Deliverable 3 - Final model changes & Frontend

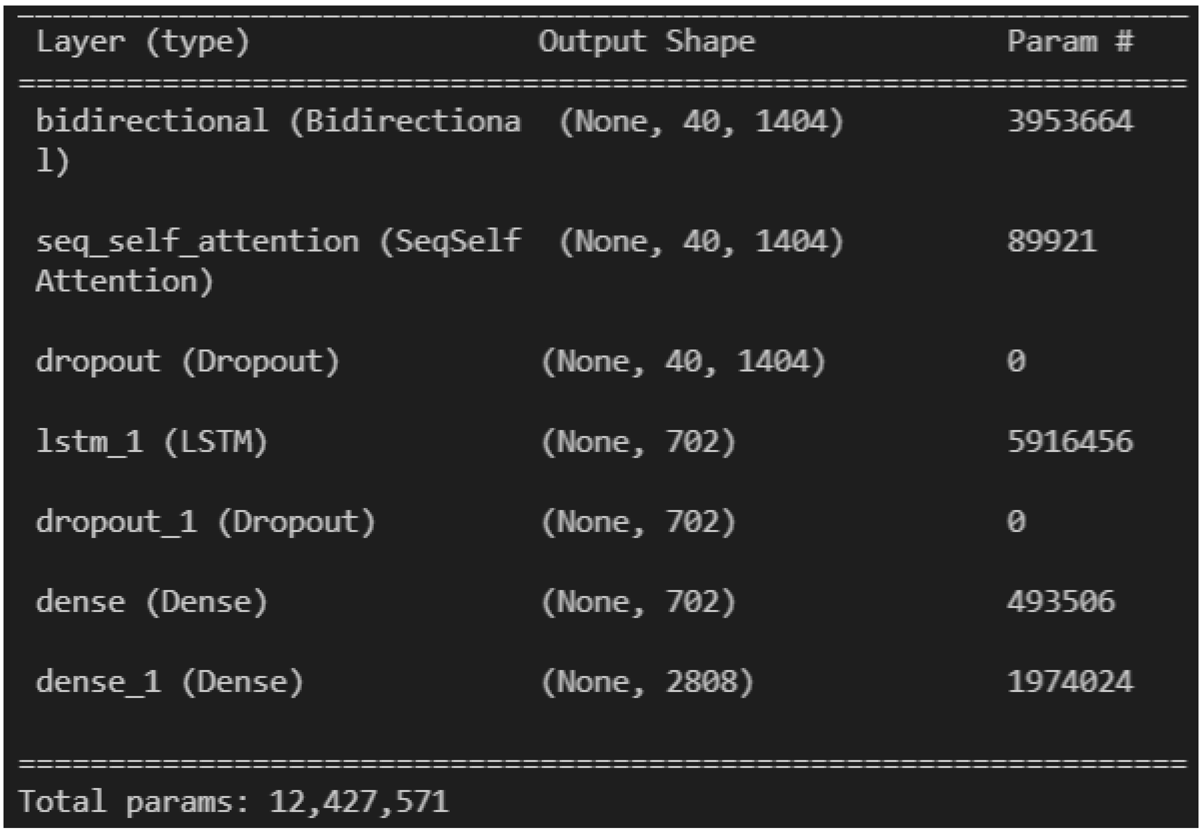
Theodore Glavas (260943207) - Wassim Jabbour (260969699)

Final model changes

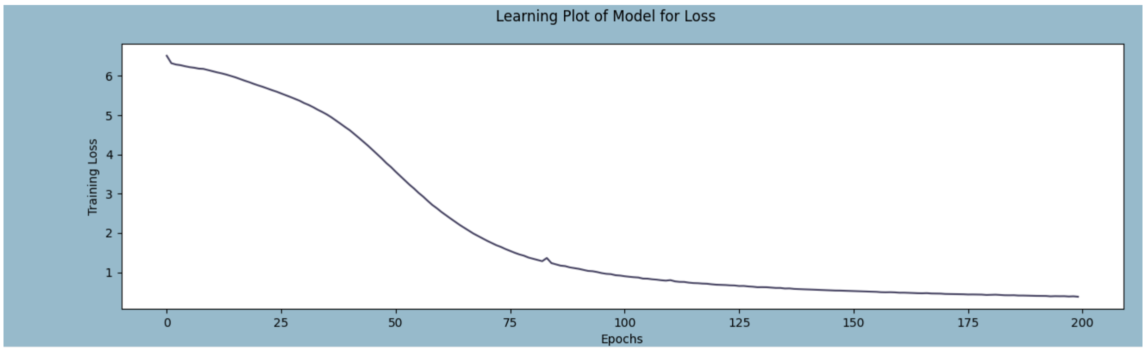
We ended up modifying the pre-processing of the midi files by eliminating rest notes (ie. silences) since we found that multiple midi files had rest notes that were hidden. However, the problem with that is that we lost the flow of the generated music, with all the notes being spaced equally, something very uncharacteristic of classical music which is what we are trying to generate. Consequently, we instead used the offset between notes to implement the notion of flow.
Additionally, we increased the size of the dataset after fixing technical issues that distorted the music of some training data. Our final dataset contains 2808 unique note/chords/durations, and a total of 165227 notes.

On the model side, we experimented with different architectures based on papers and blogs on the subject. We tried using bi-directional lstm layers, self-attention layers and Root Mean Squared Propagation learning rate optimizer in Model V5, but the results were underwhelming.

Model V5:



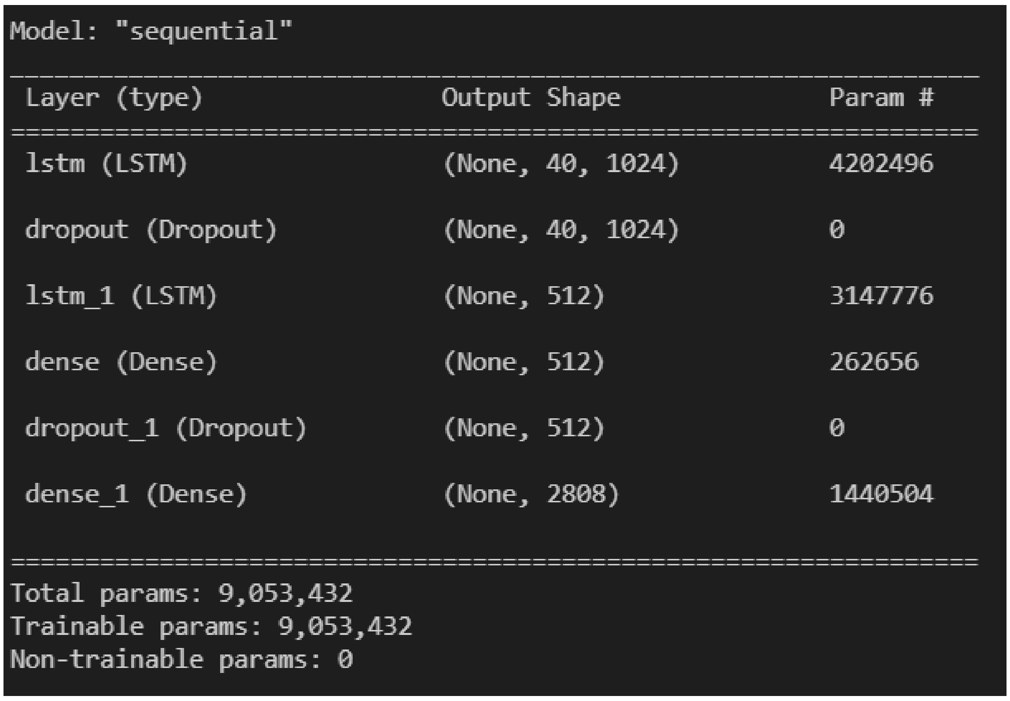
```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 bidirectional (Bidirectiona  (None, 40, 1404)         3953664
 l)

 seq_self_attention (SeqSelf  (None, 40, 1404)         89921
 Attention)

 dropout (Dropout)           (None, 40, 1404)          0

 lstm_1 (LSTM)               (None, 702)               5916456

 dropout_1 (Dropout)         (None, 702)               0

 dense (Dense)               (None, 702)               493506

 dense_1 (Dense)             (None, 2808)              1974024

=================================================================
Total params: 12,427,571
```



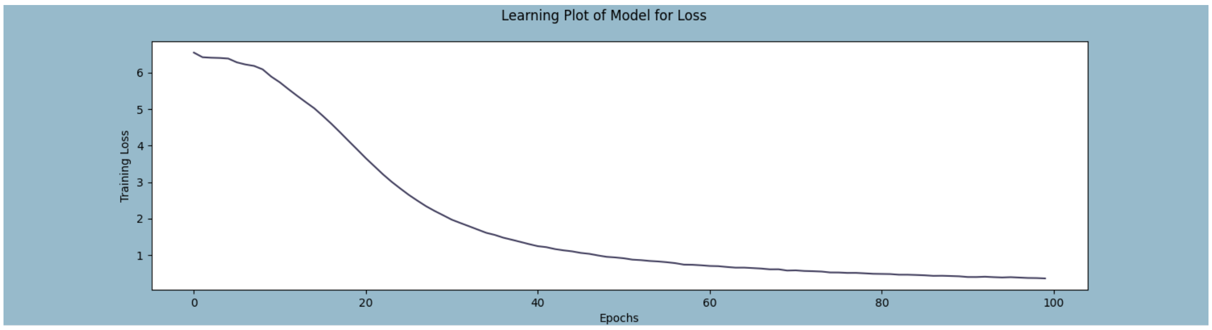
Min: 0.377

Our most successful model is Model V4, using conventional lstm and dense layers with an Adamax optimizer.

Model V4:



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 40, 1024)          4202496

 dropout (Dropout)           (None, 40, 1024)          0

 lstm_1 (LSTM)               (None, 512)               3147776

 dense (Dense)               (None, 512)               262656

 dropout_1 (Dropout)         (None, 512)               0

 dense_1 (Dense)             (None, 2808)              1440504

=================================================================
Total params: 9,053,432
Trainable params: 9,053,432
Non-trainable params: 0
```



Min: 0.36

We finally proceeded to save the weights of our model and generate a few tracks which are also be uploaded to GitHub.

Final Results:
Our final generated music was a large improvement over the preliminary results. Re-working the data-preprocessing method improved the quality of the training data and produced much more music-like results. The final categorical cross entropy loss for model V4 was 0.36 after 200 epochs. Qualitatively, the model produces music with a perceptible rhythm and notes which form short melodies. However, when the predictions start going run, it often spirals out of control and produces gibberish. To combat this, we periodically reset the seed to a random part of our dataset every 100 or so notes to reset the model state and produce better results. There is not really a quantitative metric to evaluate music, so the feedback from people who will listen to the pieces through the webApp will give us a better indication of our success.

Deploying the application

We decided to follow the advice of our TPM and use Flask to create a website that will allow us to showcase our backend which was written in tensorflow. Our goal is to have the website showcase the music tracks we generate based on different classical music composers, such as Chopin, Beethoven, etc… Furthermore, another more difficult to achieve goal is to generate the music live based on midi files of piano pieces passed by the user, although the feasibility of this second part remains to be seen. Our hope is to run the website on one of our machines to achieve the second part.