ANNA-T

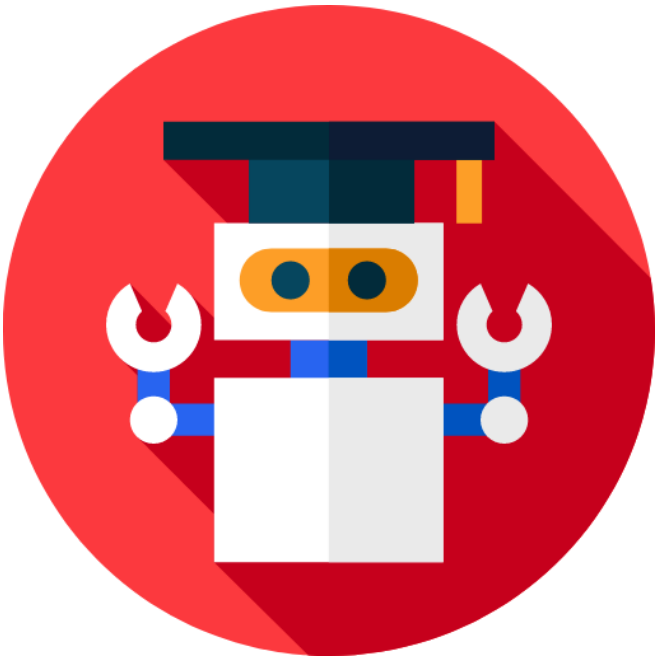
By Keshav Nath

Track:

Machine Learning

Problem Statement:

Creating Sheet Music From Audio File
(Automatic Music Transcription)



Tagline

Converting audio files to MIDI files using Machine Learning, and then converting those MIDI Files to Sheet Music

The Problem It Solves

Sheet music, while exists in abundance for classical pieces, is sparse for modern music. Additionally, if someone has a song in mind they want to play / cover, they'd like to just create the sheet music from the audio instead of rummaging through the internet to piece together the resources. My solution solves this problem by loading audio files and producing its sheet music using machine learning.

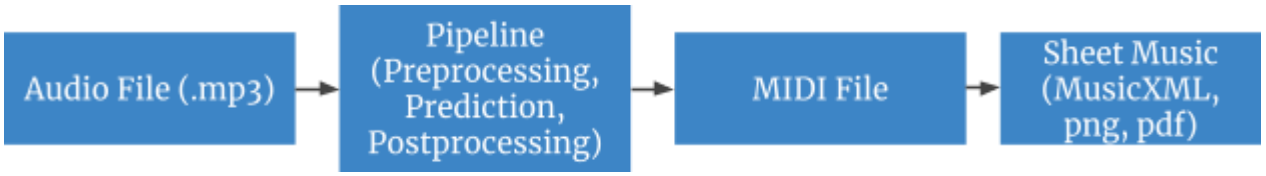
Technologies Used

The main technologies, languages, libraries and software used are mentioned below:

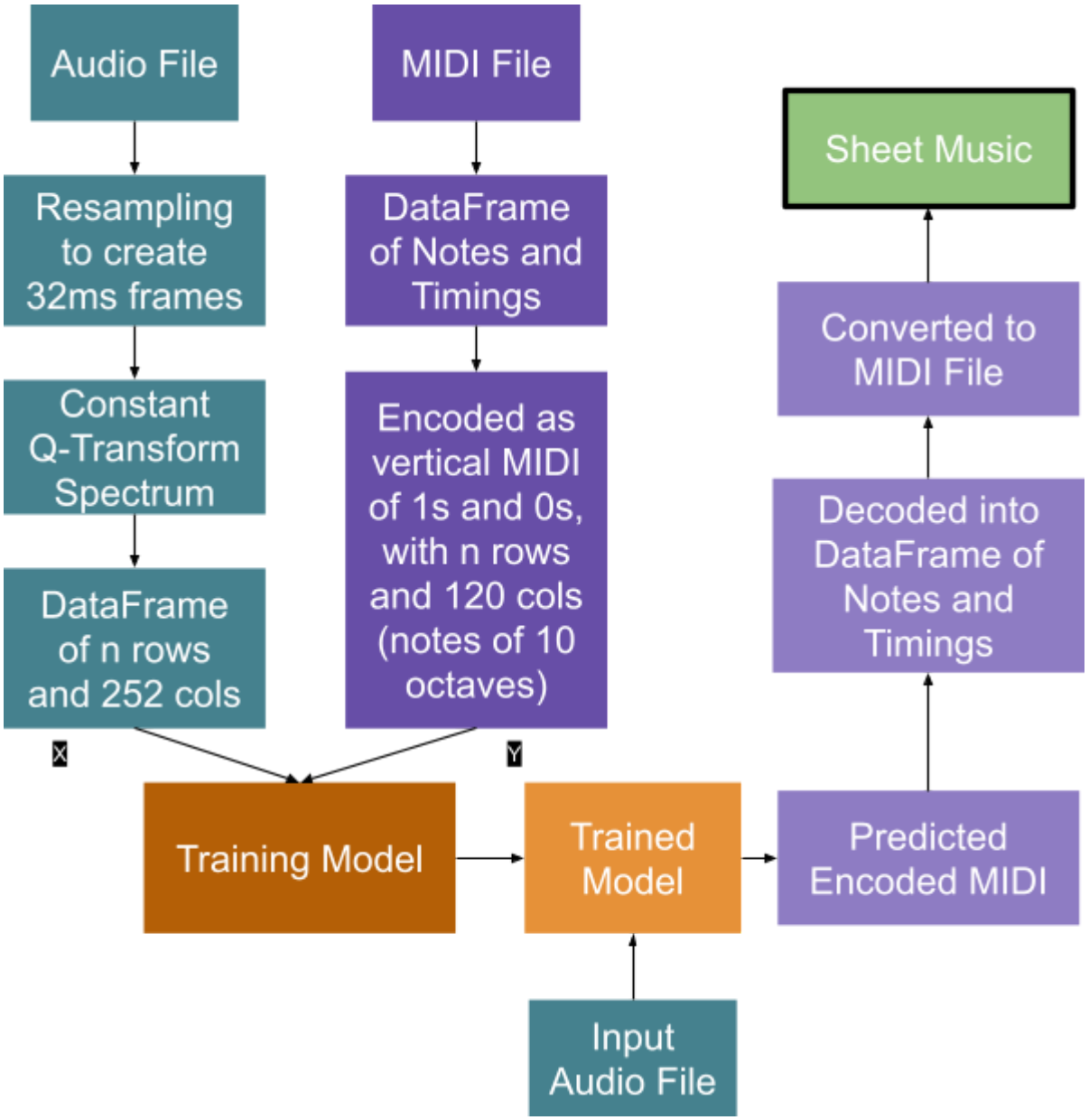
- Python
- Pandas
- Numpy
- Tensorflow
- Keras
- SK-Learn
- Librosa
- Music21
- MuseScore
- SciPy
- Matplotlib

Solution

A brief outline of the front-end is given below:



A brief outline of the procedure is given below:



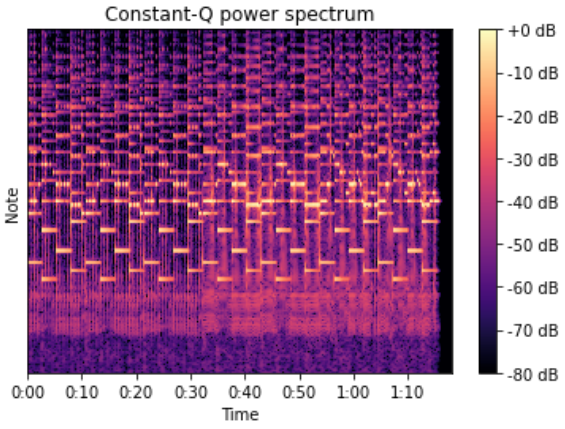
A more detailed explanation is given below:

File Loading:

Audios are loaded into python using Librosa. In general, audio files are encoded at 44.1kHz, but this creates too many frames, so all audios are resampled to a modest 16kHz. This means that the audio file is split into frames of 32 ms each, which allows efficient processing with minimum data loss. The corresponding MIDI files are loaded into python using Music21, and then converted to a Music21 Stream object. This object represents the MIDI as a stream of notes with their name/pitch (e.g. C#5), their onset (starting) time, and their offset (ending) time.

Audio Preprocessing:

Audio files can be expressed using spectrograms of Fourier transforms. I selected Constant Q-Transform (CQT) as our transformation over the other commonly used methods (MFCC, STFT) due to CQT's ability to correctly process multiple notes occurring simultaneously. I generate 36 bins per octave for 7 octaves, resulting in 252 features.



MIDI Encoding and Decoding:

The salient feature of our solution is our innovative, custom-built MIDI Encoder and Decoder. My algorithm allows us to train our model on any form of music, irrespective of dataset availability, as long as an audio file and its corresponding MIDI file are available. Most of the research done in this field relies on datasets like MAPS that provide the required data in .txt, .csv or .xml files. This limits training possibilities to pre-constructed datasets, making it difficult to use the model for modern genres like Pop and EDM.

My encoder converts a MIDI Stream into somewhat of a multilabel one-hot format of 1s and 0s, corresponding to the presence of each note in a 32ms frame. It is better imagined as a “vertical midi”, where the columns are the 120 notes and each row is a frame. A “1” would mean that that specific note is being played in that frame.

The decoder converts this DataFrame back to a MIDI Stream by parsing each frame row-wise, hunting for notes (1s) column-wise, computing their duration by smartly parsing the note’s column, and adding the note and its timings to the stream.

Model:

My data creates a multi-label binary classification problem. The concatenated audio data is used for training, and the concatenated MIDI data is used as the labels.

To prove the validity of our processes, before delving into complex CNN and LSTM models, we have already established a baseline model using Multi-Out Decision Trees and obtained baseline accuracies of >40%. As reference, 2017 research in Automatic Music Transcription from Stanford University obtained a baseline model with ~36% accuracy, and used an LSTM model to get >55% accuracy (on the MAPS Dataset). The current state-of-the-art AMT, created by Google’s Magenta Team achieves accuracies of >65%.

My final model plans to use LSTM instead of DNNs and CNNs because, in existing research, it has consistently shown the best scores across various datasets.

Sheet Music:

The generated MIDI Files are finally converted to the corresponding sheet music automatically using MuseScore, which was selected due to its easy integration with Music21. Sheet Music can be saved as PDFs, PNGs or MusicXMLs.

Challenges and Limitations

As is expected, our methodology is far from perfect. Some of the main challenges and limitations we face are:

- Very long preprocessing and postprocessing times for music with a significantly large number of notes (classical music and orchestral music for example)
- Tempo Detection - I use Librosa to detect the tempo of the input audio to generate accurate-length MIDI Notes, which may not be 100% accurate. However, even with incorrect estimation of tempo, the relative position and duration of notes remain consistent. It can be imagined as: If the tempo of a song is detected as twice of the real tempo, it will be as if we create sheet music for a 2x sped-up version of the song. Additionally, it should be noted that the average error in Librosa's BPM detection was found to be under 3%.
- Datasets are limited to 2-3 major sets (MAPS, Maestro and MusicNet), which tend to focus on long-duration recordings of classical music pieces (mostly Piano). As a result, to make our model applicable to modern music, we have to download MIDIs manually from various sources or write a scraper and create our own diverse dataset.

Future Scope

My work can act as a great Music Transcription Service that focuses on modern synth-based music (like Pop and EDM) instead of working primarily on classical Piano / String music.

Additional Features include:

- Performing separate transcription for multiple MIDI Instrument Tracks to produce a collection of sheet music, one for each instrument
- Using MFCC Spectrums alongside CQT to take into account harmonics and timbres of more complex sounds

ANNA-T

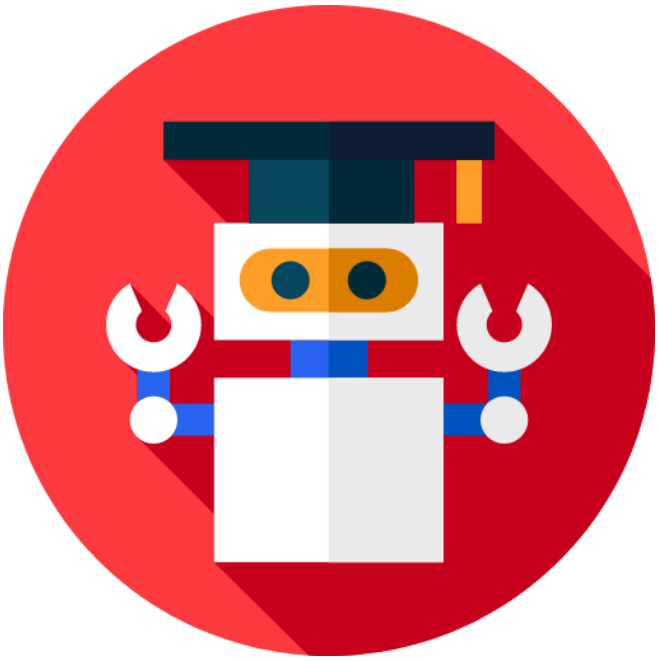
By Keshav Nath

Track:

Machine Learning

Problem Statement:

Creating Sheet Music From Audio File
(Automatic Music Transcription)



THANK YOU