Audio key estimation of digital music with CNNs

Domain Background

The art of mixing recorded music in real time is known as DJing and performed by a Disc Jockey (DJ). The initial steps of DJing were made on broadcast radio stations in the 1930s, where individual selections of recorded music were played. From there, DJing spread out to dance parties, nightclubs, music festivals and even events like marriages [1]. DJs use specialized equipment that can play at least two sources of recorded music simultaneously to create smooth transitions from one song to another [2].

The way of how the transitions are made evolved over the years and became extremely versatile nowadays. Hereby one DJ technique experienced a renaissance in 2006: harmonic mixing. The goal of harmonic mixing is to transition between songs of the same or related key. This technique enables a DJ to make smooth continuous mixes and prevents unstable tone combinations, known as dissonance [3]. But therefor the key of a song must be known.

Music Information Retrieval (MIR) is the field of research which deals with the complex structure of music and tries to develop systems to solve tasks like Audio Artist Identification (AA), Audio Genre Classification (AG) and Audio Key Finding (AK) [4]. There exists a community where the MIR systems and algorithms are going to be evaluated - the Music Information Retrieval Evaluation eXchange (MIREX) [5].

Problem Statement

This project deals with the learning task to estimate audio keys of digital music with the help of convolutional neural networks (CNNs). A key describes a scale that forms the basis of a music composition and consists of a tonic note and a mode of the scale (major or minor). The learning task is limited to diatonic scales which employs seven pitches per octave and is typically used in western music. Hereby an octave is tuned with a system called twelve-tone equal temperament (12-TET) [6]. By that, there are 24 possible keys per octave (12 major keys and 12 minor keys). The full list of possible tonic notes and their pitch frequencies per octave to be estimated is given by the scientific pitch notation [7]. By that, the learning task is a multiclass classification problem.

Datasets and Inputs

digital audio files for signal processing

The dataset, consisting of 30 second samples of common songs in digital format, is going to be self-created. To select appropriate songs, the Million Song Dataset (MSD) [8] is utilized. It provides data to one million analyzed songs, including information about their key and mode as well as how confident both are. This confidence is considered while choosing songs. Further information on how the data selection was done can be found in [14] (Jupyter Notebook of MSD data extraction).

The notation of key and mode for each analyzed song of the MSD is as follows:

- key: int number [0, 11], where 0 = C key, 1 = C# key, …, 11 = B key
- mode: int number [0, 1], where 0 = minor, 1 = major

By that, the chosen songs are going to be separated into folders named 'key-mode' (e.g. 0-0, …, 11-1), leading to 24 folders in total.

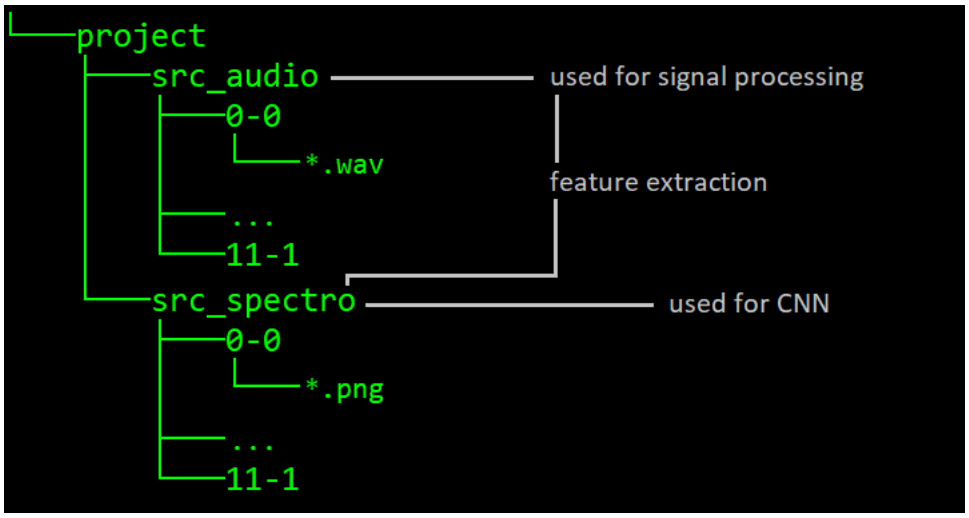
Currently every folder holds 10 song samples. Thus, the whole dataset consists of 240 files. The dataset shall be expanded up to 1000 songs at least, but it takes time due to the long preparation work for each sample. Since every ‘key-mode’ folder currently has the same amount of data, the dataset is balanced.

The format of one digital music file will be Waveform Audio File Format (WAV) with a sampling rate of 44100 Hz, a sample size of 16 Bit and 2 channels (stereo). The length of each song shall be 30 seconds. Thus, every audio file has a size of ~5 MiB.

feature extraction for CNN

The songs itself cannot be fed into the CNN. An image holding information about the pitches needs to be created per song (spectrogram). This leads to another folder with 24 subfolders containing the spectrogram images per ‘key-mode’.

For better understanding, the setup of the dataset and its usage is shown below:



```
└──project
    ├──src_audio
    │   ├──0-0
    │   │   └──*.wav
    │   ├──...
    │   └──11-1
    └──src_spectro
        ├──0-0
        │   └──*.png
        ├──...
        └──11-1
```

Solution Statement

Since input data and target data are given, the problem will be solved by a supervised learning algorithm. For this task a convolutional neural network is used which is commonly applied to visual imagery. Therefor the audio file is going to be transformed from a time-domain signal into a frequency-domain signal. Doing this in short time steps, by slicing the audio file in many parts, will result in a spectrogram of the audio file which can be fed into the neural network. The whole process is known as Short-time Fourier Transformation.

Benchmark Model

The results of the learning algorithm will be compared against key estimations by a free software named KeyFinder [12]. The same testing data will be fed into this software and compared. The MIREX evaluation procedure for Audio Key Detection [13] will be used to uniformly compare the results.

Evaluation Metrics

The metric to use depends on the following question: Out of all estimated keys for given audio files, how many were classified correctly? The metric to evaluate this is known as accuracy and is given by the formula for binary classification problems:

$$Accuracy_{bin} = \frac{TP + TN}{TP + TN + FP + FN}$$

where TP = True Positives (instances that are actually true and were predicted as true), TN = True Negatives (instances that are actually false and were predicted as false), FP = False Positives (instances that are actually false and were predicted as true) and FN = False Negatives (instances that are actually true and were predicted as false)

For this project, there exist C = 24 output classes. TP and TN are considered as correct classified, whereas FP and FN are considered as wrong classified. This leads to a (C, C) confusion matrix where C_i is the actual class and C_j is the predicted class. The TPs are all entries where i = j, and the FPs are all other entries of a column j. By that, the accuracy formula for this multi-class problem is:

$$Accuracy_{mc} = \sum_j \frac{TP}{TP + FP}$$

The above shown formula reminds of the metric precision (also known as F0-score) and can be used equivalently.

Important remark: If the dataset is going to get unbalanced, the F1-score will be used instead.

Project Design

The project is divided into several process steps:

- Data Preprocessing
    - Signal Processing (read audio signal, reduce signal, decimate signal)
    - Feature Extraction (Short-time Fourier Transformation, image generation)
- Model Preparation (CNN)
    - split up data into training and testing sets for both features and targets
    - design the model architecture
    - set up model parameters for evaluation metric and batch learning
- Model Training and Evaluation
    - train the model
    - evaluate and compare results to benchmarks

Data Preprocessing

The preprocessing is the most important part in this project, since the results of the learning algorithm strongly depend on it.

The digital audio signal will be read in with the result of having arrays containing all 16 Bit samples of the signal per channel. Then the signal is going to be reduced to one channel (mono) without facing loss of information. To continue the size reduction, the signal will be decimated which includes low-pass filtering and down sampling. All these tasks are aimed on reducing the amount of data to process.

With the help of Short-time Fourier Transformation, the time-domain signal is transformed into a frequency-domain signal. Hereby the signal will be split in a certain number of parts and on each part the transformation is applied. By that the result is a consecutive frequency-domain signal over the length of the signal, called a spectrogram. Certain parameters, including number of parts, used window function and zero-padding affect the results. Finally, the spectrogram will be preprocessed and saved.

The data preprocessing steps are done for every single digital audio file with having spectrogram images of the signals for the use with a convolutional neural network.

Model Preparation

As usual the data is split up in a training set and testing set. Furthermore, a convolutional neural network is built with keras or underlying tensorflow. Model parameters for accuracy metric and training in batches are created and connected to the model.

The network itself will consist of at least 3 convolutional layers, 2 max pooling layers, a flatten layer and a fully connected layer at the output with a softmax activation.

Model Training and Evaluation

Model training is done in batches and several times with shuffled training set and testing set. Best weights are saved dependent on the metric. Besides the accuracy metric, the model will be benchmarked against the software KeyFinder with the help of the MIREX evaluation procedure.

References

[1] https://en.wikipedia.org/wiki/History_of_DJing

[2] https://en.wikipedia.org/wiki/Disc_jockey#History

[3] https://en.wikipedia.org/wiki/Harmonic_mixing

[4] https://doi.org/10.1250/ast.29.247

[5] http://music-ir.org/mirex/wiki/MIREX_HOME

[6] https://en.wikipedia.org/wiki/Equal_temperament#Twelve-tone_equal_temperament

[7] https://en.wikipedia.org/wiki/Scientific_pitch_notation#Table_of_note_frequencies

[8] https://labrosa.ee.columbia.edu/millionsong/

[9] https://api-signup.7digital.com/

[10] http://music-ir.org/mirex/wiki/2005:Audio_and_Symbolic_Key_Finding

[11] https://www.beatport.com/

[12] http://www.ibrahimshaath.co.uk/keyfinder/

[13] http://www.music-ir.org/mirex/wiki/2018:Audio_Key_Detection#Evaluation_Procedures

[14] https://github.com/damilo/keyestcnn/blob/master/00.hlp/msd/msd.ipynb