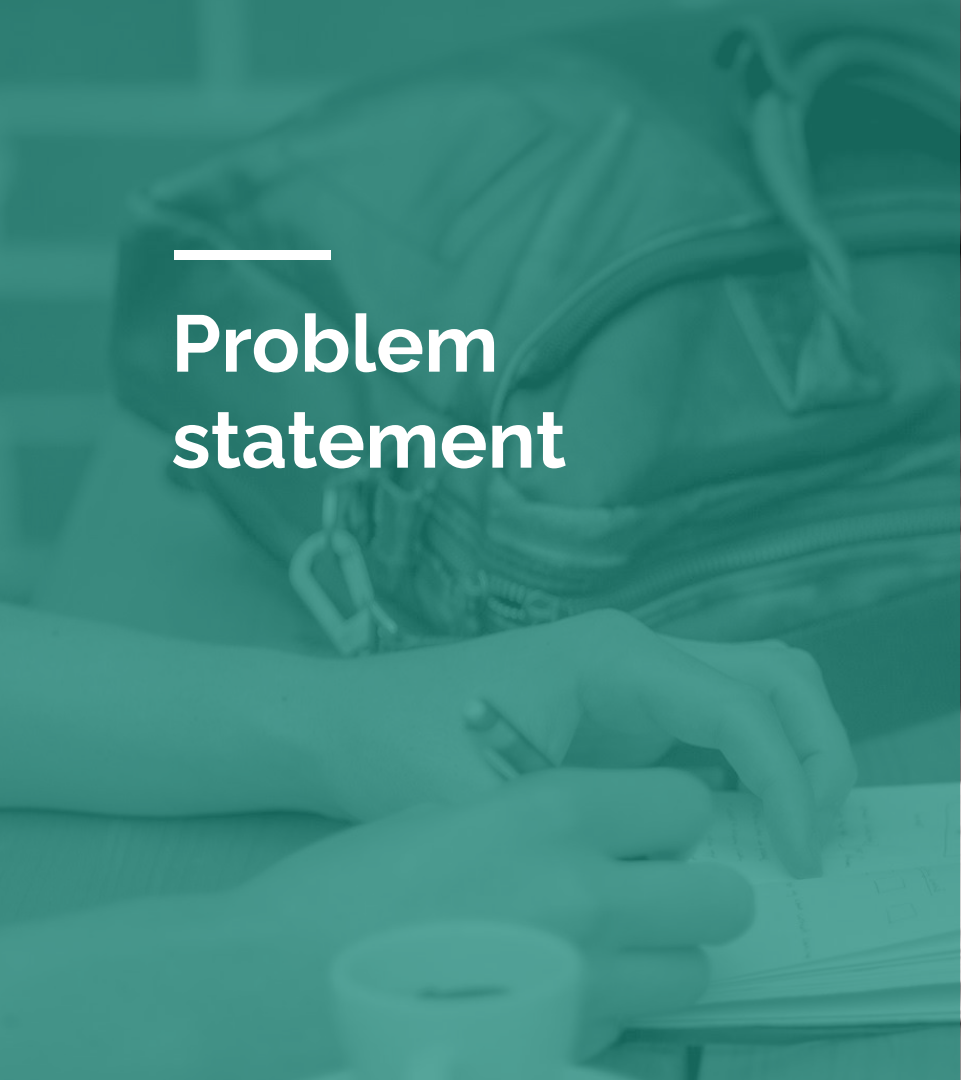
Music Generation

Yu Shen, Zhongyu Chen, Qiuli Liu, Gege Qian,Yujie Wang



Problem statement

Compose your own music without any musical-experience

Look back the productive famous composer along the history, we could find that each of them has unique musical style. To help those with zero musical talent and experience to generate their own work piece, we are taking advantage of machine learning to use the computer to generate music.

Generating Music Using LSTM Network

Data Set

From Classical Piano Midi Page

We picked Beethoven's and Mozart's works due to their very distinct and beautiful melodies, distinguishable style and large available MIDI files consisting of only single instrument for simplification purpose. We picked MIDI file over mp3 or wav file to generate our training data for a specific reason.

So what is MIDI file format?

A MIDI(short for Musical Instrument Digital Interface) file is not an audio recording. Rather, it is a set of instructions – for example, for pitch or tempo or what instrument to play the music notes.

Therefore, by using music-related library in python, we can easily extract the pitch and tempo information from a piano song that is in MIDI format, which is exactly what we need to train our model.

Data Preparation

We use Music21 to load Midi Files and get information :

The data is in the form of a sequence of either Notes or Chords

- Note contains information about the pitch, octave, and offset
    - Pitch: String that denotes sound frequency (“A4”, “D3”, etc.)
    - Octave: The set of pitches you use on a piano
    - Offset: Represents where the note is located in the piece.
- Chord is a set of Note objects

Data preperation

```python
def get_notes():
    """ Get all the notes and chords from the midi files in the ./midi_songs directory """
    #feature 1
    duration = []

    for file in glob.glob("Mozart/*.mid"):
        midi = converter.parse(file)

        print("Parsing %s" % file)

        notes_to_parse = None

        try: # file has instrument parts
            s2 = instrument.partitionByInstrument(midi)
            notes_to_parse = s2.parts[0].recurse()
        except: # file has notes in a flat structure
            notes_to_parse = midi.flat.notes

        for element in notes_to_parse:
            if isinstance(element, note.Note):
                duration.append(element.quarterLength)

            elif isinstance(element, chord.Chord):
                duration.append(element.quarterLength)

    with open('data/M_duration', 'wb') as filepath:
        pickle.dump(duration, filepath)
    return duration
```

```python
def prepare_sequences(notes, n_vocab):
    """ Prepare the sequences used by the Neural Network """
    sequence_length = 100
    pitchnames = sorted(set(item for item in  notes))
    note_to_int = dict((note,number) for number,note in enumerate(pitchnames))
    network_input = []
    network_output = []

    # create input sequences and the corresponding outputs
    for i in range(0, len(notes) - sequence_length, 1):
        sequence_in = notes[i:i + sequence_length]
        sequence_out = notes[i + sequence_length]
        network_input.append([note_to_int[char] for char in sequence_in])
        network_output.append(note_to_int[sequence_out])
        #network_input.append(sequence_in)
        #network_output.append(sequence_out)

    n_patterns = len(network_input)

    # reshape the input into a format compatible with LSTM layers
    network_input = numpy.reshape(network_input, (n_patterns, sequence_length, 1))
    # normalize input
    network_input = network_input / float(n_vocab)

    network_output = np_utils.to_categorical(network_output)
    print(network_input.shape)
    return (network_input, network_output)
```

Our Algorithm to generate music

Basic Implementation Steps

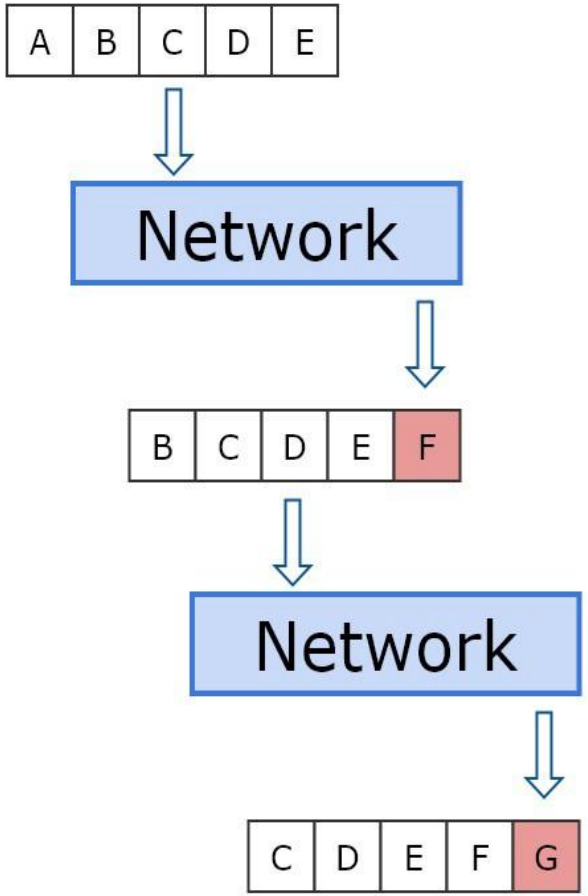
1. Selecting data based on composer/ period of time/ types of music
2. Converting music files to data set
3. Selecting training method
4. Train on big data sets to grab duration and note features
5. Generating music

Training Algorithm

- We extract the notes and durations from both datasets, save them both as 1d vectors of string and float respectively.
- We map the string and float to integers. Then we prepare the training data to feed in the neural network by looping the vectors and creating many sequences of 100 notes with the 101th note as their label.
- Due to time constraint, we trained separately for duration and notes for each person on 200 epoch for duration of Beethoven and 170 epoch for notes of Beethoven, and 110 epoch for duration of Mozart and 130 epoch for notes of Mozart, to get both loss down to 0.03 for duration and 0.1 for notes. We save to a file the weight of the neuron in neural network.

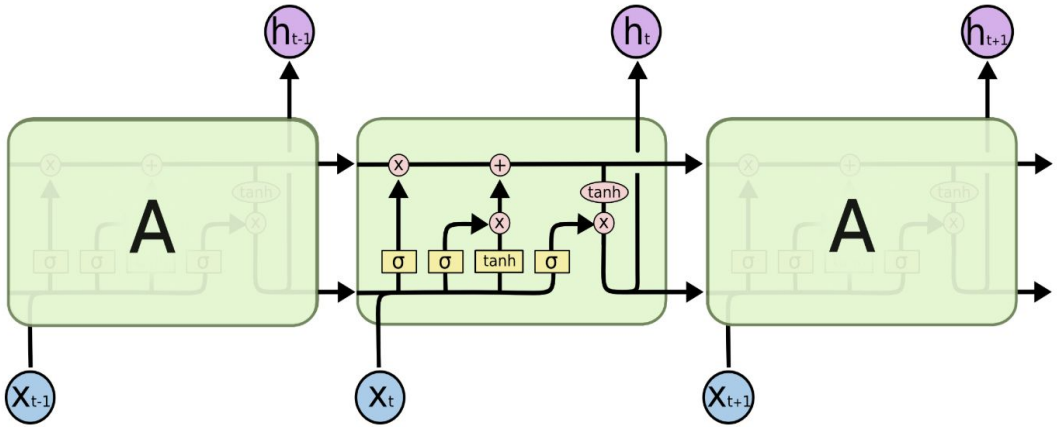
Generation Algorithm

- We take the neuron weight from the training result and reconstruct the exact same neural network.
- We then randomly choose 100 notes from the original piece to do a forward pass into the neural network to the get the prediction of their succeeding notes, then we use the 2nd to the 101st notes as the next input and predict the 102nd notes. We keep on doing this to generate a music piece of 500 notes.

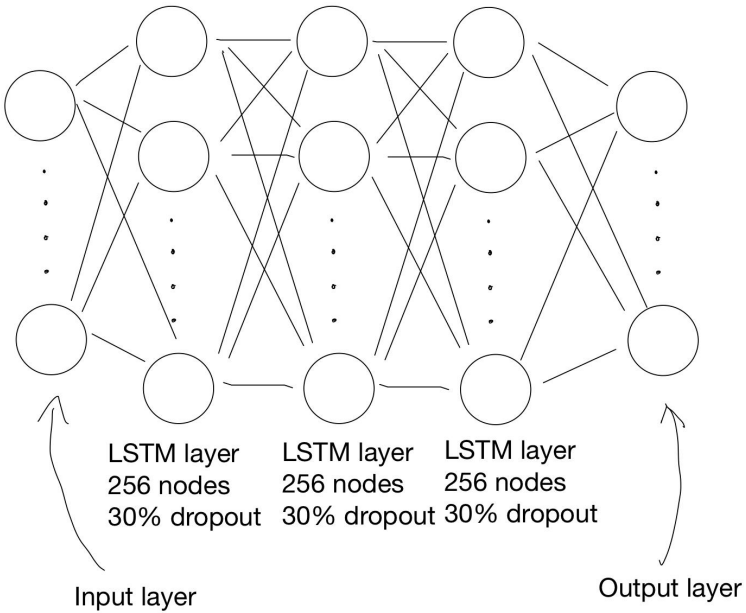


Recurrent Neural Network, LSTM

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time.



Structure of our neural network

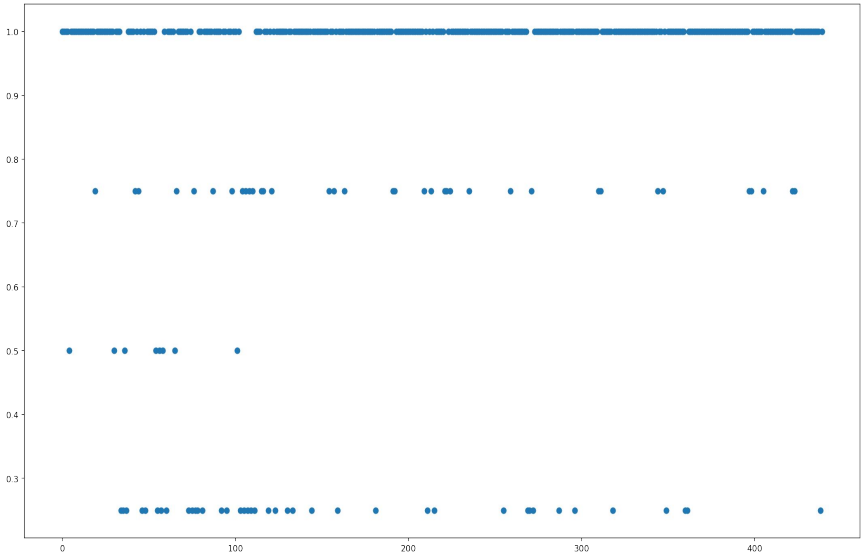


Results

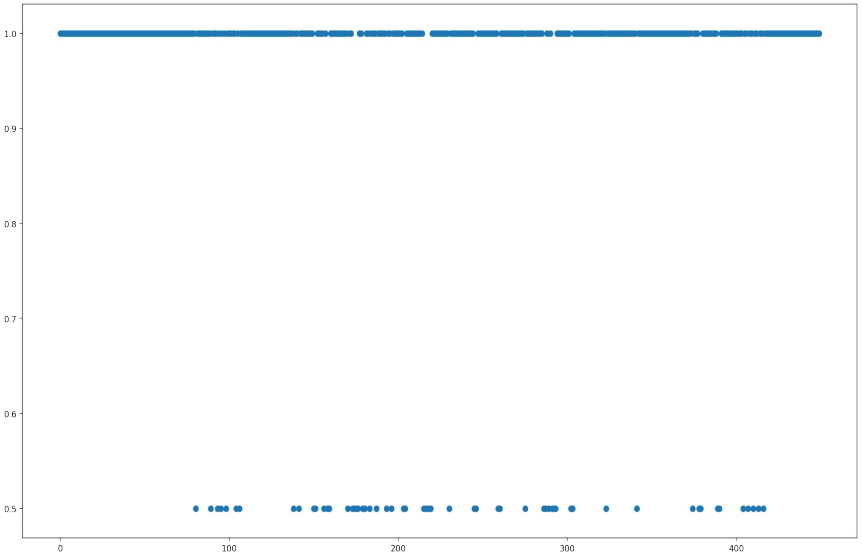
Training without Duration

Generate music with constant Duration

Comparison between generated music ---Duration

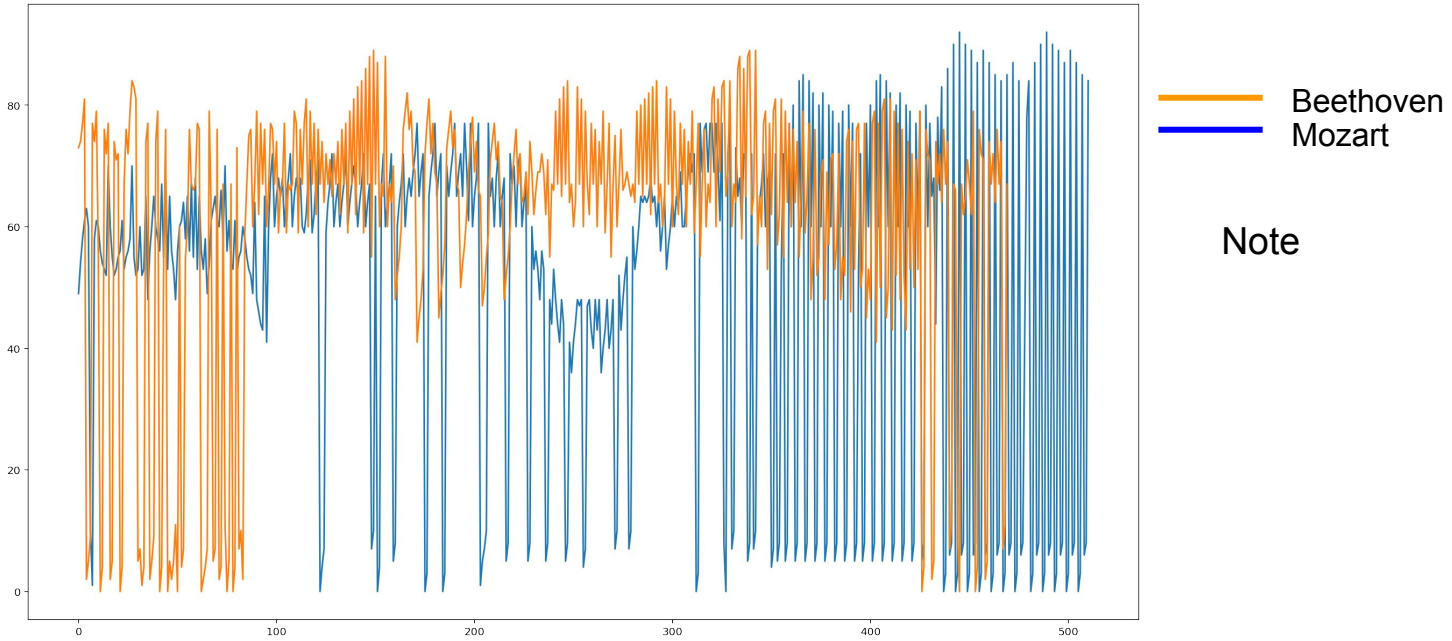


Music with trained duration



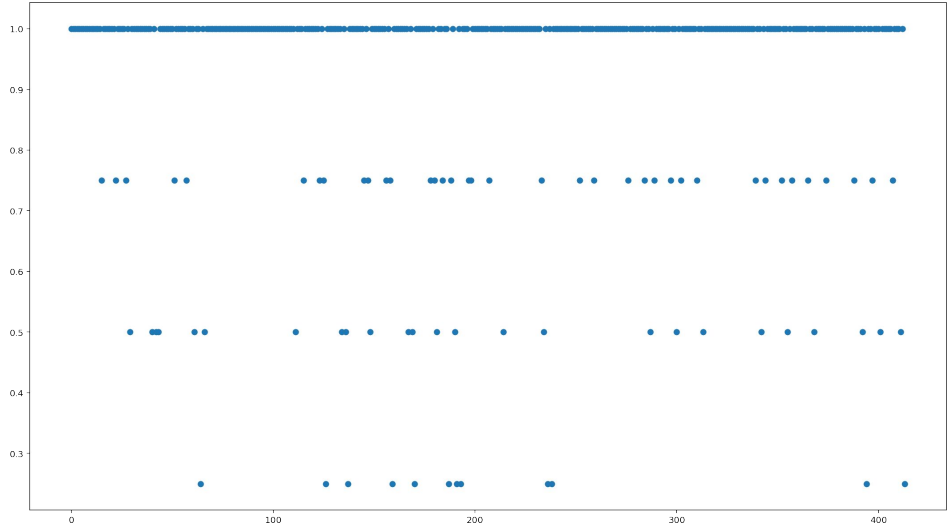
Music with uniform duration

Comparison between Mozart & Beethoven

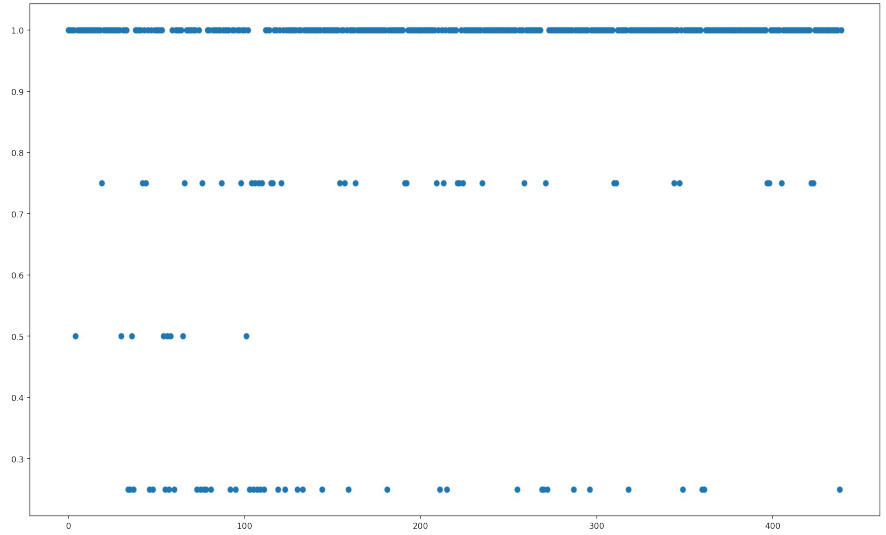


Comparison between Mozart & Beethoven

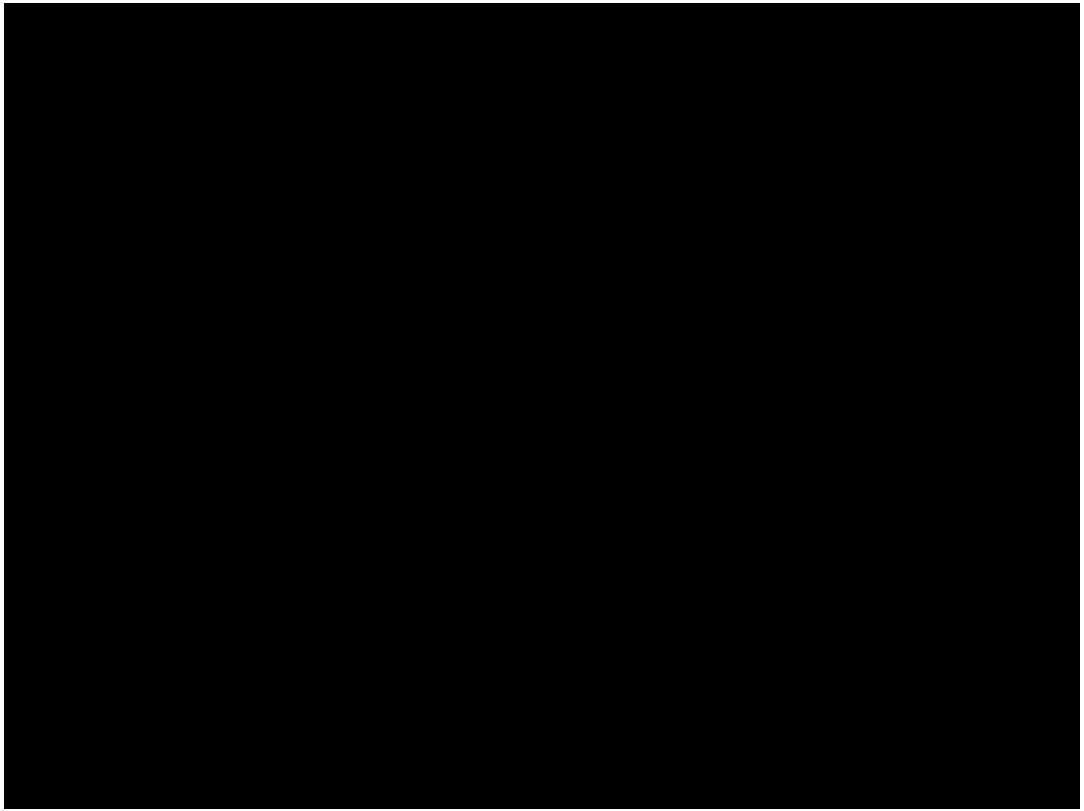
Duration



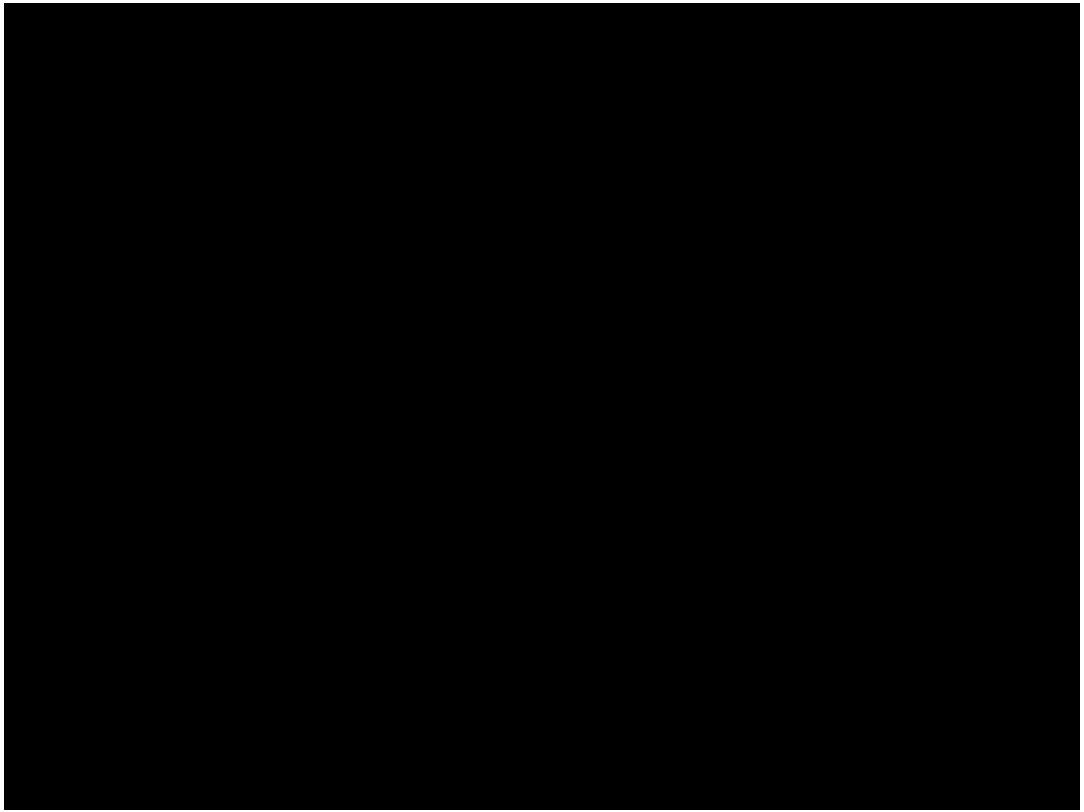
Mozart



Beethoven



Beethoven



Mozart

Discussion

Why it's better than existing solutions

Our project takes reference from previous blog posts and makes some improvements. The change that worth noticing the most is that aside from the actually pitch of each note, we also trained the duration of each note and use them later in our music generation, leading to less tedious melodies.

Future Work

What to do next

- ➔ Train duration and notes together
  - ◆ In our project, we trained the duration and notes separately. However, duration and notes are not two independent features. We believe if train ing these two features together, we would be able to generate music with rhythm.
- ➔ Train different segment of music
  - ◆ Currently, we are training the classifier with segments of music, but we did not label them with parts, such as prelude, climax, or ending. With labeling, we could train our classifier with different parts, and add the output together, so that the complete music would be smoother.
- ➔ Classify music by type before training
  - ◆ In our project, we trained our classifier with music from different composers.Since they have strong personal style, we could simply get different result from training their work. However, we do believe that if further classify the music by genre and only train music belongs to same type, the generated work would be better.

References

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

http://www.piano-midi.de/

https://cs224d.stanford.edu/reports/allenh.pdf

https://towardsdatascience.com/how-to-generate-music-using-a-lstm-neural-network-in-keras-68786