Music Generation using LSTM

Abhijeet Dubey(16305R006), Nidhi Singh(163059004), Rishabh Dabral(164050002)

April 30, 2017

1 Task Definition

1.1 Introduction

Music generation through algorithm is a difficult task on which a lot of work is going on. This is a difficult problem since we can't produce music with a limited context. We need to take into account all the frames from the beginning till now to get a meaningful output. But, In Artificial Neural Nets(ANN), we have limited context that we can give as input. Moreover, we have to specify the context size beforehand.

Neural Networks has been widely used for solving the real world problems in a way that human would. But, Neural network can not have infinite memory, i.e. it cannot model time range dependencies. To solve this, Recurrent neural Networks came into the picture. But, they also have the problem of exploding and vanishing gradients which was solved by LSTM RNN(Long Short Term Memory RNNs).

1.2 Problem Statement

The problem can be broadly stated as generating music on the basis of short audio given as input. An input seed will be given, according to which , the musical fragments are synthesised.

Frames after frames needs to be generated in a manner to make the music produced sound like it is in continuation with our input seed. The input can be anything ranging from piano notes to EDM(Electronic Sounds).

More technically, we can define our problem as follows:
Given acoustic vectors representing input audio frames $X_0, X_1, X_2, X_3....X_{t-1}$ for time *0,1,2...t-1*, generate the most proabable next frame X_t representing the acoustic vector for tth time step.

2 Related Works

A good amount work has been done in the field of algorithmic music generation using Convolutional Neural networks and Recurrent Neural networks.

WaveNet[4], by Google, uses Convolutional Neural Networks for synthesising music from input seed. They deal with raw audio waveform instead of mfcc features. They have used a different type of convolution knowns as dilated convolution. In this model, the prediction for any audio sample at timestep t is dependent on previous audio samples *1,2..,t-1*.

Dilated convolutions help increase the receptive field of the model without increasing the network parameters. A dilated convolution is a convolution ap-

plied to input with gaps. These gaps can be thought of as 0 value in the filter. This helps to exponentially grow depth to model long-range temporal dependencies in audio signals.

Wavenet has been trained using Youtube's piano dataset which consists 60 hours of solo piano music obtained from YouTube videos. Also, the MagnaTagATune dataset has been used for training. It consists of 200 hours of music audio. Each 29-second clip is tagged with tags describing genre, instrumentation, tempo, volume and mood of the music.

GRUV[3](Algorithmic Music Generation using RNNs) has compared the performance of two different RNN models. They are comparing the performance of LSTM models and Gated Recurrent units (GRU) models. Their observations were that LSTMs generated better music than GRUs.

Their motivation behind using LSTMs was to model long time-range dependencies and produce a unique musical composition.They are also directly working on raw audio waveforms. They are taking input audio in WAV format. Further, they are processing it to convert into mono-channel waveform sampled at 44.1 kHz.

After training from the same corpus, it was observed that training and validation loss was more with GRU than LSTMs. GRUs were adding noise to the music, while LSTMs output was musically plausible. Hence, even after training on same corpus, the outputs produced by both are highly different.

DeepJazz[1] is another music generation model using deep learning libraries, Keras and Theano. It is a note-based synthesis toolkit. As the name suggests, it is trained to produce *jazz* music. This model also deals with a variant of Recurrent Neural Networks i.e. two-layer LSTM. This model is learnt using the MIDI files. The network is then asked to produce MIDI notes in sequence. The output generated was musically plausible.

John Glover[2], also tried to generate music with Recurrent Neural Network. This model, instead of taking raw audio waveforms as input, takes audio input after Phase Vocoder analysis. A phase vocoder can scale both frequency domain and time domain of audio signals by using phase information. Training is done using stochastic gradient descent with AdaGrad optimizer. The model is implemented in Torch7. The output generated by the model was not at all good. He tested his model on a single sine wave, clarinet sound and piano notes.

3 Model Description

Since the data is a signal of audio waveforms, we were tempted to use RNN based approaches for our problems. We experimented with RNNs and LSTMs to synthesize music frames. Our approach can be visualized using the graph below. At test time, we input a seed

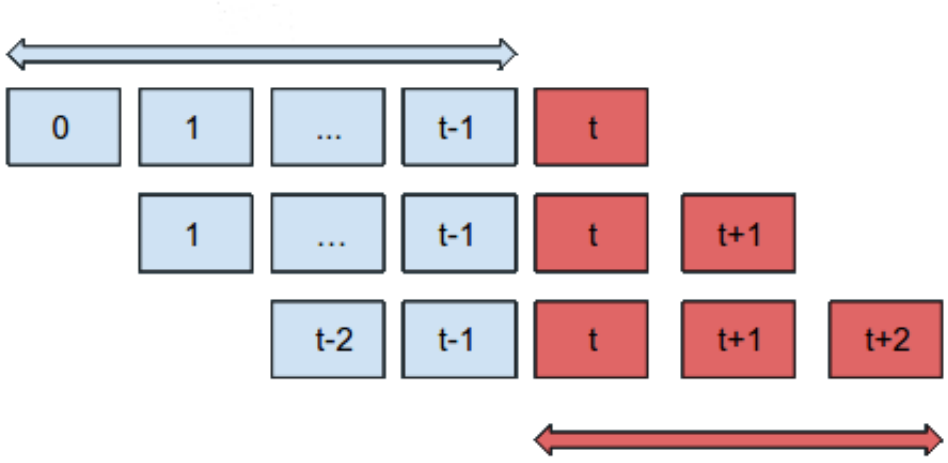

Figure 1: Expected behaviour

We extracted 512 Short Time Fourier Transform(STFT) features for every quarter-second audio waveform (called a frame). These features were hopped with a hop length of 512. Taking cues from the literature, we took logs of the squared FFT features. We trained our model on audio samples from two widely different genres/form of music, namely Piano music and Electronic Dance Music(EDM). These two forms of music correspond to the two extremes of music in terms of number of frequencies involved in the music.

We used the above features to train an LSTM to synthesize music. More formally, given a sequence of frames, the model is trained to predict the features (512 log sq. STFT features in our case). We then compute the mean squared loss between the ground truth feature vector and the predicted feature vector. The loss was minimized using AdaDelta optimization method which claims to be robust to the choices of hyper-parameters. The loss was backpropagated to 50 BPTT time steps. The configuration of our best model is tabulated below:

Parameter	Value
Number of features	512
Hop length	512
Number of Layers	1
Number of Recurrent Units	1500
BPTT unfolding	50
Optimizer	AdaDelta
Training Epochs	2000

4 Implementation Details

Our entire implementation was carried on using several tools in Python. We used *Librosa*, a python library for audio and music analysis to perform feature extraction. In addition to this, we used the software Audacity to perform certain high-level audio operation like audio clipping, file format conversion, etc.

The LSTM model was implemented using TensorFlow. We implemented a python class for our LSTM model and other utility scripts were also written in python which contains useful functions for feature extraction, training our LSTM network and testing it. We used TensorBoard for vizualizations. The model was trained for 10000 epochs which took us approximately 5hrs to train without a GPU.

In the flow of our project, we also evaluated implementations of WaveNet and DeepJazz. We took Keras implementations of both of them. WaveNet took us around 6 hrs to train for 15 epochs and approx. 90 minutes to synthesize a 1 minute long audio stream.

5 Experimental Setup

Our LSTM model was trained on EDM and Piano music separately. We used 2hrs of audio samples for both the kinds of music. The files were broken into shorter chunks of 30 secs and then fed to the feature extractor module of the code. The note-based music synthesis tool, *DeepJazz* takes as input MIDI files of piano tunes. We retrained their architecture on a 1000 This does not require too much processing as the audio is not in waveforms.

The design of of descriptive features was challenging and we explored timbral as well as pitch and rhythmic features. We have also explored spectral rolloff and spectral centroid. The calculated timbral features are based on short term fourier transform and are calculated for every short-time frame of sound. Using STFT we extracted mfcc and pitch related features.

6 Results and Discussion

We explored three broad approaches towards synthesizing music. In our first approach, we contemplated upon synthesizing music using LSTM network. We implemented our LSTM based model in Tensorflow and fine tuned various hyper-parameters. Generated samples from our LSTM based network can be found here.

Next, we looked into existing RNN based model *DeepJazz* which is a note-based music synthesis toolkit and made some tweaks in its code base. Generated samples from DeepJazz can be found here.

Finally, we looked into *Wavenet* by Google for generating raw audio waveforms. We retrained wavenet on EDM and piano data and training took a lot of time because we had limited computing resources. The output generated by wavenet was gibberish and we expected this as we could not process enough training data.

As can be heard, note-based music was the easiest to synthesize. This is particularly not challenging as the complications arising from processing audio waveforms are conveniently avoided by design. LSTMs should, ideally, be the line of attack for this kind of problem, given the temporal aspect of the predictions. The generated audio suffers from the bull-whip effect of errors. We need a quality correcting mechanism to ensure that the frames far away from the seed do not accumulate errors. Further, as discussed during the demo, a single (or double) layer LSTM would be clearly insufficient for the task. We might need attention-based models to better model long term dependencies. Wavenet, on the other hand, has proven to be good in music synthesis. The concept of dilated causal convolutions greatly eases modeling temporal dependencies in raw audio samples. However, we could not witness the goodness of WaveNet due to the lack of training data.

7 Summary

In this report, we discussed the details of our efforts to synthesize music using Long Short Term Memory architecture. We presented our methods for feature extraction and LSTMs and compared our results with note-based music synthesis and raw waveform based music synthesis (using ConvNets). In future, we plan to work with deeper LSTM architectures and more data to see the quality of synthesis. Also, as discussed during the presentation, it would be interesting to see how pre-training of weights fares.

References

[1] Deepjazz: Using keras theano for deep learning driven jazz generation. https://deepjazz.io/. [Online; accessed 28-April-2017].

[2] John Glover. Generating sound with recurrent neural networks. http://www.johnglover.net/blog/generating-sound-with-rnns.html. [Online; accessed 28-April-2017].

[3] Aran Nayebi and Matt Vitelli. Gruv: Algorithmic music generation using recurrent neural networks.

[4] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alexander Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. In *Arxiv*, 2016.