A Supervised Approach To Chord Recognition

Brad Girardeau Takatoki Migimatsu Pranav Rajpurkar

November 17, 2013

1 Introduction

In classical and popular music, knowing the progressions of chords underlying the melodies is an essential part of understanding, playing, and building on the music. However, analyzing chords by hand can be a time consuming process. An automated tool for this process will save time when learning and analyzing new songs, allowing the development of new interactive musical applications.

Recognizing chords in popular music is a multifaceted challenge, involving segmenting signals of different instruments, determining the location of chord changes, and performing chord or pitch analysis. This project focuses on the last step of this process by recognizing and classifying chords. A learning system is first trained on sythesized segmented chords representing a variety of instruments, then will be expanded to directly recognize chords in popular music.

2 Data Generation

Chord data for the learning system is created programmatically. The data set consists of each inversion, octave, and major or minor type of every chord, giving 288 basic chord examples. A MIDI file containing the pitches for each chord type is generated in Python. Audio data is then created by sythesizing the MIDI files using a given instrument. This allows flexibility in the duration and instrument type of the resulting data set. Training and testing data is generated for several instruments, though for simplicity only one instrument dataset is currently being used.

After exploring results using computer sythesized MIDI chords, non-computer generated audio data will be used. Chords will be recorded played on instruments, and eventually databases of annotated popular music like the McGill billboard corpus will be incorporated. A combination of these datasets will allow creation of a robust, accurate chord recognition system.

3 Feature Extraction

The choice of features greatly influences the success of musical information retrieval algorithms. Typically, an audio sample is loaded and divided into frames of a given window size. A feature extraction engine can then compute a variety of metrics or characteristics of each frame. In this system, the Yaafe python package was used to extract Mel-frequencies cepstrum coefficients (MFCC). While MFCC are not optimal for chord recognition, as MFCC discards much of the tonal pitch data and focuses on timbre, it has been effective in speech recognition as well as other music information retrieval tasks, such as artist or genre classification and pitch detection. As such, MFCC features are useful as part of a baseline system before further feature exploration.

The next step is extracting pitch based features. In particular, the chroma of a pitch refers to the root pitch class that the note belongs to, invarient to octave. For example, the C chroma class contains the C pitch in each octave. Chroma features are ideal for chord recognition because a chord type is also invarient to octave shifts, determined only by the chroma of the contained notes. The Chroma Toolbox is a Matlab toolkit developed by Meinard Müller and Sebastian Ewert that calculates advanced pitch and chroma based features. The Chroma Toolbox inputs WAV audio files and divides them into frames to compute the energy in all MIDI pitches for every frame. It can then calculate the energy in a chroma from the energy in the corresponding pitch bins. More complicated approaches are also supported and will be explored after learning with basic chroma features.

After features are computed for each frame, they can be combined to form a feature vector for a collection of frames or used independently. Training with individual frames does not allow learning about interactions between frames in a given chord, but gives a lower dimensional feature vector, so less data is needed. However, since the window of each frame is short, it is useful to concatenate feature vectors for several frames into one higher dimensional feature vector. Concatenating the features is chosen as a first approach, but training on individual frames will also be attempted to see if it improves performance.

4 Model Training and Testing

Support vector machines (SVM) are used as the learning model for the chord classification task. A testing dataset is created by holding out 20% of the dataset for one instrument, with training on the other 80%. To begin experimenting with classification systems, the chord recognition problem is simplified to the binary classification problem of separating major chords from minor chords. An SVM with RBF kernel

$$K(x, z) = \exp(-\gamma \|x - y\|^2)$$

is initially trained using MFCC features. With typical regularization and kernel parameters ($\gamma = \frac{1}{\#\text{ features}}$ and $C = 1$), the SVM has both high training and testing error, performing little better than chance. Increasing the penalty for misclassifying incorrect training points by increasing γ or C can give a training error of 0, but this results from overfitting, with higher test error. This suggests that MFCC features are not suited to chord recognition tasks, which is explained by the fact that they mostly discard pitch information.

Next an SVM with RBF kernel is trained using basic chroma features. Default γ and C again perform only slightly better than chance in determining major vs minor chords. In this case, however, increasing γ and C increases both training and testing accuracy. With $\gamma = 1$ and $C = 100$, error against the test set is 3.3% (with training error 0). This indicates that chroma features perform much better than non-pitch features. A multiclass SVM is then trained, using a one vs many approach, to classify specific chords (choosing between 24 major and minor chords, ignoring inversions) on the same data. With similar parameters for γ and C, test error is 8% (with training error 0), again indicating good performance with chroma features. There is still a possibility of overfitting, as only a specific computer synthesized instrument dataset is used.

5 Next Steps

We will next use testing and training datasets with different instruments to see if similar accuracy can be obtained in a more general setting. Multiple instruments will also give a much larger dataset to work with. We will also investigate different features, in particular features built on top of the basic chroma representation. In addition, we will train and test with audio samples of a chord played for different durations. Finally noisy chord data recorded or extracted from popular music will be added to the dataset to test the generality of the chroma based approach with SVM classification.