Eyas Hassan MAIS 202 Winter 2021 March 3, 2021

DELIVERABLE 2

Problem Statement:

The aim of this project is to produce a deep learning model for automatic chord recognition (ACR). The idea is to predict chord voicings given a waveform of an audio signal.

Data Preprocessing:

The dataset used to train the model is the IDMT-SMT-Chords dataset.[1] This dataset contains 16 MIDI generated audio files of various chord classes, both guitar and non-guitar instruments. The Non-Guitar category includes 10 wav files consisting of 576 2 second segments of all chord types in all possible root note positions and inversions. The Guitar category includes 6 wav files consisting of 273 2 second segments of chord voicings based on barŕe chords with root notes on the low E, A, and D strings. A text file containing the corresponding labels for each 2 second segment was provided per guitar and non-guitar audio types. A summary of the dataset is given below in Table 1.

Table 1: *Summary of dataset*

	Non Guitar	Guitar
Chord Types	7	7
Chord Inversion	17	14
Pitch Range	C2-C5	Fret 0 – Fret 12
# wav files	10	6
# Chord Segments (One instrument)	576	273
Duration (One Chord)	2 sec	2 sec
Duration (One wav File)	19.2 min	9.1 min

Only the Non-Guitar subset of the dataset was used to train the model. Each wav file was loaded at a sampling rate of 22050 Hz, a common value for simple audio processing applications. Each chord segment is 2 seconds long, meaning there are 44100 sample per chord segment. In order to obtain information about the notes present in each chord segment, the spectral (frequency domain) features of the audio signals need to be extracted. This is achieved through performing a short-time-Fourier-transform (STFT) on each 2 second (44100 sample) chord segment with a frame size of 2048 samples and a hop size of 512 samples. A STFT is essentially numerous Fourier transforms (FT)s performed on snippets of the signal. The frame size is the number of samples considered for each FT while the hop size represents how many samples the frame is shifted by for the next FT. These two parameters are shown in Figure 1. A windowing formula, specifically a Hann window, is applied to each FT in order to ensure continuity of the STFT (the end points at each FT decay to zero).

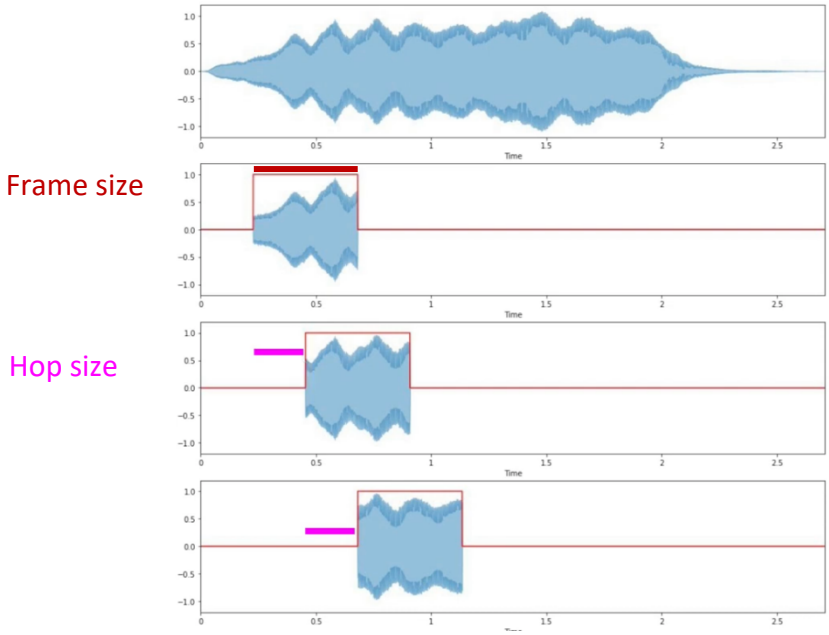


Figure 1: *Frame size and hop size*

The STFT of a waveform provides information about the development/change in frequency over time. The number of time bins (87 bins) is determined by the number of samples per chord segment (44100 samples) divided by the hop size (512 samples). Similarly, the number of frequency bins was chosen to be 133 bins, thus a matrix of shape (133 x 87) can be used to describe the spatial and temporal features of each chord segment. For a specified frequency delta, humans are able to easily distinguish between lower frequencies but have difficulty with high frequencies. This is because humans perceive sound on a logarithmic scale. So, the frequency of the STFT is also transformed to a Mel scale (log frequency scale). Finally, the logarithm of the squared amplitude is taken and multiplied by a factor of ten in order to obtain decibels (dB). The final result can be visualized in the form of a Mel spectrogram as shown in Figure 2.

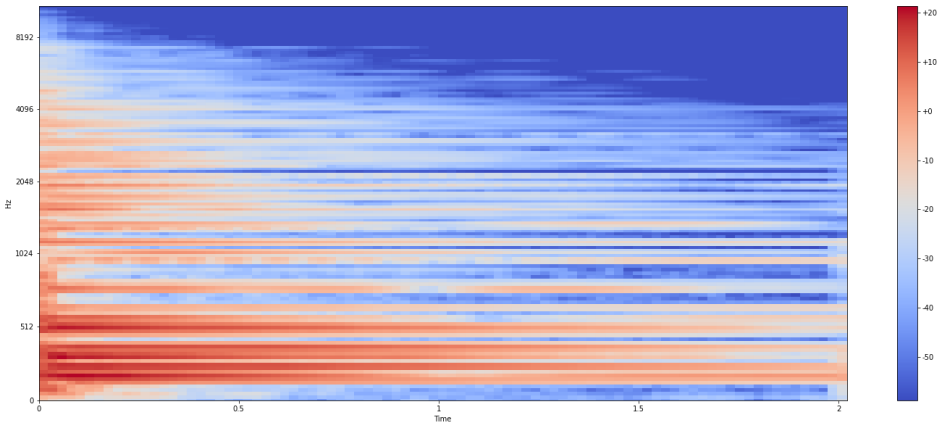


Figure 2: *Mel spectrogram of Cmaj/3 chord where intensity is measured in dB*

The reason for choosing a spectrogram as a feature as opposed to pure frequencies/pitch classes is motivated by data augmentation (i.e. artificially creating more data). Such that each column (out of the 87 time bin columns) contains information about frequency and has the same corresponding chord voicing label. So for each of the 288 chord voicing labels, there are 87 (# of time bins) x 2 (# of octaves) x 10 (# of instrument wav files) samples, i.e. 1740 samples. The train, validation, and test sizes represented 0.56, 0.19, and 0.25 respectively (default 0.25 test size value of scikit learn's test_train_split method).

Machine Learning Model:
A Mel spectrogram can be thought of as an image, and so convolutional neural networks (CNN)s are found to be really strong for ACR applications. The model architecture is presented below in Figure 3.

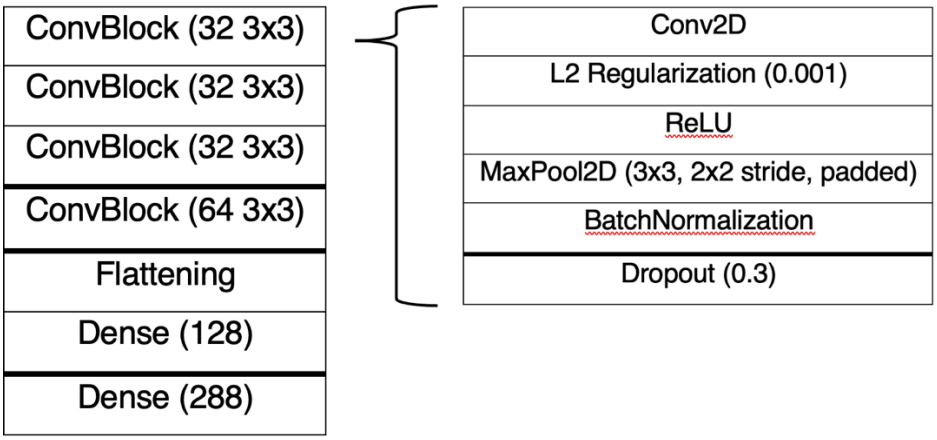


Figure 3: *Model architecture*

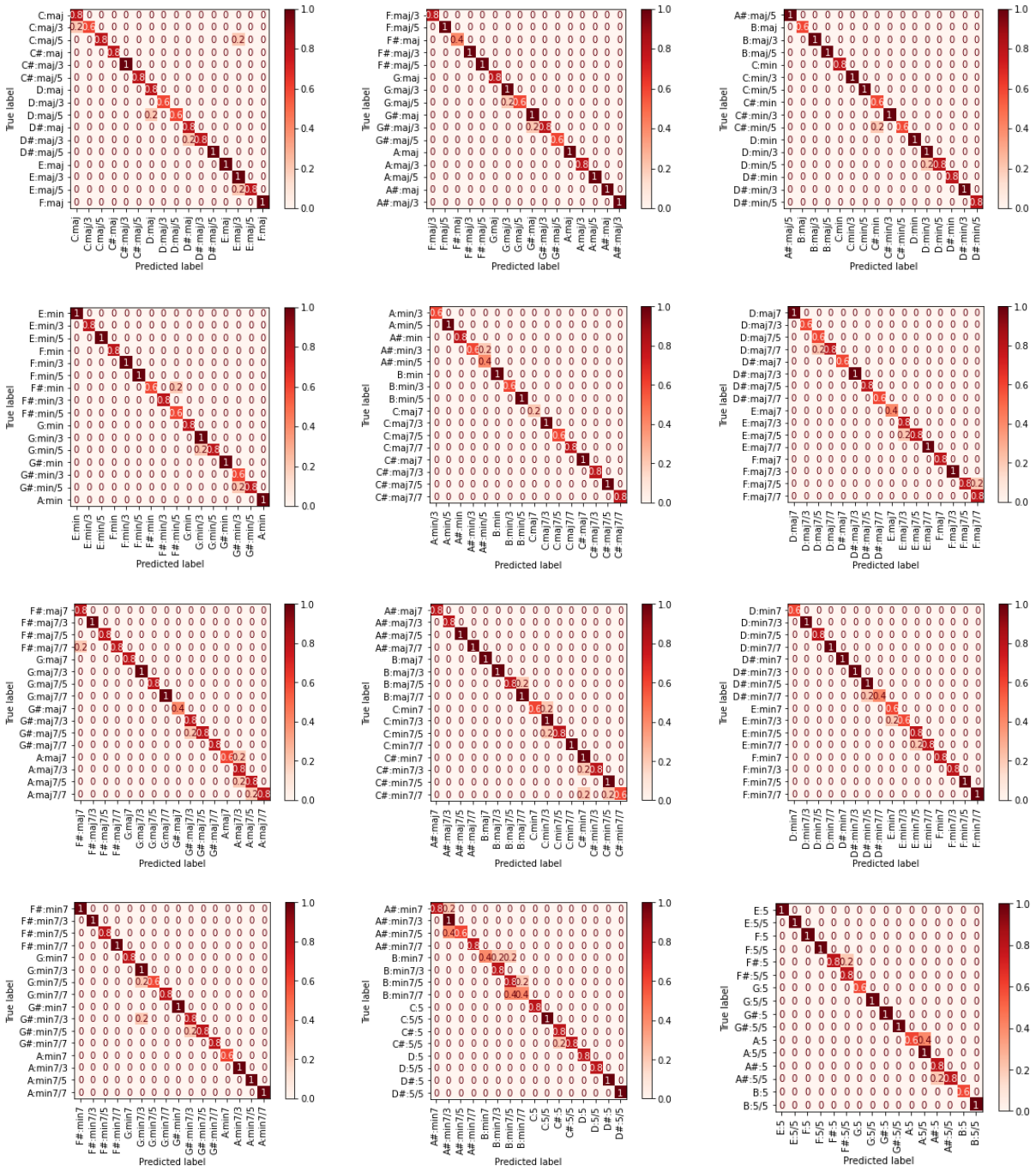
The final dense layer (CNN output) has a softmax activation. The model's optimizer was chosen to be Adam's with a learning rate of 0.0001 as advised by the governing journal article. The loss was calculated by means of sparse categorical crossentropy for the same prior reason. Lastly, the batch size was 32 with 80 epochs. These values were chosen empirically as they showed promising training accuracy and adequate computational efficiency.

The model was validated based on the accuracy metric, both for training and validation data. The model appeared to be overfitting since it achieved 98.18% training accuracy but a mere 61.31% test accuracy.

Initially, the model was performing at ~94% and ~40% training and test accuracy respectively when the inputs were chromagrams instead of Mel spectrograms. The model took less time to train but the inadequate spatial resolution gave rise to poor model performance.

Preliminary Results:

As previously stated, the model achieved 98.18% training accuracy and 61.31%. Since there are 288 classes of unique chord voicings, 18 16x16 normalized confusion matrices were obtained. They can be found below in



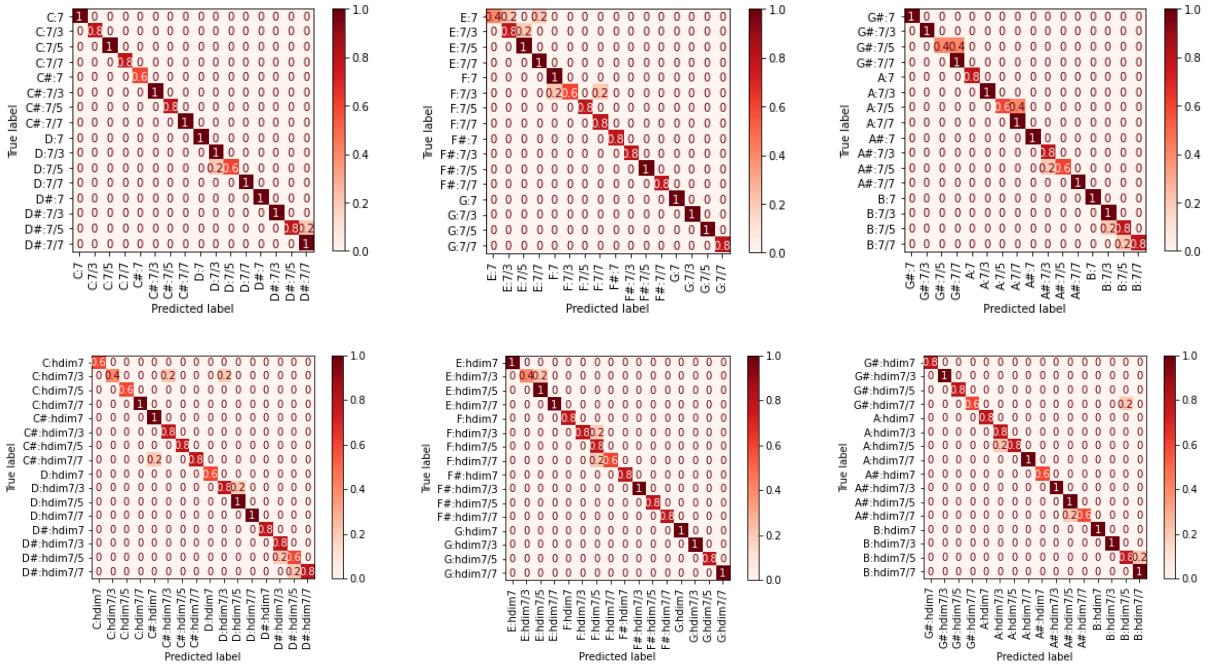


Figure 4: *Subsets of size 16x16 of the 288 parent confusion matrix.*

Next Steps:

Most of the time, the incorrect predictions were cases of predicting the incorrect inversion of the same chord. This can be tackled by either increasing the spatial resolution and/or removing the harmonics present (as it is difficult to distinguish between chords and their inversions at high frequencies and lower spatial resolution). Another possible solution would be to change the prediction strategy. Instead of directly predicting one of the 288 chord classes, two models can work simultaneously to predict one of the twelve root notes (C, C#, D,…,B) and one of the seven chord types (major, first inversion, second inversion,…, major 7th flat 5th) respectively. This would also simplify the analysis of the performance as only two confusion matrices would be produced, a 12x12 and 7x7 for root note and chord type respectively. It is important to note that this strategy also assumes that a chord and its inversions are the same (potentially yielding better test performance). Lastly, I will expand the dataset to also include the guitar dataset, hopefully solving the overfitting issue.

References:

1. Nadar, C.-R.; Abeßer, J.; Grollmisch, S. In *Towards CNN-based acoustic modeling of seventh chords for automatic chord recognition*, International Conference on Sound and Music Computing. Málaga, Spain, 2019.