Music Generator Based on MIDI files

Sun Mao

Columbia University
1300 S. W. Mudd, 500 West 120th Street,
New York, NY 10027, USA
mao.sun@columbia.edu

Yu Zheng

Columbia University
1300 S. W. Mudd, 500 West 120th Street,
New York, NY 10027, USA
zheng.yu@columbia.edu

ABSTRACT

This paper introduces how to generate music using auto-regressive model. The file format is MIDI, which is very concise and efficient in recording piano music. The notes in midi files can be organized in different ways, so we can make regression or classification to solve this problem. In order to learn the long-term dependencies in audio processing, this paper tries LSTM and WaveNet-like architecture. Results will be shown and compared.

Index Terms— Music Synthesis, Auto-regressive Model, LSTM, WaveNet

1. INTRODUCTION

MIDI (Musical Instrument Digital Interface) is a protocol developed in the 1980's which allows electronic instruments and other digital musical tools to communicate with each other. [1] In this project, we decide to utilize advantages of MIDI files such as conciseness and small data size, which makes it more efficient in recording piano music. As a result, it becomes easier to generate new samples based on what our model learns from the database.

In order to generate data, both regression model and classification model are tried. For regression model, we encode a MIDI unit in the format [note, velocity, duration]. Along with window size and batch size, input data is actually a three-dimensional matrix. Feeding the predicted output back to input will eventually add up to a MIDI file. For classification model, we split time axis into tiny time units (0.25s) and only consider the "notes" attribute of input data. Then we do classification for each time unit of output.

In terms of model selection, we choose LSTM because it is powerful in learning long-term dependencies. Later on, we also tried dilated causal convolutional layers because it works well in extending receptive field, as is introduced in WaveNet. Both of these two models will be discussed in detail in the following sections.

2. RELATED WORK

Up to now, state of art technologies to generate music include WaveNet and Magenta. WaveNet is a deep neural network for generating raw audio waveforms. The model is fully probabilistic and autoregressive, with the predictive distribution for each audio sample conditioned on all previous ones. [2] It is based on a casual dilated CNN architecture with residual blocks to realize the text and audio mapping. The result is quite good but it is really time-consuming to train and generate music since they are dealing with raw audio only with some sampling. Since WaveNet is quite computationally expensive (O(2^L), L is the number of hidden layers), there are some paper releasing some work to use dynamic programming method to realize the so-called Fast-WaveNet (O(L)). [3]

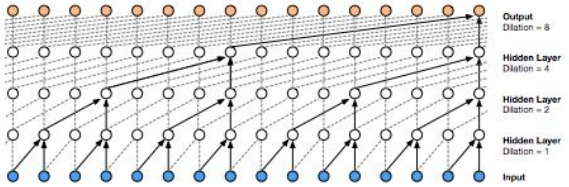

Figure 1 dilated causal convolutional layers [2]

Magenta is a project from the Google Brain team. Its main goal is to design algorithms that learn how to generate art and music. It is based on TensorFlow and is very powerful in processing and generating MIDI files. It comes with 3 pre-trained models. Basic RNN model

uses basic one-hot encoding to represent extracted melodies as input to the LSTM. Lookback RNN introduces custom inputs and labels. The custom inputs allow the model to more easily recognize patterns that occur across 1 and 2 bars. They also help the model recognize patterns related to an event position within the measure. Attention RNN allows the model to more easily access past information without having to store that information in the RNN cell's state. This allows the model to more easily learn longer-term dependencies, and results in melodies that have longer arching themes. [4]

3. MODEL ARCHITECTURE

3.1 Auto-regressive Model

A time series is a sequence of measurements of the same variables made over time. A model is autoregressive when a value from a time series is regressed on previous values from that same time series. for example,

$$y_t = \beta_0 + \beta_1 y_{t-1} + \epsilon_t. \quad (1)$$

In this regression model, the response variable in the previous time period has become the predictor and the errors have our usual assumptions about errors in a simple linear regression model. The order of an autoregression is the number of immediately preceding values in the series that are used to predict the value at the present time. [6]

In our problem, we want to generate music. We organize the note from the midi file into vectors. In the training time, we organize fixed number of such vectors as one input data, its target is the next note right after that set of notes.

When we want to generate new songs, in order to do the prediction, the model must have fixed dimension of input. The first input we choose is called seed which is several consecutive notes we randomly extract from one song. Then, we keep adding the keys we predict to the end of the original sequence and remove the first key in the seed at the same time. At the end, we concatenate these predictions and transform it back into a midi file[7]

3.2 Data Representation

From the discussion above, we already know what the input matrix and output matrix should be like. But it is still tricky to get the vector representation of each note.

In our work, both regression model and classification model are tried. For regression model, we encode a MIDI unit in the format [note, velocity, duration]. Feeding the predicted output back to input will eventually add up to a MIDI file with specified length.

For classification model, we split time axis into tiny time units (0.25s). Then we do the one-hot encoding on the notes the songs played. For example, there are 88 keys in the piano, which means there are 88 possible notes in each timestamp. We can encode the input into a 88 dimension vector. In our real implement, we actually make things a little bit more complex. A 127-dimension vector is used to include some special cases.

3.3 LSTM model

Once we have the all the training data, we should consider which model is better in this situation. Our task is to generate time series, and there is high correlation between the notes we play before and after. In order to make the music style identical, the longer sequence we can remember, the result should be better. Hence, LSTM should be a great choice.

LSTM, which represents for “Long Short Term Memory networks” is a special kind of RNN, capable of learning long-term dependencies. They were introduced by Hochreiter & Schmidhuber (1997), and were refined and popularized by many people in following work. [8]

LSTMs have chain like structure, but the repeating module has a different structure compared with traditional RNN. Instead of having a single neural network layer, there are four, interacting in a very special way.

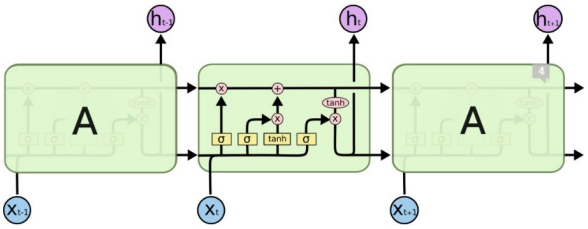

Figure 2 A LSTM neural network

A LSTM network has one main cell which can go directly through the network, it is the “memory” of this network. It decides what kind of information can be remember all the way. Such skip connection is really important because the gradient can go through and the gradient descending problem of RNN is solved this way. In each time step, there are several other gates connected to the main cell: forget gate choose which how much

memory we need to keep and update gate choose what kind of new information we need to add to our memory.

3.4 One-dimension causal dilated model(WaveNet-like Model)

In the LSTM model, we use one main cell to remember the states. It helps a lot to learn the long-term dependency. Actually, there is some more easy way to do that. We can simply extend the receptive field. This model is called one-dimension causal dilated model, which is the main idea of WaveNet raised by Deep Mind.

By using causal convolutions, we make sure the model cannot violate the ordering in which we model the data: the prediction $p(x_{t+1} \mid x_1, ..., x_t)$ emitted by the model at timestep t cannot depend on any of the future timesteps.

A dilated convolution is a convolution where the filter is applied over an area larger than its length by skipping input values with a certain step. It is equivalent to a convolution with a larger filter derived from the original filter by dilating it with zeros, but is significantly more efficient. A dilated convolution effectively allows the network to operate on a coarser scale than with a normal convolution. This is similar to pooling or strided convolutions, but here the output has the same size as the input. As a special case, dilated convolution with dilation 1 yields the standard convolution. Fig. 1 from the paper of WaveNet shows such model.

In additional to all these tricks, we also add some dropout layers and relu layers to prevent overfitting and help gradient go through more easily.

4. EXPERIMENT

4.1 Regressive Model

A simple auto-regressive model based on two layers of LSTM is our first attempt. We installed a library called “mido” to turn MIDI files into matrices for further processing and matrices into MIDI files after we obtain generated data. Though it completed the task of generating data, there is a chance to fall into infinite loops after some time. Meanwhile, due to the nature of regression, we are not able control output range, which means some keys may not be reasonable enough to form a sample of music. Here is when we want to one-hot encode the notes and move forward to the second attempt.

4.2 Classification Model with LSTM

After realizing the weakness of the previous regression model, we think it might be a good idea to do classification instead, which will restrict output value to several certain classes. We also found a better MIDI processing library called “pretty-midi”. We sample the input midi file into some unit of time and one-hot encode them. The result is a little boring when the training epochs is relatively few since each note may just have same duration. Sometimes it is even safe for the model to predict blank output. However, it turns out that a much better result is gained when we add both more training data and training epochs.

4.3 Classification Model with WaveNet-like Model

WaveNet might be the best model up to now in generating audios. Inspired by its architecture of dilated causal convolutional layers, we decided to implement it in place of LSTM. Meanwhile, we keep classification as a final goal since it outperformed regression. Input data will first go through 4 layers of one-dimensional CNN with dilations 1,2,4,8 and a constant dropout rate of 0.2, next is a global max pooling layer to reduce the dimension and two fully connected layers. Finally we use a softmax activation function for classification results. The network is relatively shallow compared with the real Wave-Net due to the restricted computational resource, but the result already make sense like the LSTM model only after hundreds of epochs.

4.4 More music related tools

We have to admit that it is somehow unfair to drop all the music data to the network to do the end-to-end training and ask the machine to generate music as good as people who have years of music knowledge. Hence, we try to teach some of the coded music knowledge such as chord. That’s when we introduce the Music21 library. Using any of the model above, with the Music 21 to better encode and decode the notes, we can see that the result is improved a lot.

5. CONCLUSION

In this project, we use 240 MIDI files as database, both regression and classification as target, LSTM and dilated causal convolutional layers as architecture to generate music. The model is relatively shallow compared with some state-of-art models because it takes

a long time for training.

Comparing regression and classification, classification has more computation amount because each tiny time unit is assigned with an output value. It also takes longer for training. However, due to the fact that output values are restricted to what actually appear in our database, they will make more sense than values generated by regression model. In fact, results of classification model sounds better to human.

LSTM and WaveNet-like architecture result are quite similar since they just learn the long-term dependency in different ways.

In addition to these, use some human knowledge such as chord to encode our data can always make the result better. With the help of Music21 library, we can see this helps both model work better.

Due to the restriction of computation resource and time, we are not able to make the network as complex as the Magenta or real WaveNet. Actually it takes over 24 hours to train these data for 400 epochs. But the result already make sense in our work. We believe results will become better if trained on a more complex model along with a larger database and more epochs.

Please refer to the attached files to listen to our results!

6. REFERENCES

[1] Amandaghassaei. "What Is MIDI?" Instructables.com. Instructables, 12 May 2016. Web. 05 May 2017.

[2] van den Oord, Aäron, et al. "Wavenet: A generative model for raw audio." CoRR abs/1609.03499 (2016).

[3] Paine, Tom Le, et al. "Fast Wavenet Generation Algorithm." arXiv preprint arXiv:1611.09482 (2016).

[4] Welcome to Magenta! N.p., 01 June 2016. Web. 06 May 2017.

[5] Collyer, Charles E., Seth S. Boatright-Horowitz, and Sari Hooper. "A motor timing experiment implemented using a musical instrument digital interface (MIDI) approach." Behavior Research Methods, Instruments, & Computers 29.3 (1997): 346-352.

[6] Deng, Li, and Dong Yu. "Deep learning: methods and applications." Foundations and Trends® in Signal Processing 7.3–4 (2014): 197-387.

[7] Gregor, Karol, et al. "Deep autoregressive networks." arXiv preprint arXiv:1310.8499 (2013).

[8] "Understanding LSTM Networks." Understanding LSTM Networks -- Colah's Blog. N.p., n.d. Web. 06 May 2017.