CMSC 25910

Dr. Blasé Ur

5/21/2024

# Artificial Song Generation

## Introduction:

Over the past couple of years, AI generated content has taken center stage in popular culture. Not only does it raise interesting questions about the nature of creativity, but also novel concerns about the ethics of training these models on an artists work without their permission. Many are concerned that the proliferation of artificial intelligence will push creators out of their medium for cheaper and more formulaic content. In this project, I hoped to investigate just how powerful artificial intelligence can be in song generation specifically.

## Dataset:

For this project, I decided to use the Nottingham Dataset. The dataset is made up of 1,034 MIDI files of folk songs from the United States and the United Kingdom. MIDI Files are a simplified version of songs that don't store any actual audio data but rather specify what notes are played, for how long, and at what time. The simplicity of the files makes them particularly easy to decipher and use as training data for machine learning algorithms. To further simplify the data, I decided to train my algorithm on only the melodies of each song. This allowed me to only look at singular notes rather than having to deals with complex chords or notes across multiple different instruments.

## Preprocessing:

In order to use these files, I first converted each note into a value from 1-128 to represent a specific note and also stored the notes duration, and when it is played in the song. I then inspected the frequency of each note across all songs (Figure 1.1). Upon inspection, I realized that there were many rare or infrequent notes. In order to further simplify the training and remove outlier or irregular progressions not typically found in folk music. Towards this end, I removed all notes from the dataset that weren't seen at least 100 times in the dataset. This removed about 100 notes from the training dataset and made training much faster. Overall the size of my dataset included around 196,864 notes from different songs.

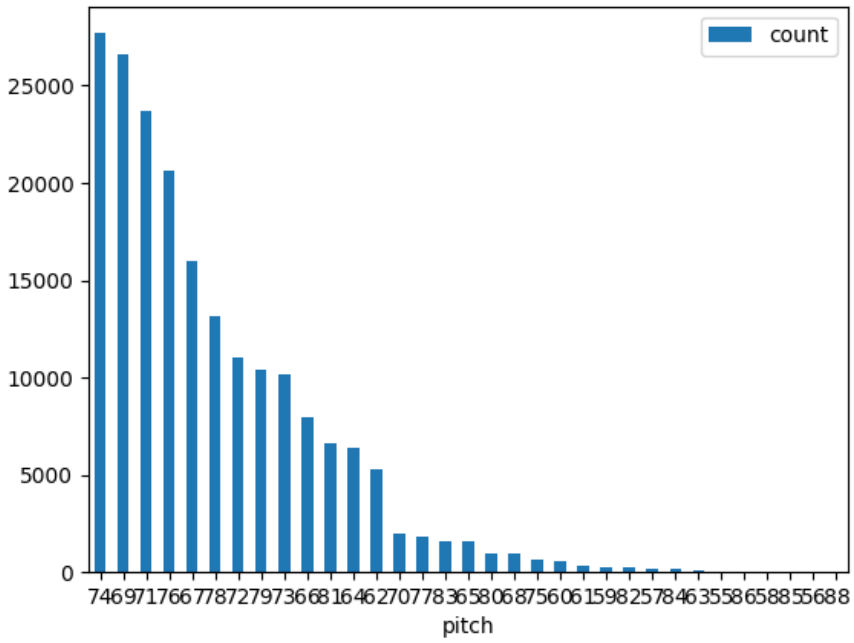


*Figure 1.1*

**Training:**

In order to train my algorithm, I first had to separate my large dataset into smaller sequences so that they could be properly ingested by the algorithm. I also scaled the pitches of each note so that their grouping would be much tighter in order to reduce inaccuracies and minimize loss. I then created a recurrent neural network with three dense layers, so that it would be able to decipher the pitch, duration and placement of a note based on the notes before it. The pitch of the note was determined using a categorical cross entropy loss function. For the note duration and placement, I used a mean squared error with a positive pressure of 100. I used this instead of a regular mean square error because I wanted to ensure that the duration and location would never be negative, and thus the positive pressure would harshly punish any predictions that were negative. I trained the model over 30 epochs with a batch size of 20,. While the model was fairly accurate at predicting pitch and duration of a note, a majority of the loss was generated from trying to predict where the note would be placed in the song. Figure 2.1 shows the loss during training over the epochs. In order to prevent overfitting, the algorithm was assigned to stop training more epochs after five iterations without a drop in overall loss of the function.

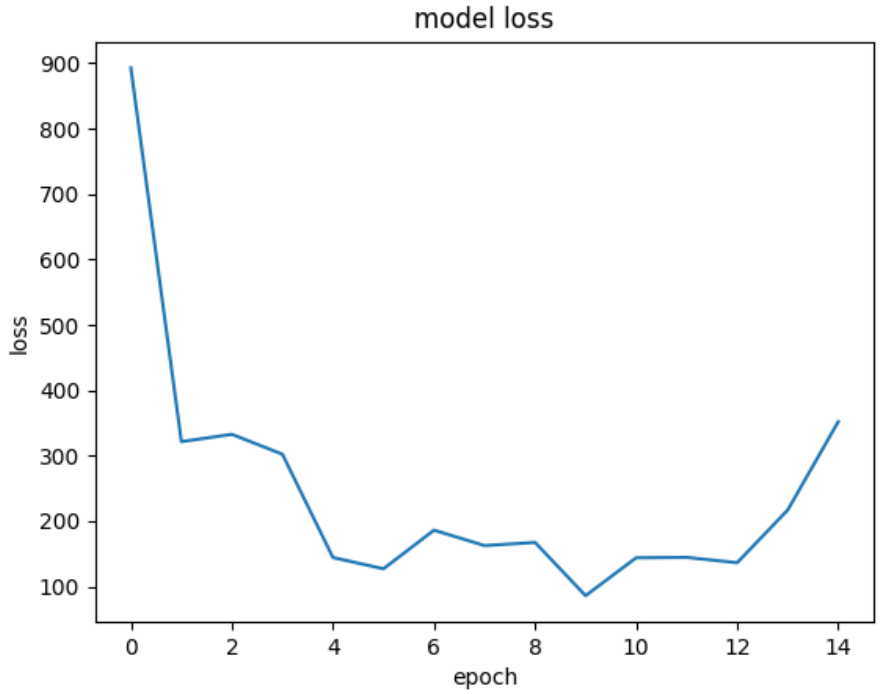


*Figure 2.1*

**Result:**

After training the model, I had the algorithm generate 120 notes seeded from the 16 notes from one of the training songs that I then transferred to a MIDI file. The resulting song was originally very sparse and was very long. This is because despite having only 120 notes, the placement of each note was quiet sparse throughout the song resulting in a single note played once every minute or so. This could largely be a result of the varied lengths of the thousands of melodies that I decided to train my model on. A more uniform song length for each MIDI file could resolve this issue, as seen by the loss generated for step prediction. Overall, this made evaluating the output very difficult, so I decided to forgo the algorithms placement of notes and decided to manipulate the output note array so that the notes would be played sequentially with the same predicted duration. The resulting MIDI was much shorter and note frequency was much higher. That being said, the song was very generic. A majority of the notes were around middle C making for a very flat song. I tried to combat this by having the predicted next note be a major $3^{rd}$ or major $5^{th}$ away from the last note (in music theory this note progression is commonly believed to sound nice, it still repeats notes often. Another hinderance was the fact that there were no chords and were just singular notes being played on a piano.

**Conclusion:**

While I certainly don't have enough background or subject knowledge on music creation or machine learning, my result tells me that wholly artificially generated songs may not yet be the industry killing tool it is often portrayed as. I surveyed 25 people and they were all able to differentiate the artificially generated MIDI file from the other training MIDI files with 100% accuracy. The output created could hardly be called a song but was more akin to a random note progression on a piano. While algorithms can certainly break down a specific genre of songs and find potential trends, or even provide inspiration to an artists, without an original idea or artistic vision, these songs are unlikely to be very popular. Introducing music theory, or even the concepts of a circle of fifths into loss functions such that they adhere to genre norms could certainly improve these outputs, but would still require an in-depth knowledge and understanding of music. The music making process is much more complicated when one considers the amount of post-production and layers that is common in modern songs of today. Without a sufficiently large dataset, artificial intelligence seems to lack the context to create a coherent and pleasing melody.

## Bibliography

Cheng, Ariel (2023). Generating lofi music with RNNs. Kaggle. https://www.kaggle.com/code/arielcheng218/generating-lofi-music-with-rnns