Melody generation with Magenta for Tensorflow

Yair Castillo and Liza Olivas
Facultad de Ingeniería y Tecnología
Universidad de Montemorelos
Apartado 16-5 Montemorelos N.L. México

Abstract - *Nowadays, it is feasible to create music through an Artificial Neural Network; since composing music is no longer something exclusive to the human being [1]. There are many artificial intelligence methods being used for music generation, such as heuristics in genetic algorithms, neural networks, stochastic methods, generative models, agents, decision trees, declarative programming, grammatical representation, among others [2]. For this project we use the Recurrent Neural Network method. We decided to experiment with the Lakh MIDI Dataset, specifically the clean-midi subset, which contains tens of thousands of MIDI files with filenames which indicate their artist and title. We used Magenta, an open source research project exploring the role of machine learning as a tool in the creative process. Our model generated a batch of 10 completely new melodies (in the form of MIDI files).*

Keywords: MIDI file, Machine Learning, Audio, Music generation.

1 Introduction

Artificial intelligence is increasingly present in our daily lives. We can clearly witness its presence in our smartphones, video games, personalized recommendations on social networks and streaming services, to image and voice recognition, and robots. However, teaching an AI to perform activities that, until a few years ago, were believed to be exclusively of human beings has come to be a subject of great interest, and even controversy [3].

The motivation for using machine learning to generate musical content is how general it is. A machine-learning-based system can automatically learn a model, a style, from ar arbitrary corpus of music [4].

Generating realistic and aesthetic pieces has been considered as one of the most exciting tasks in the field of AI [5] and because of that more and more research is being carried out on music generation with various methods, using Classical [6], Pop [7], Rock, Jazz, and many other musical genres for training.

Because musical signal is sequential, and Recurrent Neural Networks are widely used for sequential data [8], RNN seem to be a good option for the purpose of our project. The main aspect of a RNN is the existence of connections from the posterior layer to the previous one. Because of these connections, Recurrent Neural Networks become dynamic systems. Additionally, they can manage temporal information directly and naturally, alongside their capacity for optimization and for having an associative memory [9].

2 Problem Statement

As part of the Neural Networks course requirement in the Engineering in Computer Systems curriculum of the University of Montemorelos, a challenge was proposed to realize an innovative project based on Neural Networks. For its completion, we aim to experiment and understand the functioning of other types of machine learning. In this case we are focusing on Recurrent Neural Networks through Magenta. All of this through the processing of a model that generates new melodies.

3 Objectives

The main purpose of this project is to generate new melodies through machine learning from the Lakh MIDI Dataset. In addition, we seek to test the efficiency of Recurrent Neural Networks for generating new melodies. Depending on the given results, and if we find a proper dataset, the goal would be to generate hymn-like melodies.

4 State of the art

In [10], the authors implement a musical rhythm recognition system based on a neural network, that uses a variable control method to analize and compare the influence the main parameters have on the network performance. The extraction of music's rhythmic

information is an important research content in the retrieval of content based information. Rhythm is a relatively static and stable factor, therefore, muic's rhythm perception starts from its analysis. Looking at existent research methods, some models proposed by the researchers only consider the spectral energy withouth considering the phase, and some algorithms that are not in real time. Given that most of these rhythm detection system do not track precisely the position of the rhythm through velocity analysis or peak energy, this project uses the LTSM network model to track and extract rhythms, which solves the problem of phase and real time.

In [1], the use of mathematics and its applications for music generation through ANN is discussed. It says that music, in musical-mathematical terms, is divided in three aspects: tone, volume and rhythm. Given that music is a physical phenomenon, these aspects come to be frequency, amplitude, and periodicity. Artificial neural networks only take numerical inputs. Therefore, music must be assimilated with its three cardinals. These are compatibles with a learning network, and a neural network ability to recognize patterns can be taught to recognize these patterns. All sounds can be separated into its characteristic components through mathematical methods.The spectral components of music can be separated through Fourier analysis, and wave components can be obtained through other transformations like Hilbert's. Once obtained the output from the numerical values, it is possible to input them in the neural network. Nevertheless, it is not recommended due to inaccuracy caused by a poor grouping or an inadequate distribution of weights in the neural links of the neural adjacency. The authors were able to solve this problem using data segmentation in sufficiently large groups. Using these methods, and through other mathematical tools, it was possible to create an artificial intelligence system capable of generating music.

In [2], the main goal was to evaluate the tools, models, solutions, and techniques that codify human creativity through musical composition without human interaction during the composition process in the evaluation steps. This is because one of the computational issues is the lack of objective and evaluating methods accepted as a reference point for the generated pieces. The methods were divided in two groups: soft computing methods based on musical composition and symbolic AI methods based on musical composition. Each one of these with subareas. The first one has heuristic composition methods, deep learning composition methods and stochastic composition methods. In the second area we find agent composition methods, declarative programming composition methods and composition methods with grammar. The results were that the main issue of the complete musical composition is the performance measurement. Heuristics and human evaluation were the most used methods for evaluation. Other models, such as the generative model, are still under development and face serious issues. Some of these issues are the stopping criteria during training and optimization using the "theory of the game" model. In the near future, new technologies could emerge and establish standard metrics. Nevertheless, most of these models still depend on human intervention.

In [11], the authors presented that some recent improvements in generative adversarial network (GAN) training techniques prove that progressively training a GAN drastically stabilizes the training and improves the quality of outputs produced. Analysis showed that the generated results of the progressively trained model with deterministic binary neurons exhibits lesser fragmentation of the notes and improved periodicity and melodic perception. Their current model does not produce music equivalent to that composed by professionals, but it has definitely proven to be an improvement over the previous models.

In [12] the authors used a representation to train an LSTM Neural Network, then in a unique approach, applied Reinforcement Learning to select high-level network configurations that maximize a set of attributes that we expect from pleasant algorithmic music. The system demonstrates its ability to emulate some of the notable features of music composed by human composers such as harmony, melodic complexity, tonality, counterpoint, rhythm, and the proper use of treble and bass components. While some of these properties have been displayed by previous research work in this field, the compositions produced by Amadeus present all of these as a coherent whole, thus producing music that is one more step closer to human-level composition.

5 Underpinnings of our approach

Recurrent neural networks, or RNNs, are a family of neural networks for processing sequential data [14]. The hallmark of an RNN, in contrast to feedforward neural networks, is the existence of connections from posterior layer(s) to anterior layer(s) or connections among neurons in the same layer. Because of these connections, the network becomes a dynamic system, which brings many promising capabilities that the feedforward counterparts do not possess. One of the obvious capabilities of RNNs is that they can handle temporal information, whereas feedforward networks must convert the patterns from a temporal domain into a spatial domain first for further processing [9].

Music performance can be represented in various ways, depending on the context of use: printed notation, such as scores or lead sheets, audio signals, or performance

acquisition data, such as piano-rolls or MIDI (Musical Instrument Digital Interface) files. Each of these representations captures partial information about the music that is useful in certain contexts, with its own limitations [15]. MIDI files are a kind of musical files that contain commands for musical performance. They are small in size and widely used over the Internet[16].

Magenta is an open-source research project that explores the role of machine learning as a tool in the creative process. It was created by the Google Brain team. Magenta has two goals. First, it is a research project to advance the state of the art in machine intelligence for music and art generation. Second, Magenta is an attempt to build a community of artists, coders and machine learning researchers. The core Magenta team built an open-source infrastructure around TensorFlow for making art and music [17].

TensorFlow is an end-to-end open source platform to help develop and train machine learning models. It has compehensive, flexible ecosystem of tools, libraries and community resources that let researchers push the state-of-the-art in machine learning and developers easily build and deploy machine learning powered applications [18].

6 Methodology

We created a virtual environment for the project in Anaconda with Python 3.7[1]. Anaconda is a package manager, and environment manager, and a Python/R data science distribution [13]. Then, Magenta was installed in the project environment.

Magenta's Melody RNN model applies language modeling to melody generation using an LSTM. They provide 4 configurations: basic, mono, lookback and attention. We used the attention configuration method to train our model. Attention allows the model to access the information without storing said information in the state of the RNN cell. This results in melodies that have longer arching themes.

The first step to train our model was to convert our MIDI files into NoteSequences, which makes it faster, more efficient and easier to work with the format of the data.

```
convert_dir_to_note_sequences \ --
input_dir=/users/lizii/anaconda3/envs/musi
cproj/clean-midi \ --
output_file=/users/lizii/anaconda3/envs/mu
sicproj/tmp/notesequences.tfrecord \ --
recursive
```

We converted 569 midi files (folders from 'A' to 'Al' of the clean-midi dataset from the Lakh MIDI Dataset) into NoteSequence.tfrecord file for this first experimental phase.

After generating our TFRecord file of NoteSequences, the melodies were extracted from NoteSequences and saved as SequenceExamples. SequenceExamples were introduced into the model during training and evaluation. Each SequenceExample shows a sequence of inputs and a sequence of labels that represent the melody.

```
melody_rnn_create_dataset \ --
config=attention_rnn \ --
input=/users/lizii/anaconda3/envs/musicpro
j/tmp/notesequences.tfrecord \ --
output_dir=/users/lizii/anaconda3/envs/mus
icproj/tmp/melody_rnn/sequence_examples \
--eval_ratio=0.10
```

We generated two collections of SequencesExamples, one for training and another for evaluation. The ratio was set to 0.10 for the eval collection, and 0.90 was saved for the training collection.

For the training stage, we used the attention configuration. The model was fed with the TFRecord file of SequenceExamples that was created. We set the batch size (number of samples to be propagated through the network) to 64. We used a 2-layer RNN with 64 neurons each for faster training. The number of training update steps is optional and, if not specified, the training cycle will run until it is manually terminated. We opted for 2000 training steps.

```
melody_rnn_train \ --config=attention_rnn
\ --
run_dir=/users/lizii/anaconda3/envs/musicp
roj/tmp/melody_rnn/logdir/run1 \ --
sequence_example_file=/users/lizii/anacond
a3/envs/musicproj/tmp/melody_rnn/sequence_
examples/training_melodies.tfrecord \ --
hparams="batch_size=64,rnn_layer_sizes=[64
,64]" \ --num_training_steps=2000
```

We carried out an evaluation job in parallel, to evaluate the model without taking into account any of the weights being updated.

[1] Anaconda distribution download link: https://www.anaconda.com/distribution/

```
melody_rnn_train \ --config=attention_rnn
\ --
run_dir=/users/lizii/anaconda3/envs/musicp
roj/tmp/melody_rnn/logdir/run1 \ --
sequence_example_file=/users/lizii/anacond
a3/envs/musicproj/tmp/melody_rnn/sequence_
examples/eval_melodies.tfrecord \ --
hparams="batch_size=64,rnn_layer_sizes=[64
,64]" \ --num_training_steps=2000 \ --eval
```

This evaluation job is optional. The training and evaluation data is displayed in TensorBoard.

```
tensorboard --
logdir=/users/lizii/anaconda3/envs/musicpr
oj/tmp/melody_rnn/logdir
```

Melodies can be generated during or after training, using the latest checkpoint file of the trained model. The hyperparameters must be the same as those used for the training stage. We can specify the number of melodies that we want to generate and the duration of each melody in 16th steps (1 measure = 16 steps).

```
melody_rnn_generate \ --
config=attention_rnn \ --
run_dir=/users/lizii/anaconda3/envs/musicp
roj/tmp/melody_rnn/logdir/run1 \ --
output_dir=/users/lizii/anaconda3/envs/mus
icproj/tmp/melody_rnn/generated \ --
num_outputs=10 \ --num_steps=128 \ --
hparams="batch_size=64,rnn_layer_sizes=[64
,64]" \ --primer_melody="[60]"
```

We have to feed our model with at least one note so it can generate consecutive notes. The model can be fed with a melody using a Python list in string format or we could provide our model with a melody stored in a MIDI file. If we do not provide a MIDI or a melody, a random note will be chosen as the first note.

Broadly speaking, following the tutorial of Magenta's team, this is how our model was trained to generate melodies using RNN.

7 Experimental results

Using the previous steps, we generated 10 MIDI files with a length of 8 bars (with a duration of 16 seconds each). We fed our model with a single note-on event for the note C4. This means that all the generated melodies start on a C4 note. Beforehand we carried out an evaluation job to test the results in terms of accuracy, which after 2000 steps returned an accuracy of 82%.

8 Conclusions and future work

This article describes a RNN model that allows the composing of new MIDI files using Magenta for Tensorflow. We generated new melodies from the Lakh MIDI Dataset through Magenta, obtaining an accuracy of 82%. We plan on obtaining or even creating a new dataset consisting of hymns. This would allow us to study if hymns have a special structure or even a pattern in their melodies. We hope to develop a model that generates new hymn-like melodies from said dataset. If successful, this would cause an impact inside the Church because from our understanding, it would be the first time hymns would be composed by an ANN

9 References

[1] V. Kavya, 2017. Feasibility of Music Composition using Artificial Neural Networks, DOI: 10.1109/ICCMC.2017.8282520.

[2] R.L. Omar, S.A. Gerardo, 2018. Algorithmic Music Based on Artificial Intelligence: A survey. DOI: 10.1109/CONIELECOMP.2018.8327197.

[3] K.D. Sean, 2019. A philosopher argues that AI can't be an artist. Available in: https://www.technologyreview.com/s/612913/a-philosopher-argues-that-an-ai-can-never-be-an-artist/.

[4] J. Briot, F. Pachet, 2018. Deep learning for music generation: challenges and directions. Neural Computing and Applications, DOI: 10.1007/s00521-018-3813-6.

[5] H-W. Dong, W-Y. Hsiao, L-C. Yang, and Y-H. Yang, 2017. MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment. arXiv:1709.06298v2.

[6] G. Hadjeres, F. Pachet, F. Nielsen, 2017. DeepBach: A steerable model for Bach Chorales generation. arXiv:1612.01010v2.

[7] A. Shin, L. Crestel, H. Kato, K. Saito, K. Ohnishi, M. Yamaguchi, M. Nakawaki, Y. Ushiku, T. Harada, 2018. Melody generation for Pop Music via word representation of musical properties. arXiv:1710.11549v1.

[8] W. Wenli, H. Fang, S. Guangxiao, W. Zhijie, 2018. Music Genre Classification Using Independent Recurrent Neural Network. DOI: 10.1109/CAC.2018.8623623

[9] H. Xiaolin, P. Balasubramaniam, 2008. Recurrent Neural Networks. ISBN: 978-953-7619-08-4

[10] Sun, Y., Jin, C., Zhao, W. and Wang, N. 2018. Design of real-time rhythm tracking system based on neural network. 2018 5th International Conference on Systems and Informatics (ICSAI).

[11] M. Oza, H. Vaghela, K. Srivastava1, year. Progressive Generative Adversarial Binary Networks for Music Generation, arXiv: 1903.04722.

[12] H. Kumar, B. Ravindran, 2019. Polyphonic Music Composition with LSTM Neural Networks and Reinforcement Learning, arXiv:1902.01973v2.

[13] Anaconda Distribution. Anaconda Distribution - Anaconda 2.0 documentation. [Online]. Available in: https://docs.anaconda.com/anaconda/.

[14] G. Ian, B. Yoshua, C. Aaron, 2016. Deep Learning. MIT Press, Available in: http://www.deeplearningbook.org

[15] R. Pierre, P. François, 2019. Smart Edition of MIDI files. arXiv:1903.08459

[16] A. Alexander, N. Zensho, 2005. Three steganography algorithms for midi files. DOI: 10.1109/ICMLC.2005.1527346

[17] E. Douglas, 2016. Welcome to Magenta!. Official post available in: https://magenta.tensorflow.org/blog/2016/06/01/welcome-to-magenta/ See official website available in: https://magenta.tensorflow.org/

[18] See official website available in: https://www.tensorflow.org/