Music Generation using Deep Learning - A comparison study

Benedith Mulongo
940311-0050
benedith@kth.se

Max Turpeinen
941107-5196
maxtu@kth.se

Kartik Mudaliar
930319-6811
mudaliar@kth.se

Abstract

The goal of this project is to investigate the application of deep learning in a creative task such as music generation. This investigation is performed by comparing two different music representations piano-roll and notes . Furthermore a comparison between two music generation architectures is done. The network used is Long-short term memory and Recurrent neural network Restricted Boltzmann machine. The notes representation gives better result and the piano-roll model is shown to be harder to perform good result without overfitting the data. Furthermore the RNN-RBM model's performance need to be improve

1 Introduction

With the derivation and advent of back-propagation and the possibility it gives for training neural networks, good applications have been found (Perez et al.). Back-propagation enables training of bigger multi-layer neural networks usually used in deep learning. Deep learning have shown good results in artificial intelligence, computer vision and in creative tasks.

In this project we want to investigate the application of deep learning in a creative work such as music generation. Creative work such as music generation, text generation, writing etc is a very human specific task, that is why it is important to investigate the performances of neural networks in such domains. Two different representations of the music is used with LSTM networks and two different systems is compared LSTM and RNN-RBM. The comparison is performed with respect to the generated melodies.

2 Related work

There is a lot that have been done in the area of computer music generation systems. The application of deep learning in this area is also widely studied. A quite exhaustive evaluation of deep learning applications in music generations is performed by Jean-Pierre Briot, Gatan Hadjeres and Francois-David Pachet in their paper *Deep Learning Techniques for Music Generation A Survey* (Briot et al., 2019). The paper goes through most the techniques used up to now and investigate their weakness and possible solutions for improvement.

A shorter but nonetheless interesting paper with respect to this report is *Deep Learning for Music Generation Challenges and Directions written by Jean-Pierre Briot and Francois Pachet* (Fran, 2018). They discuss about different techniques used such Markov models, auto-encoder, variational auto-encoder and transfer learning in the project DeepHear, Restricted Boltzmann machines in the system C-RBM Mozart Sonata Generation etc.

A very similar project is the bachelor thesis *Generating Music in Different Genres using Long Short-Term Memory Networks* (Vranken) by Jeroen Vranken, University of Amsterdam. He used a 2-layer LSTM neural network with the piano-roll representation of the music in order to generate two types of music : classical and jazz.

3 Data

The main dataset used in this project is the Midi piano music from Google project magenta (Hawthorne et al., 2018), but also other music in Midi format from the internet websites [1]. Given that music is a quite heavy data, the quantity of the data do not have to be very big depending of the

[1] The website https://freemidi.org/ has been used in training

representation used. A music snippet of 6 minutes depending of the representation can be represented as a time-series of around 6000 points which is already quite big.

The data used is a few similar songs in order to generate coherent music. There is no preprocessing of the music data, other than the file format must be in Midi format. However two different presentations of the midi file is used, one with piano-roll and the other with only notes extracted from the file .

4 Methods

4.1 Music Basic concepts

As we are working with music generation systems with deep learning, a little notion of basic music terms and concepts is nonetheless important. The following terms (Pilhofer and Day, 2007) is the key concepts used in this project :

- **Rhythm** : A pattern of regular or irregular pulses in music
- **Beat** : A series of repeating, consistent pulsations of time that divide time into equal lengths. Each pulsation is called a beat
- **Tempo** : The speed of the beat
- **Note** : A notation that describes the repetition rate and the length of a certain musical pitch played within the beat
- **Pitch** : A perceived subjective frequency of a sound
- **Chord** : A set of three or more notes played simultaneously

4.2 Models

4.2.1 Recurrent neural network

A recurrent neural network is a sort of neural network for modelling dependent sequences data. It is recurrent because the same function $h_t = f(h_{t-1}, x_t)$ is applied over the sequence recurrently.

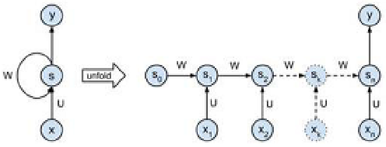

Figure 1: RNN unit

The following operations summarise the recurrent neural network :

$$h_t = \tanh\Big(W_{hh}h_{t-1} + W_{xh}X_t + b_h\Big) \tag{1}$$

$$z_t = softmax\Big(W_{hz}h_t + b_z\Big) \tag{2}$$

The training is performed by calculating the back-propagation through the gradient of the equations below with respect to its parameters :

$$\mathcal{L}(x, y) = -\sum_t y_t \log(z_t) \tag{3}$$

4.2.2 Long-short term memory

LSTM is also a recurrent neural network, but it tries to reduce the problem of vanishing and exploding gradients in vanilla RNN.

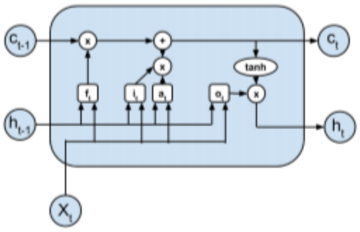

Figure 2: LSTM unit

LSTM is a more complex and complicated networks, however the operations in each steps can be summarised by the following calculations :

$$f_t = \sigma\Big(W_f\Big[h_{t-1}, x_t\Big] + b_f\Big) \tag{4}$$

$$i_t = \sigma\Big(W_i\Big[h_{t-1}, x_t\Big] + b_i\Big) \tag{5}$$

$$\tilde{C}_t = \tanh\Big(W_C\Big[h_{t-1}, x_t\Big] + b_C\Big)$$

$$\tilde{C}_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{6}$$

$$o_t = \sigma\Big(W_o\Big[h_{t-1}, x_t\Big] + b_o\Big) \tag{7}$$

$$h_t = o_t * \tanh C_t$$

4.2.3 Restricted-Boltzmann machine RNN

Restricted-Boltzmann machine is two-layer neural network, where the first layer is the input layer or the visible layer and the second layer is the hidden layer. The restriction is that there is no connections between nodes belonging to the same layer which simplifies the training by enabling conditional Independence between each nodes in the same layer given the nodes in the opposite layer is hold constant. Figure 3 shows a simple RBM network.

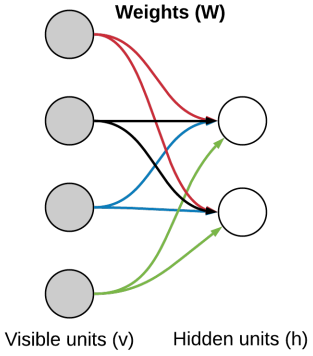

Figure 3: Basic RBM

The likelihood of the weights matrix W given the data **v** is given by :

$$\mathcal{P}(v^n|W) = \sum_h P(v^n, h|W) = \sum_h \frac{1}{Z(W)} \exp\left\{\frac{1}{2} - E(v^n, h)\right\}$$

$$-E(v^n, h) = h^T W v + h^T b + v^T c$$

The learning is performed by modifying the weights such as the likelihood matches the data, doing gradient descent with Gibbs sampling. An detailed explanation is given in (Fischer and Igel, 2012).

The output of the recurrent neural network in the hidden layer is feed into the restricted Boltzmann machines whose output is used for calculating the Back-propagation through time for re-estimating the parameters of the recurrent neural networks.

Figure. 4 shows the model for Recurrent neural network - Restricted-Boltzmann machine [2]. The inputs to the restricted-Boltzmann network layer come from equation. 1 of the recurrent neural network.

An detailed algorithm for RNN-RBM is given in the article (Boulanger-lewandowski and Vincent, 2009).

4.3 Music representation and methods

As with any machine learning the representation of the data is critical not only for the learning process but also for its performances and its interpretability. Here we investigate 2 different representations of the music data :

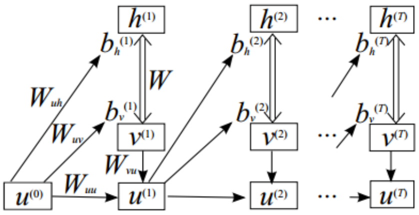

Figure 4: RNN-RBM model

- **Dictionary representation LSTM**
- **Piano-roll representation RNN-RBM**

4.3.1 Dictionary representation LSTM

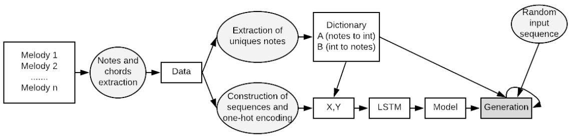

Figure 5: Model 1

The dictionary approach as its name indicates, build a dictionary or map in order to convert notes to integer and vice versa. We have the following steps in this approach :

- **Notes extraction** : Using the python library Music21 [3], we parse each midi files and extracts notes and chords in array X representing the data.
- **Dictionary** : Each unique notes from X is extracted and a map is create to convert each unique notes to a unique integer and vice versa.
- **Reshaping** : The data X is divided in equal length sequence of time step N. Where the notes coming after each sequence is collecting in a array Y. Y is converted to one-hot encoding.
- **Training** : The data X and target Y is used for training the LSTM networks. Y is one-hot encoded for the softmax layer in LSTM.
- **Generation** : A new melody is generated by randomly chose a input-vector and feed it in to the network and using the class output from the softmax layer and convert the

[2] The model used here is highly inspired by the website http://deeplearning.net/tutorial/rnnrbm.html

[3] It is a toolkit for computer-aided musicology and composition. It can be found at : https://web.mit.edu/music21/

number to a notes using the dictionary created before. The new predicted notes is again feed into the network. This process is iterated for a fixed number of times representing the length of the melody generated.

4.3.2 Piano-roll representation RNN-RBM

The piano-roll approach uses a time-series representing each notes played at each time-step. Where previous notes are used to predict coming notes. Theoretically this approach is better because the time-dependency is taking into account and but it have nonetheless a risk to over-fit the training data.

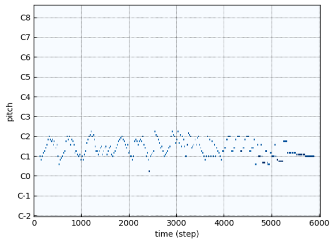

Figure 6: Piano roll

The piano-roll is a matrix where each rows **t** represent the notes played at time t. This matrix is retrieved for each song and feed into the model. The generated updates in the restricted Boltzmann is then used to generated new songs.

The training is performed by the following steps :

- **RNN bias** : The weight matrix between the RNN and RBM layer and the bias is used to estimate the bias for RBM using :

$$b_h^t = b_h + u_{t-1} W_{uh}$$

$$b_v^t = b_c + u_{t-1} W_{uv}$$

- **Gibbs sampling** : Using the ouput of the recurrent neural network, the visible layer v_t of RBM is initialized and gibbs sampling is performed given v_t in order to estimate h and v using :

$$h_i \sim \sigma\Big(W^T v_t + b_h^t\Big)_i$$

$$v_{t_i}^* \sim \Big(Wh + b_v^t\Big)_i$$

- **Back-propagation** : The gradient descent is performed in order to update the weights and the bias. The loss is computed through :

$$loss = F(v_t) - F(v_t^*) \leq -\log(P(v_t))$$

- **Parameter updates** : all parameters are updated and the recurrent neural network described in equation. (1) is calculated using the new values.

The generation is performed almost in the same way as the training but the visible state is initialised by zeros and the Gibbs sampling is performed only for a predefined K steps. Furthermore each generated v_t is added to the generated output matrix.

5 Experiments

The approach used in this project is described in the methods section above. We will here compare the melodies generated by two different networks. The difficulty with the generative systems is the evaluation of the output, because a model with a high accuracy can nonetheless generate poor quality melodies. Nonetheless an experiment is performed on the hyper-parameters and architectures.

5.1 Dictionary-based LSTM

The hyper-parameters that influence the network is : number of epochs, sequence length, architectures, batch size etc.

Figure 7and 8 shows the accuracy and the loss curve for one randomly chosen song, for a very few number of epochs the accuracy is very high. However when using a big subset of the data with different songs the accuracy drops drastically as shown inter alia in figure. 9. The reason is that a big amount of data without any commune trends or coherence in the notes sequences is used for training, which makes the prediction ability of the network very poor.

The experiments done on the number epoch shows that the accuracy become higher with increasing number of iterations, which in the other hand can leads to over-fitting. Figure 9 shows the accuracy for a small subset of the data, the accuracy is increasing for each epochs.

The sequence length also influences the accuracy of the network also. Figure 9 and 10 shows the difference between a sequence length of 100

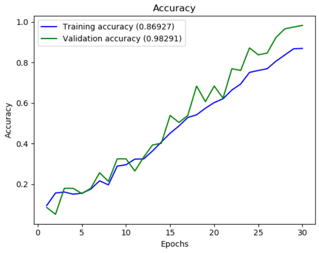

Figure 7: Accuracy curve for one song

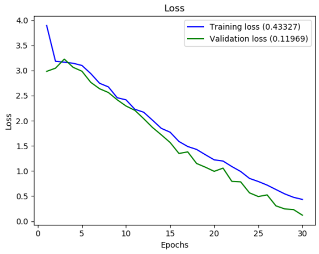

Figure 8: Loss curve for one song

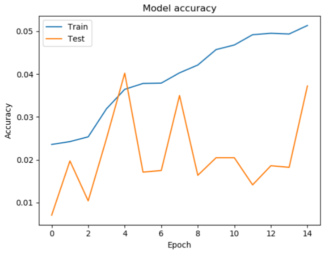

Figure 9: Accuracy 1

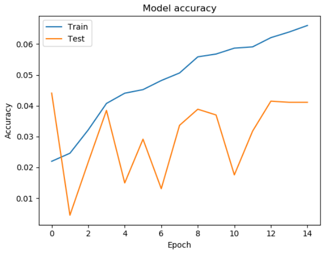

Figure 10: Accuracy 2

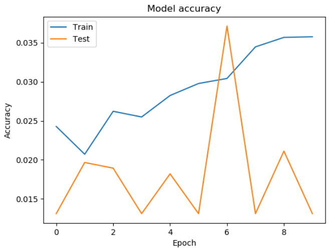

Figure 11: Accuracy 1

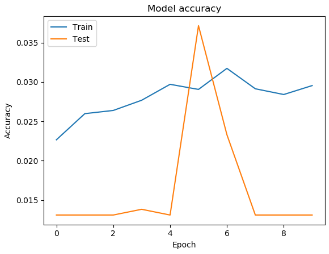

Figure 12: Accuracy 2

and 200. A longer sequence length increases also the accuracy of the network.

Experiment on the batch size shows that the accuracy increases when the batch size is smaller. However all those hyper-parameters increase the computation time for training which is a big problem.

Experiments are done on a small subset of the data using a 15 epochs, 2-Layer and 3-layer LSTM. The 2-layer LSTM has a slightly better performance and the training time is much faster.

The final architecture is a 2-layer LSTM network trained for 50-100 epochs with batch-size = 64 and the sequence length is 100.

5.2 RBM-RNN

Given that the Restricted-Boltzmann machine recurrent neural network is composed of two different parts, it is difficult to estimate the accuracy of the combined network. However the loss function can be calculated.

Figure. 13 shows the RNN-RBM loss curve training for 100 epochs.

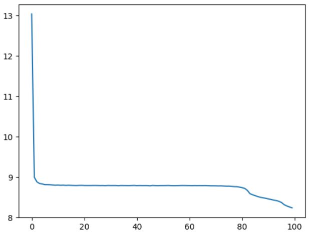

Figure 13: RNN-RBM loss

Figure 14 and 15 show the piano-roll for two randomly chosen samples. As the pictures show, the piano-roll has not clear structure, and the difference between the two samples is not big. The samples seem very similar to each other.

5.3 Comparison LSTM and RBM-RNN

The accuracy of the network is not the best method to evaluate the performance of the two architectures. The generative methods is usually evaluated

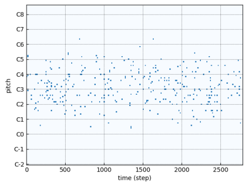

Figure 14: Piano-roll for sample 1

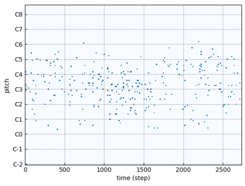

Figure 15: Piano-roll for sample 5

by their end-goal. For example by using Turing test, asking people about their opinion about the melodies generated, we can evaluate the generative system. However the method used for comparing the two architectures is the guidelines explained in the paper (Lerch, 2018), which tries to be more objective.

Figure 16 and 17 shows the piano-roll for the a sample generated respectively by LSTM and RNN-RBM network. The melody generate by LSTM seems to have a structure and a pattern that is recognisable.

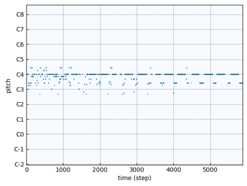

Figure 16: Piano-roll for LSTM sample

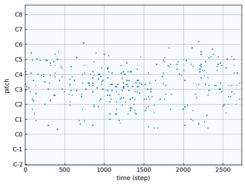

Figure 17: Piano-roll for RNN-RBM sample

Tables 1 and 2 show respectively the different features extracted (Lerch, 2018) for the LSTM and RNN-RBM networks. We can observe from the tables that the mean features on samples from RNN-RBM networks is in general higher than that of LSTM networks. That means the RNN-RBM networks have a higher variety in the melodies, however this variety is no so structured which make the melodies less harmonious.

Figure 18 and 19 show the Pitch class transition matrix for LSTM and RNN-RBM network. The PCTM of RNN-RBM is highly unstable and dispersed, that shows that there is a constant transition between different pitch class, which makes the melody less enjoyable. On the other hand, the self-transition is stronger for the melodies generated by LSTM networks, which create a more harmonious melody.

However when looking at the notes played in the melodies generated by LSTM or RNN-RBM, there not a big variety. The notes played are confined on a small area as shown on the figures below.

Features	**Mean**	**Std**
Total used pitch	13.1	10.42
Bar used pitch	1.72	0.809
Total used note	524.3	77.3
Bar used note	8.32	1.22
Total pitch class histogram	0.083	6.206
Pitch class transition matrix	3.987	1.657
Pitch range	21.80	16.55
Average pitch shift	1.099	1.26
Average Inter-Onset-intensity	0.2419	0.0246
Note length hist	0.0833	1.38e-17
Note length transition matrix	3.63	0.537

Table 1: Features for LSTM generated melodies

Features	**Mean**	**Std**
Total used pitch	63.54	2.63
Bar used pitch	11.2	1
Total used note	366.1	38.84
Bar used note	12.20	1.294
Total pitch class histogram	0.083	8.7e-18
Pitch class transition matrix	4.64	1.06
Pitch range	73.0	3.68
Average pitch shift	14.84	0.93
Average Inter-Onset-intensity	0.164	0.017
Note length hist	0.0833	1.38e-17
Note length transition matrix	2.53	0.269

Table 2: Features for RNN-RBM generated melodies

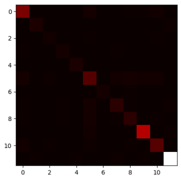

Figure 18: Average pitch class transition matrix for LSTM samples

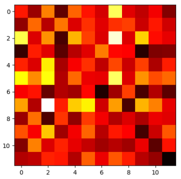

Figure 19: Average pitch class transition matrix for RNN-RBM samples

6 Conclusion

When using a data-set with different songs, the piano-roll models seem to performs worse because it relies on a time-series approach which underlining assumptions is that there is trend which obviously does not exist when using the piano-roll for different songs as the training data-set.

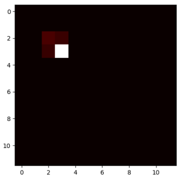

Figure 20: Average note length transition matrix for LSTM samples

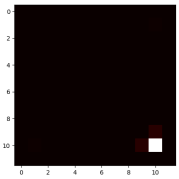

Figure 21: Average note length transition matrix for RNNRBM samples

However when extracting solely the notes, they composition seems more harmonious to listen to, because each notes is generated based on previous notes so the underlining trends is easily incorporated by predicting one note at time. In our case the result of LSTM is more harmonious.

Further extensions can be to try different architectures such as auto-encoder, implementing reinforcement learning in order to generate a specific type of music etc.

References

Nicolas Boulanger-lewandowski and Pascal Vincent. 2009. Modeling Temporal Dependencies in High-Dimensional Sequences : Application to Polyphonic Music Generation and Transcription. (Cd).

Jean-pierre Briot, Gaëtan Hadjeres, Francois Pachet, Jean-pierre Briot, Gaëtan Hadjeres, Francois Pachet, Deep Learning, and Music Generation. 2019. Deep Learning Techniques for Music Generation - A Survey.

Asja Fischer and Christian Igel. 2012. An Introduction to Restricted Boltzmann Machines. pages 14–36.

Jean-pierre Briot Fran. 2018. Deep Learning for Music Generation Challenges and Directions. pages 1–22.

Curtis Hawthorne, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse Engel, and Douglas Eck. 2018. Enabling factorized piano music modeling and generation with the maestro dataset.

Li-chia Yang Alexander Lerch. 2018. On the evaluation of generative models in music.

Javier Andreu Perez, Fani Deligianni, Daniele Ravi, and Guang-zhong Yang. Artificial Intelligence and Robotics.

Michael Pilhofer and Holly Day. 2007. *Music theory for dummies*, 1 edition. Wiley Publishing, Inc.

Jeroen Vranken. Generating Music in Different Genres using Long Short-Term Memory Networks.

A Learning outcomes

The Code can be found at : https://github.com/benedictmulongo/DeepMusic

During this project we have learned a little more about :

- **Benedith Mulongo:** How to use Deep Learning library such as Keras, basic understanding of recurrent neural networks and LSTM. Neural networks, deep learning.
- **Max Turpeinen:** Complexity of Deep Learning and its further implementation possibilities not only for music generation. The possibilty to combine different methods, parameter tweaking etc. In different combinations
- **Kartik Mudaliar:** Better understanding of Deep learning models(RNN, LSTM). How to integrate two different models together for better output (RNN-RBM). Better understanding of packages like Tensorflow, Keras and midi