A Supervised Approach To Musical Chord Recognition

Pranav Rajpurkar PRANAVSR@STANFORD.EDU
Brad Girardeau BGIRARDE@STANFORD.EDU
Takatoki Migimatsu TAKATOKI@STANFORD.EDU
Stanford University, Stanford, CA 94305 USA

Abstract

In this paper, we present a prototype of an online tool for real-time chord recognition, leveraging the capabilities of new web technologies such as the Web Audio API, and WebSockets. We use a Hidden Markov Model in conjunction with Gaussian Discriminant Analysis for the classification task. Unlike approaches to collect data through web-scraping or training on hand-labeled song data, we generate symbolic chord data programmatically. We improve the performance of the system by substituting standard Chroma features with a novel set of Chroma DCT-Reduced log Pitch features to push test accuracy on clean data to 99.19%. We finally propose a set of modifications to have the system predict with speed and accuracy in real-time.

1. Introduction

There is significant value in an automated tool to determine chords from audio. Knowing the progressions of chords underlying the melodies is an essential part of understanding, playing, and building on the music. To a curious learner of music, such a tool creates the opportunity to play a new pop song without meticulously hand-labelled chord tags. Equally useful to a learner is being able to receive feedback concerning the accuracy with which a chord was played, making such a system a good automated feedback tool, capable of being plugged into an online music course. To a song writer, the system is useful for exploring chords supporting the melodic content of a song.

Furthermore, the use of such a system extends into other machine learning tasks. The tasks of identifying a song from its waveform data, and of classifying its genre can be linked to finding the chord progressions underlying its harmonic content. Hand-labelling chord names and marking chord changes in a song takes a lot of manual time and effort. An automated tool for this process saves time, and allows the development of new musical tools and research.

There has been progress in chord recognition research. A few have built real-time systems that have shown to achieve promising results (Fujishima, 1999; Cho & Bello, 2009). However, these have not leveraged the web to make a chord-recognition system accessible online. We build a real-time online chord recognition system that makes use of modern HTML5 capabilities such as the WebAudio API and WebSockets, and detail the offline training strategies and online challenges posed by the novel adaptation.

2. Data Generation

Chord prediction is a multiclass classification task. In music, a chord is a set of notes played simultaneously. We choose the minor and major chords, the two most common sets of chords in popular music to classify on. Using the traditional twelve pitch scale (C, C#, D, D#, E, F, F#, G, G#, A, A#, B), we have 24 such distinct chords.

There are different ways of playing the same chord. The C Major triad, for instance, is the set of three notes C, E, and G played simultaneously. On a piano, these chords can be played on different octaves. For example, C Major in the fourth octave would have the notes C4, E4, and G4. Each chord also has inversions defined by the lowest note of the chord - E4, G4, and C5 make up the first inversion of the same C Major chord, and G4, C5, E5 make up the second inversion.

To train the system, we generate training data programmatically. This has been found to have advantages over hand-labelling song data in its ability to generate sufficient training data (K. Lee, 2006). We generate MIDI files for each of the 24 chords, taking into account 8 octaves, and 3 inversion forms, to generate a total of 576 MIDI files. We then use an audio synthesizer "Timidity++" to convert the 576 generated MIDI files, in conjunction with 3 soundfonts for piano, guitar, and violin, to generate a total of 1728 audio files in WAV format.

In musical chord recognition, feature extraction operates over frames. The generated WAV files, which are an average of 4 seconds in length, are first split into frames of window size 100ms each, and an n-dimensional feature vector is extracted for each frame. We label each frame with the label of the chord on its corresponding sound file, to generate 69,120 examples. We use 80% of the data as our training set, and 20% as our testing set.

3. Paralleling Speech Recognition

The pipeline of a chord recognition system is similar to that of a speech recognition one and relies on techniques that were originally applied to speech recognition tasks. The use of the Hidden Markov Models (Young, 1994), and the division of a sound file into frames on which the prediction task is performed, are two such techniques which have been reproduced in identifying chords. However, the task of finding chords is also different from speech tasks in a few ways, and these differences can be exploited to specialize a system in the task of chord recognition.

3.1. Feature Extraction

One important difference between the two surfaces in the choice of features for the tasks. Mel-frequency cepstrum coefficients (MFCC) have been the dominant features in speech recognition systems. These represent the short-term power spectrum of a sound. It has been found that MFCCs are closely related to timbre - a characteristic that captures the quality or tone of sound, e.g., the tonal difference between an oboe and a cello. Since they discard the pitch content of the sound, MFCCs have traditionally been seen as poor features for chord recognition, but are useful in setting a baseline benchmark for such a system.

Chroma features are commonly used for chord recognition tasks (Fujishima, 1999). It is a representation in which the the entire spectrum of sound frequencies is distributed into 12 bins representing the traditional twelve pitch scale. An advantage of Chroma features is that they are invariant to octaves and inversions. We use the Matlab Chroma Toolbox to extract Chroma features for the frames (Muller & Ewert, 2011). Figure 1 shows the extracted Chroma features for the C Major chord. The energy spikes at C, E, and G, the notes constituting the C Major chord, supporting the idea that Chroma features encode the harmonic content of a chord.

Table 1. Accuracies for MFCC and Chroma on the binary classification task of distinguishing between major and minor chords

Features	Test Accuracy
MFCC	51.0%
Chroma	97.7%

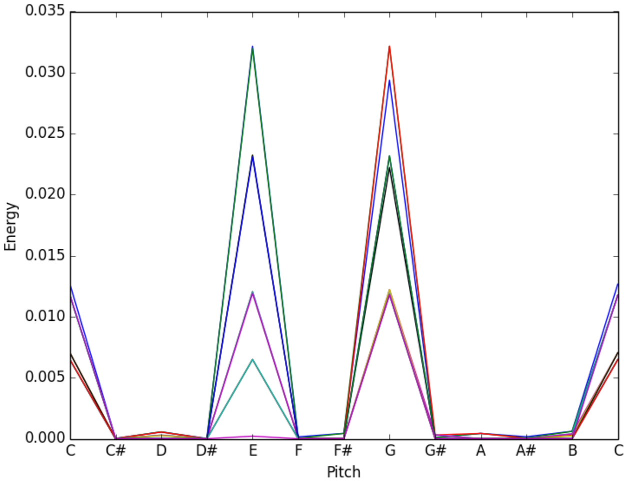

Figure 1. Chroma features for C Major in various octaves and inversions

To test the performance of Chroma features against MFCC features, we start with a binary classification problem of distinguishing major chords from minor chords. An SVM with RBF kernel

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

is trained, with regularization and kernel parameters ($\gamma = 1$ and $C = 100$). Table 1 summarizes the results, and confirms that chroma features are much better suited to the task of chord recognition than MFCCs.

4. Initial Models

4.1. Frame Model

With Chroma established as good features for the chord recognition task, we can now extend to the multiclass classification problem of determining the exact

Table 2. Softmax Regression frame model train and test accuracies

Set	Accuracy
Traning	51.2%
Testing	32.1%

Table 3. Comparisons of accuracies of mixer models with softmax frame model

Model	Test Accuracy
Middle Frame	6.7%
Max Count	33.3%
Independence Mixer	48.3%

chord played. We first use multinomial logistic regression, also called softmax regression, as our initial frame model. The frame model is responsible for making predictions on individual frames. Table 2 shows the accuracies achieved by the softmax classifier on the training and test set.

4.2. Mixer Model

Our frame model outputs a prediction for each frame. Our final classification task, however, is on an audio file, which consists of a sequence of f frames. Let us first make the simplifying assumption that a test sound file consists of a single chord being played.

We now define a mixer model, which is a model for collecting and using the results on individuals frames outputted by the frame model. A simple mixer model, the Middle Frame model, outputs the result for the entire file based on the frame model's output for middle frame in the file. Another simple model, the Max Count model, counts the most frequent prediction made across all of the frames.

Consider another such model, we call the Independence Mixer model, which assumes that the prediction on each frame is independent of the prediction on other frames. The probability that chord y is the single chord played in the file is calculated by considering the probability that y is the chord played at each frame. For a test example, our predicted output is $y_p = \underset{y}{\text{argmax}} p(y|X) = p(y|x_1, x_2, ...x_f) = \prod_{i=1}^{f} p(y|x_i)$.

Note that each $p(y|x_i)$ is given by our softmax frame model. The accuracies achieved with different mixer models are summarized in Table 3.

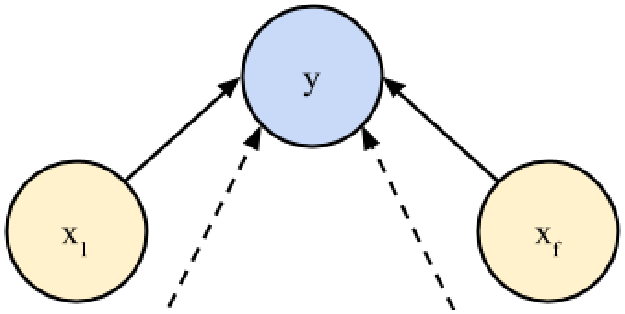

Figure 2. Bayesian network of Independence Mixer model

Table 4. GDA frame model accuracies

Set	Accuracy
Traning	67.8%
Testing	53.9%

5. Improved Models

5.1. Improved Frame Model

Softmax regression is a learning algorithm that models $p(y|x)$, the conditional distribution of the chord given the extracted frame features. We now look at a frame model that tries to model $p(x|y)$ and $p(y)$: Gaussian Discriminant Analysis (GDA) (Jiang et al., 2011). We assume all of the gaussians share the same covariance matrix: $x|y = i \sim \mathcal{N}(\mu_i, \Sigma)$. Furthermore, since we aim to make the system independent of any specific genre, we model $p(y) = 1/24$, a model in which all chords are equally likely. Table 4 summarizes the classification accuracies.

5.2. Improved Mixer Model

Earlier, we had imposed the constraint that chords could not change in a WAV file. Our next model allows us to drop that constraint. We now use a Hidden Markov Model(HMM) to predict the chord sequence in sound files, allowing us to determine chord changes in a file (A. Sheh, 2003).

Firstly, we use our our GDA frame model to model the emission probabilities $p(x|y)$ for the HMM. While state transitions for the HMMs are usually learned in chord recognition tasks (K. Lee, 2006), since each genre of music has a different distribution of transitions, assuming uniform state transitions allows us to remain flexible to any genre of music. We determine the most likely state sequence by Viterbi decoding. Table 5 summarizes the accuracies achieved by the improved

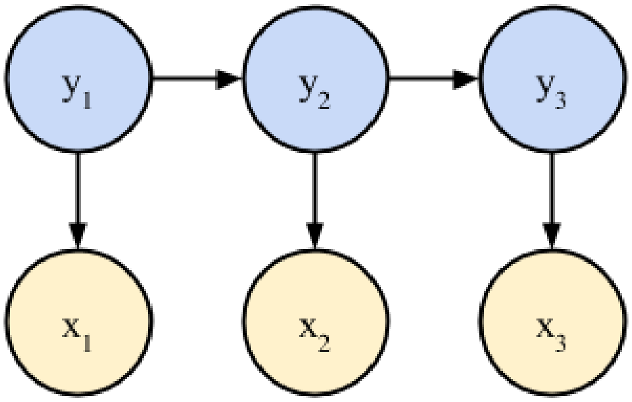

Figure 3. Hidden Markov Model to determine most likely chord state sequences

Table 5. HMM accuracies training and testing on different data sets

Training Data	Testing Data	Accuracy
Piano	Piano	97.01%
Piano	Guitar	99.46%
Piano	Violin	72.15%
All	All	90.68%

mixer model trained and tested on different sets of instruments.

Figure 4 shows the confusion matrix for the Mixer Model trained and tested on generated Piano audio. The most common misclassifications are ones between major chords and the corresponding minor chords. This is explained by the fact that a major and a minor chord of the same key have two out of three notes in common.

6. Improving Features

Chroma features, in their invariance to octave and inversions, make good features for the chord recognition task. To boost the accuracy further would require features invariant of instruments. CRP (Chroma DCT-Reduced log Pitch) has been recently introduced as a chroma-based audio feature that boosts the degree of timbre invariance (Jiang et al., 2011). The general idea is to extract Chroma features, and then discard timbre-related information similar to that expressed by MFCCs, in effect leaving the information related to pitch. Table 6 summarizes the accuracies achieved by the new CRP features in relation to the Chroma features.

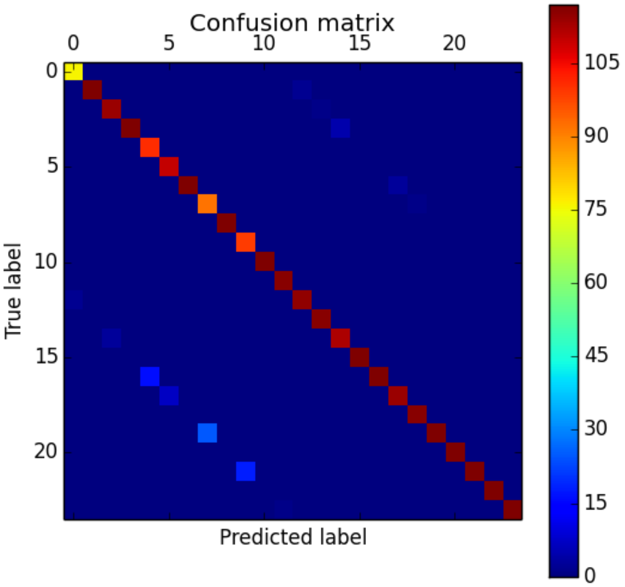

Figure 4. Confusion matrix for HMM training and testing on Piano

Table 6. New scores after replacing Chroma with CRP

Training Data	Testing Data	Accuracy
Piano	Piano	98.72%
Piano	Guitar	99.96%
Piano	Violin	98.54%
All	All	99.19%

7. Live System Considerations

Our real-time system makes use of the Web Audio API to capture live audio. Every 800ms, we encode the audio in WAV format, and transfer it to a server using WebSockets. We then extract features for every 100ms frame in the WAV file, and predict the most likely chord sequence in the 800ms using the HMM.

7.1. Handling Noise

A live online system, while pushing the extents of possible uses of the system, presents new challenges for chord recognition. One of the most important considerations for a live system to take into account is noise. It is ideal for a system to not predict any chord in when there are no chords being played.

Once the HMM returns that most likely chord sequence, we are able to determine whether the 800ms segment is a noise or a chord. Using the knowledge that it usually takes a few seconds before chords

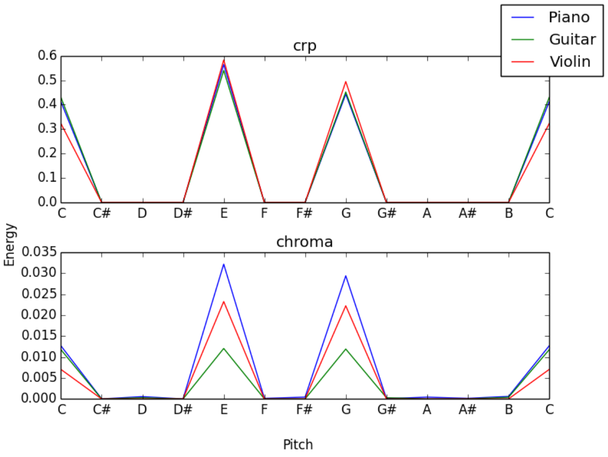

Figure 5. CRP vs. Chroma Features for different instruments. Note the invariance of CRP to instruments.

change, we can then post-process the output by looking at the number of times chord changes were predicted by the HMM in the 800ms segment. If any chord lasts for more than 400ms in the prediction, then we output the chord as our prediction for the 800ms segment. Otherwise, we understand that segment of sound as consisting of noise.

800ms was found to be the optimal time interval over which to process the audio. Increasing the interval for collecting the recording from 800ms was found to create a noticeable delay in the live system. On the other hand, reducing the window time not only decreased accuracy, but also made it harder to distinguish between noise and chords.

8. Conclusion

In this paper, we present an implementation of an online, real-time chord recognition system using modern web technologies. We show the effectiveness of Hidden Markov Models with Gaussian emissions in classifying chords. Furthermore, we show how timbre-invariant CRP features can improve robustness compared to Chroma. Finally, we detail a strategy for noise detection to create an effective live recognition system. With this system, we improve the feasibility of live automatic chord recognition, and pave the way for making this technology more accessible to the public.

References

A. Sheh, D. P.W. Ellis. Chord segmentation and recognition using em-trained hidden markov models. Technical report, LabROSA, Columbia University, 2003.

Cho, T. and Bello, J. P. Real-time implementation of hmm-based chord estimation in musical audio. Technical report, Music and Audio Research Laboratory, New York University, 2009.

Fujishima, T. Realtime chord recognition of musical sound: a system using common lisp music. In *Proceedings of the 1999 International Computer Music Conference (ICMC 1999)*, pp. 464–467, 1999.

Jiang, N., P. Grosche, V. Konz, and Muller, M. Analyzing chroma feature types for automated chord recognition. Technical report, Saarland University and MPI Informatik, 2011.

K. Lee, M. Slanley. Automatic chord recognition from audio using a supervised hmm trained with audio-from-symbolic data. Technical report, CCRMA, Stanford University, 2006.

Muller, M. and Ewert, S. Chroma toolbox: Matlab implementations for extracting variants of chroma-based audio features. In *Proceedings of the International Conference on Musical Information Retrieval (ISMIR)*, 2011.

Young, S. J. The htk hidden markov model toolkit: Design and philosophy. Technical report, Entropic Cambridge Research Laboratory, Ltd, 1994.