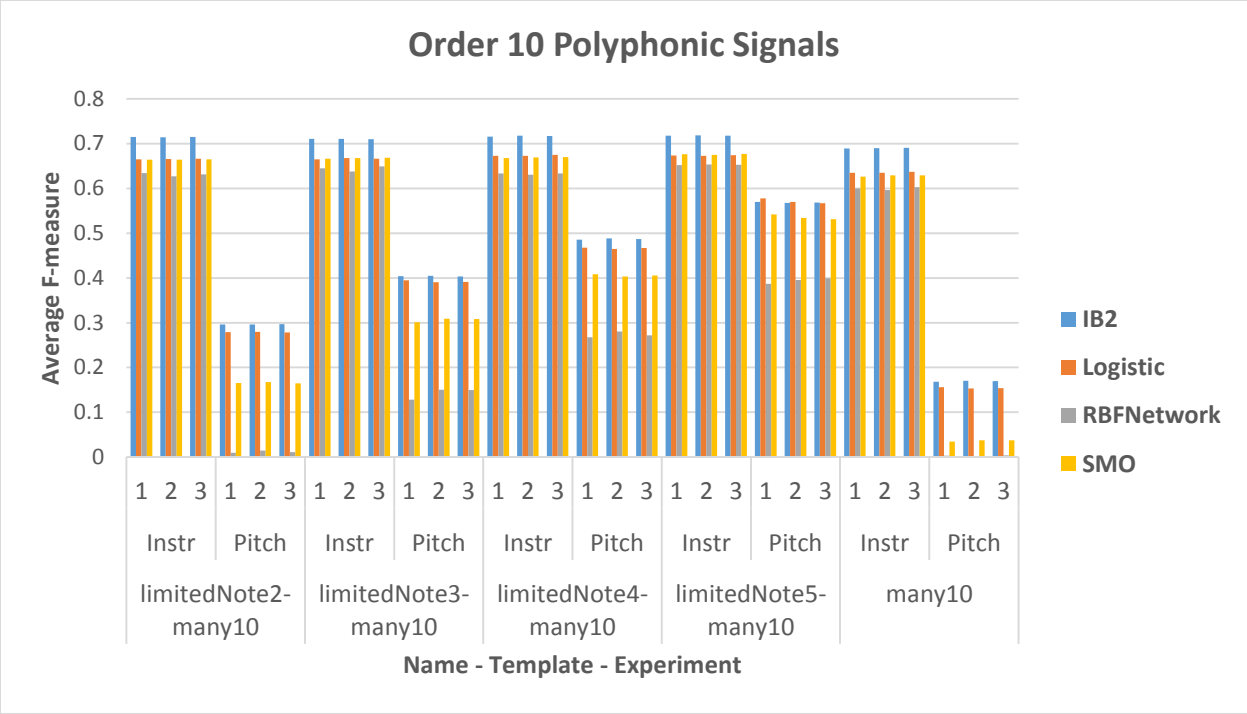

Figure 5.2 The average results for identifying groups of instruments (template Instr) and pitches (template Pitch) in the many10 batches. The y-axis represents the average f-value. The x-axis (from bottom up) represents the batch, template type, and experiment (power spectrum experiment 1, UMAGE experiment 2, and smooth UMAGE experiment 3). Each colored bar represents a different WEKA inducer.

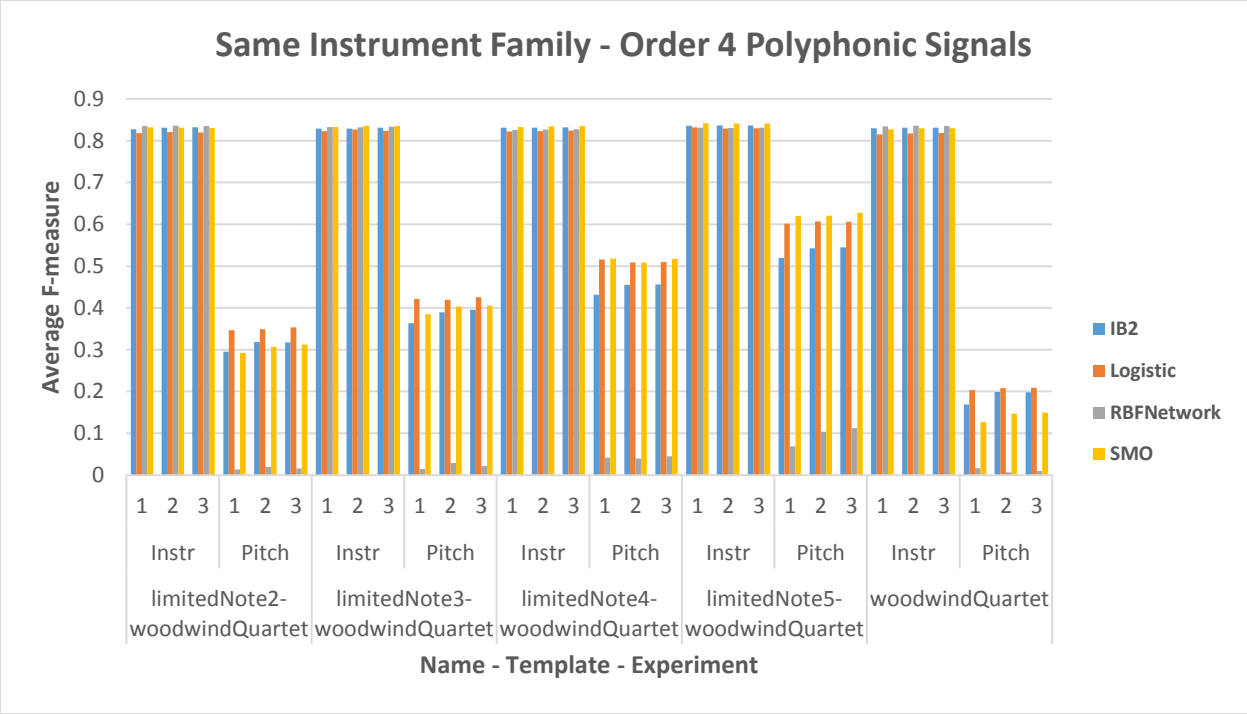

Figure 5.3 The average results for identifying groups of instruments (template Instr) and pitches (template Pitch) in the woodwindQuartet batches. The y-axis represents the average f-value. The x-axis (from bottom up) represents the batch, template type, and experiment (power spectrum experiment 1, UMAGE experiment 2, and smooth UMAGE experiment 3). Each colored bar represents a different WEKA inducer.

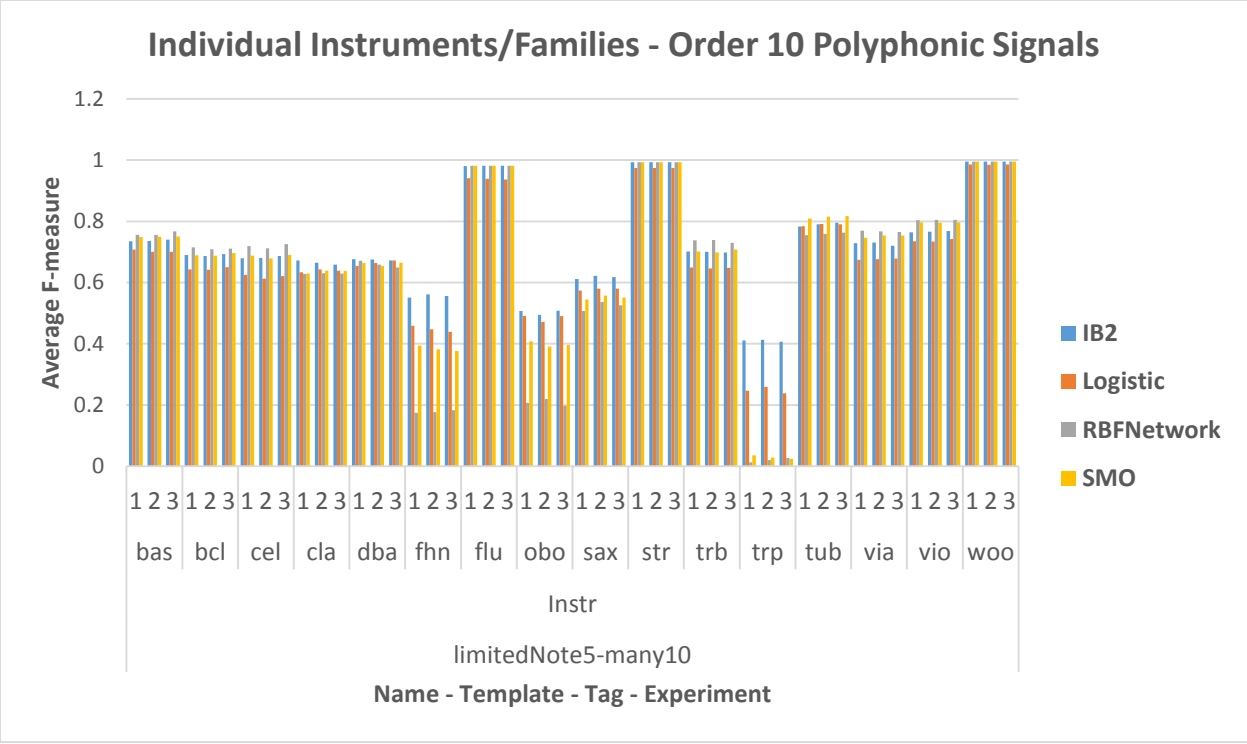

Figure 5.4 The average results for identifying individual instruments in batch limitedNote5-many10. The y-axis represents the average f-value. The x-axis (from bottom up) represents the batch, template type, instrument tag, and experiment (power spectrum experiment 1, UMAGE experiment 2, and smooth UMAGE experiment 3). Each colored bar represents a different WEKA inducer.

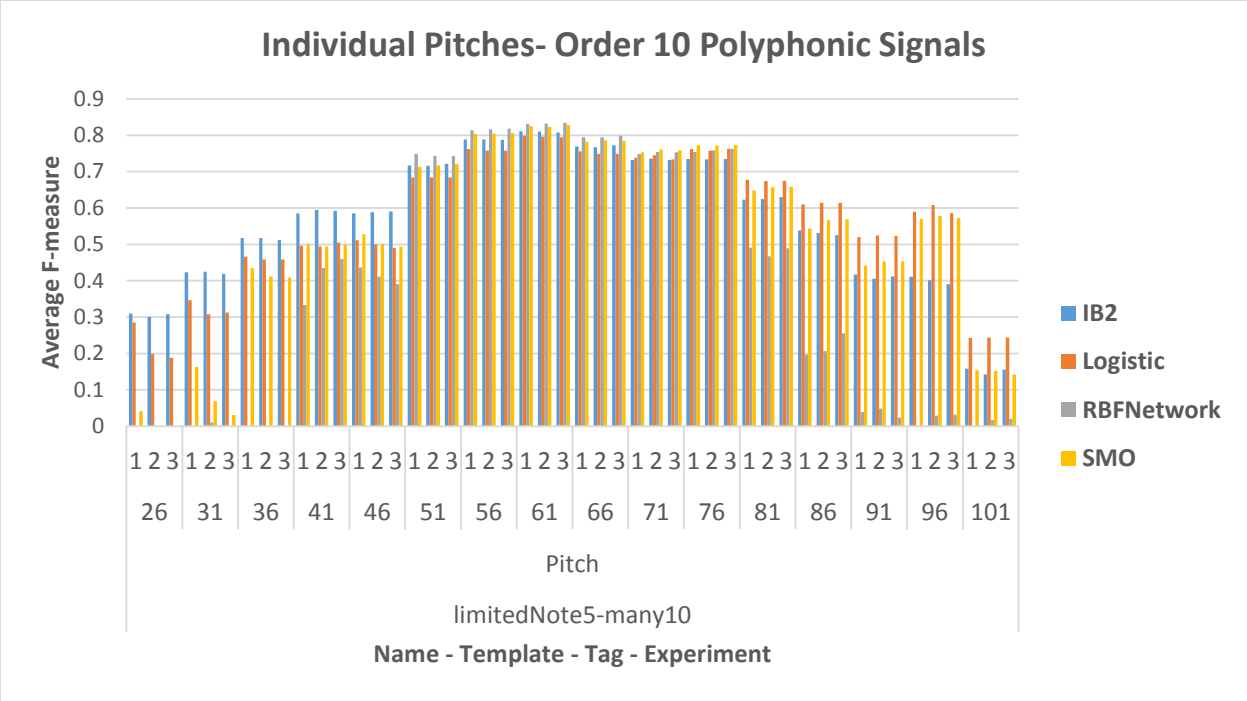

Figure 5.5 The average results for detecting individual pitches in batch limitedNote5-many10. The y-axis represents the average f-value. The x-axis (from bottom up) represents the batch, template type, instrument tag, and experiment (power spectrum experiment 1, UMAGE experiment 2, and smooth UMAGE experiment 3). Each colored bar represents a different WEKA inducer.

Instrument(s)	Strings	Cello	Double-Bass	Viola	Violin
Tag	str	cel	dba	via	vio
Instrument(s)	Woodwinds	Saxophone	Bass Clarinet	Bassoon	oboe	clarinet	flute
Tag	woo	sax	bcl	bas	obo	cla	16
Instrument(s)	Brass	French Horn	Trombone	Trumpet	tuba
Tag	brs	fhn	trb	trp	tub

Table 5.1 A lookup table for the meaning of each instrument tag used in the result graphs.

Table 5.1 serves as a look up table for the instrument codes used in Figure 5.4. The three different experiments are labeled as 1, 2, and 3 on Figure 5.2 through Figure 5.5. Group instrument identification on order 10 polyphonic signals, Figure 5.2, was surprisingly high,

reaching 71.82% using IB2 in experiment two, limitedNote5-many10 batch. If the task of identifying a specific group of instruments is broken down into individual instruments, as seen in Figure 5.4, the performance can be better represented. Some instruments, like the tuba, are identified more accurately than others in the limitedNote5-many10 batch, like the trumpet. Audiences are able to pick out individual instruments easier than others because of the niche that those instruments occupy. The tuba has a very low range and will generally stand out, just like a piccolo would in the higher range of pitches (frequencies); however, humans' ability to detect very high pitches (especially when they are only one or two pitches apart) is worse than that of low pitches. Instruments like the bassoon are also able to be detected easily due to their unique sound and partial structure. A suspicion is that the trumpet received low marks because it was covered up by the violin, which has a similar harmonic structure. Less complex signals, like the 222 batch in Figure 5.1, also performed well. The 222 batch responded positively to limiting notes contained in the signals where the many10 batch stayed relatively constant. Since the 222 batch only contained a maximum of two voices, limiting the notes gave a clearer model (spectral envelope) to distinguish instruments. Limiting polyphonic signals to a single family of instruments, like woodwinds in Figure 5.3, increased the instrument identification to upwards of 83%.

While pitch detection performed poorly when compared to instrument identification, UMAGE performance, statistically, outperformed the original power spectrum. This reveals that using the smooth UMAGE to model the spectrum produces better results as compared to instrument identification on average. Pitch detection achieves upwards of 30-35% F-measure from about MIDI note 50-83 overall in order 10 signals. This can be explained by Figure 4.1, the distribution of MIDI notes in the batch. The center range of notes are better represented since more instruments are able to play that range of notes. It is more difficult to detect an outlier instrument, like the tuba, because few are able to play at those pitches at those frequencies. This is opposite of instrument identification, where sticking out of the crowd actually increases the chances of being identified. When the signals are simplified, pitch detection achieves upwards of 70-83% F-measure for MIDI notes 51-76 in intervals of 5 for the limitedNote5-many10 batch shown in Figure 5.5. Overall pitch detection is best in the woodwindsQuartet, with an F-measure of 62.75% when using the limitedNote5-woodwindQuartet batch.

Chapter 6

Conclusions

Subjectively, UMAGE is shown to exhibit properties of goodness discussed in Section 3.3 and is also shown to model the true spectrum more accurately without dependence on periodicity or peak picking compared to envelope methods discussed in Chapter 1 and Section 2.4. Objectively, UMAGE competes with the state of the field by exhibiting up to 83% F-measure in identifying instrument groups in polyphonic signals. Even more so, UMAGE beats previous methods outlined in (Donnelly, 2012). Where most systems achieve less than 50% and even less than 40% accuracy on high order polyphonic signals, UMAGE successfully identifies order 10 polyphonic signals with an average F-Measure of 69.1%. UMAGE is able to detect pitches in high order polyphonic signals; however, the number of notes possible in training/testing data must be limited in order to achieve results better than random. When comparing UMAGE and smooth UMAGE to the baseline power spectrum for the 222 batch, a right tailed t-test give a p value of 10×10^{-5} for a .007% increase in F-measure in both instrument and pitch templates. Even though the improvement is extremely small, UMAGE and smooth UMAGE statistically win over the base line. This trend is more prominent when comparing some individual instruments and pitches, although, some exhibit the opposite relation (UMAGE and smooth UMAGE lose versus the power spectrum).

UMAGE shows promise for future research. Overall, UMAGE and smooth UMAGE are observed to perform better than the original power spectrum; however, this varies when results are broken down to individual pitches and instruments. Some spectrum, especially in pitch detection, are too complicated to learn with current features. Preliminary experiments revealed that creating a higher dimensionality feature space improves accuracy for pitch detection but decreases accuracy for instrument identification causing the model to over fit. Future works will include further exploration in parameter estimation for UMAGE, as well as different filters and sophisticated classification models like NMF and HMMs. As shown in Chapter 5, results can vary quite differently between each instrument and pitch. Look-ahead control policies are a viable candidate to select the best set of features, filters, and classifiers for each instrument and pitch. A similar approach has been implemented for tree classification from aerial images in (Bulitko, Levner, & Greiner, 2002).

Chapter 7

References

Abdallah, S. A., & Plumbley, M. D. (2003). Probability as Metadata: Event Detection in Music using ICA as a Conditional Density Model. *International Symposium on Independent Component Analysis and Blind Signal Separation.* Nara,Japan. Retrieved March 11, 2013, from http://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&sqi=2&ved=0CDAQFjAA&url=http%3A%2F%2Fciteseerx.ist.psu.edu%2Fviewdoc%2Fdownload%3Fdoi%3D10.1.1.10.5491%26rep%3Drep1%26type%3Dpdf&ei=NKo-UaXbIcfM2AWq7YGoBw&usg=AFQjCNF865ZjKMt8dfCysMe_6iz6bVnCS

About The Music Genome Project. (2013, February 7). Retrieved from Pandora.com: http://www.pandora.com/about/mgp

Alpaydin, E. (2010). Dimensionality Reduction. In E. Alpaydin, *Introduction to Machine Learning* (2ND ed., pp. 113-120). Cambridge, Massachusetts: MIT Press.

Alpaydin, E. (2010). Nonparametric Classification. In E. Alpaydin, *Introduction to Machine Learning* (2nd ed., pp. 171-172). Cambridge, Massachusetts: MIT Press.

Aucouturier, J.-J., & Sandler, M. (2001). Segmentation o Musical Signals Using Hidden Markov Models. *Audio Engineering Society 110th Convention.* Amsterdam.

Aucouturier, J.-J., Pachet, F., & Sandler, M. (2005, December). "The Way It Sounds": Timbre Models for Analysis and Retrieval of Music Signals. *IEEE Transactions on Multimedia, 7*(6).

Bay, M., & Beauchamp, J. W. (2006). Harmonic Source Seperation Using Prestored Spectra. *ICA LNCS*, 561-568.

Bello, J. P., Daudet, L., Abdallah, S., Duxbury, C., Davies, M., & Sandler, M. B. (2005). A Tutorial on Onset Detection in Music Signals. *IEEE Transactions on Speech and Audio Processing*.

Benetos, E., & Dixon, S. (2010). *Multiple-F0 Estimation and Note Tracking Using a Convolutive Probabilistic Model.* Queen Mary University of London, Centre for Digital Music. London: The Authors.

Bulitko, V., Levner, I., & Greiner, R. (2002). *Real-time Lookahead Control Policies.* American Association for Artificial Intelligence.

Chittka, L., & Brockmann, A. (2005). Perception Space - The Final Frontier. *PLoS Biol, 3(4)*(e137). Retrieved February 11, 2013, from http://www.plosbiology.org/article/info:doi/10.1371/journal.pbio.0030137

Cont, A., Dubnov, S., & Wessel, D. (2007). Realtime Multiple-pitch and Multiple-instrument Recognition for Music Signals using Sparse Non-negative Constraints. *10th International Conference on Digital Audio Effects.* Bordeaux.

Cooley, J. W., & Tukey, J. W. (1965). An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computaion 19*, 297-301. Retrieved from http://www.ams.org/journals/mcom/1965-19-090/S0025-5718-1965-0178586-1/S0025-5718-1965-0178586-1.pdf

Donnelly, P. J. (2012). *Bayesian Approaches to Musical Instrument Classification using Timbre Segmentation.* PhD Dissertaion Proposal, Montana State University, Bozeman. Retrieved from http://www.cs.montana.edu/files/techreports/1112/Donnelly.pdf

Echeverri, C. G. (2011). *Detection of genre-specific musical instruments: The case of the mellotron.* Universitat Pompeau Fabra, Barcelona.

Fastl, H., & Zwicker, E. (2007). Psychoacoustics: Facts and Models. Springer.

Fritts, L. (2011). *Musical Instrument Samples*. Retrieved November 1, 2012, from University of Iowa Electronic Music Studios: http://theremin.music.uiowa.edu/MIS.html

Heath, M. (2013, February 26). *NAudio*. Retrieved from CodePlex: http://naudio.codeplex.com/

Houton, B. (2003, October 5). *Exocortex.DSP*. Retrieved from Exocortex: http://www.exocortex.org/dsp/

Lee, D. D., & Seung, H. S. (2001). Algorithms for Non-negative Matrix Factorization. *Advances in Neural Information Processing Systems 13*, 556-562.

Leman, M. (2008). *Embodied Music: Cognition and Mediation Technology*. MIT Press.

Multiple Fundamental Frequency Estimation Tracking Results. (2010, August 6). Retrieved from MIREX WIKI: http://www.music-ir.org/mirex/wiki/2010:Multiple_Fundamental_Frequency_Estimation_%26_Tracking_Results

Noxon, A. (2012). *How TubeTraps opened up a Whole New Realm of Precision in the Performance of Audio Playback Systems*. Retrieved March 10, 2013, from Acoustic Sciences Corporation: http://www.acousticsciences.com/how-tubetraps-opened-whole-new-realm-precision-performance-audio-playback-systems

Oppenheim, A. V., & Schafer, R. W. (2010). "Computation of the Discrete Fourier Transform". In A. V. Oppenheim, & R. W. Schafer, *Discrete-Time Signal Processing* (3rd ed., pp. 716-762). Upper Saddle River: Pearson.

Oppenheim, A. V., & Schafer, R. W. (2010). "Sampling of Continuous-Time Signals". In A. V. Oppenheim, & R. W. Schafer, *Discrete-Time Signal Processing* (3rd ed., pp. 153-273). Upper Saddle River: Pearson.

Oppenheim, A. V., & Schafer, R. W. (2010). "The Discrete Fourier Transform". In A. V. Oppenheim, & R. W. Schafer, *Discrete-Time Signal Processing* (3rd ed., pp. 623-683). Upper Saddle River: Pearson.

Patin, F. (2011). *Matrix convolution filters.* Retrieved March 16, 2013, from Gamedev.net: http://archive.gamedev.net/archive/reference/programming/features/imageproc/page2.html

Philharmonia Orchestra. (n.d.). *The Sound Exchange*. Retrieved March 20, 2013, from Philharmonia Orchestra: http://www.philharmonia.co.uk/thesoundexchange/make_music/samples/

Rabiner, L. R. (1989, February). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *IEE, 77*(2).

Resources. (2013). Retrieved from MIDI Manufacturers Association: http://www.midi.org/aboutmidi/resources.php

Rossing, T. D. (1990). *The Science of Sound* (2nd ed.). Addison Wesley.

Schwarz, D., & Rodet, X. (1999). *Spectral Envelope Estimation and Representation for Sound Analysis-Synthesis.* Paris: IRCAM.

Smith, L. I. (2002). *A tutorial on Principle Components Analysis.*

Smith, S. W. (1997). "Convolution". In S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing* (1st ed., pp. 107-122). California Technical Pub. Retrieved March 14, 2013, from http://www.dspguide.com/ch6.htm

Smith, S. W. (1997). "Linear Systems". In S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing* (1st ed., pp. 87-106). California Technical Pub. Retrieved March 14, 2013, from http://www.dspguide.com/ch5.htm

Smith, S. W. (1997). "The Fast Fourier Transform". In S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing* (1st ed., pp. 225-242). California Technical Pub. Retrieved March 10, 2013, from http://www.dspguide.com/ch8.htm

Song, B. L. (2013, August 17). *Demonstrates gaussian blur with circles showing radius*. Retrieved March 18, 2013, from Wikipedia: http://en.wikipedia.org/wiki/File:Gaussian_blur_example.jpg

Stuttle, M. N. (2003). *A Gaussian Mixture Model Spectral Represntation for Speech Recognition.* Cambridge University, Engineering Department.

Thomas, J. M. (2012). *Robust Realtime Polyphonic Pitch Detection.* Fairfax: George Mason University.

Vaseghi, S. (2013, February 11). *Multimedia Signal Processing Lecture Notes: Chapter12: Music.* Retrieved from Brunel University London: http://dea.brunel.ac.uk/cmsp/home_saeed_vaseghi/chapter12-music.pdf

Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing, 17*(4), 395-416.

Wang, H. (2009). *FFT Basics and Case Study using Multi-Instrument.* Virtins Technology.

Weisstein, E. W. (2013). *Fast Fourier Transform*. Retrieved from MathWorld--A Wolfram Web Resource: http://mathworld.wolfram.com/FastFourierTransform.html

Chapter 8

Appendix

Instrument Name	Instrument Tag
banjo	1
bassclarinet	2
bassoon	3
cello	4
clarinet	5
contrabasson	6
doublebass	7
englishhorn	8
flute	9
frenchhorn	10
guitar	11
mandolin	12
oboe	13
saxophone	15
trombone	16
trumpet	17
tuba	18
viola	19
violin	20

Table 8.1 The instrument tags/codes used in each experimental batch.

Notes Tag	Notes (MIDI #)
ALL	21-109
SPACED x2	21;23;25;27;29;31;33;35;37;39;41;43;45;47;49;51;53;55;57;59;61;63;65;67;69;71;73;75;77;79;81;83;85;87;89;91;93;95;97;99;101;103;105;107
SPACED x3	21;24;27;30;33;36;39;42;45;48;51;54;57;60;63;66;69;72;75;78;81;84;87;90;93;96;99;102;105;108
SPACED x4	21;25;29;33;37;41;45;49;53;57;61;65;69;73;77;81;85;89;93;97;101;105
SPACED x5	21;26;31;36;41;46;51;56;61;66;71;76;81;86;91;96;101;106

Table 8.2 The note tags/codes used in each experimental batch.

Batch Name	# VOICES	Instruments	Notes
222	2	17;16;5;15;4;20	ALL
333	3	16;17;18;5;15;9;4;19;20	ALL
444	4	10;16;17;18;5;15;9;4;19;20;3;7	ALL
455	5	8;10;16;17;18;5;3;15;9;13;4;19;20;7;12	ALL
brassQuartet	4	10;16;17;18	ALL
brassTrio	3	10;16;17	ALL
limitedNote2_222	2	17;16;5;15;4;20	SPACED x2
limitedNote2_333	3	16;17;18;5;15;9;4;19;20	SPACED x2
limitedNote2_444	4	10;16;17;18;5;15;9;4;19;20;3;7	SPACED x2
limitedNote2_455	5	8;10;16;17;18;5;3;15;9;13;4;19;20;7;12	SPACED x2
limitedNote2_brassQuartet	4	10;16;17;18	SPACED x2
limitedNote2_brassTrio	3	10;16;17	SPACED x2
limitedNote2_many10	10	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x2
limitedNote2_many6	6	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x2
limitedNote2_many7	7	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x2
limitedNote2_many8	8	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x2
limitedNote2_many9	9	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x2
limitedNote2_stringQuartet	4	4;19;20;7	SPACED x2
limitedNote2_stringTrio	3	4;19;20	SPACED x2
limitedNote2_woodwindQuartet	4	5;3;15;2	SPACED x2
limitedNote2_woodwindQuintet	5	5;3;15;2;9	SPACED x2
limitedNote2_woodwindTrio	3	5;15;2	SPACED x2
limitedNote3_222	2	17;16;5;15;4;20	SPACED x3
limitedNote3_333	3	16;17;18;5;15;9;4;19;20	SPACED x3
limitedNote3_444	4	10;16;17;18;5;15;9;4;19;20;3;7	SPACED x3
limitedNote3_455	5	8;10;16;17;18;5;3;15;9;13;4;19;20;7;12	SPACED x3
limitedNote3_brassQuartet	4	10;16;17;18	SPACED x3
limitedNote3_brassTrio	3	10;16;17	SPACED x3
limitedNote3_many10	10	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x3
limitedNote3_many6	6	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x3
limitedNote3_many7	7	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x3
limitedNote3_many8	8	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x3
limitedNote3_many9	9	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x3
limitedNote3_stringQuartet	4	4;19;20;7	SPACED x3
limitedNote3_stringTrio	3	4;19;20	SPACED x3
limitedNote3_woodwindQuartet	4	5;3;15;2	SPACED x3
limitedNote3_woodwindQuintet	5	5;3;15;2;9	SPACED x3
limitedNote3_woodwindTrio	3	5;15;2	SPACED x3
limitedNote4_222	2	17;16;5;15;4;20	SPACED x4
limitedNote4_333	3	16;17;18;5;15;9;4;19;20	SPACED x4

limitedNote4_444	4	10;16;17;18;5;15;9;4;19;20;3;7	SPACED x4
limitedNote4_455	5	8;10;16;17;18;5;3;15;9;13;4;19;20;7;12	SPACED x4
limitedNote4_brassQuartet	4	10;16;17;18	SPACED x4
limitedNote4_brassTrio	3	10;16;17	SPACED x4
limitedNote4_many10	10	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x4
limitedNote4_many6	6	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x4
limitedNote4_many7	7	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x4
limitedNote4_many8	8	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x4
limitedNote4_many9	9	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x4
limitedNote4_stringQuartet	4	4;19;20;7	SPACED x4
limitedNote4_stringTrio	3	4;19;20	SPACED x4
limitedNote4_woodwindQuartet	4	5;3;15;2	SPACED x4
limitedNote4_woodwindQuintet	5	5;3;15;2;9	SPACED x4
limitedNote4_woodwindTrio	3	5;15;2	SPACED x4
limitedNote5_222	2	17;16;5;15;4;20	SPACED x5
limitedNote5_333	3	16;17;18;5;15;9;4;19;20	SPACED x5
limitedNote5_444	4	10;16;17;18;5;15;9;4;19;20;3;7	SPACED x5
limitedNote5_455	5	8;10;16;17;18;5;3;15;9;13;4;19;20;7;12	SPACED x5
limitedNote5_brassQuartet	4	10;16;17;18	SPACED x5
limitedNote5_brassTrio	3	10;16;17	SPACED x5
limitedNote5_many10	10	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x5
limitedNote5_many6	6	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x5
limitedNote5_many7	7	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x5
limitedNote5_many8	8	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x5
limitedNote5_many9	9	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	SPACED x5
limitedNote5_stringQuartet	4	4;19;20;7	SPACED x5
limitedNote5_stringTrio	3	4;19;20	SPACED x5
limitedNote5_woodwindQuartet	4	5;3;15;2	SPACED x5
limitedNote5_woodwindQuintet	5	5;3;15;2;9	SPACED x5
limitedNote5_woodwindTrio	3	5;15;2	SPACED x5
many10	10	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	ALL
many6	6	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	ALL
many7	7	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	ALL
many8	8	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	ALL
many9	9	2;3;4;5;7;8;9;10;11;12;13;15;16;17;18;19;20	ALL
stringQuartet	4	4;19;20;7	ALL
stringTrio	3	4;19;20	ALL
woodwindQuartet	4	5;3;15;2	ALL
woodwindQuintet	5	5;3;15;2;9	ALL
woodwindTrio	3	5;15;2	ALL

Table 8.3 The complete table of the different batches of notes and instruments used in experiments.