LARGE-SCALE MUSIC CLASSIFICATION USING AN APPROXIMATE *k*-NN CLASSIFIER

Haukur Pálmason
Master of Science
Computer Science
January 2011
School of Computer Science
Reykjavík University

M.Sc. RESEARCH THESIS

ISSN 1670-8539

Large-Scale Music Classification using an Approximate k-NN Classifier

by

Haukur Pálmason

Research thesis submitted to the School of Computer Science
at Reykjavík University in partial fulfillment of
the requirements for the degree of
Master of Science in **Computer Science**

January 2011

Research Thesis Committee:

Dr. Björn Þór Jónsson, Supervisor
Associate Professor, Reykjavík University, Iceland

Dr. Laurent Amsaleg
Research Scientist, IRISA-CNRS, France

Dr. Yngvi Björnsson
Associate Professor, Reykjavík University, Iceland

Large-Scale Music Classification using an Approximate k-NN Classifier

Haukur Pálmason

January 2011

Abstract

Content based music classification typically uses low-level feature vectors of high-dimensionality to represent each song. Since most classification methods used in the literature do not scale well, the number of such feature vectors is usually reduced in order to save computational time. Approximate k-NN classification, on the other hand, is very efficient and copes better with scale than other methods. It can therefore be applied to more feature vectors within a given time limit, potentially resulting in better quality. In this thesis, we first demonstrate the effectiveness and efficiency of approximate k-NN classification through a traditional genre classification task, achieving very respectable accuracy in a matter of minutes. We then apply the approximate k-NN classifier to a large-scale collection of more than thirty thousand songs. We show that, through an iterative process, the classification converges relatively quickly to a stable state. We also show that by weighing the classifier with the size of genres, genre distribution in the classification result is improved considerably.

Flokkun stórra tónlistargrunna með k-NN nálgunarflokkun

Haukur Pálmason

Janúar 2011

Útdráttur

Venjan er að sjálfvirk tegundarflokkun tónlistar byggð á innihaldi noti margvíða lýsinga fyrir hvert lag. Þar sem fæstar gerðir flokkunar skalast vel er algengast að nota fáa slíkar lýsinga til að spara tíma. k-NN nálgunarflokkun er hins vegar skilvirk aðferð sem ræður betur við stór gagnasöfn en margar aðrar aðferðir. Þess vegna er hægt að nota slíka flokkun á fleiri lýsinga innan ákveðinna tímamarka, sem hugsanlega skilar sér í meiri gæðum. Í þessari ritgerð sýnum við fyrst fram á árangur og skilvirkni notkunar k-NN nálgunarflokkunar á hefðbundna tónlistartegundarflokkun, þar sem við náum mjög prýðilegum árangri á nokkrum mínútum. Við keyrum síðan k-NN nálgunarflokkun á stórt lagasafn með yfir þrjátíu þúsund lögum. Við sýnum fram á að með ítruðu ferli beinist tónlistarsafnið fljótt yfir í stöðuga flokka. Einnig sýnum við fram á að með því að gefa flokkuninni mismunandi vægi eftir stærð hvers tegundarflokks, bætum við dreifingu tegundarflokka í niðurstöðunum umtalsvert.

To my wife, Sigurbjörg Ósk Sigurðardóttir, and my three kids,
Stefán Elí Hauksson, Mikael Blær Hauksson and Ísabella Sól Hauksdóttir.

Acknowledgements

Dr. Björn Þór Jónsson, my supervisor, for convincing me to take on this project, and for all his help in turning this into reality.

Dr. Laurent Amsaleg, for his contributions as co-author of the related paper.

Grímur Tómas Tómasson, a fellow M.Sc. student at the Database Lab for invaluable help with coding and debugging.

The Reykjavík University Development Fund, for a project grant supporting this work.

The Icelandic Research Fund for Graduate Students, for a project grant supporting this work.

Tonlist.is, for granting us research access to their collection of Icelandic songs.

Davíð Einarsson, an employee at Tonlist.is, for all his help.

Dr. George Tzanetakis, for granting access to his song collection, and for quickly answering all questions regarding the MARSYAS framework.

Icelandic musicians, who participated in my ground-truth experiment.

Publications

Part of the material in this thesis has been submitted to an international conference. Co-authors are Björn Þór Jónsson (Reykjavík University) and Laurent Amsaleg (IRISA-CNRS). While Björn and Laurent contributed significantly to the writing of the conference submission, the implementation and experimentation is entirely my work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Today, the majority of all music is created, stored, and played in digital format. A personal music collection may contain thousands of songs, while professional collections typically contain tens or hundreds of thousands. Browsing and searching such large song collections requires new tools as querying by artist, album name or song title becomes inadequate at that scale.

Information based on the *contents* of songs is likely to be useful, allowing genre, mood, rhythm, instrumentation or key to be part of modern browsing tools. The content based information typically relies on low-level descriptors, where each descriptor is a high-dimensional feature vector describing a particular aspect of the audio file such as the frequency distribution, key or tempo. These descriptors are often created for short intervals of the audio files, resulting in multiple descriptors per song.

The usage of these descriptors for various forms of music information retrieval has received significant attention lately, for example for genre classification (Panagakis, Kotropoulos, & Arce, 2009), for mood and theme classification (Bischoff et al., 2009), and tag prediction (Hoffman, Blei, & Cook, 2009).

In particular, classifying songs according to their genre has proved to be useful, and music genres, such as “pop”, “rock”, and “jazz” are typical browsing aids for music. As a consequence, automatic genre classification has received much attention lately (e.g., see Li, Ogihara, & Li, 2003; Bergstra, Casagrande, Erhan, Eck, & Kégl, 2006; Panagakis & Kotropoulos, 2010; Tzanetakis & Cook, 2002). This work, however, has largely focused on small collections—the collections mainly used contain 1,000–1,500 songs.

1.1 Large-Scale Genre Classification

Using state-of-the-art solutions for large-scale genre classification in the real world, however, raises two major issues.

First, as mentioned above, typical music collections are much larger than the ones used when evaluating state of art solutions, which challenges the complexity of current approaches.

Second, songs in real life song collections may not have consistent genre labels. From a nicely annotated small collection with similar number of songs belonging to each genre, that a particular algorithm classifies well, it is expected that genre labels will be propagated in such a way that the resulting classification will be of high quality. This remains to be proven, however, as no studies addressing quality issues at a large scale have been performed so far.

1.2 Contributions

The goal of this thesis is to examine, using both a small standard reference collection and a real life collection of over 30,000 songs, whether approximate k-NN classification is competitive with state-of-the-art results from the literature when using a large number of audio descriptors.

In part one of this thesis we therefore study the efficiency and effectiveness of k-NN genre classification using the collection of Tzanetakis and Cook (Tzanetakis & Cook, 2002). This collection has been studied well in the past, and the most recent methods achieve over 90% classification accuracy. Since we have not obtained access to the feature generation software of those works, however, we work with the open-source MARSYAS software, and use the resulting feature vectors for the classification.

We obtain approximately 80% classification accuracy, which ranks quite high in the literature; to the best of our knowledge it is the highest classification accuracy using these particular descriptors. More importantly, however, our results were obtained in only a few minutes using the cluster pruning indexing technique.

We also perform a user study on the Tzanetakis collection, where 20 individuals from the Icelandic music industry classify 192 songs from the set which were incorrectly classified by one version of our k-NN classifier. We also examine further 15 songs from this collection and look at their classification in iTunes, allmusic.com and last.fm. Our study shows

that in many cases it is difficult to pinpoint one genre as the "correct" one, and that often there is harmony between the top three genres chosen by our k-NN classification program and the genres used on the above web-sites.

In part two, we study the large-scale genre classification of Icelandic songs from the music collection of `http://tonlist.is`. By using cluster pruning we are able to classify a collection of over 30,000 songs in 95 minutes. Since the ground-truth of Tonlist.is is slightly suspect, it is difficult to assess the classification accuracy, but we suggest alternate ways of assessment. We look at the stability of genre classification over five iterative classification experiments, and we see that the classification quickly converges to a stable one. We also examine the genre distribution. Using the regular voting scheme of the k-NN classifier, the distribution is skewed towards pop and becomes more skewed at each iteration. By weighing votes, giving higher weight to votes belonging to genres with fewer songs, we generate a much better distribution, and thus a better classification of the smaller genres.

Our results indicate that k-NN classification is indeed a very promising approach for large-scale music classification.

1.3 Overview of Thesis

We start by providing background information on music, its representation and its classification in Chapter 2. The remainder of the thesis is split into two parts with part I providing information on our k-NN classification experiments on theTzanetakis collection, and part II dealing with our large-scale classification of Tonlist.is.

In Part I, we describe our motivation, the classification process and our two types of k-NN classifiers in Chapter 3. We then detail our experimental setup, including our methods of feature extractions from the audio files in Chapter 4. In Chapter 5 we report results from our genre classification experiments on the Tzanetakis collection. Part I concludes with Chapter 6, where we perform a user study on the ground-truth of the Tzanetakis collection.

In Part II, we start by giving a detailed information on the Tonlist.is music collection in Chapter 7. We discuss the issues in classifying this collection , including genre distribution, which genres from the selection to use, how to choose the training set, our classification methodology, and how to measure the classification results. Chapter 8 details our experimental setup for the Tonlist.is collection, including results from a set of experiments we did on a subset of the collection, in order to tune the parameter for the

large-scale classification experiments. Results from our classification experiments are provided in Chapter 9, while we conclude our overall findings in Chapter 10.

Chapter 2

Background

In this section we start by introducing music and its main representations in computers. We then introduce the feature-extraction process, including a description of the MARSYAS framework which is used in this thesis. Finally we conclude the section by a discussion of some standard reference collections used for automatic genre classification and the classification results achieved by researchers on these collections.

2.1 Music Background

It is hard to clearly define what is music. Musicality is very subjective and varies between individuals and cultures. In most cases it can be said that music is a organization of sounds and silence. Although deep music theory is outside the scope of this thesis there are some basic elements of music and sound that is beneficial to understand before continuing.

Pitch is the fundamental frequency of a sound wave. It tells how high or low a given note is. In western music pitch is organized into a 12 tone chromatic scale.

Octave is one such set of 12 tones. Given two notes x and y where y is an octave higher than x, the frequency of y is double that of x.

Scale is usually eight out of the 12 tones in an octave. Different scales have different pitch intervals.

Tonality or key describes the main or base chord in a song.

Timbre is the sound characteristics that define a given sound. It is the collection of overtones to the fundamental frequency what makes two sounds of the same pitch sound different.

Rhythm is the time interval of sounds played. Rhythmic information indicates both the *tempo* and *meter* of a song.

Tempo is the speed at which a song is played.

Meter is the time signature of a song, that is, how many notes of what length are in each bar of the music. Most popular rock and pop music is said to have the meter 4/4, which means that there are four quarternotes in one bar.

Melody is a sequence of notes, which usually has a series of pitch changes and a specific rhythm. Melodies can be monophonic, where each element in the sequence is a single note, or polyphonic, where each element in the sequence is a harmonic collection of notes.

Harmony is a collection of notes played at the same time.

The importance of each musical element varies greatly depending on the research area at hand. For example melody, (including various pitches, and perhaps rhythm) is of utmost importance to Query By Humming systems, while it usually plays a lesser role in genre classification systems. Timbre, on the other hand, is very important for genre classification, while quite irrelevant for Query By Humming. It is important for MIR (Music Information Retrieval) research to work with an appropriate representation of music.

2.2 Music Representation

MIR researchers traditionally divide music representation formats into *symbolic representation* and *audio representation* (Lidy, Rauber, Pertusa, & Iñesta, 2007). For content-based retrieval we can add *feature representation* to this list. We will in this section describe these three forms of music representation.

2.2.1 Symbolic Representation

Symbolic representation of music does not contain any actual sounds, rather symbols or messages that help either a human or a computer to play a musical piece. The symbolic

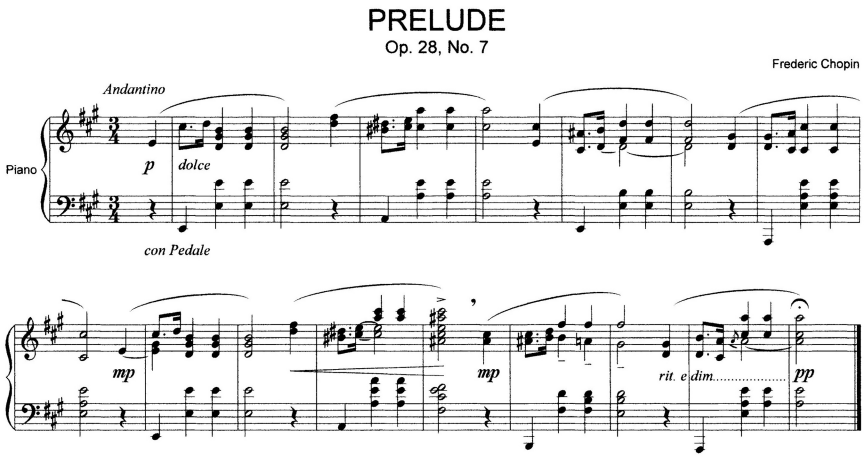

Figure 2.1: Chopin's Prelude no. 7 as represented by musical notation

form therefore does usually not represent a particular performance, but rather the composition of a song. Orio (2006) divides the parameters represented by the symbolic form into two groups:

General parameters describe the global features of a song, such as tonality, tempo, meter and structure.

Local parameters represent the position, length and volume of each tone played by each instrument. From these parameters information about melody, harmony and rhythm can be extracted.

The modern musical notation system is one form of symbolic representation of music. Figure 2.1 shows an example of musical notation, sometimes called a score. General parameters shown by this score include the following:

- The three hash signs at the beginning of each line, along with the final notes of the piece indicates that they key of the piece is A major.
- The 3/4 sign states that each bar consists of three quarter notes.
- *Andantino* is Italian and stands for moderate tempo.
- *p* is short for *piano* which is Italian for soft.

On the other hand, each note in the score is an example of a local parameter.

The *Musical Instruments Digital Interface* (MIDI) has been the symbolic form of choice for the MIR community as well as for computer games and multimedia applications. MIDI was originally created in the early 80's in order to allow digital musical instruments

from different manufacturers to communicate with each other. (Midi Manufacturers Association Inc., n.d.) One instrument, usually a digital keyboard, could control another through the MIDI protocol. The original MIDI messages did not contain any time information, but instead the receiver would act upon a message as soon as it was received from the sender. Example messages are *NoteOn*, *NoteOff*, *Pitch Bend* and *Key Pressure*.

There are many commercial and free-ware software programs that are able to scan, edit and create music via the notation system. For a long time, however, the lack of an accepted format standard that includes necessary information about how to display music limited the portability of the files from these programs. *MusicXML* changed this. MusicXML (http://www.musicxml.org/xml.html) is an XML based music interchange language that has elements that represent how music should be performed as well as how the music should be displayed in notational form. The language is now supported by over 80 programs including the popular notation programs *Sibelius* (www.sibelius.com) and *Finale* (www.finale.com).

2.2.2 Audio Representation

Contrary to symbolic representation, audio representation contains a recording of a given performance of a musical piece.

Digital recordings of sounds are based on *sampling* the analog audio signal. Sampling is performed by recording the amplitude of the signal at a given *sampling rate* and storing these samples in binary format. The sampling rate and the *bit rate* (number of bits used to store each sample) are the two most relevant factors regarding the quality of the recording. Audio CDs use a sampling rate of 44.1 kHz, or 44,100 samples per second, and 16 bits for each sample. Popular sound file formats include the WAVE format adopted by Microsoft and IBM and AIFF developed by Apple. The most popular formats can easily be converted from one to another making the choice of format largely irrelevant to the MIR researcher.

Figure 2.2 shows a waveform display of 30 second audio clip. The x-axis represents time while the y-axis represents amplitude.

Disadvantages with audio files are that they do take up a lot of storage space, and that sampling of silence takes up as much space as the sampling of actual sounds. Many compression schemes have therefore been developed; the best known formats are MPEG-1 Layer 3 (MP3) and MPEG-4 Advanced Audio Coding (AAC). These schemes aim at

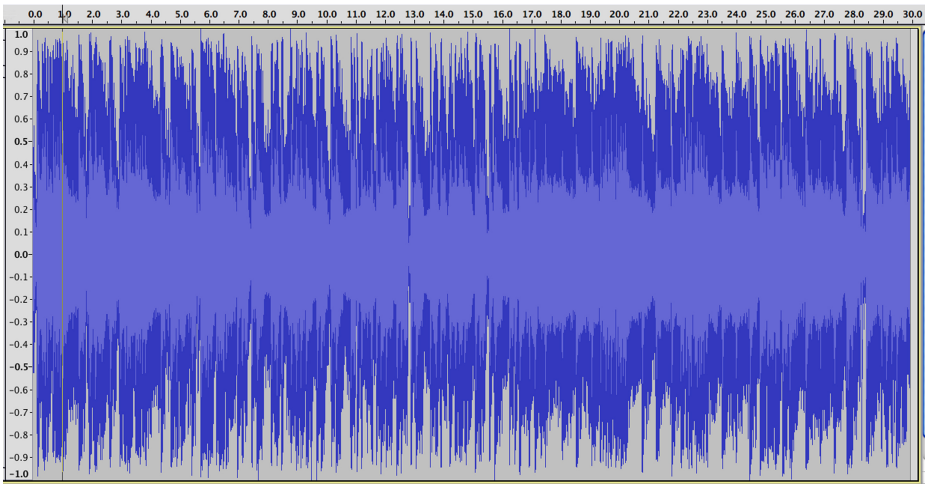

Figure 2.2: Waveform display of a 30 second audio clip

reducing the number of bits needed to represent audio without noticeable difference in sound quality.

2.2.3 Feature Representation

While both symbolic representation and audio representation are good for reproducing music they contain too much information to be used directly for content-based music information retrieval tasks. These tasks usually rely on low-level descriptors where each descriptor is a high-dimensional feature vector describing a particular aspect of an audio file such as the frequency distribution, key or tempo. These descriptors are often created for short intervals of the audio files resulting in hundreds, or thousands of descriptors per song. This thesis will focus only on this form of music representation.

Figure 2.3 shows a feature representation of a few milliseconds of a reggae song.

Sound waves can be divided into a collection of simple sine waves where the set of amplitudes and phases of the sine waves is called the frequency spectrum of that sound wave. Waveforms are typically decomposed into their basic sinusoidal waves with *Fourier Transform*. With sampled frequency and time variables the Fourier transform is called the *Discrete Fourier Transform* or DFT. When extracting spectral features from a digital audio file researchers frequently use *Short-Time Fourier Transform* or STFT to split the signal into segments and apply the Fourier transform to each segment individually. Features are then calculated for each segment.

Many researchers write their own feature extraction software, but there are also several open source platform available for researchers to use. One of them is MARSYAS which

0.001025,0.000256,0.000195,0.014124,-1.958591,0.131598,0.068298,0.049015,-0.021473,0.038111,-0.000046,0.039370,-0.000764,0.006617,0.012997,-0.000935,-0.013090,0.008483,0.0
15335,0.015836,0.011548,0.012486,0.011309,0.013594,0,0.010775,0.018429,0.015514,0.008886,0.010294,0.011915,0.012119,0.013117,0.017485,0.017364,0.010131,0.020732,0.016919,0.0
15721,0.025000,0.025000,0.025000,0.006404,0.001602,0.001215,0.008202,12.231394,0.821826,0.426524,0.306100,0.134097,0.238006,0.008285,0.245863,0.004773,0.041322,0.001164,0.
005841,0.001750,0.052979,0.095767,0.098893,0.072117,0.077978,0.070624,0.084897,0.067287,0.115088,0.096883,0.055493,0.064285,0.074409,0.075684,0.081916,0.109195,0.108437,0.
063268,0.129469,0.105656,0.098181,0.156125,0.156125,0.156125
0.003223,0.002376,0.002626,0.016454,-3.571522,0.240811,0.052753,0.019281,-0.003557,0.056462,0.042040,0.054709,-0.001963,0.032601,0.008219,0.011608,0.019615,0.025769,0.0346
40,0.034076,0.029615,0.030607,0.031319,0.034427,0.028403,0.025194,0.023633,0.016389,0.022575,0.023501,0.027261,0.028346,0.031407,0.031923,0.021289,0.030971,0.021985,0.0248
09,0.050000,0.050000,0.050000,0.014993,0.013289,0.015205,0.009025,15.644464,1.086448,0.439853,0.362067,0.406987,0.261497,0.262832,0.261568,0.008774,0.166421,0.087189,0.078
697,0.221934,0.119024,0.152032,0.148924,0.132342,0.135769,0.141956,0.153515,0.089081,0.121576,0.108191,0.071705,0.098800,0.102450,0.119595,0.092513,0.137824,0.139712,0.092
911,0.142922,0.109512,0.112135,0.217945,0.217945,0.217945
0.005078,0.003781,0.004961,0.018891,-5.186399,0.362032,0.062364,0.017793,-0.142365,0.079587,0.093348,0.080779,-0.000175,0.047903,0.009186,0.033784,0.043009,0.049168,0.0590
06,0.056533,0.048557,0.049684,0.047368,0.056696,0.028673,0.031784,0.028218,0.024456,0.034406,0.032079,0.038767,0.027557,0.037968,0.043224,0.030938,0.040361,0.030129,0.0330
48,0.075000,0.075000,0.075000,0.018631,0.015716,0.020772,0.009871,18.300803,1.274349,0.442785,0.362265,0.539156,0.294322,0.409190,0.303433,0.014448,0.189287,0.087307,0.157
664,0.263972,0.185237,0.211142,0.208788,0.174313,0.177368,0.170854,0.203405,0.101320,0.127053,0.110943,0.086113,0.121190,0.113851,0.137255,0.101454,0.142346,0.154203,0.108
869,0.152591,0.119255,0.121709,0.263391,0.263391,0.263391
0.006152,0.004487,0.007198,0.021143,-6.864252,0.486763,0.059930,0.013351,-0.177194,0.123438,0.153597,0.110367,0.010807,0.038465,0.015949,0.062402,0.058662,0.074110,0.06536
1,0.063532,0.060456,0.071591,0.061494,0.077854,0.034097,0.040546,0.038784,0.033734,0.044018,0.050050,0.050252,0.034555,0.044315,0.053609,0.044819,0.047947,0.044378,0.04426
7,0.100000,0.100000,0.100000,0.019524,0.016157,0.024587,0.090497,20.671423,1.463016,0.443388,0.363543,0.572783,0.393243,0.545669,0.348473,0.070114,0.200518,0.096344,0.2342
35,0.279088,0.236902,0.213087,0.203556,0.186418,0.219088,0.188772,0.237558,0.106862,0.136308,0.111447,0.101581,0.132773,0.156219,0.151963,0.108709,0.146122,0.164582,0.1360
43,0.157847,0.145880,0.137769,0.300000,0.300000,0.300000
0.007520,0.005145,0.009533,0.022907,-8.568431,0.618676,0.049906,0.012776,-0.219116,0.198667,0.219497,0.136770,0.010136,0.038495,0.025023,0.086264,0.072580,0.089453,0.08938
1,0.008244,0.073427,0.083385,0.072531,0.098558,0.043301,0.049659,0.033263,0.042085,0.057336,0.059586,0.065567,0.045824,0.052254,0.064055,0.058378,0.057658,0.051523,0.06134
3,0.125000,0.125000,0.125000,0.020911,0.016494,0.027991,0.090755,22.741521,1.640314,0.449125,0.363582,0.617873,0.597312,0.668490,0.377878,0.070342,0.200512,0.110472,0.2722
03,0.289503,0.251056,0.254493,0.249221,0.199360,0.227454,0.197555,0.264437,0.116565,0.145188,0.111837,0.111691,0.152885,0.164307,0.175215,0.126455,0.152007,0.173847,0.1564
05,0.166319,0.150459,0.169824,0.330719,0.330719,0.330719
0.008545,0.005739,0.011868,0.026342,-10.349760,0.766093,-0.004138,-0.006842,-0.250086,0.279912,0.265995,0.157068,0.008138,0.043732,0.051756,0.102321,0.072124,0.114201,0.11
2874,0.111515,0.087474,0.099476,0.088549,0.120897,0.052069,0.063397,0.040695,0.048319,0.071899,0.070293,0.076133,0.056277,0.061037,0.072299,0.074898,0.062824,0.062480,0.07
1101,0.150000,0.150000,0.150000,0.021514,0.016724,0.030847,0.092408,24.706326,1.831874,0.566585,0.384321,0.636868,0.762844,0.714694,0.391547,0.071724,0.202167,0.196817,0.2
85269,0.289631,0.287206,0.286517,0.281292,0.213019,0.243208,0.216129,0.291523,0.125801,0.164545,0.119026,0.116042,0.173132,0.173758,0.183491,0.138903,0.158734,0.178374,0.1
82149,0.167652,0.161836,0.177078,0.357071,0.357071,0.357071
0.009570,0.006450,0.014300,0.028796,-12.203424,0.949174,-0.009155,-0.025195,-0.271413,0.383084,0.382047,0.152813,0.033344,0.062865,0.077114,0.125074,0.101385,0.138321,0.13
6685,0.132760,0.104465,0.118468,0.106166,0.132021,0.055226,0.082463,0.058921,0.055798,0.079978,0.080391,0.087904,0.065254,0.068495,0.077255,0.087995,0.077286,0.079290,0.09
0543,0.175000,0.175000,0.175000,0.022053,0.017066,0.033533,0.092977,26.571408,2.093436,0.776016,0.400734,0.642396,0.969180,0.736409,0.394833,0.171793,0.231247,0.247366,0.3
11309,0.336241,0.315703,0.314371,0.303296,0.231653,0.263511,0.235997,0.295165,0.126039,0.197066,0.160123,0.122167,0.177083,0.180967,0.193081,0.146379,0.162650,0.179050,0.1
94693,0.185383,0.187378,0.208169,0.379967,0.379967,0.379967
0.010254,0.006924,0.014689,0.031152,-14.074503,1.143126,-0.172235,-0.044532,-0.271559,0.484840,0.334468,0.141850,0.057752,0.092610,0.098806,0.135476,0.134256,0.161167,0.15
1943,0.147536,0.125766,0.143114,0.127353,0.152234,0.061343,0.097627,0.074028,0.060174,0.084292,0.089221,0.103214,0.083676,0.082894,0.082658,0.099296,0.091734,0.099469,0.11
3724,0.200000,0.200000,0.200000,0.022169,0.017143,0.033455,0.093410,28.229571,2.341232,0.925514,0.417369,0.642335,1.124795,0.750804,0.403747,0.226096,0.290244,0.276036,0.3
13896,0.385399,0.337200,0.322083,0.310774,0.258667,0.295447,0.262113,0.312600,0.129111,0.212844,0.180995,0.123220,0.177184,0.185392,0.209116,0.179604,0.180469,0.179896,0.2
02232,0.200886,0.218613,0.245138,0.400000,0.400000,0.400000

Figure 2.3: A few milliseconds of a reggae song represented by feature vectors

we use for feature extraction. Please see Section 4.1 for a description of our usage of MARSYAS.

2.3 Classification

Automatic Genre Classification of music has received considerable attention in the music information retrieval community for the last decade. Here we will mention some of the key works.

2.3.1 Standard Reference Collection

In order to be able to compare accuracy results in genre classification, it is necessary to run the various algorithms on the same music collection. In this section we will mention two such collections which are widely used in the literature.

The Tzanetakis collection

The Tzanetakis collection was created by George Tzanetakis and first used in (Tzanetakis & Cook, 2002). It consists of 1,000 song excerpts where each excerpt is 30 seconds long. Each song is sampled at 22,050 KHz in mono. The songs are evenly distributed into the following 10 genres: Blues, Classical, Country, Disco, Hip Hop, Jazz, Metal, Pop, Reggae and Rock. We use the Tzanetakis collection for our initial experiments.

Reference	Accuracy
Panagakis and Kotropoulos (2010)	93.7%
Panagakis et al. (2009)	91.0%
Bergstra et al. (2006)	82.5%
Li et al. (2003)	78.5%
Panagakis, Benetos, and Kotropoulos (2008)	78.2%
Lidy et al. (2007)	76.8%
Benetos and Kotropoulos (2008)	75.0%
Holzapfel and Stylianou (2008)	74.0%
Tzanetakis and Cook (2002)	61.0%

Table 2.1: Classification results of the Tzanetakis collection in the literature

The ISMIR 2004 collection

The ISMIR2004 collection is another reference collection, which consists of 1,458 full songs using the following 6 genre classes: Classical (640), Electronic (229), Jazz_Blues (52) , Metal_Punk (90), Rock_Pop (203) and World (244). The set was first used at the *ISMIR2004 Audio Description Contest - Genre/ Artist ID Classification and Artist Similarity*

2.3.2 Genre Classification Results

The evolution of classification accuracy for the Tzanetakis collection is summarized in Table 2.1 (partly reproduced from Panagakis et al. (2009)).

The ground-breaking work of Tzanetakis and Cook (2002) is particularly well known. They classified the 1,000 songs based on timbral features, rhythmic content and pitch content. The timbral features included means and averages of Spectral Centroid, Spectral Rolloff, Spectral Flux, Time Domain Zero Crossings, five MFCC coefficients and a calculation of texture windows where the average RMS energy is less than the average RMS energy of windows. The rhythmic content features included the beat per minute of the first and second peak of the files's beat histogram, their relative amplitude and the overall histogram sum, which indicates the beat strength. Pitch features include the most prominent pitch class (which indicates the tonic chord of the song), complexity of harmonic changes, the octave range, the pitch interval between the two most prominent pitches and the overall pitch detection strenth. One 30-dimensional feature vector was extracted for each of the song excerpts. Classification was performed using a simple Gaussian classifier, the Gaussian mixture model and k-NN classification with 1, 3 and 5 neighbors.

Classifier	Accuracy
Real time	44%
Simple Gaussian	59%
Gaussian Mixture model	61%
3-nearest neighbor	60%

Table 2.2: Classification accuracy of Tzanetakis & Cook (2002)

Table 2.2 shows the best accuracy received for each of the classifiers used. Note that Real time only uses the timbral features.

Genre confusion matrices show that the misclassifications of the system are in many ways what can be expected. Classical music with strong rhythm is misclassified as jazz, and rock music has only 40% accuracy which is due to its broad nature.

Li et al. (2003) is heavily influenced by Tsanetakis and Cook. They note, however, that Tsanetakis and Cook only managed to achieve 61% accuracy of classification on the 10 musical genres, and raise the question whether there are better features and classifiers that can be used in order to achieve higher accuracy.

Their research includes all the timbral features, rhythmic content features and pitch content features but add *Daubechies Wavelet Coefficient Histograms (DWCH)*.

It is noted that audio is distinguished by the variation in amplitude and that identifying these variation is therefore vital when classifying music. A wavelet decomposition of the sound file is suggested since the frequency spectrum of music is divided into octaves with each octave having different characteristics. Each wavelet subband gives information about the sound characteristics (timbre) at that particular time and a histogram of these wavelet coefficients therefore gives good indication of the waveform variations over time. It can therefore be said that wavelet coefficient histogram provide both global and local information about the audio file. The average, the variance, the skewness and the energy of each subband is calculated along with the traditional timbral features.

Multi-class classifiers used included the Gaussian mixture model and k-NN in addition to Support Vector Machines and Linear Discriminant Analysis. Support Vector Machines (SVM) were extended to multi-class classification using one-versus-the-rest, pairwise comparison and multi-class objective functions.

Table 2.3 shows the results. SVM1 and SVM2 represent pairwise SVM and one-versus-the-rest SVM respectively. It can be seen that all classifiers achieved higher accuracy with Daubechies Wavelet Coefficient Histograms than when using only the timbral, rhythmic and pitch features.

Features	Methods					
	SVM1	SVM2	MPSVM	GMM	LDA	KNN
DWCHs	74.9	78.5	68.3	63.5	71.3	62.1
Timbral+Rhythmic+Pitch	70.8	71.9	66.2	61.4	69.4	61.3

Table 2.3: Classification accuracy of Li et al., (2003)

We can see that DWCHs always outperform the timbral, rhytmic and pitch features, regardless of which classifier is used. An evaluation of the effect of each of the timbral, rhythmic and pitch features was conducted. It was found that the accuracy when using the timbral features was considerably higher than when using the rhythmic or pitch features.

Bergstra et al. (2006) note that it is better to classify features aggregated over segments (a collection of frames) of audio, rather than using one feature vector per song, or to classify individual feature vectors. They focus on aggregating the frame level features into segments suitable for classification and experiment with the size of these segments. Their software won the first prize in the genre classification part of the MIREX 2005 international contest in music information extraction. The genre classification task used two collections, with approximately 1,500 full length songs in each collection. They use the following features in their experiments: The 32 lowest Fast Fourier transform coefficients (FFTCs), 256 Real Cepstral Coefficients (RCEPS), 64 Mel-frequency cepstral coefficients (MFCC), Zero-crossing rate, Spectral spread, Spectral centroid, Spectral rolloff and 32 Autoregression coefficients (LPC). Features were extracted in frames of 1,024 samples and m such frames were aggregated into segments before using AdaBoost for classification. This approach yielded 82.3% classification accuracy on MIREX 2005 using $m = 300$, which corresponds to each segment being 13.9 seconds. They achieved 82.5% accuracy on the Tzanetakis collection using the same settings. They state that the best results are achieved by using values of m between 50 and 100, but that this would not have allowed them to complete the task in 24 hours, which was a requirement of the contest.

The highest accuracy results for the Tzanetakis dataset were achieved by Panagakis et al (Panagakis et al., 2009; Panagakis & Kotropoulos, 2010) who focus on how audio information is encoded in the human auditory system, and extracted auditory temporal modulation representation of each song. In (Panagakis et al., 2009) they achieve 91% accuracy by reducing the dimensionality of the above mentioned representation using linear subspace dimensionality reduction techniques before applying sparse representation-based classification (SRC). Several other classifiers were also examined, including support vector machines and k-NN. Their results using k-NN were below 70%. In (Panagakis &

Kotropoulos, 2010) the authors improve their accuracy and achieve 93.7%. As before they use SRC for classification, but this time they use Topology Preserving Non-Negative Matrix Factorization (TPNMF) for dimensionality reduction before applying the sparse representation-based classification.

What these methods generally have in common, is that they use one vector, or a few, to represent each song. It is well known from the image retrieval community that this is insufficient for image recognition (e.g., see Lowe (2004)), and we believe that further improvements can possibly be made by using multiple descriptors similarly to what Boiman, Shechtman, and Irani (2008) show for image classification. Furthermore, classification time is very rarely reported. The only reference that mentions the time necessary for classification is Bergstra et al. (2006), who mention that they could not use the settings they would like if they were to finish the task in under 24 hours.

In our experiments we achieve up to 81.1% accuracy on the Tzanetakis dataset using an approximate k-NN classifier applied to many feature vectors per song (1292 exactly). We use feature vectors that are similar to those used by Tzanetakis and Cook (2002) and Li et al. (2003); this accuracy is thus the best accuracy reported using these features. We obtain this in a matter of minutes. We also apply the classifier to a collection of more than 30,000 songs. To the best of our knowledge, automatic genre classification of this scale has not been reported in the literature before.

2.4 Summary

We have seen that representing music for MIR work is mostly done in three ways. Symbolic representation uses symbols to represent a given song so that either a computer or a human can recreate the song. Audio representation is an actual recording of a certain performance of a song, while feature representation is based on high-dimentional feature vectors describing particular features of an audio file. Feature vectors are typically used by automatic genre classification systems. We have presented details about some genre classification methods and we have seen that genre classification results for the Tzanetakis collection are in the range of 61% – 93.7%.

In part one of this thesis we describe our k-NN genre classification system and our results from using our system to classify the Tzanetakis collection. Part two focuses on a real-world song collection of over 30,000 songs and our genre classification experiments with this collection.

Part I

Approximate k-NN Genre Classification

Since we want to show that given a large number of audio descriptors k-NN classification can be competitive to other more complex classifiers, we must start by using this classifier on a reference collection. In this part we therefore describe our experiments where we use an approximate k-NN classifier to classify the Tzanetakis collection.

Chapter 3 discusses our k-NN classifiers in detail. In Chapter 4 we describe our experimental setup, including our feature extraction process. Chapter 5 gives detailed information about our classification results. We conclude this part by Chapter 6 which describes a human classification experiment we performed on the Tzanetakis collection in order to get more information about the ground-truth itself.

Chapter 3

Classifiers

k-NN classifier is a supervised learning algorithm where each feature vector from a reference collection is treated as a point in multi-dimensional feature space and k-NN retrieval is subsequently used to classify the unclassified feature vectors.

In this chapter we first motivate why k-NN classification is a promising method in Section 3.1. We then outline the overall classification process in Section 3.2. The chapter concludes with Section 3.3 describing our two k-NN classifiers: an accurate classifier based on a sequential scan, and an approximate classifier based on clustering.

3.1 Motivation

Classifying songs can be cast into k-NN query processing for large collections of high-dimensional vectors. Such k-NN classification has two major advantages.

First, it has already been demonstrated, albeit within the context of content-based image and video retrieval, that approximate k-NN algorithms can cope very successfully with scale. Approaches such as the NV-Tree (Lejsek, Ásmundsson, Jónsson, & Amsaleg, 2009), Locality Sensitive Hashing (LSH) (Andoni & Indyk, 2008), cluster pruning (Gudmundsson, Jónsson, & Amsaleg, 2010) and visual word approaches (Philbin, Chum, Isard, Sivic, & Zisserman, 2007) all return high-quality results when indexing hundreds of millions of high-dimensional descriptors. For moderate collections, a carefully implemented exhaustive sequential scan can even perform quite well.

Second, k-NN classification has the added advantage of handling fuzzy class assignments very well, as different near neighbors can give weights to different classes. This is partic-

ularly important in browsing environments, where these weights can be used for ranking songs.

While k-NN music classification has been studied in the past, the comparison has typically been tilted in favor of its competitors. In these comparisons, a given low-level descriptor setup has been used for several classifiers, and their accuracy reported. The time required for classification has largely been ignored, however, although many of the classification methods are quite time-consuming. The conclusion has invariably been that the quality of k-NN classification is lower than that of its competitors (Tzanetakis & Cook, 2002; Li et al., 2003; Panagakis et al., 2009; Bischoff et al., 2009).

If the classification time is the main constraint, however, then the results may very well be different. Since k-NN classification is very efficient and scalable, it can be applied to more descriptors and more songs. By using more descriptors, quality is improved, making k-NN classification competitive. This latter comparison is in fact more relevant in the real world, where typical music collections are one or two orders of magnitude larger than the ones typically used in the literature. We therefore examine k-NN genre classification on a real-life music collection

3.2 The Classification Process

In a supervised genre classification process there are two sets of songs. The first set, called the *training set*, has a solid set of genre labels, which are used to guide the classification process. The features extracted from the training set are typically pre-processed in some manner. For SVM classification, e.g., the SVM is applied to the set to learn the classifications, while for k-NN classification, the training set may be indexed to facilitate the subsequent classification.

The second set is the set to be classified, or the *classification set*. For each song in this set, the classification process is applied to all features of the song, resulting in a genre assignment. If each song is represented by a single feature, the genre of the feature is automatically applied to the song. If each song is represented by multiple features, however, the genres of the features are aggregated into a genre assignment for the song, e.g., through a voting process, where the genre assigned to the most features is assigned to the song.

Of course, the songs in the classification set typically also have a robust set of genre assignments, which is called the *ground-truth*. The ground-truth allows for easy assessment of the classification accuracy; each song either agrees with the ground-truth, in which case

the classifier is said to correctly classify the song, or it does not agree, in which case classifier is said to incorrectly classify the song. The classification set is indeed sometimes called the *ground-truth set*.

A common experimental methodology is the *10-fold cross-validation*, where 90% of a collection is used as a training set for the remaining 10%; by performing 10 such classifications with distinct ground-truth sets, each song is classified exactly once. To avoid the order of songs in a collection impacting results, the preferred methodology is *randomized and stratified* 10-fold cross-validation, where each training set is composed randomly, but in such a manner that all genre classes are represented equally in each training set.

3.3 k-NN Classifiers

In k-NN classification, each query feature is assigned the genre of its k nearest neighbors. Distance between points can be calculated using any standard distance calculation such as Euclidean Distance, Manhattan Distance or Hamming Distance. When $k = 1$, the classification is based on the one nearest neighbor, but when $k > 1$ the votes of all k nearest neighbors have equal weight. The number of high-dimensional feature vectors can be hundreds or thousands per song, resulting in a large number of votes per song.

To give an example, assume that each song is represented by 1,200 descriptors, and that each feature retrieves $k = 3$ nearest neighbors. Then each song will retrieve 3,600 similar features from songs in the training set, where each feature votes for the genre of the song it came from. These 3,600 votes are then aggregated and the genre with the most votes is assigned to the song to be classified.

3.3.1 Sequential Scan

The *sequential scan* is the most straight-forward method for k-NN classification. As the name suggests, this classifier sequentially scans the collection of features from the training set and computes the distance to each feature of the song to be classified. The advantage of a sequential scan is that no training phase is required; the disadvantage is that scanning the training set features and computing distances takes time. Nevertheless, no k-NN classifier exists that is more efficient, while *guaranteeing* the same "accurate" results as a sequential scan would give.

The feature vectors typically have a dimensionality of at least 30, and it is not uncommon for the dimensionality to be much higher. In our experiments we use between 34 and

154 dimensions. Say that our example collection mentioned above has a dimensionality of 80. In this case a sequential scan algorithm makes 1,440,000,000,000 comparisons of 80 dimensional vectors. Using standard distance calculations where each dimension is compared in one instruction this means a total of 115,200,000,000,000 comparison instructions.

Our implementation uses Manhattan Distance since this allows the use of very efficient intrinsic SSE instructions. Using SSE, we can calculate the distance between 16 dimensions in one instruction, thus reducing CPU time considerably. In order to do this we convert the text file with floating point descriptors from MARSYAS to a binary file with unsigned char descriptors. Our experiments indicate no loss of classification accuracy due to this conversion.

3.3.2 Cluster Pruning

As described in Section 3.1, various approximate query processing strategies exist for high-dimensional descriptors. These strategies do not guarantee that they return the same results as a sequential scan. Instead, they attempt to return very good "approximate" results, in a very short amount of time. The approximate classifier that we use in this thesis is called Cluster Pruning (Chierichetti et al., 2007; Gudmundsson et al., 2010), and is based on grouping all feature vectors from the training set into clusters of similar features.

More precisely, the Cluster Pruning algorithm starts by randomly selecting C cluster leaders from the training set feature collection. The number of clusters is typically chosen such that the average cluster fits with one disk read. The features in the training set collection are then read, one by one, and assigned to the closest cluster leader. For large feature collections, where there are many cluster leaders, they are organized into a hierarchy for more efficient processing.

During k-NN retrieval, the features from the song to be classified are considered one by one. For each feature, the b closest cluster leaders are identified, again through the hierarchy, and the clusters they represent are retrieved and scanned for the k approximate nearest neighbors.

Experience from different application domains has shown that $b = 1$ often gives excellent results. If the training set feature collection is divided into C clusters, then the time required for k-NN retrieval is roughly b/C times the time required for a sequential scan.

Note that the clusters formed by Cluster Pruning do not correspond directly to genres. Each cluster may have features from many genres, and each genre is likely to be found in many clusters.

3.4 Summary

We have in this chapter described the k-nearest neighbor classification process. k-NN is a supervised learning algorithm where each feature vector from a reference collection is treated as a point in multi-dimensional feature space and k-NN retrieval is subsequently used to classify the unclassified feature vectors. Accurate classification is performed by sequentially scanning all descriptors from the training set and calculating distance to all descriptors from the song to be classified. By using the Cluster Pruning algorithm we can perform approximate classification which does not guarantee the same results, but achieves very good "approximate" results, in a very short amount of time.

Chapter 4

Experimental Setup

This chapter starts with a description on how we use the MARSYAS framework for feature extraction. We then provide information on our evaluation procedure, before concluding with details of the hardware used.

4.1 Feature Extraction

We use MARSYAS's *bextract* module to extract audio descriptors. See Section 5.1.3 for details about the effect of various parameters on classification accuracy.

MARSYAS (**M**usic **A**nalysis, **R**etrieval and **Sy**nthesis for **A**udio **S**ignals) is an open source software framework for audio analysis and synthesis with specific emphasis on building Music Information Retrieval systems. The system was first introduced by George Tzanetakis and Perry Cook in 1999 (Tzanetakis & Cook, 1999) and used in their influential paper, Musical Genre Classification of Audio Signals (Tzanetakis & Cook, 2002). The framework, which can be downloaded from `http://marsyas.info/` (accessed in october 2010) includes a variety of building blocks for performing common audio tasks. Some representative examples are: soundfile IO, audio IO, feature extraction, signal processing and machine learning.

In this project we only use the feature extraction module *bextract* version 0.3.3. It produces, by default, the following timbral based features:

Time Domain Zero Crossings. Measures the noisiness of the sound by computing the number of sign changes in the time domain.

Spectral Centroid. Measures the shape or brightness of the audiofile by calculating the weighted average frequency of every time frame.

Spectral Flux. The squared change in normalized amplitude between two consecutive time frames and therefore measures how much the sound changes between frames.

Spectral Rolloff. The frequency below which 85% of the energy distribution is achieved.

Mel-Frequency Cepstral Coefficents (MFCC). These are perceptually motivated features which have traditionally been used in speech recognition. MARSYAS stores 13 coefficients.

The following features can also be extracted:

Spectral Flatness Measure (SFM). Measures the noisiness of an audio signal. High values indicate high noise levels where all frequencies have similar amount of power, while low values indicate "cleaner" sound.

Spectral Crest Factor (SCF). Another measure of noisiness, closely related to SFM.

Line Spectral Pair (LSP). Represents a compressed version of the signal by two polynomials with alternating roots.

Linear Prediction Cepstral Coefficients (LPCC). This feature represents the evolution of the signal by using a linear combination of recent samples.

CHROMA. A pitch based feature that projects the frequency spectrum into 12 bins, with one bin for each of the 12 distinct pitches of the chromatic musical scale.

In order to extract features, Short Time Fourier Transform (STFT) is used to break the audio signal into short *analysis windows* or time frames, which can possibly overlap. Both the size of the analysis window, w, and the hop size, h, can be changed from the default of 512 samples. The actual features are then calculated as running means and variances over several such analysis windows. This forms another window, called *texture window*, which can be thought of as a memory of the last m descriptors (by default, $m = 40$).

For a detailed explanation of STFT and MFCC please see Miranda (2002) and Logan (2000) respectively. For more in-depth information about MARSYAS timbral features please see Tzanetakis and Cook (2002).

It is worth noting that the *bextract* module looks at the collection of audio files to be analyzed as one big audio file. This has the effect that texture windows can cross the

boundaries of audio files, resulting in the first $m-1$ descriptors of audio file n containing information from audio file $n-1$. This may affect the recall scores, either positively or negatively, and these descriptors should be ignored.

The output of this step is a text file with floating point descriptors. The number of descriptors for each song depends on the song file length and two MARSYAS parameters, *window size* and *hop size*. For most of our experiments with the Tzanetakis collection we use 1,292 descriptors for each song excerpt, so each descriptor describes 23.2 milliseconds of audio.

4.2 Evaluation

We measure both classification accuracy and classification time.

When measuring classification accuracy we used randomized and stratified 10-fold cross-validation. Stratified randomization is implemented by shuffling the 100 songs within each of the 10 genres, and then concatenating the genre files. 10-fold cross validation is implemented by ignoring, when computing distances, all descriptors from song that are located within the same 10% of the collection as the query.

Classification time is measured using wall clock time.

4.3 Hardware

All experiments reported in this part were performed on an Apple MacBook Pro machine with 2.66GHz Duo Core Intel processors, 4GB of main memory and one 300GB 5.4Krpm Serial ATA hard disk, running Mac OSX 10.5.8.

Chapter 5

Classification Results

In this chapter we report on our classification experiments on the Tzanetakis collection. In Section 5.1 we analyze various aspects of the descriptor configuration, and achieve baseline accuracy measures using sequential scan. Once sequential scan figures are reported, we use the cluster pruning indexing method to reduce the classification time considerably in Section 5.2

5.1 Impact of Feature Extraction Parameters

Recall from Section 4.1 that extracting features Marsyas breaks the audio into analysis windows, but the actual features are calculated as running means and variances over texture windows which are several analysis windows. In this section we will study the impact of various parameters such as the size of the analysis window (w), the hop size between analysis windows (h) the number of analysis windows in one texture window, or memory size (m) along with the number of neighbors retrieved (k). All experiments in this section use the sequential scan classifier.

Table 5.1 shows the parameters studied and their default values. Before looking at these parameters, however, we will compare our k-NN classification to the MARSYAS SVM classifier.

5.1.1 Comparison to the MARSYAS SVM Classifier

MARSYAS includes the machine learning module kea. In this section we compare our k-NN classifier to the SVM classifier of kea. Detailed runtime analysis is outside the scope

Parameter	Abbreviation	Default Value
Window Size	w	512
Hop Size	h	512
Memory Size	m	40
Neighbors Retrieved	k	1

Table 5.1: Parameters of the performance study on the Tzanetakis collection

of this paper, and there are more efficient SVM classifiers available elsewhere. Therefore it is important not to generalize these results to all SVM classifiers. The results only give an indication of the relative efficiency of our solution to the kea SVM classifier.

Note that this experiment is different from others in this chapter in three ways. First, it is not randomized and the first m descriptors are not ignored, leading to artificially high accuracy scores for the parameters used. Second, in order to run the SVM classifier in a reasonable time, we set $w = h = 2,048$, in order to reduce the number of descriptors. Third, the accuracy of SVM is reported in a per-descriptor basis, making the results difficult to compare. The main point of this section, however, is the efficiency.

Table 5.2 shows the accuracy obtained with each of the classifiers, and the total time required to train the classifier and classify the 1,000 song excerpts. As the table shows, our simple k-NN classifier using sequential scan achieves better accuracy than the SVM classifier of MARSYAS, using only a fraction of the time.

Note again that in this experiment, the parameters were modified to give a relatively small set of descriptors, such that the kea SVM package would complete in a reasonable time.

5.1.2 The Effect of Window Size, Hop Size and Memory

Starting with the effect of the window size, w, Table 5.3 shows the impact on accuracy and classification time. We see that accuracy increases as window size gets smaller, until we go from 512 to 256, indicating that 512 (the default value) is indeed the ideal window size. Table 5.3 furthermore shows the effect on the number of descriptors, and on the running time of the k-NN classification. As expected, the number of descriptors doubles

Classifier	Accuracy	Time
kea **SVM**	67.5%	1,078 min
k**-NN**	78.1%	10 min

Table 5.2: Comparison of k-NN and MARSYAS SVM classification (timbral features, $w = h = 2,048$)

Window Size w	**Accuracy**	**Descriptors**	**Time**
256	77.5%	2,586,641	850.5 min
512	80.4%	1,293,335	207.4 min
1,024	78.6%	647,000	60.5 min
2,048	76.3%	323,834	14.1 min
4,096	72.1%	162,082	3.3 min

Table 5.3: Effect of window size w (timbral feat. + SFM)

when the window size is halved. Running time, on the other hand, is roughly quadrupled, as both the size of the descriptor collection and the number of query descriptors are doubled.

We also experimented a little with the hop size, h, which impacts the overlap between descriptors, but this did not improve our results. For example using $w = 512$ and $h = 256$ yielded 80.2% accuracy.

Having reached our best accuracy with $w = h = 512$ samples, using timbral features along with SFM, we next experimented with the memory size. Table 5.4 shows that we improve our accuracy neither by increasing nor decreasing m. Again, the default value is optimal. Note that in this experiment the value of k is 3.

When not taking into account the fact that MARSYAS treats a collection of sound files as one large file it is easy to get artificially high accuracy measurements with large values for window size and *m*.

We can see from Table 5.5 that for a small query set of 50 songs with equal distribution of genres, we managed to get every song correctly classified by using a window size = hop size = 32,768. This is done using just the timbral features. When we ran our algorithm on this collection with all songs as queries we managed to achieve 98.5% accuracy, and by increasing *m* to 90 we managed to achieve 99.2% accuracy. It is easy to see why. We have songs that are 30 seconds long sampled at 22,050 KHz, yielding a total of 661,500 samples for each song. With the size of the analysis window as 32,768 and each texture window being 90 analysis window we see that each texture window is 2,949,120 samples or almost 4.5 songs. When looking at vote distribution we see that each song always gets votes from the songs adjacent to itself, *because they store partly the same information*.

Memory Size m	**Accuracy**
30	80.4%
40	80.8%
50	79.2%

Table 5.4: Effect of memory size m (timbral feat. + SFM)

Window Size	2,048	4,096	8,192	16,384	32,768
Accuracy	78%	86%	94%	96%	100%

Table 5.5: Effect of varying the window size when all descriptors are used

When seeing which songs did not get "correctly" classified this way we noticed that all of them were first or last songs of a particular genre. Again, this is understandable since they also get votes from adjacent songs, which happen to be in a different genre.

It is therefore very important for all researchers using MARSYAS for feature extraction to be aware of this and take measures to avoid artificially high accuracy scores. We do this by always ignoring the first *m* descriptors of each song.

5.1.3 The Effect of Feature Selection

For this experiment we started by extracting the default timbral features. We then experimented with adding features; the results are summarized in Table 5.6. The table shows that adding Spectral Flatness Measure (SFM) increases the accuracy for this test set from 75.4% to 80.4%. Adding further information to the feature vector, however, did not improve the results, and in fact we observed that it actually hurts the results slightly. We conclude that in addition to the default timbral features it is beneficial to include Spectral Flatness Measure to achieve best results.

5.1.4 The Effect of Neighbors Retrieved

Finally, we experiment with the value the k parameter, which indicates how many neighbors from the collection are considered for each query descriptor. We ran the k-NN search for values of k ranging from 1 to 10. Table 5.7 shows the results. As the table shows, vary-

Features	**Dimensions**	**Accuracy**
Timbral features (TF)	34	75.4%
TF + SFM	82	80.4%
TF + SFM + SCF	130	80.0%
TF + SFM + LSP	118	80.0%
TF + SFM + LPCC	106	79.8%
TF + SFM + CHROMA	106	79.4%
TF + SFM + SCF + LSP	166	79.8%
TF + SFM + SCF + LPCC	154	79.4%
TF + SFM + SCF + CHROMA	154	79.9%

Table 5.6: Effect of feature selection on accuracy

Neighbors k	Accuracy	Time
1	80.4%	207.4 min
2	80.7%	208.3 min
3	80.8%	209.3 min
4	80.6%	210.1 min
5	80.5%	212.3 min
10	80.5%	217.5 min

Table 5.7: Effect of varying k (timbral features + SFM)

ing k does not affect the classification accuracy much, nor does it have any significant impact on the classification time. Changing the accuracy 80.4% to 80.8% in a collection of 1,000 songs only means that four more songs are found.

5.1.5 Result Analysis

Table 5.8 shows the "confusion" matrix for our experiment of Table 5.7, for $k = 3$, where we managed 80.8% accuracy. Classic is classified with 100% accuracy and Blues, Jazz, and Metal all have over 90% accuracy. These are all well defined genres. We see that in most cases the mistakes the program does are similar to those humans commonly make, and in general these results agree with others reported in the literature. The table shows that 3 blues songs are classified as jazz, 7 country songs are classified as pop (which is a very fuzzy classification genre), 10 disco songs are classified as pop, 8 jazz songs are classified as classic, and 7 reggae songs are classified as pop. Just as in (Tzanetakis & Cook, 2002) the lowest accuracy is for the rock genre. We were, however, surprised to see how many rock songs were classified as disco songs. Country also receives a low accuracy score, and it is interesting to note how few songs are also wrongly classified as either rock or country. This indicates that these genres are very broad in nature.

	Blues	Classic	Country	Disco	Hiphop	Jazz	Metal	Pop	Reggae	Rock
Blues	**93**	0	0	3	0	3	0	0	0	1
Classic	0	**100**	0	0	0	0	0	0	0	0
Country	4	7	**66**	11	0	1	0	7	2	2
Disco	1	1	2	**80**	3	0	0	10	1	2
Hiphop	2	0	1	0	**80**	0	2	8	5	2
Jazz	0	8	0	0	0	**91**	0	1	0	0
Metal	2	0	0	1	1	0	**92**	0	0	4
Pop	1	3	3	2	1	0	1	**87**	1	1
Reggae	1	1	1	8	5	0	0	7	**75**	2
Rock	3	7	3	20	4	3	5	7	4	**44**
Total	107	127	76	125	94	98	100	126	88	58

Table 5.8: Confusion matrix (timbral features + SFM, $k = 3$)

Place	Percentage	Aggregated
1	80.8%	80.8%
2	9.2%	90.0%
3	4.6%	94.6%
4	2.8%	97.4%
5	1.6%	99.0%
6	0.4%	99.4%
7	0.2%	99.6%
8	0.2%	99.8%
9	0.0%	99.8%
10	0.2%	100.0%

Table 5.9: Placement of the correct genre

Since k-NN classification is based on aggregating votes from each of the local audio descriptors we are able to not only see which genre each song is classified to, but also which genre comes in second place and so on. Table 5.9 shows us the order when using timbral features + SFM with $k = 3$.

We see that 90% of songs have the correct genre come in 1st or 2nd place while 99% of songs have the correct genre in one of the top 5 places. Please note that the two songs where the correct genre is placed last actually received no votes from the correct genre. As discussed in Chapter 6, this is due to the fact that the song *"Tie Your Mother Down"* by Queen appears twice in the collection, each time with different genre classification.

5.2 Impact of Approximate Query Processing

The main reason for us to examine k-NN music classification is that we want to be able to classify songs very quickly, in order for the solution to scale to real-life song collections. Therefore, having reached a very respectable result of 80.8% accuracy on the Tzanetakis collection in 3.5 hours using sequential scan, we now turn our attention to reducing the classification time by employing the Cluster Pruning high-dimensional indexing method, which already has proved to be beneficial for text retrieval (Chierichetti et al., 2007) and image copyright protection (Gudmundsson et al., 2010).

As mentioned in 3.3.2 there are a few parameters both when clustering the database, and when searching the clusters. We will now present a study we did on the effect of these parameters on both search time and classification accuracy. The database used was the same that yielded 80.8% accuracy using sequential scan.

Table 5.10 shows the parameters studied and their default values.

Parameter	Abbreviation	Default Value
Clusters Created	C	100
Neighbors Retrieved	k	1
Clusters Searched (for each descriptor)	b	1

Table 5.10: Parameters of the performance study of Cluster Pruning

5.2.1 The Effect of Clusters Created

Table 5.11 shows the effect of varying the number of clusters created. The first thing we notice is that we are now able to achieve comparable accuracy results with sequential scan at only fraction of the time. We report both the clustering time and the classification time, which is the time for searching the clusters. We note that clustering time is largely independent from number of clusters created but classification time is reduced as the number of clusters increase, since more clusters means smaller clusters, which in turn means fewer distance calculations.

Our best results of 80.6% accuracy are achieved using $C = 100$. Here the accuracy is only 0.2% lower than when using sequential scan, but search time less than 8 minutes, compared with 209 minutes.

It is somewhat surprising that our lowest accuracy of 77.1% was achieved using $C = 750$. By increasing C, we get more clusters and the clusters are smaller. We therefore expected to see the accuracy drop more as C got larger, but instead we see that all C values higher than 750 result in higher accuracy, and the difference in accuracy between $C = 50$ and $C = 5,000$ is actually only 0.3%.

C	avg. # of desc.	clust. time	classification time	accuracy
50	25.867	7 sec	13 min 04 sec	80.0%
75	17.244	7 sec	8 min 45 sec	80.0%
100	12.933	7 sec	7 min 50 sec	80.6%
125	10.347	7 sec	5 min 40 sec	79.4%
250	5.173	7 sec	3 min 22 sec	79.1%
500	2.587	7 sec	1 min 54 sec	80.2%
750	1.724	8 sec	1 min 01 sec	77.1%
1,000	1.293	7 sec	55 sec	78.4%
1,250	1.035	7 sec	48 sec	79.2%
1,500	862	8 sec	42 sec	78.8%
1,750	739	8 sec	35 sec	78.8%
2,000	647	8 sec	30 sec	78.9%
3,000	431	8 sec	21 sec	78.9%
4,000	323	8 sec	17 sec	78.7%
5,000	259	9 sec	15 sec	79.7%

Table 5.11: Effect of varying number of clusters C

k	**Classification Time**	**Accuracy**
1	8 min 12 sec	80.6%
3	8 min 12 sec	80.2%
5	8 min 13 sec	80.0%
7	8 min 14 sec	80.0%
9	8 min 16 sec	79.9%

Table 5.12: Effect of varying neighbors retrieved k

While the difference between our highest and lowest accuracy is 3.5% it is clear that factors such as the random choice of cluster representatives and how close the votes of 1st and 2nd genres are causing the relationship between C and accuracy to be non-linear.

5.2.2 The Effect of Neighbors Retrieved

Table 5.12 shows the effect of varying the neighbors retrieved. We see that increasing k from the default value of 1 does not improve accuracy. This experiment was done using $C = 100$, but we also performed the same experiment using $C = 500$, $C = 1,250$, and $C = 5,000$ and never did accuracy improve by increasing k. This differs from Table 5.7 where we managed to increase accuracy from 80.4% to 80.8% by increasing k from 1 to 3 when using the sequential scan.

5.2.3 The Effect of Clusters Searched

Figure 5.1 shows the effect of varying the number of clusters examined at classification time. When b is increased we search more data, so it does not come as a surprise that we see an increase in accuracy when we increase b. However, it is quite interesting to note than with b values of 2, 3 and 4 we actually achieve higher accuracy than when doing sequential scan. This is due to the approximate nature of the cluster pruning algorithm. As with the k parameter experiment, we also ran this experiment with C values of 500, 1,250 and 5,000 and in all cases saw an increase in accuracy with higher b values.

5.2.4 Approximate Query Summary

We have seen that using the cluster pruning indexing technique we are able to achieve similar accuracy as sequential scan, but using only a fraction of the time. We have experimented with the parameters C, k, and b and seen that C has some effect on accuracy, and

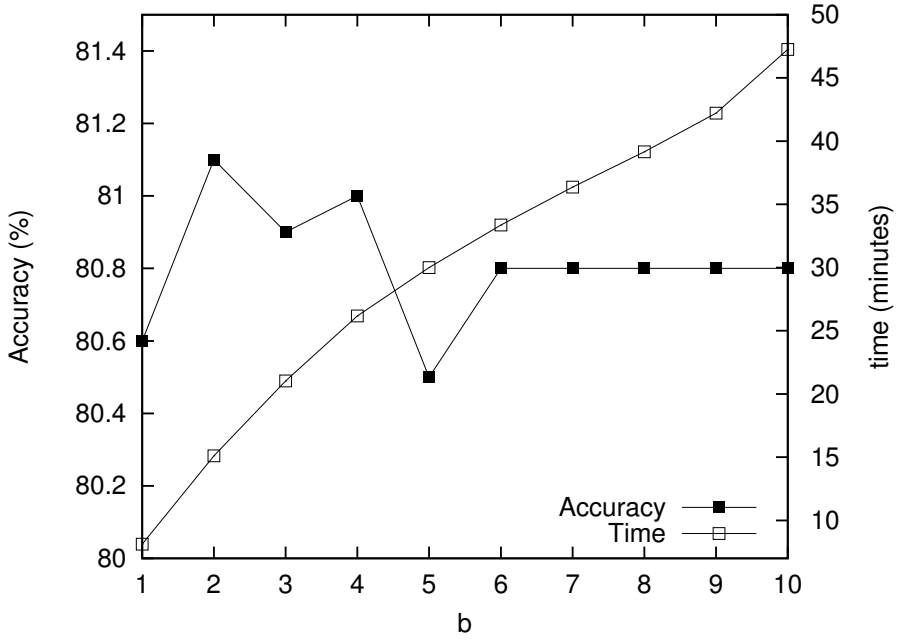

Figure 5.1: Effect of varying number of clusters, b, examined at classification time

much effect on search time, increasing k from the default 1 has not proved to be beneficial, but increasing b has yielded our best results for this collection, or 81.1% accuracy.

Table 5.13 reports a "skyline" of the results from our cluster pruning experiments, where we report results that are either faster than other classification experiments of identical quality, or give better classification accuracy than other classification experiments that take similar time. Note that the classification time does not include the clustering time.

Table 5.14 shows the confusion matrix for our cluster pruning experiment with values $C = 100$, $k = 1$ and $b = 2$. When comparing this to Table 5.8 we see that in most classification of specific genres is either the same or only 1% higher or lower. It is only reggae that goes up by 2%, from 75 to 77.

Classification Time	**Accuracy**	C	k	b
15 sec	79.7%	5,000	1	1
1 min 54 sec	80.2%	500	1	1
2 min 59 sec	80.4%	500	1	2
5 min 15 sec	80.5%	1,250	1	15
6 min 45 sec	80.6%	1,250	1	20
15 min 11 sec	81.1%	100	1	2

Table 5.13: Summary of approximate k-NN results

	Blues	**Classic**	**Country**	**Disco**	**Hiphop**	**Jazz**	**Metal**	**Pop**	**Reggae**	**Rock**
Blues	**94**	0	0	2	0	3	0	0	1	0
Classic	0	**100**	0	0	0	0	0	0	0	0
Country	2	8	**67**	09	0	2	0	7	1	4
Disco	1	2	2	**79**	4	0	0	9	1	2
Hiphop	3	0	0	0	**79**	0	2	6	8	2
Jazz	0	8	0	0	0	**91**	0	1	0	0
Metal	1	0	0	0	1	0	**92**	1	0	5
Pop	1	3	3	3	1	0	1	**87**	1	0
Reggae	0	1	2	6	5	0	0	7	**77**	2
Rock	4	6	2	19	4	3	6	7	4	**45**
Total	106	128	76	118	94	99	101	125	93	60

Table 5.14: Confusion matrix for best Cluster Pruning results

Table 5.15 displays the placement of correct genres when using cluster pruning, and when comparing this to Table 5.9 we see again that the difference is minimal.

By indexing our database of audio descriptors using cluster pruning we can achieve comparable accuracy results as with sequential scan but at only fraction of the time. Our best results of 81.1% is achieved with a time reduction by a factor of 15 compared with sequential scan, while we achieved 79.7% accuracy at a time reduction by a factor of 837. We believe we got lucky with that particular choice of C, but this nevertheless proves to us that cluster pruning is a very viable method for k-NN music classification.

5.3 Summary

We have shown our classification results on the Tzanetakis collection using two versions of k-NN classification. We have seen that although it does not achieve results comparable with the best, they are nevertheless quite respectable. In fact, we believe our classification

Place	**Percentage**	**Aggregated**
1	81.1%	81.1%
2	8.7%	89.8%
3	5.1%	94.9%
4	2.7%	97.7%
5	1.1%	98.7%
6	0.7%	99.4%
7	0.3%	99.7%
8	0.1%	99.8%
9	0.0%	99.8%
10	0.2%	100.0%

Table 5.15: Placement of the correct genre for best Cluster Pruning results

results for the Tzanetakis collection are the best that have been achieved using these particular descriptors. We now turn our attention more to the collection itself, and in Chapter 6 we provide a study of the ground-truth of this popular collection.

Chapter 6

Ground-Truth Experiments

Having reached a genre classification accuracy of 80.8% on the Tzanetakis collection using accurate k-NN classification, and 81.1% using approximate k-NN classification, we set out to examine the songs that were wrongly classified. When listening to these songs we found out that in some cases we actually agreed with our system on the classification, and disagreed with the ground-truth. On many more cases, however, we agreed with the ground-truth, and in some cases, we actually disagreed with both the ground-truth and our classification.

Classification into musical genres can be subjective and boundaries between genres are not always clear. We therefore decided to examine these songs in more detail and find out how human subjects would classify the songs.

6.1 Ground-Truth in the Literature

Although automatic genre classification has received much attention in the literature, there is not much work published about ground truth, how it is classified, and whether musicians and music listeners actually agree on this ground-truth.

The original version of Gjerdingen and Perrott (2008) from 1999 is much cited, although it has been unavailable in print until the re-release in 2008. The authors chose the following 10 genres: Blues, Classical, Country, Dance, Jazz, Latin, Pop, R&B, Rap and Rock. Fifty-two university students representing "ordinary undergraduate fans of music" listened to excerpts from eight songs of each genre. The excerpts varied in length from 250 ms to 3,000 ms. Genre classification was taken from CDnow, BMG and Tower Records, the leading web based music vendors of the nineties. When listening to the 3,000 ms excerpts

participants agreed with the ground-truth about 70% of the time. When participants were only allowed to listen to 250 ms excerpts, the accuracy varied greatly with genres. The average across all genres was 44% but the accuracy of blues songs was less than 20%, while the accuracy of classical songs was over 70%. A study with a small group of music theory majors revealed essentially the same results as with the non-musicians in the main study.

Martens, Leman, Baets, and Meyer (2004) compared the results of automatic genre classification and human genre classification on the same collection, called the MAMI collection. The MAMI collection consists of 160 full length songs, which were originally classified into 11 genres. They concluded that due to various reasons this classification was not fit for automatic genre classification and therefore conducted a user study with 27 human listeners. Each participant listened to a 30 second excerpt from all the songs and classified each song into one of six genres. The outcome from that study was as follows: 24 classical, 18 dance , 69 pop, 8 rap , 25 rock and 16 other, with the genre other being used for songs that did not fit into any of the first 5 genres. The next step was to compare the selected genre of each participant with this new ground truth. The accuracy of the 27 participants ranged from 57% to 86% averaging at 76%. A subset of the MAMI collection, called MAMI2 was then created. It included songs from the first five genres mentioned above, and only songs that had received 18 or more votes for their particular genre. This resulted in 98 tracks, and the average classification accuracy of the participants for this collection was 90%.

Craft, Wiggins, and Crawford (2007) criticized how the MAMI2 collection was created, and claimed that it was "not statistically well-founded". Their basis of argument is that the meaning of the genre "other" was undefined to the participants, resulting in different ways of using that genre. Should participants only use it for songs that do not find a home in any of the other genres or should they also use it if a song features multiple genres? They examined the songs that did not make it into the MAMI2 collection and found that only one of these songs received 10 votes for "other", one song received seven votes, but the remaining songs received five or fewer votes for the "other" genre. The authors then went on to construct a similarity graph of all songs in the MAMI collection, where songs with similar *distribution of genre votes* were grouped together. It turned out that there were groups of tracks that spanned multiple genes, and there were genres that spanned multiple groups of similar tracks. The main conclusion of the paper is that it is unrealistic to try to create a genre classification collection that is entirely unambiguous, since real life collections do not just contain unambiguous data. They proposed that all results from automatic genre classification systems should be weighted to reflect the amount of ambiguity of human classification of that same collection.

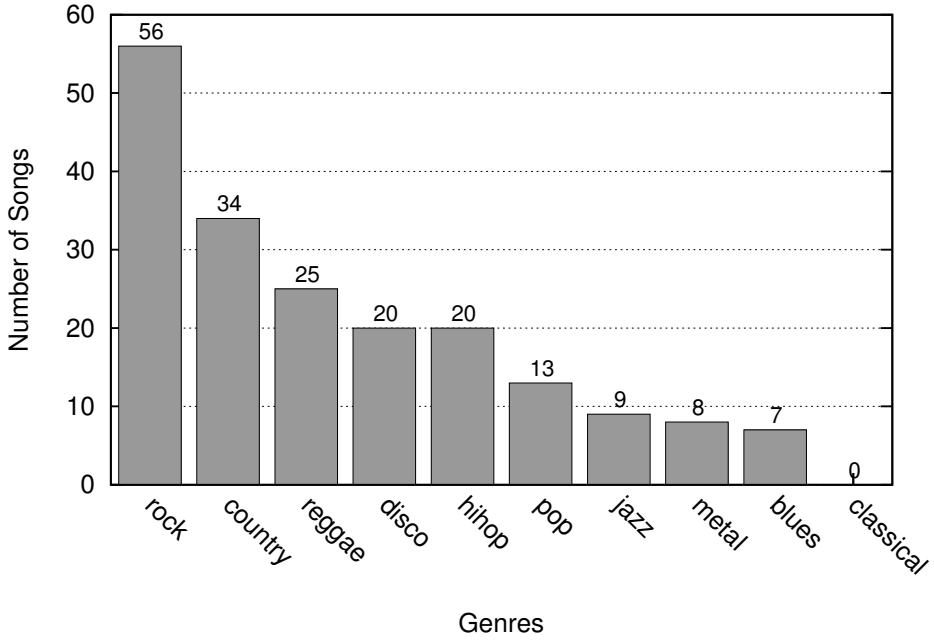

Figure 6.1: Genre distribution of songs used in our user study

6.2 User Study Setup

In order to see how human participants would classify our collection, we set up an experiment where 20 participants listened to the 192 songs that were wrongly classified by the sequential scan. The participants are all active in the Icelandic music industry, either as musicians, producers, sound engineers or DJs. They included both professionals and semi-professionals. Each participant received a list of the 10 genres of the Tzanetakis collection and then listened to the 192 30 second clips and marked each song with the genre label best describing that song. The wrongly classified songs were distributed to genres as shown in Figure 6.1. Note that even if no classical song was included in this datset, the genre itself was included on the list of possible genres.

The collection does not contain information regarding artists or song names, but listening to the songs reveals several interesting facts, including two mistakes in its creation. First, a live version of the song *"Tie your mother down"* by Queen is included two times in the collection, once labelled as rock and once as metal. During k-NN search both versions get incorrectly classified since the version labelled rock gets all its votes from the version labelled metal and vice versa. Second, one reggae sound clip is faulty, with only 6 seconds of music and 24 seconds of loud noise. It is interesting to note that the program classifies this noise as pop. George Tzanetakis, the author of the collection, has in a private e-mail confirmed being aware of these issues, but has chosen to let them be, since the

	Ground-Truth		Voted Results		k-NN	
	Songs	**Percentage**	**Songs**	**Percentage**	**Songs**	**Percentage**
Lowest agreement	101	52.6%	121	63.0%	15	7.8%
Median agreement	112	58.3%	153	79.7%	25	13.0%
Average agreement	113	59.1%	150	78.3%	25	13.0%
Highest agreement	134	69.8%	166	86.5%	34	17.7%
Voted agreement	122	63.5%	192	100.0%	24	12.5%

Table 6.1: Participants agreement with ground-truth, their voted results, and k-NN sequential scan results

collection has been used so many times and changing the files at this point would confuse comparison of results.

It turns out that the set often includes several songs by the same artists. Out of these 192 songs 7 songs are by Sting, 6 by Jethro Tull and 4 by Rolling Stones. Other artists that have multiple songs include Black Sabbath, Led Zeppelin, Beastie Boys, Bob Marley, Willie Nelson, Alanis Morrisette, Vince Gill and Guns'n'Roses. All the Sting songs are from his first two albums *"Dream of the Blue Turtles"* and *"Bring on the Night"* where Sting uses famous jazz musicians including Branford Marsalis on saxophones and Kenny Kirkland on pianos. All these Sting songs are classified as rock by Tzanetakis, whereas participants were divided between pop and jazz. The Jethro Tull songs included such diverse songs as *"Happy and I'm smiling"*, *"Bungle in the Jungle"* and *"Life is a love song"*. Again, all songs were considered rock by Tzanetakis. Most were also classified as rock by participants, while some were classified as pop. Many commented that *"Life is a love song"* is really folk or acoustic, but no such genre is included in the set.

6.3 Results

Table 6.1 displays participants agreement with the ground-truth, their voted results and the classification from our k-NN sequential scan. We see that the average agreement between participant and the ground truth is 59.08%. This number should not be compared with the results from Gjerdingen and Perrott (2008) and Martens et al. (2004) since the collection used for this experiment was specifically chosen because of our program not being able to classify the songs correctly. It is, however, interesting to note the low agreement rate on these songs. Table 6.1 also confirms that there is considerable variation in the way our participants classified the songs, with the highest agreement with the voted classification being 166 songs, or 86.46%. We also looked into whether the participants would agree with the results from our k-NN classification. Recall that these songs were

	Ground-Truth		k-NN	
Genre	**Songs**	**%**	**Songs**	**%**
pop	12	92.3	1	7.7
hiphop	18	90.0	1	5.0
blues	6	85.7	1	14.3
country	29	85.3	1	2.9
jazz	7	77.8	2	22.2
reggae	16	64.0	4	16.0
disco	10	50.0	6	30.0
rock	23	41.1	4	7.1
metal	1	12.5	4	50.0

Table 6.2: Agreement between votes from participants and ground truth or kNN

all incorrectly classified by sequential scan. It is interesting to see that the voted results of our participants agrees with our incorrect classification of 24 songs, with the range of individual agreement being from 15 to 34.

Next we compared the voted results to both the ground-truth and the results from our k-NN classification for each genre. Table 6.2 shows this agreement. We see that participants agree strongly with ground truth for pop, hiphop, blues, country and jazz. Reggae and disco have moderate agreement, while rock, and especially metal, have very low agreement. Participants agree with the incorrect k-NN classification of four out of the eight metal songs, but all of these four songs were classified as rock.

6.3.1 Comparison With the World Out There

In order to compare the classification of ground-truth, k-NN and our participants to that of the world out there we selected 15 songs randomly from the songs in the collection that we recognized. We then looked at how these songs are classified on iTunes, allmusic.com and last.fm.

Apple's on-line media store, iTunes, only classifies albums so songs actually can have multiple genre classifications if they are featured on more than one album. Two songs in our set, David Bowie's *"Space Oddity"*, and Jethro Tull's *"Life is a love song"* fall into this category, where both are classified as pop in one place, and rock in another place.

allmusic.com is a music reference web page with album and artist critique. allmusic.com classifies artists into genres and styles, where genres are usually very broad, such as "Pop-Rock" but styles are narrower. We report the genre and the two top styles of each artist.

last.fm is an internet radio station that allows users to tag songs. Tags can be anything users feel describes the songs. Most popular tags are displayed on the website. We report the three most popular tags, omitting all tags that include artists or song names.

Table 6.3 shows the comparison of the ground-truth, k-NN classification, our participants voting, iTunes, allmusic.com and last.fm. We see that iTunes agrees with the ground-truth of the collection in most cases, or 12 out of the 15 songs, if we count the two songs that have both pop and rock classification in iTunes. allmusic.com genre label is very broad, and in 12 out of the 15 songs this genre label is pop/rock. This goes for songs classified as pop, rock or metal by our ground-truth. The table also shows us that in 8 songs our k-NN classification has the correct genre in 2nd place, and in 2 songs the correct genre comes in 3rd place. From this small sample our participants only agree with ground-truth in 2 songs which is quite far from their agreement for the whole 192 songs. The participants have the correct genre in second place in 6 songs, and in third place in 3 songs.

6.3.2 Discussion of Particular Songs

Ani Difranco's *"Cradle and all"* has a very strong acoustic guitar presence and this is without a doubt the reason why our k-NN program classifies the song as country. Many country songs have this same sound characteristics. We see folk mentioned both at allmusic.com and last.fm, which also is a genre characterized by the acoustic guitar, but our ground-truth does not include this genre. iTunes uses the alternative genre for this song, but this genre is very ill defined. Our participants classify the song as a pop song, with several of them commenting that they would use folk, or acoustic pop, if either was available.

Billy Joel's *"Movin' out"* can hardly be classified as a disco song, although it has the dry 70's drum sound. Yet our k-NN classifier classifies it as disco, as too many other rock songs, with rock coming in second. 75% of our participants classify it as a pop song with the remaining votes going to rock. Both allmusic.com and last.fm use terms such as soft rock, which we believe is a synonym for pop in many people's mind.

Bob Seger's *"Against the wind"* is on all three websites considered a rock song. However, our solution does not have rock in the top three places, whereas 2 of our 20 participants classified the song as a rock song. The song has several elements of a classic country song including the acoustic guitar, the piano playing, and the vocal harmonies. This is one of the rock songs which our k-NN classifier classifies as a disco song, which is plainly wrong. We believe that if multiple genres were to be used, then pop, rock and country should all be used.

Artist **Song**	**Tzan**	**iTunes**	**allmusic**	**last.fm**	k**-NN** **genre / %**	**participants** **genre / %**
Ani Difranco	rock	alternative	pop/rock	folk	country 24	pop 85
Cradle and all			folk	female.voc	jazz 22	jazz 15
			urban folk	indie	blues 18	
Billy Joel	rock	rock	pop/rock	classic rock	disco 35	pop 75
Movin' out			singer/songwr.	pop	rock 21	rock 25
			soft rock	soft rock	country 20	
Bob Seger	rock	rock	pop/rock	classic rock	disco 24	pop 65
Against the wind			rock'n'roll	rock	country 23	country 25
			hard rock	soft rock	reggae 16	rock 10
David Bowie	rock	pop/rock	pop/rock	classic rock	country 37	pop 75
Space Oddity			hard rock	glam rock	disco 23	rock 20
			glam rock	british rock	rock 19	reggae 5
Jethro Tull	rock	pop/rock	progressive	country	disco 43	pop 85
Life is a love song			blues-rock	classic rock	rock 16	rock 10
			hard rock	70s	country 15	blues 5
Led Zeppelin	rock	rock	pop/rock	classic rock	pop 26	reggae 50
D'yer Mak'er			blues	rock	disco 23	rock 35
			blues-rock	reggae	rock 23	pop 15
Simply Red	rock	pop	pop/rock	pop	disco 28	pop 55
Freedom			soul	rock	rock 25	jazz 25
			adult.cont.	easy	blues 21	disco 15
Sting	rock	rock	pop/rock	rock	pop 31	jazz 50
Consider me gone			adult cont.	jazz	hiphop 15	pop 45
			cont. pop/rock	pop	reggae 14	blues 5
Jimmy Cliff	reggae	reggae	reggae	reggae	classical 26	pop 70
Many rivers to cross			reggae-pop	soul	jazz 23	classical 25
			roots reggae	jamaica	country 22	reggae 5
Marcia Griffiths	reggae	reggae	reggae	funk	pop 60	pop 65
It's Electric			dancehall	dance	disco 23	disco 30
			roots reggae	party	hiphop 5	hiphop 5
Cher	pop	pop	pop/rock	pop	disco 26	disco 60
Believe			dance-pop	dance	pop 24	pop 35
			adult. cont.	90s	reggae 23	hiphop 5
Madonna	pop	pop	pop/rock	pop	hiphop 32	pop 65
Music			dance-pop	dance	pop 22	hiphop 25
			adult.cont.	electronic	jazz 18	disco 15
Guns'n'Roses	metal	rock	pop/rock	rock	rock 54	rock 75
Live and let die			hard rock	hard rock	metal 38	metal 20
			heavy metal	cover	disco 4	blues 5
Living Colour	metal	rock	pop/rock	rock	hiphop 34	rock 60
Glamour Boys			alt. metal	funk rock	metal 28	pop 40
			alt. pop/rock	80s	disco 22	
Willie Nelson	country	country	country	country	blues 38	country 40
Georgia on my mind			trad.country	classic country	country 29	blues 30
			progr. country	folk	classical 16	jazz 20
Beastie Boys	hiphop	hiphop/rap	rap	hip-hop	metal 59	rock 70
Fight for your right			pop/rock	80s	hiphop 18	metal 25
			alt. pop/rock	rock	rock 17	hiphop 5

Table 6.3: Comparison of ground-truth, k-NN, our participants voting, iTunes, allmusic.com and last.fm

David Bowie's *"Space Oddity"* features the acoustic guitar very much, and this is without a doubt the reason why it gets classified as a country song by k-NN. All websites use the rock genre, sometimes with specific sub-genres of rock for this song, although iTunes classifies it as pop when it is a part of Bowie's *"Singles collection"* album. Most participants in our study classified the song as a pop song, with rock coming in second. The reggae classification of one participants must be a mistake, since there is not a single reggae element in the song. Just as it is difficult to pinpoint the boundaries between rock and metal, it is also very difficult to pinpoint exactly the difference between pop and rock.

Jethro Tull's catalog of songs is extremely diverse, so classifications on artist level are not going to be very accurate. allmusic.com classifications of blues-rock or hard rock hardly describe *"Life is a love song"* very well. last.fm tags of progressive, classic rock and 70's are more accurate, although less popular tags, such as folk rock describe it better in our opinion. Our participants classified it as pop, with rock coming in second, and one participant using the blues genre. The song is very acoustic, with acoustic guitars, mandolins and a flute. As with some other acoustic songs it gets considerable number of votes from country songs in k-NN.

Led Zeppelin is of course one of the greatest rock band in history, so it does not come as a surprise that *"D'yer Mak'er"* is classified as a rock song by ground-truth, iTunes, and two most popular last.fm tags, with allmusic.com using blues and blues-rock. Blues is indeed where the roots of Led Zeppelin lie. 50% of our participants and considerable number of last.fm users want to classify this song as a reggae song, and it cannot denied that indeed it has much more reggae feel than *"Many rivers to cross"*. At the same time it has some pop elements such as the instrumentation.

Simply Red's *"Freedom"* is classified as rock by ground-truth. This time we are not surprised with the disco classification of k-NN, since the song has in our opinion more disco element than rock elements, including the guitar sound and the prominent strings. The rhythm, although not the standard disco beat, also resembles disco, with very prominent bongo drums and tambourines. Vast majority of participants classify the song as a pop song, thereby agreeing with iTunes and the most popular last.fm tag (where rock comes in second).

Sting's *"Consider me gone"* is one of the songs he recorded with several famous jazz musicians. Our participants have almost the same number of votes for jazz and pop for this song, with one person considering it a blues song. None mentioned the rock genre used by the ground truth, iTunes and last.fm. We notice, however, that last.fm also has both pop and jazz tags, while allmusic.com concentrates on the adult-contemporary label.

This is one of these songs where it is very difficult to say that one particular genre is correct.

Jimmy Cliff's *"Many rivers to cross"* is yet another one of those difficult songs. Websites and ground-truth agree on defining the song as a reggae song, but the song does not include any trademark reggae features, such as the off-beat rhythm. Instead it has some classical characteristics, such as the prominent church organ sound. Jimmy Cliff is one of those artists that has merged reggae and pop music successfully, and as with Marcia Griffiths this song is perhaps not very representative for him. Most of our participants classify this as a pop song, with classical coming in second.

Marcia Griffiths' *"It's electric"* is an example of a song that perhaps does not represent the artist very well, and therefore there is inconsistency between genres that are created by artist or album classifications and genres that are created by song clssification. Both k-NN and our experiment participants classify this as a pop song, with disco and hip-hop coming in 2nd and 3rd places respectively in both cases. last.fm tags include funk, dance and party which can be said to be closer to the pop, disco, hip-hop, categories than the reggae genre assigned by both iTunes and allmusic.com. However, some of our participants commented that Marcia Griffiths is known as a reggae artist, but they still could not classify this particular song as a reggae song.

Cher's *"Believe"* has a the infamous disco drum beat where the high-hat opens on every offbeat. Most of the instruments are obviously programmed, which makes the sound different from the classic 70's disco songs. Participants agree with k-NN in classifying this as a disco song, but both put pop in second place with the difference in votes in the k-NN classification being very low. Perhaps the style dance-pop used my allmusic.com describes it best, but dance-pop is nothing other than a combination of disco and pop.

Madonna's *"Music"* is a very electronic song. Most, if not all instruments are electronic in nature and programmed instead of being "hand-played". It has this in common with most hiphop songs, in addition to some strange vocal effects. However, in our opinion, it lacks the hiphop beat to be classified as a hiphop song. We see that our participants agree with ground-truth, iTunes and last.fm most popular tag, in classifying it as a pop song, and indeed pop is the genre with second most votes in k-NN. Allmusic uses dance-pop which also describes the song very well.

Guns'n'Roses version of the Wings hit *"Live and let die"* is considered a metal song by ground-truth. iTunes, k-NN, participants and last.fm all agree on rock, while the first style at allmusic.com is hard rock, with metal coming in second for both k-NN and our

participants. It is difficult to say where the boundaries lie between rock and metal. This song does include a large dose of overdriven guitars, which does characterize metal, but in our opinion the overall sound and feel is much more rock.

Living Colour's *"Glamour Boys"* is classified as hiphop by k-NN with metal and disco in 2nd and 3rd place respectively. Ground-truth considers this a metal song, while participants, iTunes and the most popular last.fm tag agree on rock. Some participants commented that indeed the verse with its clean guitar sound of the song is a pop verse, while the chorus with its overdriven guitar and more aggressive voice is more rock oriented. This caused some of them to have problems deciding which genre to use. In the end it was 60/40 for rock against pop.

"Georgia on my mind" has been recorded my many artists. With Willie Nelson being a country icon, iTunes, which classifies albums, and allmusic.com, which classifies artists, of course use the country genre for his version of this song. The three most popular tags at last.fm are country, traditional country and folk. The fourth most popular tag (not counting the tag Willie Nelson) is blues. k-NN classifies the song as blues with country coming in second place, while this is reversed for our participants. The song, in our opinion, is more of a blues song than a country song, but Willie Nelson does of course bring a certain country flavor to it.

Beastie Boys' *"Fight for your right"* would probably never be classified as a hiphop song by people that heard it the first time and did not know that Beastie Boys are a hiphop/rap band. The instrumentation and rhythm are those of a typical rock/metal song, with loud overdriven guitars, and simple bass and drum beats. The vocals are the only thing that resembles rap music. k-NN strongly classifies this as metal with hiphop and rock coming in second and third, while 70% of our participants classify it as rock, and 25% as metal. One participant classified it as hiphop.

6.3.3 The Effect of Ground-truth on Classification Accuracy

Having seen that the participants in our ground-truth experiment had in many ways different opinion on which genre songs in the Tzanetakis collection should belong to we decided to change the ground truth of the songs where the voted genre from our participants differs from the ground-truth. Recall from Table 6.1 that the voted results from the experiment agrees with ground-truth for 122 songs, meaning we changed the ground-truth of 70 songs. Table 6.1 shows us that out of these 70 songs the result from the experiment agrees with the results from our k-NN classification for 24 songs.

We now ran our sequential scan program with the same parameters as our best results in Section 5.1 , extracting timbral features + SFM and using $k = 3$ when searching, with other parameters being default. To our surprise our accuracy did not improve much. It went from 80.8% to 81.5%, meaning only 7 more songs were correctly classified, even if 24 songs were changed to be as our program had previously classified them. 86 songs in addition had the correct genre in 2nd place, for a total of 90.1% in 1st or 2nd, which is only an improvement of three songs. This is an increase of only one song compared with the unmodified ground-truth. The reason for this is that in many cases the vote ratio between the genre in first place, and the genre in second place is very low, meaning they get very similar number of votes. Therefore several songs that were correctly classified when using unmodified ground-truth change to being incorrectly classified when using the modified ground-truth. It is also worth pointing out that we only had the participants of our experiment listen to the songs that were originally incorrectly classified. If we were to actually change the ground-truth in order to make each genre more coherent we would need to use all of the 1,000 songs for this kind of an experiment.

6.4 Summary

We have seen through a number of experiments that the evaluation of the results from automatic genre classification systems is not as simple as it might seem. Just because the classification of a given song does not agree with a given ground-truth classification does not necessarily mean it is incorrect. Given the subjective nature of genre classification, and how artists sometime merge two multiple known genres, there are many situations where two or more genres might be appropriate for a given song. We have also seen that changing the ground-truth increased our accuracy for 7 songs out of the 1,000. We conclude that in order to create a working automatic genre classification system, much more emphasis must be put on the ground-truth creation and analysis, and evaluation of the results of such systems needs to be much more than simply calculating a percentage of how many of the top genres agree with a given ground-truth. We agree with Craft et al. (2007) that one good method for such evaluations could be to weigh the results from such systems to reflect the amount of human classification ambiguity of the same collection.

Part II

Large-Scale Music Classification

We have shown using the Tzanetakis dataset that k-NN is a viable method for classification of music databases. However, the set is a very small one, counting only 1,000 songs, where each song is represented by a 30 second clip. In order to examine whether our method works on real life large-scale databases we contacted *D3 midlar* (D3 media), which runs the Icelandic web site Tonlist.is.

Tonlist.is was opened in 2003, and today it includes the bulk of all Icelandic music published in the past 50 years, along with music videos and music related news and blogs. In addition to being an on-line store where users can buy these songs, the web site also offers subscription packages allowing users to freely stream all the offered music. The company also runs the website `http://icelandicmusic.com` where the main focus is to introduce Icelandic music to the international market. D3 midlar was kind enough to grant us research access to their collection of Icelandic songs for our experiments.

In Chapter 7 we give detailed information about this song collection, including its usage of genres. Chapter 8 discusses our experimental setup, including some experiments we conducted on the small set, while Chapter 9 presents the classification results from the large set. Finally Chapter 10 concludes the thesis.

Chapter 7

The Tonlist.is Music Collection

In this chapter we will discuss the real world song collection used for the large scale genre classification task, and give an overview of our methods for working with this collection.

Our experiments with this collection were conducted in two steps. At first we were provided with 3,148 songs randomly selected from their song collection. We will call this set *Small Set* in this text. After experimenting with the small set we were granted access to their full collection of Icelandic songs, which includes 38,762 songs. We will refer to this set as *Large Set* in our text.

All songs in the collection are stereo mp3 files sampled at 44,100 kHz with bit rate 192,000 kbit/s.

Unlike traditional work in music genre classification, the goal of this work is not to verify classification accuracy against a robust ground truth. Instead, we have a large collection of songs with slightly suspect genre tags, and our goal is to use these labels to obtain an improved classification. In order to achieve that goal, we now consider, in sequence, the genre distribution of the songs in Section 7.1, selection of genres for automatic classification in Section 7.2, robust training set selection for bootstrapping the classification in Section 7.3, the classification process itself in Section 7.4, and finally quality measures in Section 7.5.

7.1 Genre Distribution

The Tonlist.is collection is classified into genres by D3 employees. Over time, several different employees have been responsible for the classification. Furthermore, in the clas-

Genres	**Number of Songs**
Pop	13,313
Adult/Contemporary	5,130
Funk	70
Soul	50
Eurovision	205
Pop Total	18,768
Classic	1,691
Choirs	660
Icelandic Vocals	596
Classical Vocals	579
Icelandic Folk Songs	262
Traditional Icelandic	79
Opera	27
Classic Total	3,894
Children	2,190
Adventure	259
Children Total	2,449
Folk/Country	1,498
Poetic	354
Folk/Country Total	1,892
Soundtracks	421
Theatre	689
Soundtracks Total	1,110
Instrumental	621
Accordion	254
Instrumental Total	875
Electronic	266
Experimental	278
Electronic Total	544
Christian	188
Gospel	165
Christian Total	354
World	0
Latin	185
World Total	185
Rock	4,148
Christmas	1,797
Jazz	1,327
Comedy	672
HipHop/Rap	365
Blues	135
Relaxation	108
Total	38,762

Table 7.1: Genre distribution in Tonlist.is song collection

sification interface, the pop category is the default option. Due to these issues, the genres have not seen much use in the web-site. As a result, it is our understanding that the

genre classification is slightly suspect, and in particular the pop category is too large and diverse.

Table 7.1 shows the genre structure of the Tonlist.is music collection. The table shows the hierarchical structure of their genres where sub-genres of a given parent genres follow the parent genre and are indented. The table only shows genres that have songs either assigned to them or to their sub-genres. Their database includes some additional genres that are not used. We see that the collection has a very skewed distribution of genres with the pop genre being prominent, regardless of whether we examine top-level genres or root-level genres. Several genres are very poorly represented, while we also notice that some genres are unusable in automatic genre classification systems, since they are based on other factors than the actual sound of the recording. These include Children, Christmas, Theatre, Comedy, Soundtracks, Adventure, Eurovision and Christian. We also see that in most cases the top-level genres stand on their own, with the only exception being the world genre. In this case no song is assigned to the top-level world genre, while the sub-genre of latin has 185 songs.

7.2 Genre Selection

Table 7.1 includes several genres that are not good candidates for automatic genre classification. We therefore removed the following genres before doing our experiments:

Christmas. Although there are some sound characteristics common with many Christmas songs, such as the Christmas bells, this class of songs includes pop, rock, jazz and more genres.

Comedy. This class includes pop and rock songs as well as spoken word.

Children includes pop, choir music and spoken word.

Christian. Our classification is not based on lyrical content at all. We can therefore not use any genres focusing on that aspect.

Soundtracks. We saw that this includes many pop and rock songs which have been featured in movies. In many cases these same songs are differently classified when appearing on the artists album, than when featured in the movie soundtrack.

Theater. Just as with Soundtracks this includes many pop and rock songs.

Icelandic Folk Songs. Many of these songs have either pop or classical arrangement, making the genre not coherent.

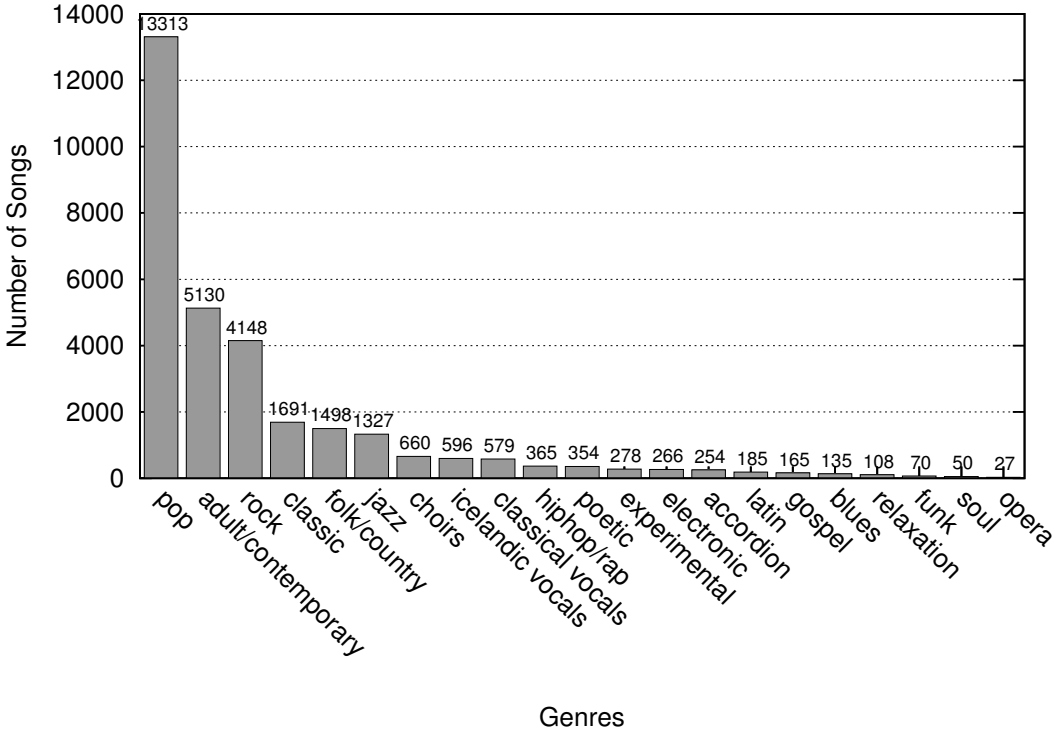

Figure 7.1: Genre distribution of songs used in our large set experiment

Eurovision. These songs are mostly pop songs, and as we can see from Table 7.1 it is actually a sub-genre of pop. However, this is a bit of a simplification since it is not completely limited to pop. We therefore elected not to use it.

The following genres were used in our small set experiments. Based on the results we observe that there was not enough to differentiate them from other genres.

Adventure. Most of the songs are pure pop songs.

Instrumental. The lack of vocal parts is not enough to differentiate the genre. This class includes mostly jazz and pop songs.

Traditional Icelandic. Although this is a sub-class of classic, it includes very broad types of sound characteristics.

After removing these 11 genres we were left with 31,199 songs of 21 genres shown in Figure 7.1 for our large set experiments. We see that pop songs dominate this collection with 42.67% of the songs being pop songs.

7.3 Training Set Selection

For any supervised classification task, it is necessary to have a well classified *training set*, from which to learn how to classify the remaining songs. For the Tonlist.is collection, selecting this training set is not a trivial task, as a) the collection is too large to use user input, and b) as mentioned above, the current genre classification is slightly suspect.

We have therefore chosen to use the following methodology to select a training set that we believe is robust. Recall that the k-NN classifier returns not only the classification of each song, but also the number of votes that the various genres receive; these votes can then be used to measure the robustness of the classification. In order to select the initial training set, we indexed the entire collection, and ran the approximate k-NN classifier for each song. For the training set, we then selected only those songs where the k-NN classification agrees with the Tonlist.is genre, and the correct genre has at least three times as many votes as the next genre. The rationale is that if the classification agrees with the Tonlist.is classification with such a high confidence, then that classification is most likely quite accurate. This methodology is similar to edited nearest neighbor (ENN) (Guan, Yuan, Lee, & Lee, 2009), where data points that have been misclassified by k-NN are removed from the training set. We do, however, also remove data points that are correctly classified, if the confidence of their classification is low. Using this methodology, 10,280 songs were included in the training set, which is indexed and used to classify the entire collection. Section 9.1 provides more details of the training set selection.

7.4 Classification Methodology

Unlike traditional genre classification, where a training set with a solid ground-truth is in place, the goal of our work is to improve the existing classification as much as possible. We therefore propose an iterative methodology, where we start with the initial training set and classify the entire collection. We then take the output of the classification as the next training set, and re-classify the entire set. This process can be repeated, until the output converges to a stable classification.

We hope that, in each iteration, some songs that are incorrectly classified will switch to the correct genre. In the next iteration, this song may then help to attract other incorrectly classified songs. In particular, the expectation is that the pop genre will shrink between iterations, while most other genres will grow.

One concern, however, is that when a few large genres dominate the collection, they will impact the classification in the other direction, incorrectly attracting songs from smaller genres, resulting in a degradation of classification quality. We therefore propose to weigh the votes with the size of the genre, in order to give less weight to larger genres, which are more likely to contain incorrectly classified songs.

The inverse document frequency is often used in information retrieval to correlate the value of a term with its frequency in the document collection, since more frequent terms tend to be less distinctive than less frequent terms (Jones, 1972). We define the inverse document frequency of a genre g, or idf_g, as:

$$idf_g = log_2(N/f_g) \tag{7.1}$$

where N is the number of songs in the collection and f_g is the frequency of genre g. This gives a vote from a small genre more weight than a vote from a common genre.

7.5 Quality Measures

The standard method for measuring classification accuracy of automatic genre classification systems is to compare the output of the classifier with a given ground-truth set, which is typically distinct from the training set. Each song either agrees with the ground-truth, in which case the classifier is said to correctly classify the song, or it does not agree, in which case classifier is said to incorrectly classify the song.

We can, indeed, compare the output of the k-NN classifier to the Tonlist.is genre, but we believe that this would result in a poor measure of quality. We can also compare to our training set, but we expect classification accuracy to be high for that set. In the following, we report both measures.

In addition to these standard accuracy measures, however, we report the following two quality measures.

Genre Distribution. Since pop has been used as a default genre at Tonlist.is it includes a number of songs that should belong to other genres. We therefore believe that the smaller the pop genre is in the classification results, the better the classification is. More generally, we believe that the more evenly distributed the genres are, the better the classification. We therefore calculate the *skew* of the distribution of songs into genres.

We note that the distribution of songs into genres for this collection resembles a Zipf-like distribution. Zipf's law was originally concerned with appearances of words in languages

and states that the n-th most popular words appears $1/n$ times as often as the most popular word. More generally, in Zipf-like distributions, the popularity of the n-th most popular rank can be estimated with K/n^{α} where K represents the frequency of the most popular term and α characterizes the distribution. The lower the α value is, the less skewed the distribution is. We used least-squares fitting of our data for the classification output of each iteration to find the α value of the Ziph-like distribution.

Convergence. Although we hope that incorrectly classified songs will move between genres, it is important that they converge to a stable distribution. In order to measure convergence, we measure classification accuracy with respect to the previous iteration, a_p, by comparing the output input of each iteration to the input of that iteration. Convergence is then defined as $100\% - a_p$. The expectation is thus that after a few iterations, this measure should converge to 0%.

7.6 Summary

We have in this chapter given detailed information about the Tonlist.is collection and its usage of genres. We see that the distribution of genres in the collection is skewed, and that pop is by far the largest genre. We have also provided information about which genres from the collection we use and how we establish a training set for the collection by choosing songs where the k-NN classification agrees with the Tonlist.is classification with high confidence. Finally we discussed our classification methodology and some ways we can measure classification accuracy for this collection.

We will now discuss our experimental setup and give details on the experiments we conducted on the small set in order to fine tune our parameters for the genre classification of the large set.

Chapter 8

Experimental Setup

We start this chapter by Section 8.1, which provides information on some experiments we performed using the small set. The results from these experiments affect our final parameter settings for the classification of the large set discussed in Chapter 9. The chapter concludes with discussion of the feature extraction process of the large set in Section 8.2.

All experiment reported in this part were conducted on a Linux machine with eight 2.67GHz Intel Core i7 processors, 3GB of main memory and three 7.2Krpm hard drives (500GB, 300GB and 150GB), running CentOS 5.5 operating system.

8.1 Small Set Experiments

In this section we discuss the genre classification experiments conducted on the small set of songs from the Tonlist.is music collection. The goal of these experiments is to obtain good settings for our classifier for the large set experiments. All accuracy measurements here are simply the percentage of songs that agree with the Tonlist.is classification.

As before MARSYAS was used for all feature extractions. Our approximate k-NN classifier described in Section 3.3.2 was used for classification.

8.1.1 Comparing Tonlist.is to the Tzanetakis collection

In our search for a robust training set we decided to try to classify songs from the small set using the Tzanetakis collection as a training set. Note that out of the 10 genres in

the Tzanetakis collection only seven are represented by Tonlist.is. These are pop, classic, country, jazz, rock, hiphop and blues. Two of these seven seven genres are actually broader in Tonlist.is where country is called folk/country and hiphop is called hiphop/rap. 1,620 songs from the small set belong to these seven genres.

Please note that while the Tzanetakis collection is sampled at 22,050kHz the Tonlist.is collection is sampled at 44,100kHz. In order to have comparable window sizes and hop sizes in samples we have two choices; to downsample the Tonlist.is collection by a factor of 2 before extracting the features, or to use twice as high values for window size and hop size when doing the extraction. We tried both methods, and it turned out that the latter was slightly better.

We extracted 120 seconds from each of the small set songs, starting 30 seconds into each song. This resulted in 1,599 excerpts being used since some of the songs are actually less than 30 seconds, and are therefore skipped when using these parameters. We used our cluster pruning program with parameter settings of $C = 100$, $L = 2$, $k = 1$, and $b = 1$. The results were very disappointing, with only 26.95% of the songs being correctly classified, and 14.01% being in second place, for a total of 40.96% of songs being in 1st or 2nd place. When examining the data we observe that the seven blues songs of the small set all get correctly classified, and the 226 classical songs get a classification accuracy of 80%. Other genres got much lower accuracy. We therefore decided not to pursue this any further.

It would, however, be interesting to examine this further in future work. Does the different audio format (uncompressed audio files vs. mp3 files) have this much to say, or is some information lost in the downsampling of the Tzanetakis set? The difference between the genres of the two collections and the fact that the Tzanetakis collection is generally older than the Tonlist.is collection may also influence this somewhat. At any rate we feel that these results should be much higher. To make sure that the problem does not lie within this subset of the small set of Tonlist.is we ran our k-NN classification on this collection. We again used cluster pruning, now with parameters of $C = 1,000$, $L = 2$, $k = 1$, and $b = 1$. Now we achieved much better results with 70.86% of the songs being correctly classified and 20.58% being in second place for a total of 91.44% being in either first or second place compared to the Tonlist.is classification.

Table 8.1 shows us the confusion matrix of this experiment. The numbers show the percentage of songs, while the numbers in parenthesis are the actual number of songs. We see from it that there are only two of the seven genres that are well classified. Classic achieves 94% accuracy and pop achieves 95% accuracy. Other genres range from 0 to 34%. Please note that the seven blues song that achieved 100% accuracy against the

	blues	**classic**	**country**	**hiphop**	**jazz**	**pop**	**rock**
blues	**0**	0	0	0	0	100(7)	0
classic	0	**94(213)**	0	0	0	4(9)	2(4)
country	0	13(15)	**19(22)**	0	0	68(79)	0
hiphop	0	0	0	**4(1)**	0	92(21)	4(1)
jazz	0	21(18)	0	0	**34(29)**	45(38)	0
pop	0	2(15)	1(5)	0	0(1)	**95(751)**	2(15)
rock	0	3(11)	0	0	0	64(227)	**33(117)**

Table 8.1: Confusion matrix of Tonlist.is songs having same genres as Tzanetakis collection

Tzanetakis set now are all wrongly classified. However, due to the fact that they are only seven the results in this context are not very reliable. These results are in perfect harmony with our information from Tonlist.is that the pop category is used very much. 787 songs out of the 1,599, or 49.2% are classified as pop, resulting in most genres getting a high number of votes from the pop genre. Five genres out of these seven get the majority of their votes from pop songs, with the only exceptions being classic and jazz.

We now turn our attention to using the 24 genres detailed in in Section 7.2. A total of 2,689 songs from the small set belong to these genres.

8.1.2 The Effect of Song Excerpt Location

Our first experiment with these 2,689 songs was to see which 30 seconds clips from the songs would give highest classification accuracy compared to the Tonlist.is classification. We started by using only the first 30 seconds, then the next 30, and so on. For all experiments we used the parameter settings of $L = 2$, $k = b = 1$, and $C = 5,000$.

Table 8.2 shows us the effect of the song excerpt location on the classification accuracy. We see that the highest accuracy is received by using the 30 seconds starting 60 seconds into each song. Notice how the number of songs drops as we move further into the songs, indicating that some songs files are less than 30 seconds long.

Song part	**Songs**	**Accuracy**
0-30 sec	2,689	49.2%
30-60 sec	2,662	52.9%
60-90 sec	2.620	53.7%
90-120 sec	2.620	52.4%
120-150 sec	2.408	50.1%

Table 8.2: Effect of song excerpt location on genre classification accuracy

sec.	C	Accuracy	class. time
30	5,000	52.9%	1:41
60	10,000	54.1%	4:11
90	15,000	54.1%	5:51
120	20,000	53.4%	9:32

Table 8.3: Effect of soundclip length on genre classification accuracy and classification time

We conclude from this experiment that if using a 30 second clip then the part from 60 seconds to 90 seconds should be used.

8.1.3 The Effect of Song Excerpt Length

Next we examined the effect of the length of each clip on the classification accuracy.

Here we start 30 seconds into the soundfile since this allows us to do more experiment than if we start 60 seconds into the soundfile. We then work with 30 seconds, 60 seconds, 90 seconds and 120 seconds clip. In order to keep the average cluster size similar we increase C as we increase the excerpt length.

Table 8.3 shows the effect of soundclip length on both genre classification accuracy and classification time. We see from table that accuracy improves a little bit when going from 30 second excerpts to 60 seconds, but classification time more than doubles. Increasing the size again from 60 seconds to 90 does not provide further improvements and increasing the size any further seems to actually decrease accuracy slightly. When using a length of 120 seconds that start 30 seconds into the song we have actually come to the end of some songs as can be seen in table 8.2 meaning that included in the descriptors are descriptors with near or complete silence, which decrease the accuracy.

When comparing tables 8.2 and 8.3 we notice that using a 30 second excerpt that starts 60 seconds into the song yields accuracy results very close to using either 60 seconds or 90 seconds excerpts. Since the classification time is so much less for the 30 seconds excerpts we conclude that we will use 30 seconds clip starting at 60 seconds into the songs for our large set classification experiments.

8.1.4 Using Only Top Level Genres

As mentioned in Section 7.1 the genres at Tonlist.is have a hierarchical structure. We decided to run our k-NN on the small set using only the top level genres to compare the

results to other results in this chapter. For this experiment we used the following 11 top level genres: pop, classic, country/folk, rock, jazz, hiphop, blues, instrumental, choirs, electronic and relaxation. We included choirs in this although it is not top-level genre according to Tonlist.is, since our experiments have shown this genre to be coherent and able to stand on its own. Using 30 seconds excerpts starting 60 seconds into the songs and the same parameter settings as used in Table 8.2 we achieved 67.78% accuracy with 86.45% of songs having the correct genre in either 1st or 2nd place.

We see that this is considerably higher than when using the more detailed genre labels. However, this also provides much less information, since we use only 11 genres. We therefore conclude that even with the lower accuracy we will use the more detailed genre labels for the iterative large set experiments.

8.2 Large Set Feature Extraction

Feature extraction of the large set took place at Tonlist.is offices. We extracted features from the entire song files, for maximal flexibility in the experimentation. We extracted all the timbral features, as well as Spectral Flatness Measure, resulting in descriptors of 82 dimensions. Window size was set to 1,024 and all other parameters of the MARSYAS package were given default values.

Since MARSYAS only runs on one processor, but our hardware is a multi-processor machine we opted to run six instances of MARSYAS with each instance extracting features from different songs. We also used the three hard-drives for writing the resulting files. With this setup it took 515 minutes or roughly 8.5 hours to extract our features from the 31,199 songs. The resulted text files are a total of 218GB. Converting these files to binary database files, where only 30 seconds excerpts starting 60 seconds into the songs are used, took 65 minutes, and the resulting binary file is 4GB. 603 songs turned out to be shorter than 60 seconds, so the final number of songs used in the experiments is 30,596. The number of descriptors used are roughly 39 million.

8.3 Summary

In this chapter by detailed the classification experiment done on the small set from Tonlist.is. We saw that we can not use the Tzanetakis collection as part of our training set. We also saw that location and length of the song excerpt can influence classification accu-

racy. Length also has considerable effect on classification time. Taking both accuracy and classification time into account, we decided to use 30 seconds clips starting 60 seconds into the songs for our large set experiments. We also performed a small experiment with the top level genres, but concluded that even with higher accuracy we loose too much detail in the classification due to limited number of genres. The chapter concluded with a description of our feature extraction process for the large set.

We will now look at the results from our k-NN genre classification experiments using these features.

Chapter 9

Classification Results

In this chapter we give the results of our genre classification experiments using the large set of Tonlist.is. Section 9.1 discusses our training set used, while Section 9.2 gives details of our iterative experiments. Finally we summarize the results in Section 9.3.

9.1 The Training Set

As mention in Section 7.3 it is imperative to use a robust training set in any classification experiments. We decided to run our k-NN classification on the complete collection, and select songs from the results that were classified correctly compared to the Tonlist.is classification with a vote ratio of at least 3. We believe that the songs that have this much confidence in the classification are most likely classified correctly by Tonlist.is

We did some experiments with the number of clusters for the collection (the C parameter), with 40,000 turning out to be a good number. Other clustering parameters are default.

Clustering our collection took 29 minutes. We then ran our approximate k-NN classification program on these clusters. Classification time was 95 minutes. The results was that 18,687 songs, or 61.08% were classified correctly according to the Tonlist.is classification with further 21.46% having the correct genre in 2nd place, for a total of 82.54% of songs having the correct genre in either 1st or 2nd place. When comparing this with our results from the small set in sections 8.1.2 and 8.1.3 we notice that these results are considerably better. We believe there are two main reasons for this. First, cluster pruning, like any other k-nn classification scheme, is likely to provide better results the more data

Genres	Number of Songs
Pop	6,752
Adult/Contemporary	1,091
Classic	1,013
Rock	599
Choirs	231
Folk/Country	132
Jazz	123
Classical Vocals	119
Icelandic Vocals	92
Accordion	55
Poetic	23
HipHop/Rap	22
Gospel	9
Latin	6
Experimental	5
Electronic	5
Blues	2
Relaxation	1
Total	10,280

Table 9.1: Genre distribution of the training set for large set experiments

you have. Second, the genre distribution of the small and large set are not exactly the same. This might effect the accuracy somewhat.

From the correctly classified songs we selected as our training set to be the 10,280 songs where the correct genre has at least three times many votes as the next genre. Three genres were lost in this experiments, since no songs from them managed to be classified with a vote ratio of three or higher. The genres are funk, soul and opera.

Table 9.1 shows the distribution of songs into genres in this training set.

9.2 Iterative Experiments

In our iterative experiments we start with the training-set described above. We classify our song collection based on this, and take the output of that classification to be the input, or training-set, for the next iteration. We run a total of five iterations in this matter. We conduct two lines of these iterative experiments. In one line of experiment we use regular voting, where each vote counts as one vote irrespective of the genre it comes from. In the other line of experiments we weigh the voting by *idf* described in Section 7.4

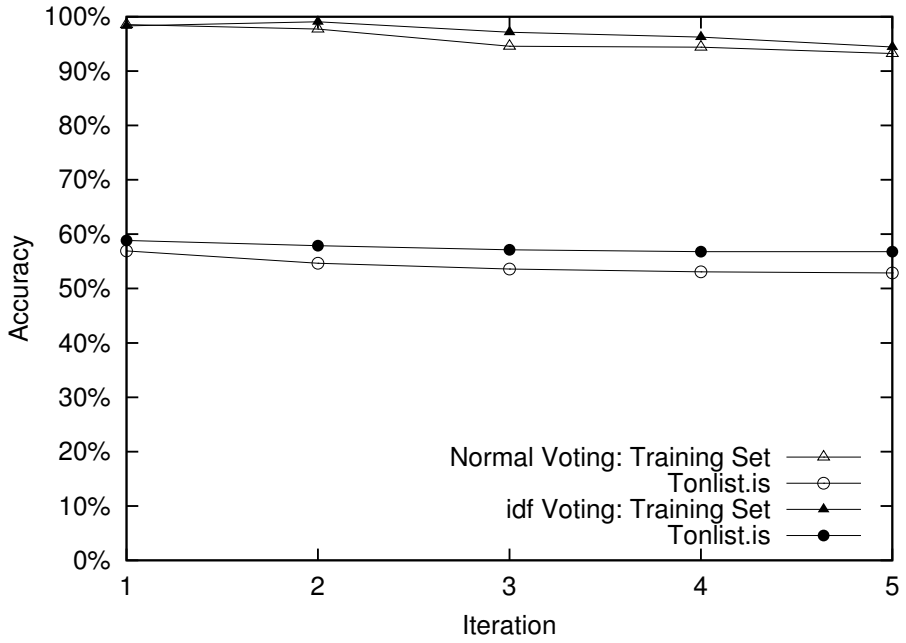

Figure 9.1: Classification accuracy across iterations

9.2.1 Accuracy

Figure 9.1 shows the development of classification accuracy across the iterative classification process, based on both the training set and the entire collection. As mentioned in Section 7.5, neither is a particularly good measure of the classification quality, but they are included here for completeness. The x-axis of Figure 9.1 shows the iteration, while the y-axis shows the classification accuracy.

As expected, the accuracy for the songs in the training set is very high, near 100%, while the accuracy for the entire set is much lower, only between 50% and 60%. Furthermore, we observe that the accuracy decreases across the iterations, e.g., from 98% to 93% for the songs in the training set. Again, as mentioned in Section 7.4, this may be a good sign, as it may mean that incorrectly classified songs are shifting between genres. Finally, in Figure 9.1, we observe that the idf based voting scheme performs better than the regular voting scheme for both metrics.

9.2.2 Genre Distribution

Figure 9.2 shows the distribution of the songs across different genres. The x-axis shows the genre rank (different classifications may result in different rankings), while the y-axis shows the percentage of songs that belongs to that genre. Note the logarithmic scale on

the y-axis. The figure compares the distribution in the training set to the distribution after five iterations using either the regular voting scheme or the idf coting scheme.

Both voting schemes give pop as the most common genre. With regular voting, 25,186 songs are assigned to pop, which is about 82.3%. This is severely skewed when compared to the training-set (65.7% of the songs are labeled as pop) and to the results of the idf voting scheme which classifies 15,512 songs (50.7%) in the pop genre. The distribution of the other genres is also much closer to the training-set when applying the idf voting scheme. For example, the two next most popular genres receive 9.4% and 4.4% of the songs with regular voting while they get 13.7% and 11.2% of the votes with the idf scheme, which is closer to the training-set (10.6% and 9.9% respectively). Overall, Figure 9.2 clearly shows the improvement in genre distribution when using the weighted voting scheme: it sticks quite close to the distribution of genres defined by the training-set. In contrast, the distribution of the regular voting scheme is worse than that of the training set, and the smallest genres are completely empty with this voting scheme. With idf, these genres are also under-represented but the indeed very small differences are somehow emphasized due to the y-logscale.

Figure 9.3 shows the same data as Figure 9.2, but this time the genres are always ordered the same, so we can see the effect of the two voting schemes on each genre. We see that the genre ranks change considerably based on the voting scheme. For example Adult/Contemporary, which is the second largest genre in training set, is the fourth largest in idf voting. Jazz, which is the seventh largest in the training set is the fifth largest in idf voting.

As mentioned in Section 7.1 we believe that a number of non-pop songs are incorrectly classified as pop by Tonlist.is. However, we find it likely that the quality of the genre classification of other genres is better. We therefore examined the number of songs from each genre where the k-NN classification agrees with Tonlist.is. Table 9.2 clearly shows that at each iteration more and more songs get classified as pop. By the fifth iteration 12,637 songs out of the 12,838 songs classified as pop by Tonlist.is are also classified as pop by the regular voting scheme. In Table 9.3 we see that the same number for the idf voting scheme is 9,697. By using idf voting, we get improved accuracy for most of the non-pop genres, except for the smallest ones where it is the same. The difference is most notable for jazz, where the fifth iteration of idf classifes 579 songs or 43.86% correctly, compared to only 24 songs or 1.82% when using normal voting. This is an improvement by a factor of 24. The classification of poetic improves by a factor of 13.8, and Icelandic vocals by a factor of 10.5.

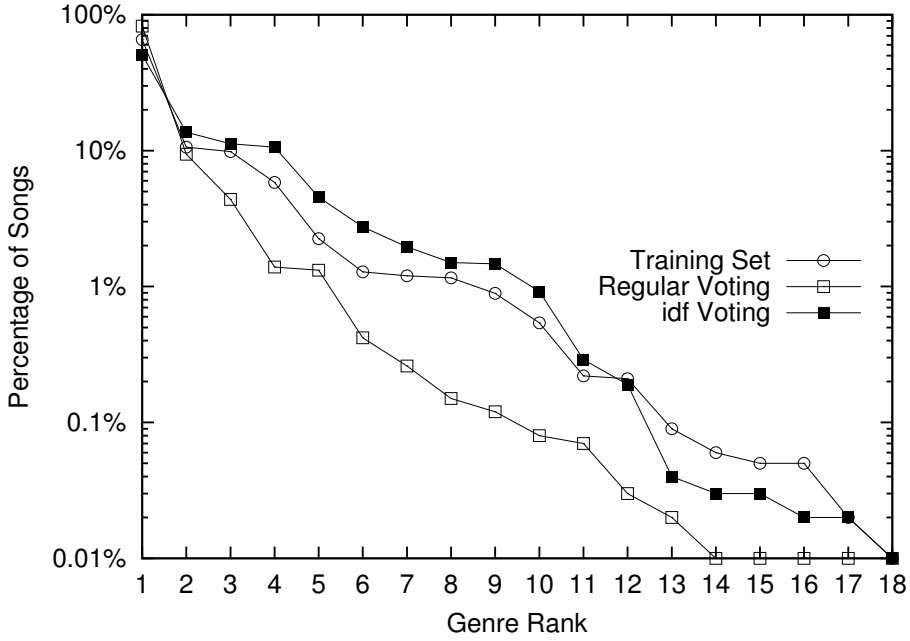

Figure 9.2: Genre distribution after five iterations

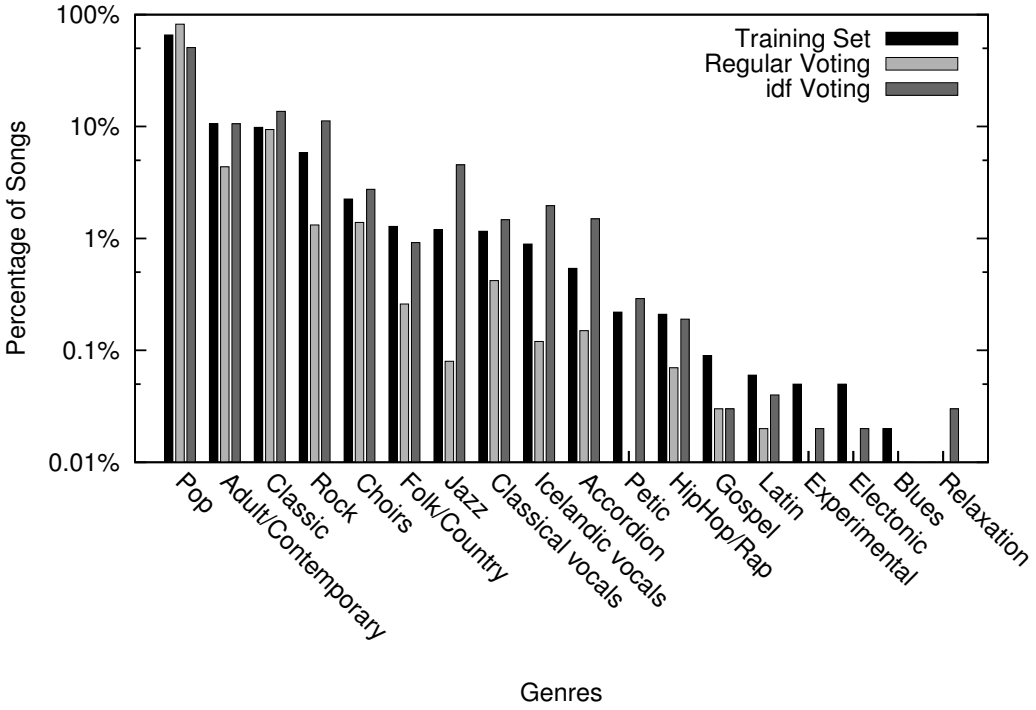

Figure 9.3: Genre distribution after five iterations

	Iterations									
genre	**1**		**2**		**3**		**4**		**5**	
pop	12,425	96.78%	12,549	97.75%	12,600	98.15%	12,620	98.30%	12,637	98.43%
classic	1,453	87.58%	1,433	86.38%	1,383	83.36%	1,330	80.17%	1,304	78.60%
adult/cont.	1,445	27.46%	1,296	24.63%	1,235	23.47%	1,202	22.84%	1,193	22.67%
rock	849	20.95%	592	14.61%	464	11.45%	398	9.82%	375	9.25%
choirs	382	57.97%	352	53.41%	341	51.75%	335	50.83%	331	50.23%
jazz	194	14.70%	68	5.15%	27	2.05%	24	1.82%	24	1.82%
clas. voc.	182	31.49%	139	24.05%	114	19.72%	109	18.86%	108	18.69%
country	181	12.47%	121	8.33%	92	6.34%	89	6.13%	80	5.51%
ice.voc.	148	24.96%	47	7.93%	37	6.23%	35	5.90%	32	5.40%
accordion	73	28.74%	59	23.23%	50	19.69%	47	18.50%	45	17.72%
poetic	38	10.73%	19	5.36%	7	1.97%	4	1.30%	4	1.30%
hiphop	22	6.16%	22	6.16%	22	6.16%	22	6.16%	22	6.16%
gospel	9	5.45%	9	5.45%	9	5.45%	9	5.45%	9	5.45%
latin	6	3.24%	6	3.24%	6	3.24%	6	3.24%	6	3.24%
experim.	5	2.04%	5	2.04%	3	1.22%	3	1.22%	3	1.22%
electronic	4	1.55%	4	1.55%	4	1.55%	4	1.55%	4	1.55%
blues	2	1.50%	2	1.50%	2	1.50%	2	1.50%	2	1.50%
relaxation	0	0.00%	0	0.00%	0	0.00%	0	0.00%	0	0.00%

Table 9.2: Number of songs from each genre where the k-NN classification agrees with Tonlist.is using the regular voting scheme

genre	Iterations									
	1		**2**		**3**		**4**		**5**	
pop	10,839	84.43%	11,077	86.28%	10,251	79.85%	9,883	76.98%	9,697	75.53%
adult/cont.	2,084	36.87%	1,751	30.98%	1,869	33.07%	1,949	34.48%	1,939	34.31%
rock	1,585	39.11%	1,487	36.69%	1,819	44.88%	1,982	48.90%	2,045	50.46%
classic	1,418	85.47%	1,455	87.70%	1,411	85.05%	1,374	82.82%	1,348	81.25%
jazz	512	38.79%	465	35.23%	549	41.59%	579	43.86%	579	43.86%
choirs	487	73.90%	498	75.57%	517	78.45%	528	80.12%	540	81.94%
ice.voc.	294	49.58%	275	46.37%	314	52.95%	332	55.99%	337	56.83%
country	281	19.35%	215	14.81%	226	15.56%	222	15.29%	220	15.15%
clas. voc.	244	42.21%	240	41.52%	252	43.60%	250	43.25%	255	44.12%
accordion	125	49.21%	137	53.94%	152	59.84%	159	62.60%	161	63.39%
poetic	56	15.82%	55	15.54%	56	15.82%	56	15.82%	55	15.54%
hiphop	41	11.45%	28	7.82%	30	8.38%	28	7.82%	30	8.38%
gospel	9	5.45%	9	5.45%	9	5.45%	9	5.45%	9	5.45%
experim.	8	3.27%	6	2.45%	7	2.86%	6	2.45%	7	2.86%
latin	7	3.68%	6	3.24%	6	3.24%	6	3.24%	6	3.24%
electronic	6	2.33%	4	1.55%	4	1.55%	4	1.55%	4	1.55%
relaxation	4	7.23%	4	4.82%	4	4.82%	4	4.82%	4	4.82%
blues	3	2.26%	2	1.50%	2	1.50%	2	1.50%	2	1.50%

Table 9.3: Number of songs from each genre where the k-NN classification agrees with Tonlist.is using the idf voting scheme

9.2.3 Genre Skew

As described in Section 7.5, the popularity of the n^{th} most popular rank in a Ziphian distribution can be estimated with K/n^{α} where K represents the frequency of the most

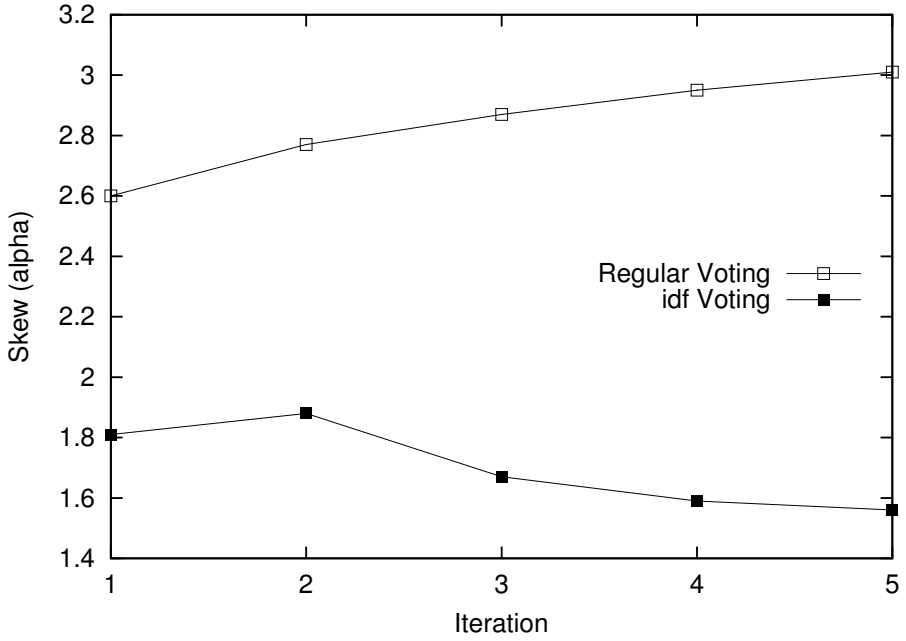

Figure 9.4: The evolution of skew (α) across iterations

popular term and α characterizes the distribution. The lower the α value is, the less skewed the distribution is.

Figure 9.4 shows the evolution of the α parameter of the genre distribution across the five iterations. The figure clearly confirms that the regular voting scheme results in a considerably more skewed distribution than the idf voting scheme. We also observe that while the distribution gets more skewed with each iteration of the regular voting scheme, the converse is true for the idf voting scheme. The distribution is slightly more skewed in the second iteration, but subsequently improves with each iteration.

Note that for the training set, $\alpha = 2.18$. We therefore see that the distribution of the idf voting scheme is always better than that of the training set, while the distribution of the regular voting scheme is always worse.

Table 9.4 shows the confusion matrix for the fifth iteration of the weighted experiments. Please note that the x-axis does not include the genres opera, funk and soul, since no songs from these genres where included in the training set. We see that although the distribution is much less skewed than before, songs are still getting too many votes from the pop genre, with 15,515 songs being classified as pop including the majority hiphop, blues, gospel, funk and soul songs. Several other genres, including blues, adult/contemporary, folk/country and poetic receive very low accuracy scores. There can be three reasons for this:

1 Our features vectors do not manage to describe these genres adequately. However, in our experiments with the Tzanetakis collection, blues songs were classified with 93% accuracy, hiphop songs with 80% accuracy, and country with 66% accuracy, suggesting that this is not the case, at least not for the first two genres. However, here the folk/country genre is of course broader than the country genre of Tzanetakis.

2 The genres themselves are not unique enough to warrant being genres of their own. Again, our earlier experiments suggest that this is not the case for the blues, hiphop and folk/country genres, although it might possibly be the case for the others.

3 Weak training set. For the experiment we selected the training set to be only songs classified correctly according to the Tonlist.is classification with high confidence. No funk and soul songs were included. Only two blues songs were included, and these are indeed the only two blues songs that are correctly classified by the end of the fifth iteration. The same goes for the nine gospel songs and the six latin songs. Five experimental and five electronic songs were included in the training set, and four of each genre survived five iterations. Only 132 folk/country songs were included in the training set, out of 1,452 in the collection.

We also notice that 25 out of the 17 opera songs are classified as either classical vocal or icelandic vocal. The two latter genres receive considerable amount of votes from each other, suggesting that these three genres should perhaps be considered as only one genre.

9.2.4 Convergence

Figure 9.5 shows the convergence of the classification process. Recall that convergence is essentially the inverse of the classification accuracy of a particular iteration with respect to its input. As the figure shows, the regular voting scheme very quickly converges to a stable classification; as shown above, however, that classification is not very good. The idf voting scheme, however, has significantly more transitions between genres, which is the reason for the improved classification. After five iterations, however, less than 4% of the songs move between genres, and the classification is thus relatively stable.

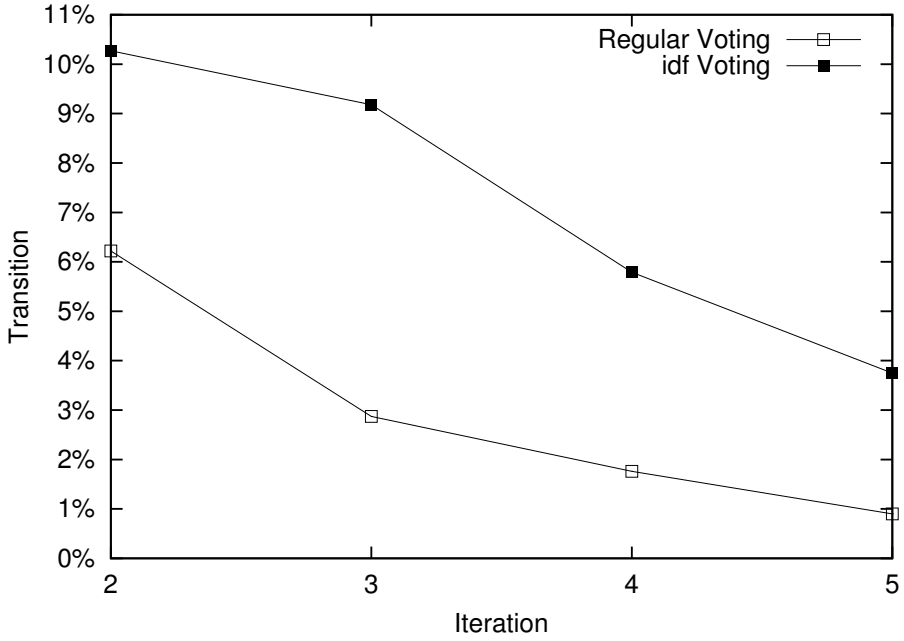

Figure 9.5: Convergence across iterations

9.3 Summary

We have in this chapter described our iterative genre classification experiments on the Tonlist.is collection. We started by explaining how we selected our training-set, and then went on to describe two lines of iterative experiments with one line using regular voting scheme and the other using weighted voting scheme. We saw from this that by using the regular voting scheme our classes converged very quickly, meaning that songs quickly settled on a genre class. However, the distribution was very skewed with too many songs being classified as pop, resulting in the pop genre having a very high classification accuracy and other genres very low. This got worse with each iteration. Using weighted voting changed this. The overall classification accuracy compared to Tonlist.is classification improved, and the distribution was much better.

	pop	**clas.**	**cou.**	**rock**	**jazz**	**hip.**	**blu.**	**ad.c.**	**acc.**	**poe.**	**exp.**	**cl.voc.**	**i.voc.**	**cho.**	**lat.**	**ele.**	**rel.**	**gos.**	**tot.**
pop	**9,967**	878	16	1,140	305	14	0	660	81	7	0	9	3	26	2	0	0	0	12,838
clas.	28	**1,348**	9	8	19	0	0	10	2	7	0	72	54	102	0	0	0	0	1,659
cou.	540	288	**220**	35	46	0	0	238	42	6	0	1	7	28	0	0	1	0	1,452
rock	1,451	270	7	**2,045**	91	2	0	157	14	0	0	7	1	6	1	0	1	0	4,053
jazz	303	296	2	23	**579**	1	0	35	65	1	0	1	9	3	1	1	0	0	1,320
hip.	265	8	0	26	15	**30**	0	14	0	0	0	0	0	0	0	0	0	0	358
blu.	73	16	0	8	16	0	**2**	19	0	0	0	0	0	0	0	0	0	0	133
ad.c.	2,553	388	12	82	139	1	0	**1,039**	66	5	0	5	22	48	2	0	0	0	5,262
acc.	41	12	6	0	10	0	0	23	**161**	0	0	0	0	0	1	0	0	0	254
poe.	61	102	2	0	41	0	0	47	4	**55**	0	10	26	6	0	0	0	0	354
exp.	34	125	2	20	32	1	0	9	3	0	**7**	1	1	6	0	0	4	0	245
cl.voc.	13	139	4	2	4	0	0	17	1	6	0	**255**	120	17	0	0	0	0	578
i.voc.	5	108	0	0	2	0	0	15	0	2	0	71	**337**	53	0	0	0	0	593
cho.	35	53	0	7	1	0	0	10	0	0	0	8	5	**540**	0	0	0	0	659
lat.	68	45	0	0	18	0	0	32	16	0	0	0	0	0	**6**	0	0	0	185
ele.	115	37	0	27	53	10	0	6	3	0	0	0	0	3	0	**4**	0	0	258
rel.	18	43	0	1	11	0	0	4	0	0	0	0	0	2	0	0	**4**	0	83
gos.	124	20	0	7	1	0	0	3	0	0	0	0	0	1	0	0	0	**9**	165
ope.	1	1	0	0	0	0	0	0	0	0	0	10	15	0	0	0	0	0	27
funk	49	4	0	5	5	0	0	5	2	0	0	0	0	0	0	0	0	0	70
soul	41	0	0	5	0	0	3	0	0	0	0	0	0	0	0	0	0	0	50
tot.	15,515	4,182	280	3,436	1,392	59	2	3,246	460	89	7	450	600	841	13	5	10	9	30,596

Table 9.4: Confusion matrix of iteration 5 using the weighted voting scheme

Chapter 10

Conclusions

Many classification tasks and methods are based on low-level audio feature vectors. Traditionally, due to efficiency considerations, each song has been described using a small number of vectors (often only one vector). k-NN classification, on the other hand, is able to use hundreds or thousands of feature vectors per song and still cope well with the scale.

In the second part of this thesis we have examined k-NN genre classification system on a large-scale real life music collection. We have examined the Tonlist.is collection and its usage of genres, paying particular attention to the distribution of genres. We have observed that a considerable part of the collection is classified as pop, and that when using k-NN genre classification on this collection, too many songs get classified as pop.

We started by removing from the Tonlist.is collection some genres that are not suitable for automatic genre classification, since they do not depend on the sound characteristics of the songs. We then went on to define alternative measurements of classification accuracy, since we know that the Tonlist.is classification is not always correct. We chose to examine the convergence, or stability of the classes created by the system, and the genre distribution.

We ran a simple k-NN classification on the collection and achieved 61.08% accuracy measured against the Tonlist.is classification, using only 95 minutes of classification time for over 30,000 songs. From the results we chose only the songs that were correctly classified according to Tonlist.is by with a high confidence. The songs thus chosen where then used as the training set for the first round of our iterative experiments. The results from the first iteration was then used as the training set for the second iteration and so on.

We used this iterative k-NN classification to examine the stability of the classification. We used five iterations and found that by using a regular voting scheme the genres quickly converge to stable classes. However, the distribution of genres was very skewed towards pop, with the skew getting worse at each iteration. We therefore performed another iterative experiment where the votes were weighed by *idf*, giving more weight to genres with very few songs than to the more popular genres like pop. This improved genre distribution and classification accuracy when measured by the Tonlist.is classification. However, using this technique the genres do not converge as quickly. The accuracy measured against Tonlist.is classification was 56.34% in the fifth iteration using the weighted voting scheme compared to 52.87% using the standard scheme.

It is difficult to accurately evaluate the quality of our proposed techniques. The best way to would be to conduct user studies, where users of tonlist.is would be asked to manually classify a subset of the collection. We would then compare this classification to the results of our iterative classifications. Unfortunately this is something we did not have time to do for this thesis, so this is left as future work.

While we have learned that the efficiency of the k-NN classification algorithm allows us to use it on a sizable real world music collection, we have also seen that automatically classifying such collections is not without difficulties and in order for it to work better the following issues should be addressed:

Training Set Selection. Our training set did not represent all genres very well. We saw that the genres that were represented by very few songs in our training set usually received very low accuracy scores in the iterative experiments. We believe that it is necessary to listen to a large number of songs and manually classify with care. This training set collection should include a similar number of songs from each genre, and each genre should include the whole spectrum of songs that we want to be classified to that genre.

Genre Selection. The Tonlist.is collection includes some genres that include very few songs, and it includes some genres that are very related to each other. We recommend that classical vocals, icelandic vocals and opera are merged into one genre. We also recommend looking at the difference between pop and adult/contemporary very carefully. When asking Tonlist.is employees about these genres they could not identify exactly what the difference is, or which songs should go into each genre. Poetic seems to be a genre encompassing a wide variety of music, and we are not sure it warrants being a genre on its own. Soul and funk are examples of genres that either should be skipped or defined more clearly.

Feature Extraction. In the first part of this thesis we saw that the quality of the k-NN classification of the Tzanetakis reference collection is about 12% below the best results in the literature. We believe that the fundamental reason for this lies in the (lack of) quality of the low-level features used to describe the songs. There is a wide collection of feature extraction software available, and many researchers write their own. In order to achieve higher accuracy results it could be beneficial to explore this in much more depth than we were able to do in this thesis. Note, again that the accuracy we achieved is, to the best of our knowledge, the highest accuracy reported using the MARSYAS features.

The focus on this thesis is k-NN genre classification and its efficiency on large music collections. We fully realize that we have made no attempts to examine the plethora of other classification methods. It was simply outside the scope of this work. However, it would be very interesting to examine whether any of the other classifiers can cope with a large music collection like Tonlist.is, and whether they could achieve similar or better results.

Bibliography

Andoni, A., & Indyk, P. (2008, January). Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communications of the ACM*, *51*(1), 117–122.

Benetos, E., & Kotropoulos, C. (2008). A tensor-based approach for automatic music genre classification. In *Proceedings of the 16th Signal Processing Conference(EUSIPCO).* Lausanne, Switzerland: IEEE Press.

Bergstra, J., Casagrande, N., Erhan, D., Eck, D., & Kégl, B. (2006, December). Aggregate features and Adaboost for music classification. *Machine Learning*, *65*(2-3), 473–484.

Bischoff, K., Firan, C., Paiu, R., Nejdl, W., Laurier, C., & Sordo, M. (2009). Music mood and theme classification - a hybrid approach. In *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR).* Kobe, Japan.

Boiman, O., Shechtman, E., & Irani, M. (2008). In defense of Nearest-Neighbor based image classification. In *Proceedings of the 11th IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* Anchorage, Alaska, USA.

Chierichetti, F., Panconesi, A., Raghavan, P., Sozio, M., Tiberi, A., & Upfal, E. (2007). Finding near neighbors through cluster pruning. In *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems.* New York, NY, USA: ACM.

Craft, A., Wiggins, G., & Crawford, T. (2007). How many beans make five? the consensus problem in music-genre classification and a new evaluation method for single-genre categorisation systems. In *Proceedings of the 8th International Symposium on Music Information Retrieval (ISMIR).* Vienna, Austria.

Gjerdingen, R. O., & Perrott, D. (2008). Scanning the dial: The rapid recognition of music genres. *Journal of New Music Research*, *37*(2), 93–100.

Guan, D., Yuan, W., Lee, Y.-K., & Lee, S. (2009, June). Nearest neighbor editing aided by unlabeled data. *Information Sciences*, *179*, 2273–2282.

Gudmundsson, G., Jónsson, B. T., & Amsaleg, L. (2010). A large-scale performance study of cluster-based high-dimensional indexing. In *18th ACM International Conference on Multimedia - Workshop on Very-Large-Scale Multimedia Corpus, Mining and Retrieval.* Florence, Italie.

Hoffman, M., Blei, D., & Cook, P. (2009). Easy as cba: A simple probabilistic model for tagging music. In *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR).* Kobe, Japan.

Holzapfel, A., & Stylianou, Y. (2008). Musical genre classification using nonnegative matrix factorization-based features. *IEEE Transactions on Audio, Speech & Language Processing*, *16*(2), 424-434.

Jones, K. S. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, *28*, 11–21.

Lejsek, H., Ásmundsson, F. H., Jónsson, B. T., & Amsaleg, L. (2009). Nv-tree: An efficient disk-based index for approximate search in very large high-dimensional collections. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*(5), 869–883.

Li, T., Ogihara, M., & Li, Q. (2003). A comparative study on content-based music genre classification. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR).* Toronto, Canada.

Lidy, T., Rauber, A., Pertusa, A., & Iñesta, J. M. (2007). Combining audio and symbolic descriptors for music classification from audio. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR).* Vienna, Austria.

Logan, B. (2000). Mel frequency cepstral coefficients for music modeling. In *Proceedings of the First International Symposium on Music Information Retrieval (ISMIR).* Plymouth, Massachusetts.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. Journal on Computer Vision*, *60*(2), 91–110.

Martens, J. P., Leman, M., Baets, B., & Meyer, H. (2004). A comparison of human and automatic musical genre classification. In *Proceedings of the 2004 IEEE International Conference on Acoustics, Speech, and Signal Processing. (ICASSP)* (Vol. 4). Montreal, Quebec, Canada.

Midi Manufacturers Association Inc. (n.d.). *Midi Manufacturers Association.* Retrieved October 2010, from `http://www.midi.org/`

Miranda, E. (2002). *Computer sound design: Synthesis techniques and programming with cdrom*. Burlington, MA, USA: Rebound by Sagebrush.

Orio, N. (2006). Music retrieval: a tutorial and review. *Found. Trends Inf. Retr.*, *1*(1), 1–96.

Panagakis, Y., Benetos, E., & Kotropoulos, C. (2008). Music genre classification: A multilinear approach. In *Proceedings of the 9th International Symposium on Music Information Retrieval (ISMIR)*. Philadelphia, USA.

Panagakis, Y., & Kotropoulos, C. (2010). Music genre classification via topology preserving non-negative tensor factorization and sparse representations. In *2010 IEEE International Conference on Acoustic Speech and Signal Processing (ICASSP)*. Dallas, TX, USA.

Panagakis, Y., Kotropoulos, C., & Arce, G. (2009). Music genre classification via sparse representations of auditory temporal modulations. In *Proceedings of the 17th European Signal Processing Conference (EUSIPCO)*. Glascow, Scotland.

Philbin, J., Chum, O., Isard, M., Sivic, J., & Zisserman, A. (2007). Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the 10th IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Mineapolis, MN, USA.

Tzanetakis, G., & Cook, P. (1999). Marsyas: a framework for audio analysis. *Organised Sound*, *4*(3), 169–175.

Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. *IEEE Trans. on Speech and Audio Processing*, *10*(5), 293–302.

School of Computer Science
Reykjavík University
Menntavegi 1
101 Reykjavík, Iceland
Tel. +354 599 6200
Fax +354 599 6201
www.reykjavikuniversity.is
ISSN 1670-8539