Computer-executed Genre Classification of Music
Computer Systems Lab 2009-2010

Alex Stabile

April 7, 2010

Abstract

The goal of this project is to write code that can accurately group given pieces of music into appropriate genres. Genres are often generalities that may not perfectly fit a given piece of music, but by analyzing different musical qualities we may determine what genre best describes it and what other pieces of music it is most similar to. Computer code that can accomplish this task could have applications in sorting large libraries of music or suggesting music to individuals based on musical qualities, rather than manually entering information or comparing with common likes and dislikes. For ease of analyzing musical qualities, midi format music files are used as input for the program. Python was used to write classes that can read and store the information contained in midi files, such as note value, duration, and tempo. These data are organized by grouping notes into their appropriate beats. Organizing them in such a manner allows for harmonic analysis, which provides insight into how a piece is written and and what it sounds like. We theorize that this information would be enough to distinguish among basic genres of music. For future research, analyses of other low-level musical qualities may be implemented to provide a fuller picture of a given piece. A neural network will be trained by analyzing harmonic data from a set of music (whose genre is known) to learn how to appropriately categorize music it has not seen before . This project will focus on music written for solo piano, so that harmonic analysis rather than instrumentation will be used to determine genre.

Keywords: music, genre, naive, bayes

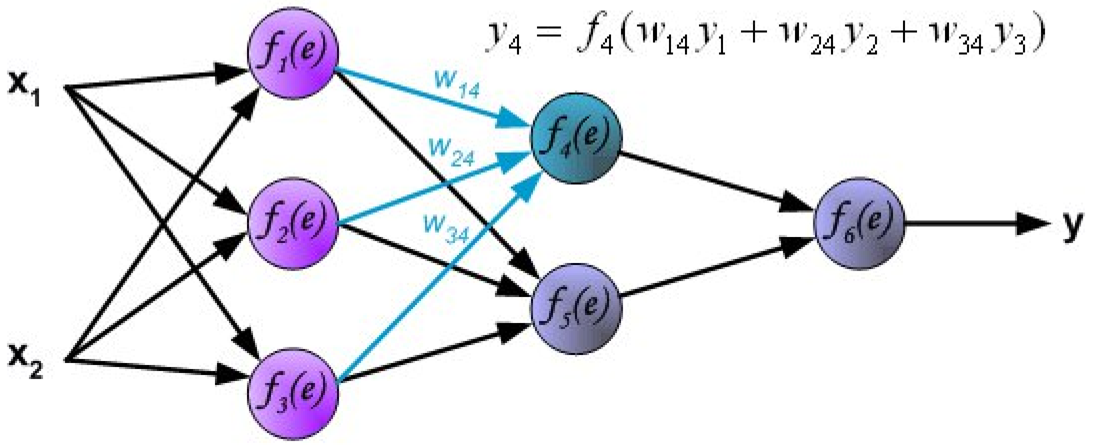
1 Introduction and Background

Current research often uses statistical models to determine how a given piece of music should be categorized. One approach has been to use Inter-Genre Similarity modeling [2]. This project attempted to categorize music by analyzing the timbral textures of a short sample of music. These textures differ due to differences in instrumentation and rhythm. Gaussian Mixture Models were used to create the statistical model, and IGS to cluster similar groups together. Their algorithms grouped a 0.5 to 30.0 second sample of music into one of nine genres. Longer samples yielded better results, up to 64 percent correct. This somewhat low success rate is attributed to the difficulty of machine analysis of sound samples. Music has also been represented through a model of rhythmic complexity [5]. With this method, rhythm was represented in a tree structure and its overall complexity was determined. Results were moderately successful, but the method of organizin notes could be useful in other attempts at classifying music. Another approach attempted an automatic method of classification that is completely general [3]. This was done by looking for mathematical similarity, rather than features specific to music. Midi files were used for musical analysis, and very successful results were shown when grouping music into rock, jazz, or classical genres. Results were moderately successful when attempting to group pieces by composer, but got worse as sample size increased. Interestingly, the algorithm could even cluster like file types together (sorting out java class files, gene sequences from different species, and widely different styles of music). One especially successful experiment was "On musical stylometrya pattern recognition approach" [1]. Different musical aspects of pieces were analyzed, and a statistical model was created to group new pieces into their appropriate period. By analyzing more musical characteristics, their model became more fine-tuned. These characteristics included harmonies, dissonance, note entropy, and types of intervals. No method has yielded or should be expected to yield perfectly successful results even many people cannot successfully place music into its correct genre.

2 Project Design

The first step of this project was to write code for reading and organizing the information stored in a midi file. A midi file is a sequence of commands preceded by how long to wait until they are executed (delta-time). For the purposes of this project, I will only need the note-on and note-off commands and their delta-time values. This will allow the code to determine which notes are playing at the same time and what rhythms are found in the piece. This information is organized by storing it in a Beat class. Each instance of a Beat knows which notes sound on its downbeat and off the beat. It also contains which beat it represents from the piece (beat 1, beat 2, etc.), and can be sorted according to this number. When the notes that sound within a beat are passed to a chord identification method, it returns what chord most accurately represents that combination of notes. This is roughly how harmonic analysis is performed by a human, and harmonies present in a piece of music are a good indicator of what style or genre the music belongs to. Harmonic analysis is now working smoothly. Chord identification code accurately returns the key, quality, and inversion of a given combination of notes (e.g., B Flat Dominant Seventh, Third Inversion). One issue I addressed was the possibility of the notes combining to form more than one type of chord. I did this by modifying my code to prioritize the results based on the lowest note in the chord, allowing it to return consistent results. I also have a method of dealing with non-chord tones. When a group of notes does not form any known chord, my code analyzes every possible combination of three notes from the group. This chooses the most representative key for that beat, yielding an appropriate result (major, minor, dimished, etc.) without being thrown off by non-chord tones. I tested my code by having it analyze a Bach chorale and print out information about each beat. This output includes the beat number, the notes within the beat, and what chord they form. I could verify the information by looking at the music. I have found and fixed many issues with my code thanks to this method of testing. To categorize the music, there are a couple of options. One is using a Nave Bayes classifier. This model treats each probability vector as independent and not correlated with any others. While this is rarely true in data sets, Bayesian classifiers have nevertheless been shown to provide useful results. To apply the Bayes classifier to music, probability vectors would be calculated by analyzing the combinations of notes and harmonies a piece of music contains. However, I initially constructed a neural network to analyze these data. Machine learning methods have been shown to be the most reliable and useful in analyzing music due to their flexibility. Contains different layers of "perceptrons:" nodes that contain an activation value and that are associated with a weight. The input layer of the network is at the top, and consists only of the different inputs the network will receive. In my project, these inputs are the ratios of different types of chords found in a piece. These ratios become the values of the input-layer nodes, and weights are generated randomly for all the nodes. The given values are then "fed" through the network, and are used to calculate values for all the other nodes. The output layer consists of just one node, whose final value will be matched with a genre.

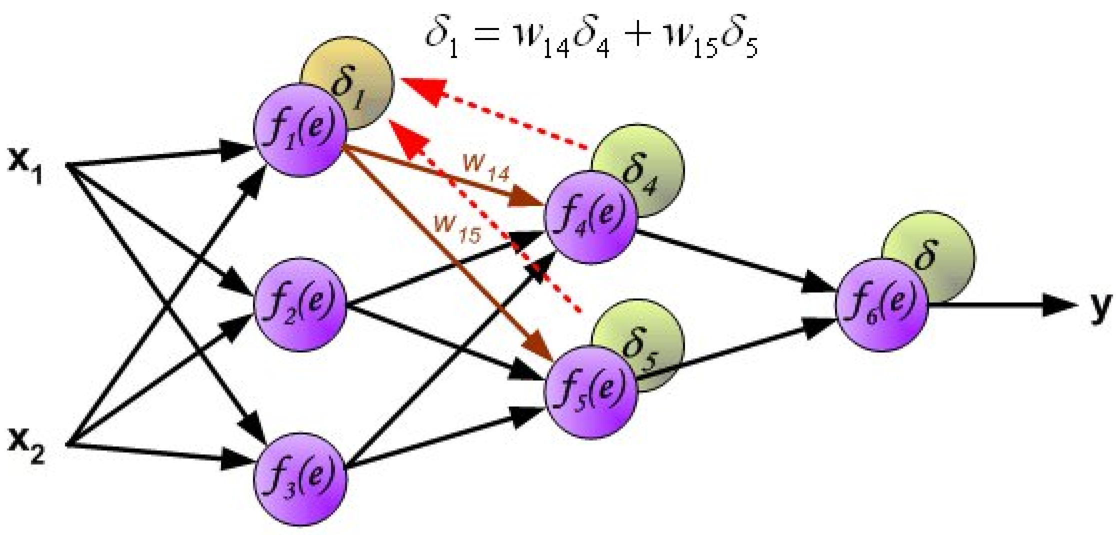
2.1 A Simple Neural Network



Source: http://galaxy.agh.edu.pl/ vlsi/AI /backp_t_en/backprop.html

The network is "trained" by giving it a large data set containing inputs and their respective outputs. When the network produces an output far off from its target, its weights are adjusted to achieve better results. Theoretically, analyzing enough data will yield a network consistent not just with the training data, but also any inputs that may be thrown at it. My project's success will be evaluated by how well the network produces outputs for musical samples it did not analyze in training. Because of the necessary size of the network, a fast learning algorithm is necessary to obtain good results. This project will use the backpropagation method of minimizing error.

2.2 Propagation of weights



Source: http://galaxy.agh.edu.pl/ vlsi/AI /backp_t_en/backprop.html

3 Results

Code successfully inputs a midi file, identifying how many tracks it has, and what commands they contain. The command types, arguments, and delta-time values (how long to wait before executing a given command) are stored. Code then iterates through the commands, identifying note-on and note-off commands and using their delta-time values to group them into their appropriate beats. Each instance of the Beat class knows which notes sound during the beat, and which only sound initially (for harmonic analysis purposes). Once finished, each beat passes the notes it contains to the chord identification method, which returns what chord best represents that combination of notes. When the best harmonies for every beat in the piece have been determined, the total number of beats is counted to find a ratio of each kind of chord to the total number of chords. These ratios will form the values of the input nodes in the neural network. However, my learning algorithm does not yet work correctly.

4 Discussion

This project could be a valuable tool in organizing libraries of music. Genres are often generalities that do not necessarily fit a piece exactly, but this project could determine what music really is similar and should be grouped together. It could also provide interesting insights into how we interpret music and what gives different styles their distinct feel. Another application could be composer identification. Sometimes, new manuscripts of music are discovered, but their composer is not known. Successful implementation of this project could aid in identifying the composer of any newly discovered composition, or composition whose actual composer is in question.

References

[1] Backer, Eric, and Peter van Kranenburg. *On musical stylometry - a pattern recognition approach.* N. pag. Delft University of Technology, 21 Jul. 2004. Web. 19 Jan. 2010

[2] Bagci, Ulas, and Engin Erzin. *Inter Genre Similarity Modelling for Automatic Music Genre Classification.* N. pag. N.p., 18 July 2009. Web. 24 Oct. 2009.

[3] Cilibrasi, Rudi, Paul Vitanyi, and Ronald De Wolf. *Algorithmic Clustering of Music.* N. pag. University of Amsterdam, 24 Mar. 2003. Web. 24 Oct. 2009.

[4] Dannenburg, Roger B., Thom, Belinda, and David Watson. *A Machine Learning Approach to Musical Style Recognition.* 344-347. pag. Carnegie Mellon University, Sep. 1997. Web. 19 Jan. 2010

[5] Liou, Cheng-Yuan, Tai-Hei Wu, and Chia-Ying Lee. *Modelling Complexity in Musical Rhythm.* N. pag. National Taiwan University, 26 Mar. 2007. Web. 24 Oct. 2009

[6] Lu, Cheng-Che, and Vincent Tseng. *A Novel Method for Personalized Music Recommendation.* N. pag. National Cheng Kung University, 2009. Web. 6 Apr. 2010