Deep Learning Music Generation

Kinbert Chou, Ryan Peng

{klchou, pengryan}@stanford.edu

Stanford
CS 230

Introduction

- Music is deeply embedded in our everyday lives from listening to the radio to YouTube music videos. With everyone having a distinct music taste, the area of music is only expanding.
- In recent years, deep learning has reached a level of generating words such as natural language processing (NLP), however, there is less research done on generating music.
- We are interested in music generation using classical music. We plan on modeling musical data similarly to human language in our projects.

Data and Preprocessing

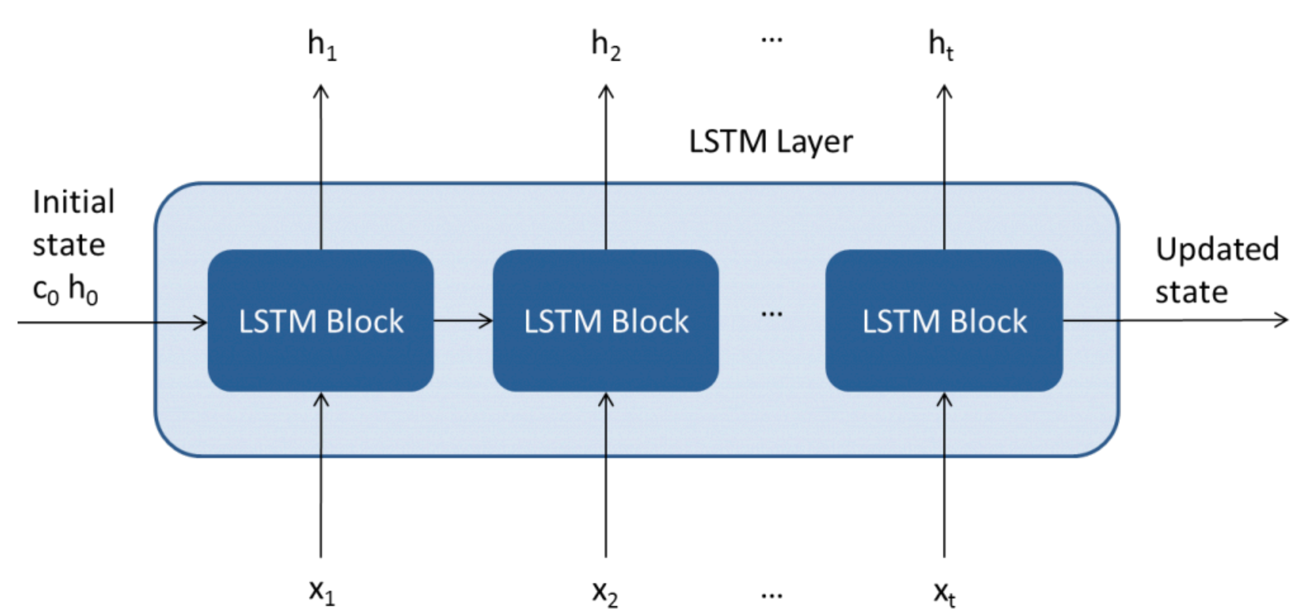
- We used the MAESTRO dataset (6) for our project which consists of over 200 hours of piano music.
- The data contains the notes' object type and this contains information about the pitch, octave, and offset of the note. For the base model there are 967 samples for the train set, 137 samples for the validation set, and 178 samples for the test set.
- We load each file into a Music21 stream object and get a list of all notes in the MIDI file. We append the pitch of every note object using its string representation and tokenize those string outputs to feed it into the network. We continue this "sliding window" process until we have seen all notes in the file. These encodings allows us to easily decode the output generated by the network into the correct notes.

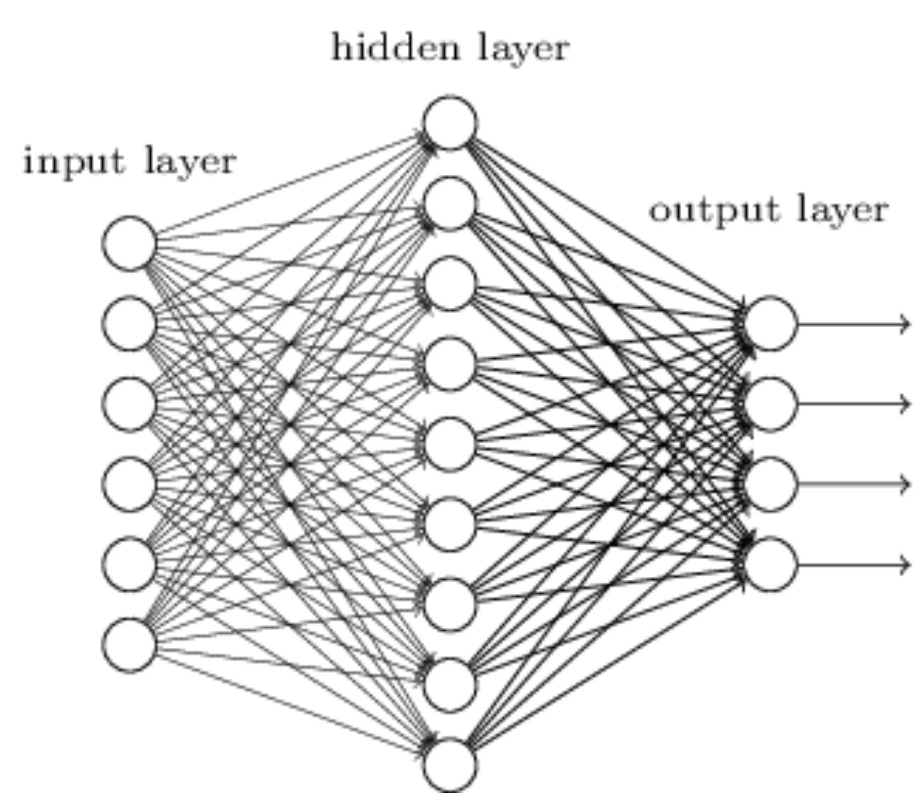
Model and Approach

Dataset

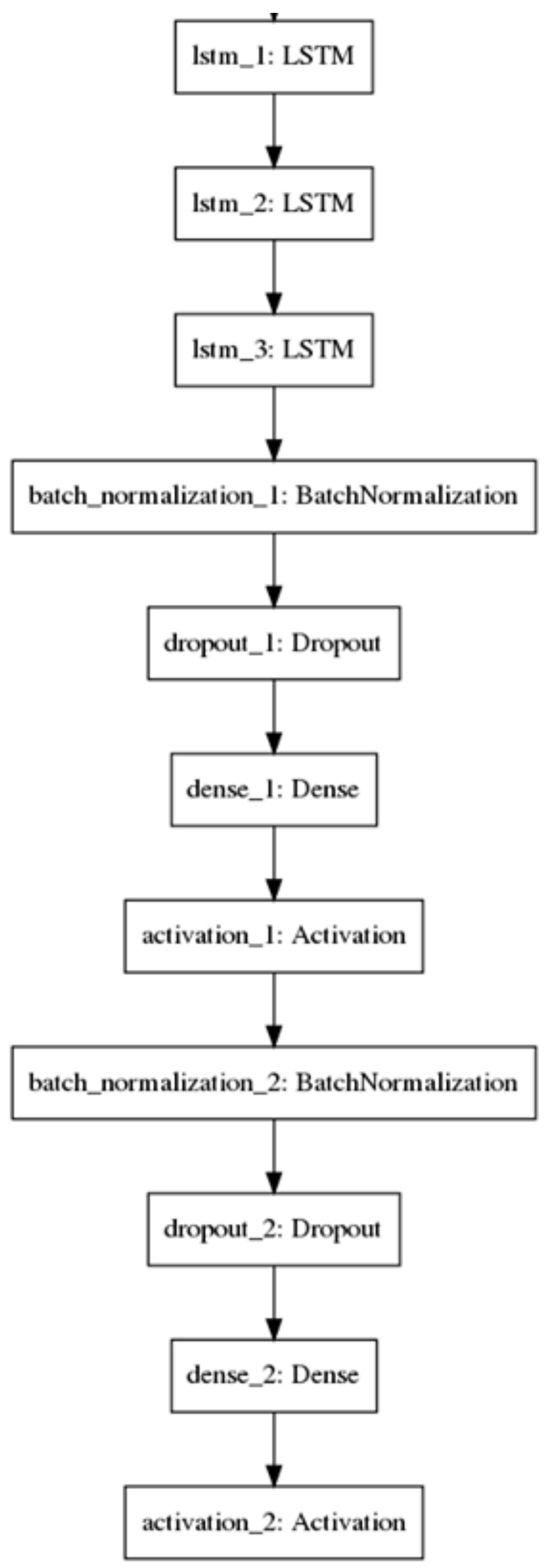
- The base model consists of 967 train samples, 137 validation samples, and 178 test samples.
- Our dataset consists of 2839786 examples (length-100 sequences, next note) pairs generated from our 967 midi files.

LSTM Model





$$\mathrm{softmax}(y)_i = \frac{\exp(y_i)}{\sum_j \exp(y_j)}$$



Results

Hyperparameter Testing

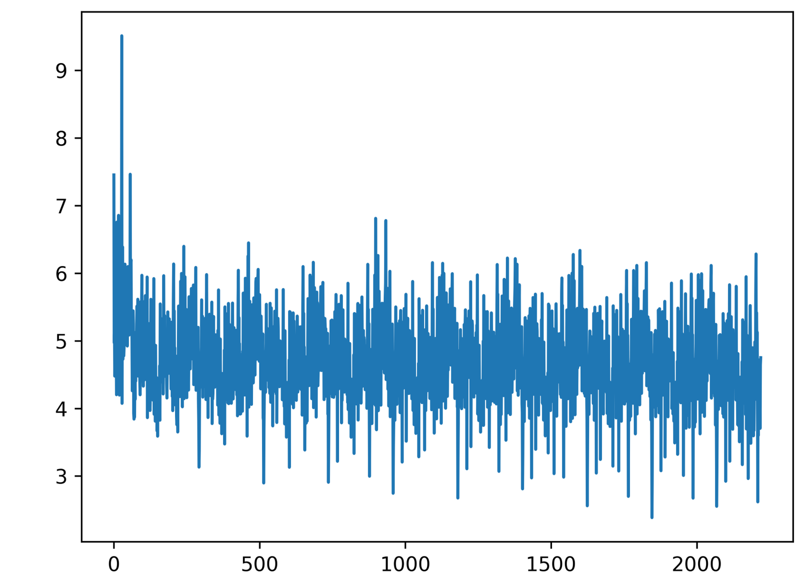
- We trained using a batch size of 32, 64, and 128 and also trained the model for 30, 50 and 100 epochs to compare.
- The model with the best results contained a batch size of 128 with 100 epochs. This model achieved lower loss and higher accuracy compared to other models.

Accuracy and Loss

Accuracy and Loss Table

Model	Accuracy	Loss
128/100 Train	0.0460	1.62
128/100 Validation	0.0152	
128/100 Test	0.0146	
64/85 Train	0.0108	4.27
64/85 Validation	0.0134	
64/85 Test	0.0124	
32/30 Train	0.0108	4.81
32/30 Validation	0.0134	
32/30 Test	0.0124	

Loss Graph



Conclusion and Future Work

- The model overall observes an oscillating loss during each iteration of training due to the difficulty of learning individual songs. However, the loss overall does generally decrease after every epoch.
- Adding more hidden layers to this model may reduce underfitting and improve the overall accuracy.
- Future work would include training for more epochs, expanding to train on different genres of music, and building a deeper and more complex architecture.