# 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 system by substituting the usually tried Chroma features with a novel set of Chroma DCT-Reduced log Pitch features to push test accuracy on clean data to 99.96%. We finally propose a set of modifications to have the online system achieve a good balance between speed and accuracy.

## 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 be able to play a new pop song without requiring good-quality 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 in to an online MOOC class. To a song writer, the system is useful in being able to explore chords supporting the melodic content of the song.

*Proceedings of the $30^{th}$ International Conference on Machine Learning*, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28. 

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 the genre of song can be linked to finding the chord progressions underlying the harmonic content of the song. 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 INSERT PAPER LINKS. 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 the novel 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

The chord prediction task is a multiclass classification task. In music, a chord is a harmonic set of notes sounding 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 Cmaj chord, for instance, is the set of three notes C, E, and G played simultaneously. On a piano, these notes can be played on different octaves. Played on the fourth octave, the Cmaj would have the notes C4, E4, G4. It is also possible to play Cmaj in with E as the lowest note - E4, G4, C5 (first inversion form) and with G as the lowest note G4, C5, E5 (second inversion form).

To train the system, we generate training data programmatically in Python using INSERT TOOLKIT INSERT REFERENCE. 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 these 576 midi files in conjunction with 6 soundfonts for piano, guitar and violin to generate a total of 3456 sound files.

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 10ms each, and a 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  1 million training examples.

## 3. Paralelling Speech Recognition

The pipeline of a chord recognition system is very 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, 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) are commonly used as features in speech recognition systems INSERT CITATION WHERE USED BEFORE. These represents the short-term power spectrum of a sound. It has been found that MFCCs are closely related to timbre - quality that is associated to being able to distinguish between a voice, string instrument, wind instrument, and percussion instrument. These have traditionally been seen as bad features for chord recognition, since they discard the pitch content of the sound, and are simply used to set benchmarks for other features.

Chroma features are commonly used for chord recognition tasks. INSERT CITATION WHERE USED BEFORE. 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 invariant to octaves and inversions. These features also possess a reasonable level of robustness to changes in instruments. We use the Matlab Chroma Toolbox to extract Chroma features for the frames.

*Table 1.* Classification accuracies for MFCC vs Chroma on binary classification task of distinguishing major and minor chords

| Features | Test Accuracy |
|---|---|
| MFCC | 51.0% |
| Chroma | 97.7% |

*Table 2.* Softmax Regression frame model accuracies on an 80/20 train/test split

| Set | Accuracy |
|---|---|
| Traning | Insert% |
| Testing | Insert% |

Figure 1 shows the extract Chroma for C major. Note that the spikes at INSERT correspond to spikes at C, E, and G, the notes constituting the C major. This supports the idea that Chroma encodes the harmonic content of sound.

INSERT CHROMA IMAGE FOR CMAJ 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 shows the results, and confirms that chroma features are much better suited to the task of chord recognition than MFCCs are.

## 4. Initial Models

### 4.1. Frame Model

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

Table 3. Comparisons of accuracies of mixer models

| Model | Test Accuracy |
|---|---|
| Middle Frame | Insert% |
| Max Count | Insert% |
| Independence Mixer | Insert% |

### 4.2. Mixer Model

Our frame model outputs a prediction for each frame. Our final classification task, however, is on a sound file, which is a collection of frames. 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 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. Let us first assume that a test sound file consists of a single chord being played. We will drop this assumption later, but for now, the simple case allows us to build up to the more complex one. Having imposed the constraint that there are no chord changes within a wave file, we now assume that the prediction on each frame is independent of the prediction on other frames. The probability that chord y is played in the file is calculated by considering the probability that y is the chord at each frame. For a test example, our predicted output is $y_p = \underset{y}{\text{argmax}} p(y|X) = \prod_{i=1}^{f} p(y|x_i)$, where f is the number of frames in the file. Note that $p(y|x_i)$ is given by our softmax frame model. The accuracies with different Mixer Models are shown in table 3.

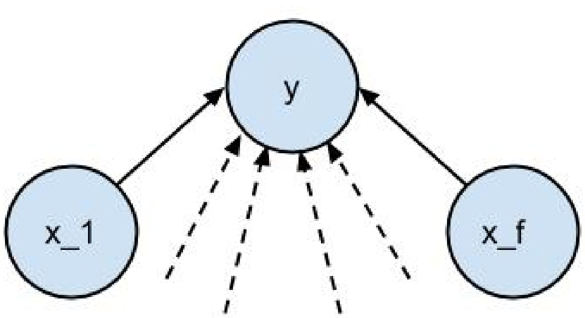


Figure 1. Bayesian network of mixer model

## 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 model that tries to model $p(x|y)$ and $p(y)$: Gaussian Discriminant Analysis (GDA). INSERT CITATION WHERE USED BEFORE. We model $p(x|y)$ using a multivariate normal distribution. Although we model each gaussian with different means, we assume they all share the same covariance matrix: $x|y = i \sim \mathcal{N}(\mu_i, \Sigma)$. Since our aim is to make the system independent of any specific genre, we model $p(y) = 1/24$, a model in which all chords are equally likely. With this new generative model, we can now expand our mixer model.

### 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 loosen that constraint. We now use a Hidden Markov Model to predict the chord sequence in sound files, allowing us to determine chord changes in a file. INSERT CITATION WHERE USED BEFORE. Firstly, the emission probabilities $p(x|y)$ are modelled by our GDA frame model. While state transitions are usually learned in chord recognition tasks, INSERT CITATION WHERE USED BEFORE, 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 the Viterbi decoding. Table 4 summarizes the accuracies acheived by the improved mixer model trained on different sets of instruments.

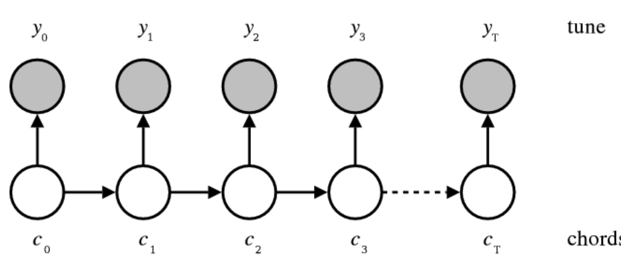


Figure 2. Hidden Markov Model to determine most likely chord state sequences

## 6. Improving features

Chroma features, in their invariance to octave and inversions, make great features for the chord recognition

*Table 4.* HMM accuracies training and testing on different data sets

| Training Data | Testing Data | Accuracy |
|---|---|---|
| Piano | Piano | Insert% |
| Guitar | Guitar | Insert% |

*Table 5.* Chroma vs CRP

| Training Data | Testing Data | Accuracy |
|---|---|---|
| Piano | Piano | Insert% |
| Guitar | Guitar | Insert% |

task. To boost the accuracy further would require features were invariant of instruments. CRP (Chroma DCT-Reduced log Pitch) is a chroma-based audio feature that boosts the degree of timbre invariance. 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 5 summarizes the accuracies achieved by the new CRP features in relation to the Chroma features.

INSERT GRAPH SHOWING CHROMA INVARIANCE TO INSTRUMENTS VS CRP INVARIANCE

## 7. Live system consideration

A live system presents new challenges for chord recognition, but also expands the possibilities of In a live system, one of the most important considerations to take into account is noise. Even while running the system on songs, there are segments that consist of percussion, or silence. It is important for a system to not predict any chord in this timespan. Challenging is that we don't know what's about to come next.

## References


Author, N. N. Suppressed for anonymity, 2011.

Duda, R. O., Hart, P. E., and Stork, D. G. *Pattern Classification.* John Wiley and Sons, 2nd edition, 2000.

Kearns, M. J. *Computational Complexity of Machine Learning.* PhD thesis, Department of Computer Science, Harvard University, 1989.

Langley, P. Crafting papers on machine learning. In Langley, Pat (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Michalski, R. S., Carbonell, J. G., and Mitchell, T. M. (eds.). *Machine Learning: An Artificial Intelligence Approach, Vol. I.* Tioga, Palo Alto, CA, 1983.

Mitchell, T. M. The need for biases in learning generalizations. Technical report, Computer Science Department, Rutgers University, New Brunswick, MA, 1980.

Newell, A. and Rosenbloom, P. S. Mechanisms of skill acquisition and the law of practice. In Anderson, J. R. (ed.), *Cognitive Skills and Their Acquisition*, chapter 1, pp. 1–51. Lawrence Erlbaum Associates, Inc., Hillsdale, NJ, 1981.

Samuel, A. L. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):211–229, 1959.