MIDI GENERATION

Cade Stocker

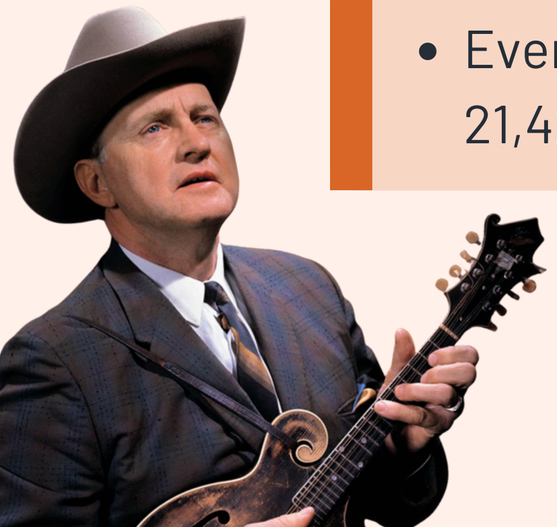
MOTIVATION

- Motivation for this project is to explore lightweight, controllable symbolic models.
  - Explore how sampling strategies impact music created by the model (more abstract note choices have the potential to sound better than predictable choices).
  - Explore the output differences between models using naive tokens vs Miditok tokens (which have more context/info)
  - Evaluate output via metrics like note density, pitch range, and polyphony to quantify how different settings affect the music.
- Changed motivation slightly since I'm working alone now.
- Want to follow architecture shown in Musenet paper (detailed in later slide)
- Eventually, I'd like to explore the original motivation (User inputs text and model creates song matching their description)

DATASET

- Current models have been trained on the Nottingham dataset, a collection of 1200 British and American folk tunes.
- Using this small dataset allows quick training for multiple models, making it easy to compare sampling strategies.
- Bluegrass songs!

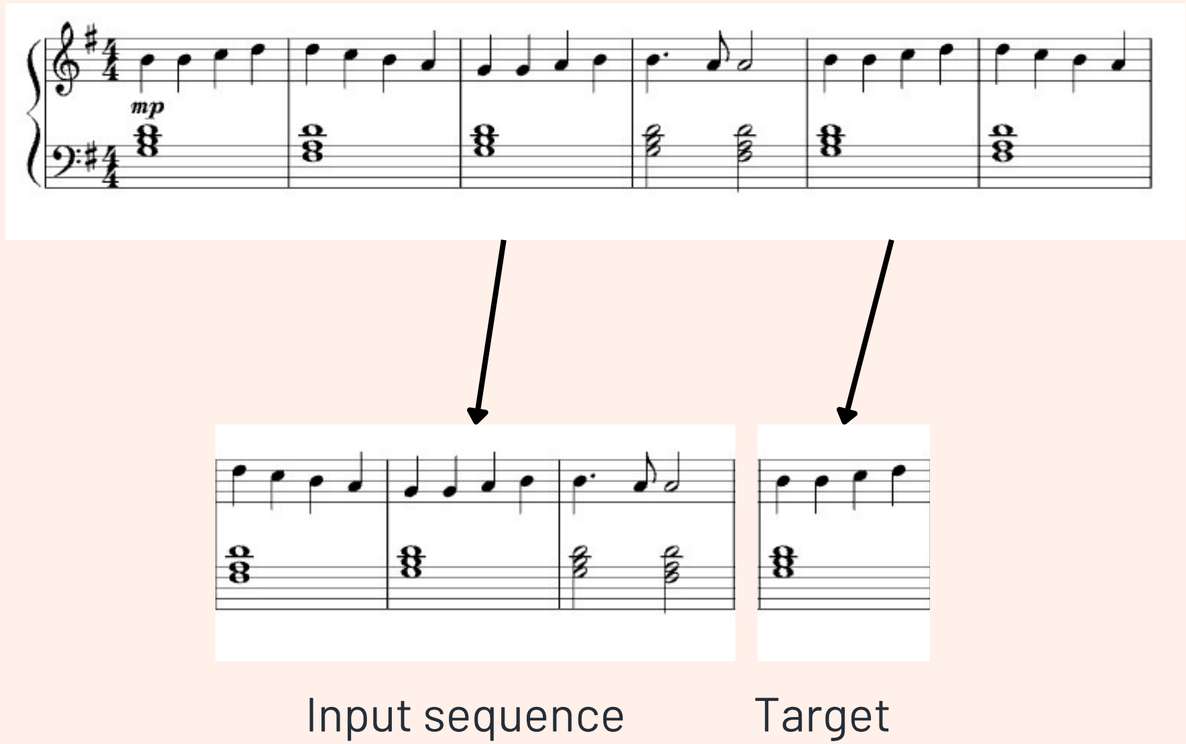
- Eventually, I would like to use the Lakh Piano Roll dataset, which contains 21,425 multitrack piano rolls



PREPROCESSING

- Currently using two methods to preprocess the songs from the Nottingham dataset:

- Naive approach: notes are represented by their letter (has yielded much better-sounding results so far)
  - Cons: output represented as all 8th notes (want to fix this so that notes can be different lengths)
- Miditok: using miditok allows you to use REMI (Revamped MIDI-derived events). Provides more info such as location of the note within the bar, duration of the note, etc.



This is also how a seed is selected for generation (grab a certain amount of a song then predict the rest of it)

Stocker

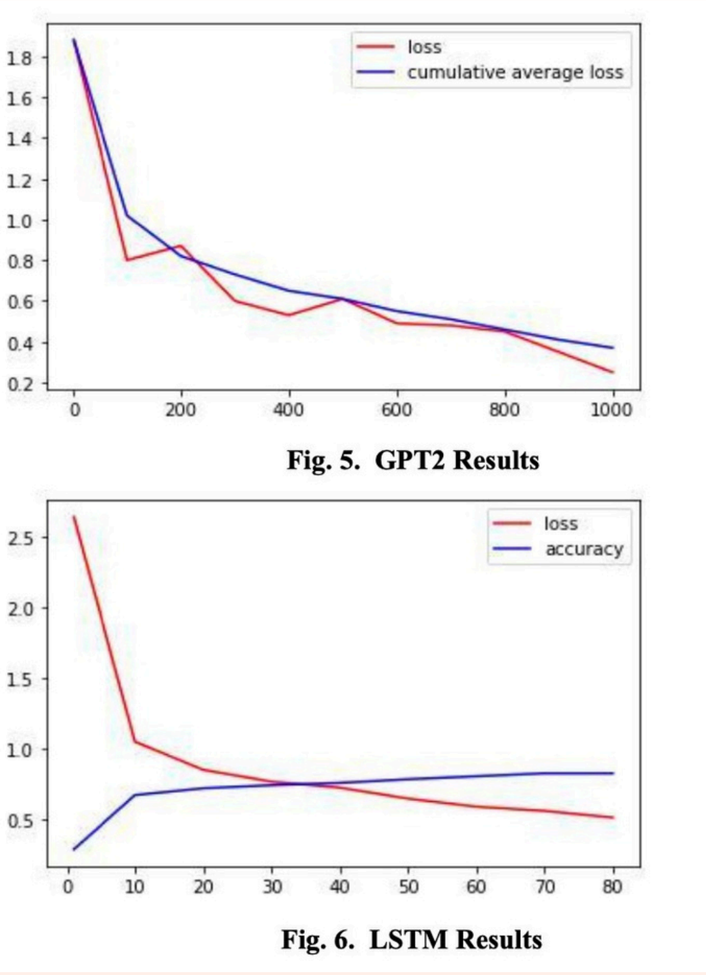
MUSENET

I’m aiming to follow the architecture laid out in **Musenet: Music Generation using Abstractive and Generative Methods**. Musenet uses a discriminator to predict the chord for the next measure, then a generator to create the notes for that measure based on the chord selection.

Currently, I have only trained the generator, which in its current form doesn’t select notes based on a chord. Right now, it grabs a sample from a song in the dataset, then generates more measures based on the that measure.

MUSENET CONTINUED...

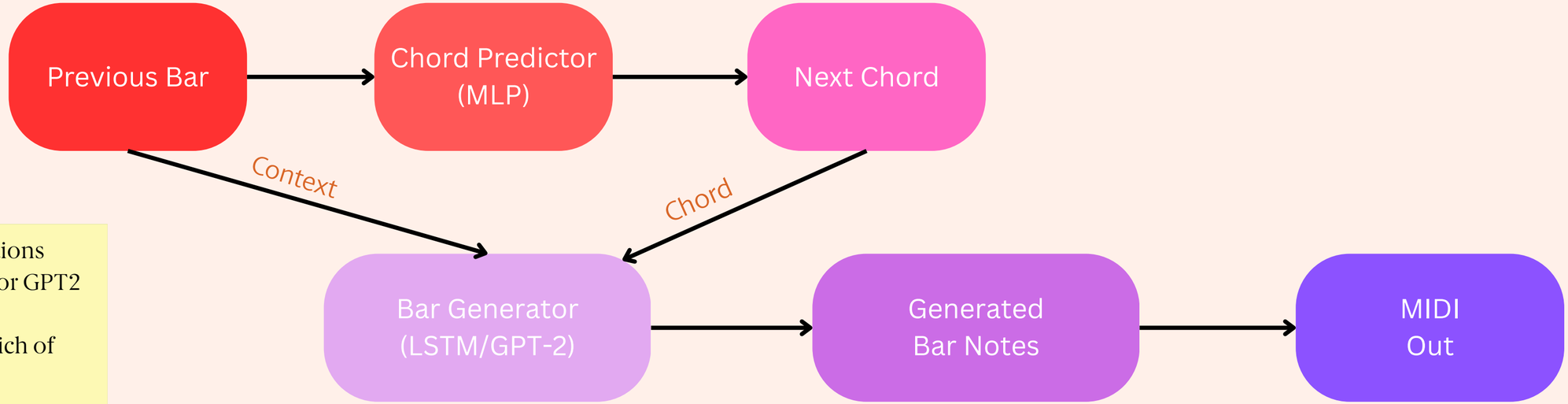
- They tried both LSTMs and Transformer (GPT2) for the generator
  - They state that GPT2 architecture “generates more coherent music than the vanilla LSTM since the GPT2 decoder can extremely easily learn long term dependencies”
- Musenet trained GPT2 on the Nottingham dataset (same one I’m currently using)
- They also used three layers of stacked LSTM



From Musenet paper

CURRENT

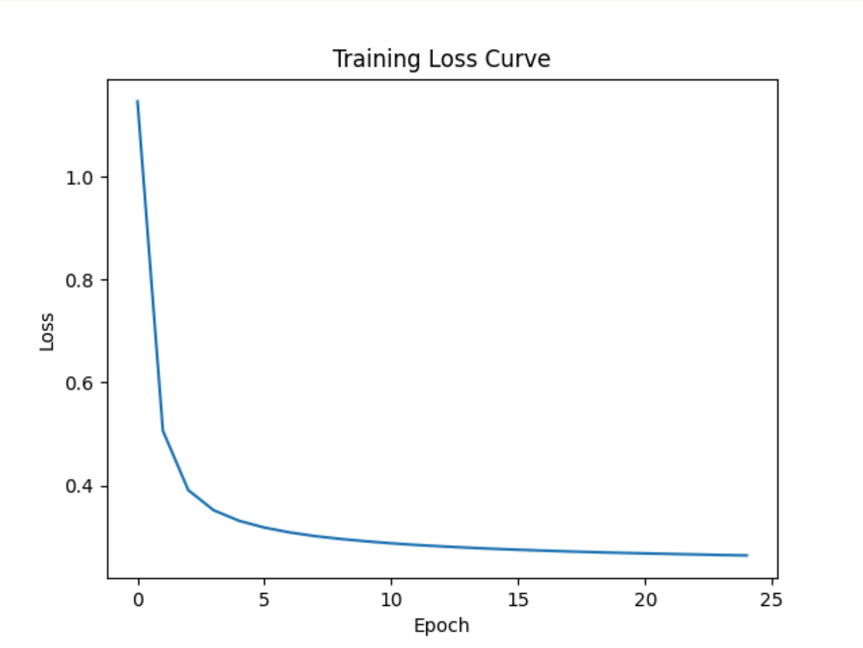
MUSENET



Musenet paper mentions using stacked LSTM or GPT2 for generator.
(I need to decide which of these to do)

Stocker

TRAINING LOSS



Naive Model

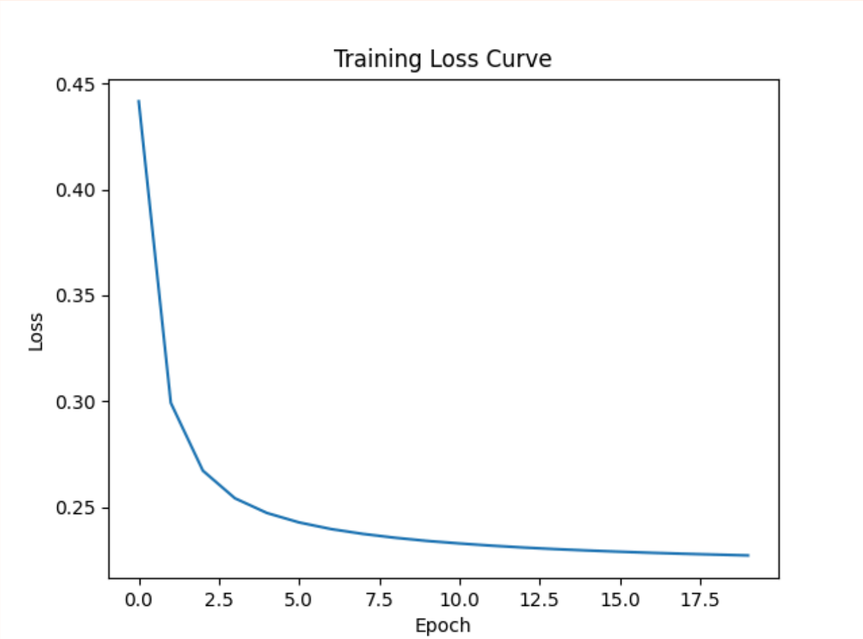
Epochs: 25

Batch Size: 16

Hidden Size: 512

Number of Layers: 3

Learning Rate: 0.0005



Miditok Model

Epochs: 25

Batch Size: 16

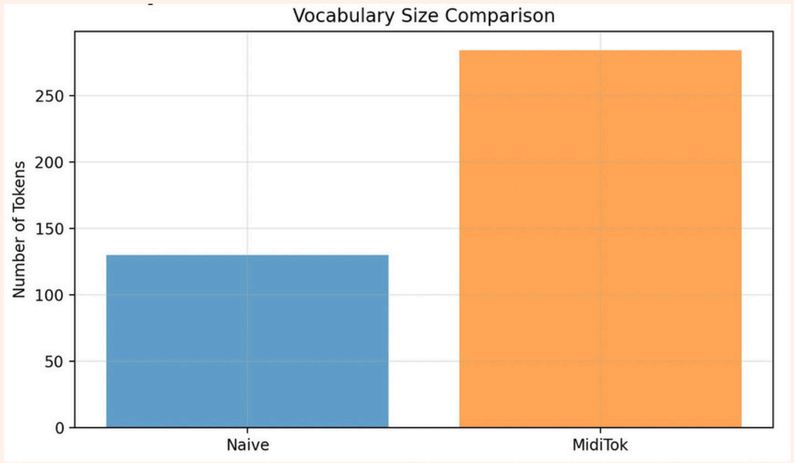
Hidden Size: 512

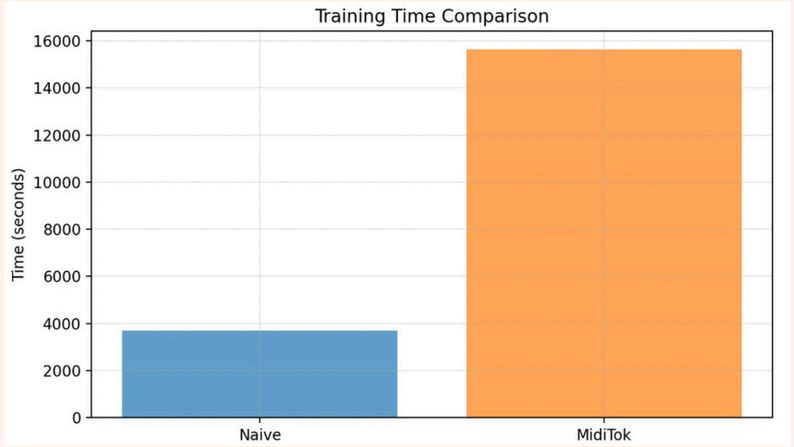
Number of Layers: 3

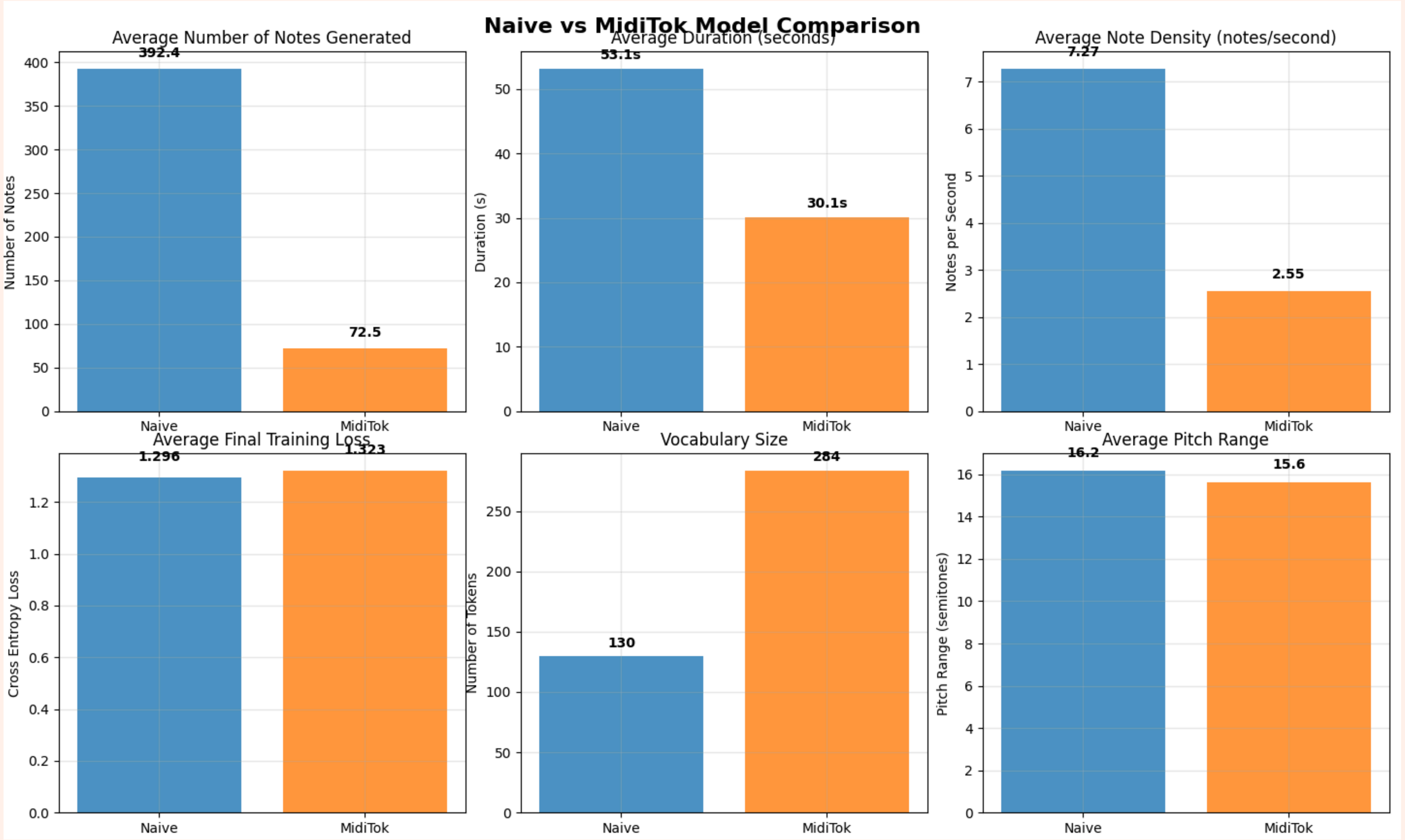
Learning Rate: 0.0005

TRAINING

- Info about models logged into csv files after training.
- Miditok vocab is much larger than naive
- Despite much smaller vocabulary and shorter training time, naive has been consistently producing better music






Naive vs MidiTok Model Comparison
Average Number of Notes Generated
Number of Notes
392.4
72.5
Naive
MidiTok
Average Duration (seconds)
Duration (s)
53.1s
30.1s
Naive
MidiTok
Average Note Density (notes/second)
Notes per Second
7.27
2.55
Naive
MidiTok
Average Final Training Loss
Cross Entropy Loss
1.296
1.323
Naive
MidiTok
Vocabulary Size
Number of Tokens
130
284
Naive
MidiTok
Average Pitch Range
Pitch Range (semitones)
16.2
15.6
Naive
MidiTok

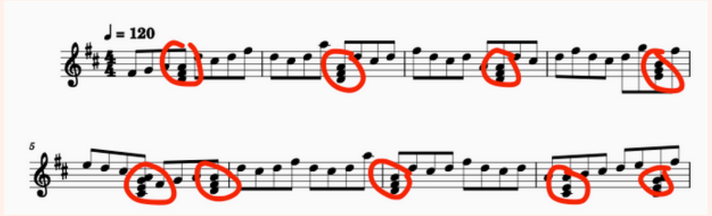
NAIVE TUNES

MIDI files generated by the naive models have sounded much better in the project's current form. I think this is because I need to use a transformer to properly utilize the miditok tokens.

Naive keeps outputting recognizable melodies.

Another weird thing I noticed was the seemingly random output of chords in the middle of melodies from the naive model.

Stocker



Currently, generator grabs random seed from dataset. It just grabbed Old Joe Clark by coincidence.

Stocker

SOMETHING COOL

I had the temperature set higher than usual when it made this song, explaining some of the funky/not optimal sounds.

Stocker

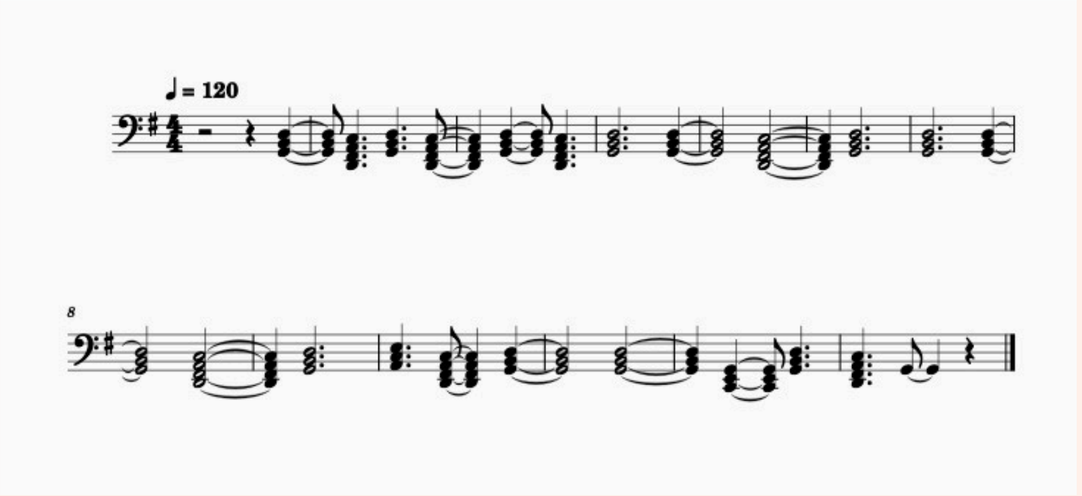
The naive model generated a song that I immediately recognized! It output a traditional instrumental song (Old Joe Clark). The melody at the beginning of the song is very obvious, then it seems that the model added in some random notes that worked within the key. After this, though, it repeated the melody, which I didn't expect to happen.

I found this really interesting because this is how musicians really play these songs (plays some variation of the melody with some sort of improvised ending, then play the melody again slightly differently with another improvised ending).

Song is in AABB form, midi generated only makes it through the A part.

MIDITOK TUNES

Miditok outputs so far have mostly been chords (with a few exceptions). I'm not exactly sure yet why this is, but I want it to produce melodies.



CONCLUSION

- In the project's current form (with one LSTM and a small dataset), naive is performing better. I recognize that this is a very small dataset, and is likely not sufficient to make good/longer music.
  - Able to experiment with different parameters for generation

- Up next:
  - Chord discriminator
  - Transformer instead of LSTM?
  - Larger dataset