

MUSICAL CHORD RECOGNITION

GdKI Project report



20. SEPTEMBER 2022

POLINA SHAKHOVA

NGOC THAO TRAN

Inhalt

[Introduction](#_heading=h.gjdgxs) 2

[Project methods](#_heading=h.30j0zll) 3

[Overall method:](#_heading=h.1fob9te) 3

[Model input features:](#_heading=h.3znysh7) 3

[Model creation:](#_heading=h.2et92p0) 4

[Convolutional neural network architecture (CNN)](#_heading=h.tyjcwt) 4

[Recurrent neural network (RNN) / Long Short-term Memory model (LSTM)](#_heading=h.3dy6vkm) 5

[Experiments and results](#_heading=h.1t3h5sf) 6

[Experiment](#_heading=h.4d34og8) 6

[Dataset](#_heading=h.2s8eyo1) 6

[Library](#_heading=h.17dp8vu) 6

[Evaluation of results](#_heading=h.3rdcrjn) 7

[CNN Model](#_heading=h.8vmp7lez1ro1) 7

[Signal :](#_heading=h.4adq8kdxnflx) 7

[STFT:](#_heading=h.m3i9ubefs1c6) 8

[Comparison between STFT and Signal:](#_heading=h.r2g1gi1attya) 8

[RNN/LSTM Model](#_heading=h.xwtfq8i6549i) 9

[MFCC:](#_heading=h.hqq6s7t7ob5e) 9

[STFT:](#_heading=h.urkgsuk1hehr) 10

[Discussion](#_heading=h.26in1rg) 11

[References](#_heading=h.lnxbz9) 11

# Introduction

The problem of automatic chord recognition has been known since the last century. Automatic chord recognition systems are widely used in many areas, for example for music generation, for classification of the music into various categories (genre, mood), for song identification, etc.

Nowadays there are many conferences, competitions, and other events in the field of Music Information Retrieval (MIR), including the annual ISMIR conference to exchange ideas and innovations related to this field, MIREX competition of algorithms for musical chord recognition, etc. However, there is no unambiguous and completely optimal solution of this problem at the moment and the existing solutions need improvements.

A lot of research in this area has focused on deep neural networks: convolutional (CNN) [4, 5], recurrent (RNN) [2], long short-term memory (LSTM) [7]. The following issues were noticed in reviewed works:

• training datasets usually consist of songs of one artist or one genre, that can affect the model’s quality of recognition on the real data [1, 2];

• model input is usually audio features, that require to store a large amount of data [2, 4, 7];

• some of the models are able to recognize only a small number of chords [3].

The purpose of the project is to investigate the application of the convolutional neural networks (CNN) or Recurrent Neural Network (RNN) to recognize musical chords from music audio recordings based on the work done by Polina Shakhova in her bachelor thesis. Her model is able to classify 25 types of chords: 12 minor and 12 major chords and 1 “non-chord” type. A set of Mel-Frequency Cepstral Coefficients or MFCC extracted from each audio signal is used as model input features in this model and it is implemented using Python, Librosa and TensorFlow libraries.

In this project the model will be modified in 3 ways. In the first way, model input features are a set of **Short-time Fourier transform** (**STFT)** which are extracted from each audio signal. In the second way, the model input features are raw audio signals without transforming. In the third way, RNN /LSTM model will be used for training MFCCs and STFTs features dataset. Apart from that, CNN model network structure can also be changed accordingly to yield better training results.

# Project methods

## Overall method:

In general, the task of recognizing song chords can be defined as: for a given sound signal *x(t)*, where *t ∈* *[tstart, tend]* and a set of possible chord classes *Y*, for each moment of time *t* it is necessary to determine the chord that sounds at that moment.

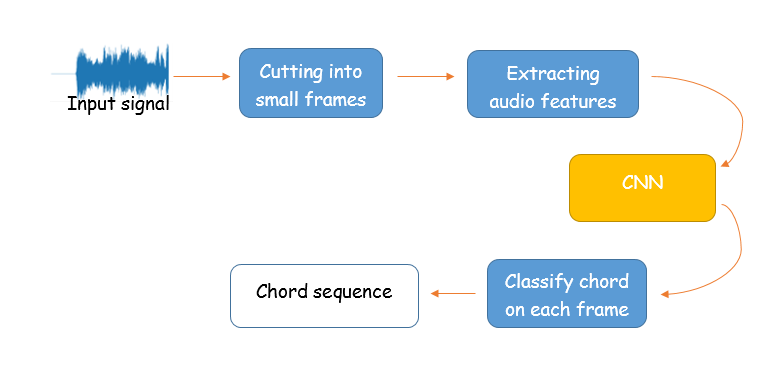


Figure 1. General system structure

Chords in each song transcription file were simplified to 25 (or 28) [[1]](#footnote-1)basic chord classes. Both audio and chord transcription files were aligned by time and cut into small ( 0,2 seconds) frames. Below are the values of metrics used to extract the audio file:

SAMPLE\_RATE = 22050 (every second, 22050 samples are taken, default value by STFT)

N\_FFT = 2048 (number of samples per FFT, default value by STFT)

HOP\_LENGTH = 512 (length of the non-intersecting portion of window length)

WINDOW\_SIZE = 0.2 (length of frames)

After preprocessing audio data, results were saved into JSON-files.

## Model input features:

* STFTs (**Short-time Fourier transform)**:

STFTS involves dividing a longer time audio signal into shorter segments of equal length (frames) and then compute the Fourier transform separately on each frame. This reveals the Fourier spectrum of each shorter segment. STFTs is visualized as spectrogram. [8]

Since STFTs require so much more memory space than MFCC and the original dataset is quite large, the total numbers of songs are reduced, from 379 songs to around 30 songs.

* Raw audio signal

Signals are not usually used as model input features for audio deep learning, because of the heavy memory space it needs. Therefore the total number of songs are reduced to around 30.

* MFCCs (**Mel** **frequency cepstral coefficients)**

There are around 370 songs used for MFFCs model training.

## Model creation:

### Convolutional neural network architecture (CNN)

Convolutional Neural Networks (CNNs) are widely used in image classification, but they also have shown very good results in audio processing (speech recognition, music identification). It consists of an input layer, hidden layers and an output layer. In CNN, hidden layers include layers which perform convolutions.

Below are the structures of the CNN model for different types of input features:

* STFT as input features:

|  |  |
| --- | --- |
| Convolution | 9 × 1025 × 16 |
| × 2 layers | |
| Max pooling | 2 x 2 |
| Convolution | 5 × 513 × 16 |
|  | |
| Max pooling | 2 x 2 |
| Flatten | 12336 |
| Fully-connected | 16 |
| Softmax | 25 |

The output of each convolutional layer is activated with the ReLU function. Batch normalization is performed after two first max pooling layers. Dropout with probability 0.2 is applied after the second batch normalization layer and fully connected layer.

* Signal as input features:

|  |  |
| --- | --- |
| Convolution | 6615 × 1 × 32 |
|  | |
| Max pooling | 2 x 2 |
| Convolution | 1654 × 1 × 32 |
|  | |
| Max pooling | 2 x 2 |
| Convolution | 414 × 1 × 32 |
|  | |
| Max pooling | 2 x 2 |
| Flatten | 3328 |
| Softmax | 25 |

The output of each convolutional layer is activated with the ReLU function. Batch normalization is performed after each max pooling layer.

### Recurrent neural network (RNN) / Long Short-term Memory model (LSTM)

Recurrent neural network (RNN) is an artificial neural network which uses sequential data or time series data. Its deep learning algorithms are commonly used for ordinal or temporal problems, such as language processing, speech recognition or image captioning. But it can be difficult to train standard RNNs to solve problems that require learning long term temporal dependencies.[9]

Long Short Term Memory networks (LSTMs) are a type of Recurrent Neural Network (RNN) that uses special units in addition to standard units. LSTM units include a ‘memory cell’ that can maintain information in memory for long periods of time. This memory cell lets them learn longer-term dependencies. [9]

In the project LSTM will be used as RNN instead of RNN standard model.

Below are the structures of the RNN model for different types of input features:

* MFCC as input features:

|  |  |
| --- | --- |
| lstm | 9 × 32 |
|  | x 2 layers |
| Fully-connected | 32 |
| Fully-connected | 25 |

Dropout with probability 0.3 is applied after the first fully connected layer.

* STFT as input features:

|  |  |
| --- | --- |
| lstm | 9 × 32 |
|  | x 2 layers |
| Fully-connected | 32 |
| Fully-connected | 25 |

Dropout with probability 0.3 is applied after the dense layer.

# Experiments and results

## Experiment

### Dataset

In addition to the Isophonics collection of 180 transcribed The Beatles' songs, which is traditionally used for chord recognition tasks, the training set in this work is extended with also popular Isophonics Queen dataset and a collection of popular rock and pop songs presented at one of the stages of MIREX competition. The total dataset consists of 379 songs with their transcriptions stored in the format: Start\_time – End\_time – Chord with .lab format. The audio are stored in .wav format.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Figure 2. Example of chord transcription file

### 

### Library

**TensorFlow** (keras: for training model)

**Librosa** (for preprocessing audio data before giving them to the model)

**Python**

## Evaluation of results

The main metrics for evaluating the results of training Neural Network are accuracy function and loss function. Loss function shows the average cost of training for the epoch, and accuracy function - the amount of correctly classified data. For both CNN and RNN models loss function is “categorical\_crossentropy”

Dataset is split into a train and test dataset. Test dataset is around 20-30 percent derived from the whole dataset, and is used to unbiased evaluate the final model fit on the training dataset. Test dataset is not trained by the model.

In the figures below, blue line represents value from train dataset, and orange line represents value from test dataset.

### CNN Model

#### Signal :

Number of Epochs: 10

Batch Size : 64

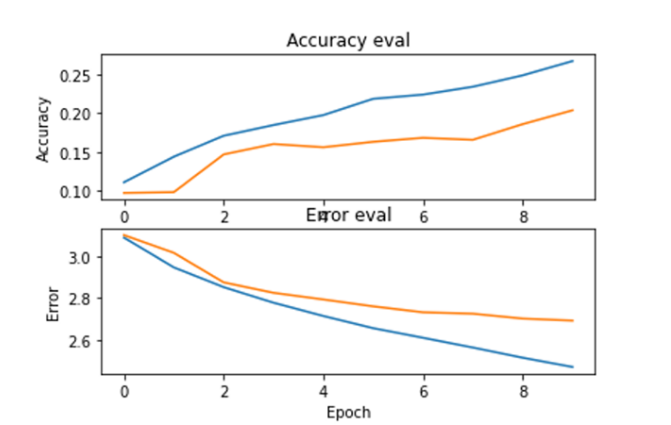


Figure 3 : Output of training CNN model with signal as input

The train and test accuracy value are low, just around 0.20, but they seem to have a tendency to grow. If more epochs are executed, the result would possibly be better. Training dataset has better accuracy value and lower error value than test dataset, which can be explained by a small number of training dataset.

#### STFT:

Number of Epochs: 30

Batch Size : 32

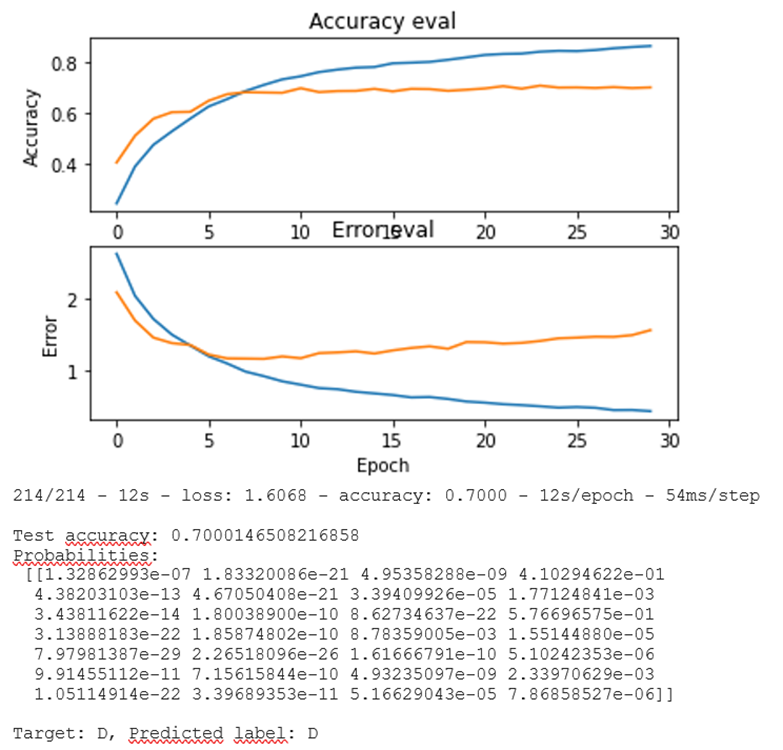


Figure 4 : Output of training CNN model with STFT as input

The model has the test accuracy around 0.7, and train accuracy around 0.8, which is quite good. It also correctly predicted the target chord. However it can be seen from the figure that the train accuracy increases over epochs, but the test accuracy has the tendency to decrease. It is clearer in the error graph, where test error in the last epochs has the increase pattern.

#### Comparison between STFT and Signal:

Comparing 2 models with input as STFT and raw audio signal , it can be seen that STFT model works better on predicting the data. On the other hand, STFT test error increases while signal test error decreases over the last epochs, which means the result from signal model is more reliable. Increasing the number of epochs in the model of signal can yield better accuracy value.

### RNN/LSTM Model

LSTM models were built for MFCCs original dataset, and STFTs reduced dataset. Generally 2 LSTM layers, 1 dense layer and an output layer are built.

#### MFCC:

callbacks function is used to stop the model when the test error decreases.

Number of Epochs: 50

Batch Size : 64

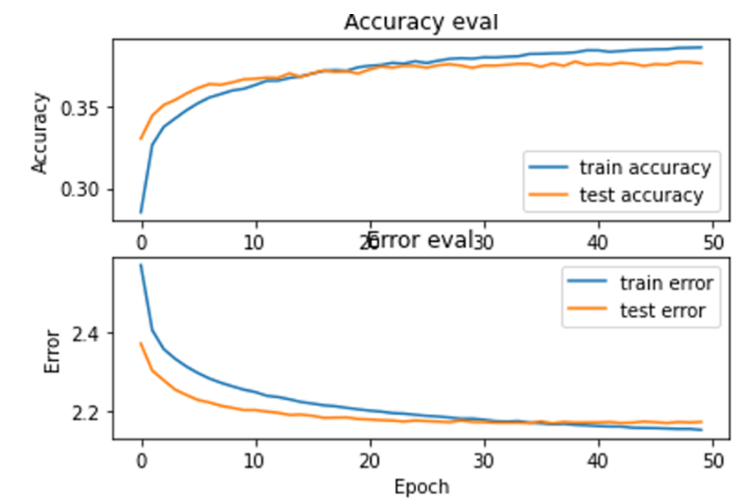


Figure 5 : Output of training RNN model with MFCC as input

It can be seen from the graph that the model does not yield a good training result, with just around 40 percent of training and 35 percent of test accuracy value. There is no overfitting to be seen here, but test accuracy value and errors seems to not increase in the last epochs. Compared to CNN model results made by Polina, as can be seen in the below graph, the accuracy value from CNN model is much higher. A solution for this problem can be increasing the number of neuron layers, since the dataset is very large and complex.

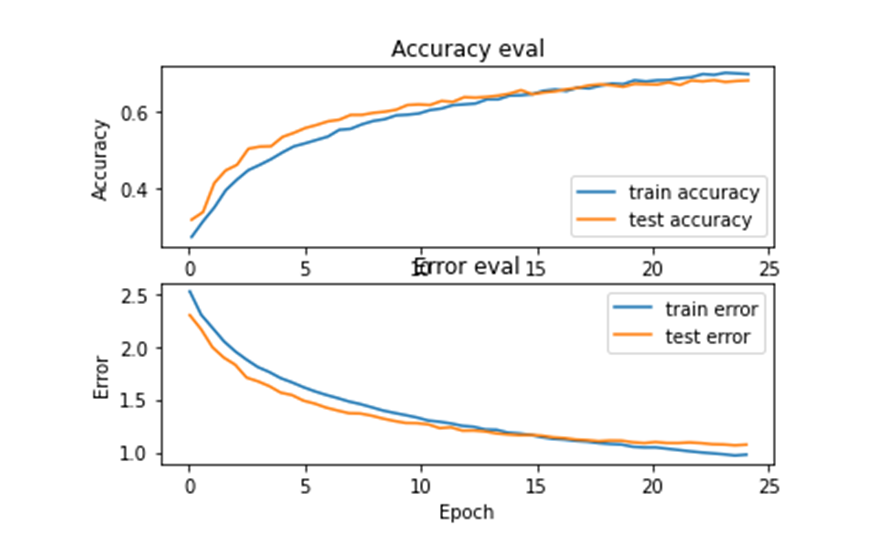


Figure 6: Output of training CNN model with MFCC as input

#### 

#### STFT:

Number of Epochs: 30

Batch Size : 32

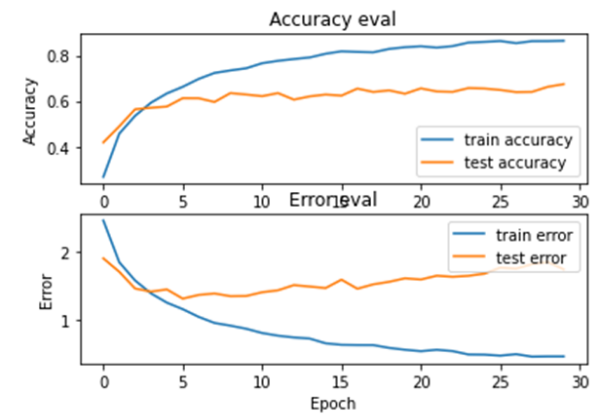


Figure 7: Output of training RNN model with STFT as input

As can be seen from the graph, the model can correctly predict around 80% of the training dataset, while it can only do around 60 percent for the test dataset. Moreover, the test error seems to increase, while training dataset error declines over epochs, which indicates that the model is overfitting. Possible solution to this can be an increase in the number and quality of the dataset.

# Discussion

By trying different types of audio extractions for the input of Neural Network model to recognize musical chords in different songs, it can be concluded that MFCC is more efficient to use for deep learning as STFT and raw audio signal due to its small memory space usage and it is closer to human hearing perception than other types.

Using a signal instead of an extracted audio features can facilitate pre-processing of the data before connecting to the neural network. However, it requires more computing power, and the signal without transformation provides less information than, in particular, a spectrogram or a cepstrogram.

The best result was shown by CNN with STFT-input - test accuracy 0.7. RNN with MFCC-input has not very high accuracy of around 0.4 and overfitting, the model can be improved by changing the hyperparameters of the model, adding a dropout layer, changing the current dataset, etc. CNN model with signal-input has accuracy less than 0.2. The model did not show its maximum accuracy (there is no overfitting), so the result can be improved by increasing the number of epochs.

Overall, possible improvement for the project is to try building models with better parameters and dataset to produce better training results. Problems such as overfitting should be diminished in future work.

# References

1. H.-T. Cheng, Y.-H. Yang, Y.-C. Lin, I.-B. Liao, and H. H. Chen. (2008). *Automatic chord recognition for music classification and retrieval.* In 2008 IEEE International Conference on Multimedia and Expo. (pp. 1505–1508).
2. N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent. (2013). *Audio chord recognition with recurrent neural networks.* In ISMIR. (pp. 335–340).
3. J. Osmalskyj, J-J. Embrechts, S. Piérard, M. van Droogenbroeck. (2012). *Neural networks for musical chords recognition.* In Journées d’Informatique Musicale, hal ID: hal-03041758.
4. F. Korzeniowski, G. Widmer. (2016, Sept. 13–16). *A fully convolutional deep auditory model for musical chord recognition*. In IEEE International Workshop on Machine Learning for Signal Processing.
5. E. J. Humphrey, J. P. Bello. (2012). *Rethinking automatic chord recognition with convolutional neural networks*. In 11th International Conference on Machine Learning and Applications IEEE.
6. M. McVicar, R. Santos-Rodriguez, Y. Ni, T. D. Bie. (2014). *Automatic Chord Estimation from Audio: A Review of the State of the Art*. In IEEE ACM Transactions on Audio, Speech, and Language Processing. (pp. 556 – 575).
7. S. Nakayama and S. Arai. (2018). *DNN-LSTM-CRF model for automatic audio chord recognition*. In Proceedings of the International Conference on Pattern Recognition and Artificial Intelligence (pp. 82–88).
8. G. Mendels, (2019) Towards Data Science. How to apply machine learning and deep learning methods to audio analysis. https://towardsdatascience.com/how-to-apply-machine-learning-and-deep-learning-methods-to-audio-analysis-615e286fcbbc

9. A. Tripathi NLP, „WHAT IS THE MAIN DIFFERENCE BETWEEN RNN AND LSTM | NLP | RNN VS LSTM“, *Data Science Duniya*, 2. Juli 2021. https://ashutoshtripathi.com/2021/07/02/what-is-the-main-difference-between-rnn-and-lstm-nlp-rnn-vs-lstm/

1. When running the chord\_transcript file on google colab, the output of simplified chords are 28, not 25, like on the computer. [↑](#footnote-ref-1)