AUTOMATED MUSIC MAKING WITH RECURRENT NEURAL NETWORK

You Peng[1], Ariel Jiang[2] and Qi Lu[3]

[1]Department of Computer Science
California State Polytechnic University, Pomona, CA, 91768
[2]Department of Computer Science University of California, Irvine
[3]Department of Social Science University of California, Irvine
Irvine, CA, 92697

ABSTRACT

Today, the growing market of entertainment has placed a higher demand for music. Quality music is essential for video making, video game making, or even in any public places. However, sometimes finding a suitable list of music can be hard and expensive. This may be solved by automatic, deep-learning based music making. Using Recurrent Neural Network, computers are able to learn the patterns from existing music pieces and convert them to a possibility map. Companies like Google, Sony, and Amper are creating their applications for music generation. We plan to set up a platform where generating music can be done and retrieved directly online. With different options for genre and length, the users can conveniently generate music that fits their needs.

KEYWORDS

Music Generation, Machine Learning, RNN, Web Service

1. INTRODUCTION

The traditional definition of music is similar to what is described as: a set of notes that correlate, harmonize, and express emotion. Most classical music [1] was composed of sophisticated structure and arrangement to better convey ideas and emotions. However, modern music [2] has evolved to serve as more of a leisure-time entertainment with a more rhythmic, patterned, and less complex style which may be learned and generated with machine learning [3][4].

While many companies are doing such businesses: for example, Ample, which provides enterprise-oriented music generation service, they are designed to serve large enterprises and are quite expensive for an individual or small group. On top of that, their registration processes are long and complex which limits accessibility. The purpose of the musicgen System [5] is to provide convenient and quick access to AI [6][7] generated music that is open to everyone as well as testing models to test out potential ways to improve the quality of AI-generated music. The system consists of two parts: the front web page and the core code to generate music.

The front page is created using bootstrap [8]. The layout is simple; every parameter is put into a single block to be selected, and the user just have to go from top to bottom to finish selecting. After clicking the button “generate now”, a playable music file will be returned at the bottommost to be previewed or downloaded.

Natarajan Meghanathan et al. (Eds) : CSEIT, CMLA, NeTCOM, CIoT, SPM, NCS, WiMoNe, Graph-hoc - 2019
pp. 183-188, 2019. © CS & IT-CSCP 2019 DOI: 10.5121/csit.2019.91315

The core code is implemented with Python using Google's open source machine learning library magenta [9]. We trained models with hundreds of sample music online categorized into 4 genres: piano, drums, rock, and electrical. Then, we used flask to retrieve user request and input it into the core code and return it to the user. Finally, we built the website on Amazon's AWS cloud server [10].

2. CHALLENGES

2.1. Music is Complex.

Music is intrinsically complex. It composes of various of elements such as pitches, harmony, rhythm, tempo, etc. It cannot be easily expressed with some combinations of known rules or formula. Unlike taught computers to play chess or Go, where there are clear and fixed rules, it is extremely difficult for a computer to evaluate how well a piece of music is. Therefore, it is difficult to define an objective function to optimize for. The variety and subjectivity of music makes automatic music generation a very challenging task.

2.2. How to build an automatic music generation system that is user-friendly

There have been many approaches for algorithmic music generation, which uses statistical models or artificial neural network. However, these mathematical concepts are difficult for a mature music fans to understand. It is unrealistic to require users to understand how the number of hidden layers and the choice of activation function will affect generated music. It remains a challenge how to enable users to create music with high-level control parameters such as style and rhythm instead of low-level parameters such as number of hidden layer and units, activation function.

3. SOLUTION

In this section, we describe the architecture of our auto music generation system [11] and the algorithms behind that powers this system.

3.1. Overview of the Solution

We have built a web-based [12] interactive music generation system. Figure 1 shows the architecture of the system. For the front-end, we built a web page in html and Javascript. The web UI [13] is shown in Figure 2. It allows users to specify the genres, types and length of generated music. The web page will make Ajax call to the backend server. For the backend, we use Flask to build a server and execute music generation scripts. For the AI part, we use magenta library. Magenta is an open source Python library powered by TensorFlow [14]. It uses Recurrent neural network (RNN) [15] to generate music in MIDI format. The generated music can be played from web page or downloaded for processing using GarageBand.

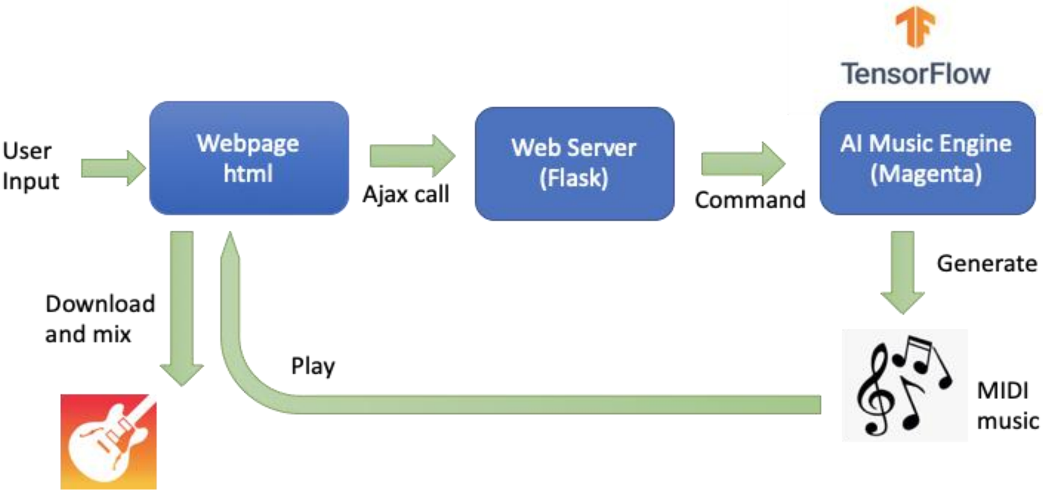
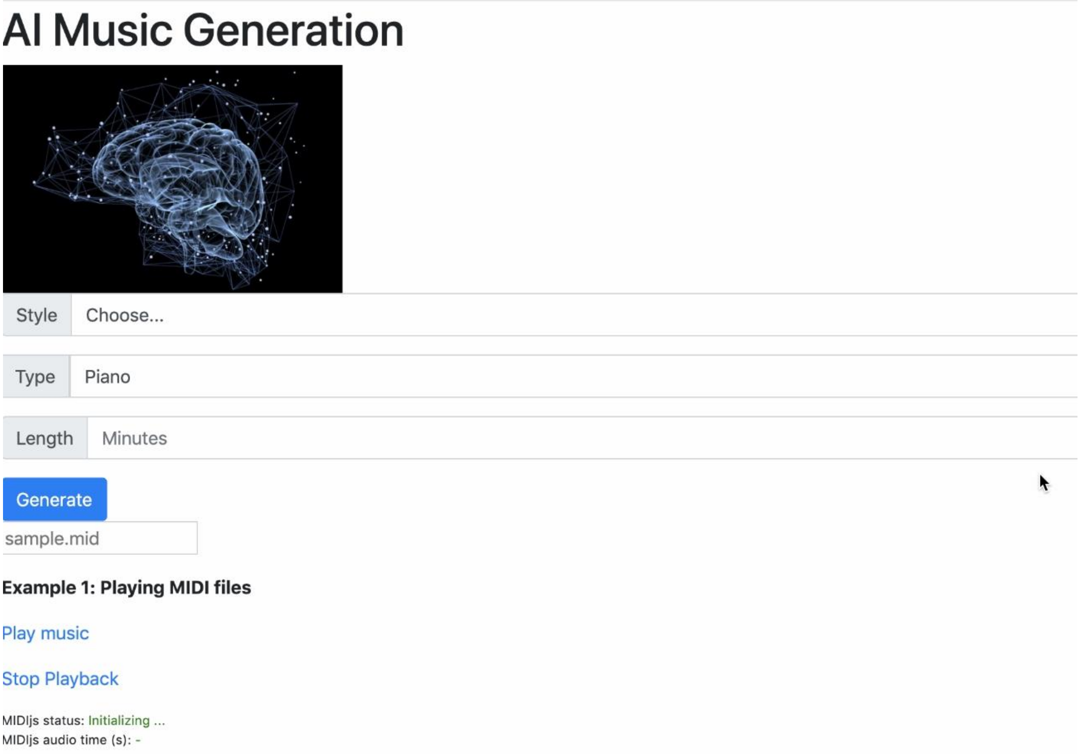


Figure 1. Architecture of auto music generation system



AI Music Generation

Style Choose...

Type Piano

Length Minutes

Generate

sample.mid

Example 1: Playing MIDI files

Play music

Stop Playback

MIDIjs status: Initializing ...
MIDIjs audio time (s): -

Figure 2. Web UI of auto music generation system

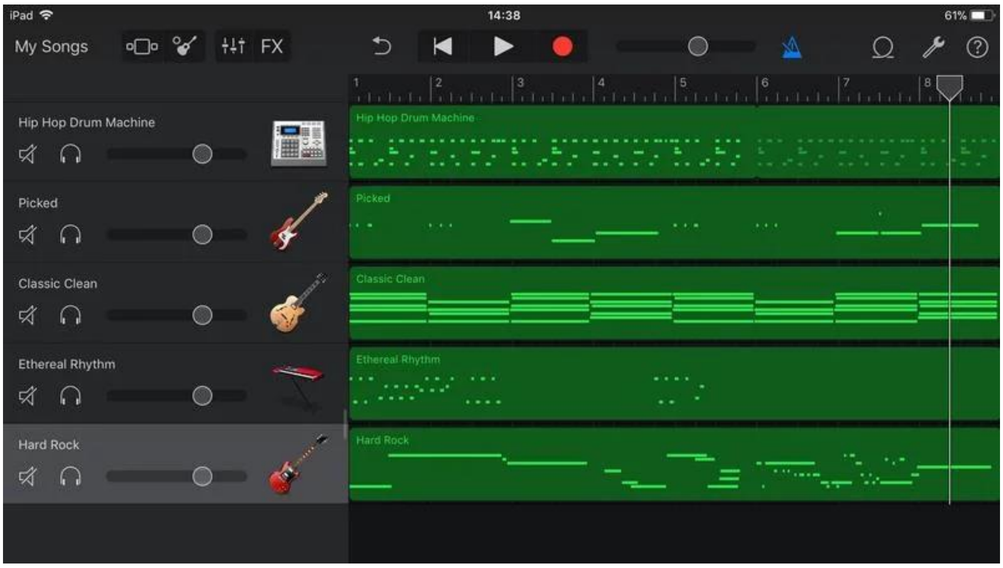


Figure 3. Using GarageBand to mix MIDI music

3.2. Recurrent Neural Network

Recurrent neural network (RNN) is a variant of neural network that can be used for sequential data. Unlike traditional feedforward neural network which only takes a fixed sized input, RNN can take input of arbitrary long. By keeping internal hidden states, the output of RNN not only depends on the input, but also influenced by what it has learnt from the past.
The structure of RNN is illustrated in Figure 4.

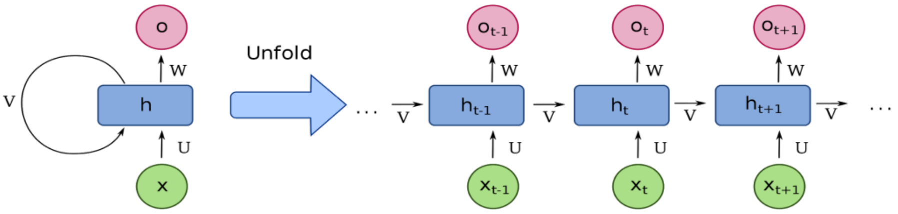


Figure 4: RNN model for auto music generation system

$h_t = \phi(Wx_t + Uh_{t-1})$ (1)

Here, the hidden state at time t is h_t. It is a function of the input x_t and hidden state of previous time h_(t-1). W and U are parameters to be learned. ϕ is the activation function, usually chosen to be sigmoid or tanh.

3.3. Music Generation with RNN

To generate music with RNN, we first need to train the network. By providing different style of training data, the generated music will also be different and conform to the training music. We have used Yamaha e-Piano Competition dataset [16], which contains MIDI captures of ~1400 performances by skilled pianists. Each piece of MIDI is composed of a stream of musical events. An event could be 128 MIDI pitches or velocities. One-hot encoding was used to encode the music into vectors. The RNN network was trained using gradient descent [17]. After the training was completed, the network can be used to generate music similar but not identical to the training music. A starting pitch is needed as the input and RNN will generated subsequent pitches and velocities according the Equation 1.

4. RELATED WORK

There have been persistent efforts for human to develop method to compose music automatically. These efforts can be categorized into different types: some use rule-based system. Some use mathematical models such as fractal, cellular automata or L-system. Boenn [18] used grammars such as harmonics to generate music. Leach et al. [19] analyzed the relationship between fractal and repetitive pattern in music and how to compose music with fractal mathematics. Worth [20] use L-system to generate music. These systems were able to produce reasonable music but the generated music in general lacks variety and novelty.

Recurrent neural networks (RNNs) was first proposed by David [21]. RNNs are able to capture temporal behavior and sequential information. With the increased computational power and algorithm improvement in recent years, RNNs have received great success in many domains such as machine translation [22], speech recognition [23], robot control [24], etc. Also, RNNs have shown surprisingly capability in text generation, such as review generation [25], poetry generation [26].

5. CONCLUSIONS

In this project, we set up a platform where generating music can be done and retrieved directly online, which uses recurrent neural network model to learn the patterns from existing music pieces and convert them to a possibility map. Based on this platform, users can conveniently create their music using proper options for genre and length. Bootstrap is used to develop front page and Google's open source machine learning library magenta is applied to implement RNN model and train it with hundreds of sample music online categorized into several genres.

As for the future work, we will investigate more music genres to update the auto music generation system and make it cover cases as many as possible. Therefore, it can serve users' needs better.

In addition, one limitation related with the auto music generation system is that it does not have enough users in test. we plan to add more features to the system in the next version and collect more feedback from users.

REFERENCES

[1] Lee, S.hyun. & Kim Mi Na, (2008) "This is my paper", ABC *Transactions on ECE*, Vol. 10, No. 5, pp120-122.

[2] Goto, Masataka, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. "RWC Music Database: Popular, Classical and Jazz Music Databases." In *Ismir*, vol. 2, pp. 287-288. 2002.

[3] Adorno, Theodor W. *Philosophy of modern music*. Bloomsbury Publishing, 2007.

[4] Michie, Donald, David J. Spiegelhalter, and C. C. Taylor. "Machine learning." *Neural and Statistical Classification* 13 (1994).

[5] Bishop, Christopher M. Pattern recognition and machine learning. springer, 2006.

[6] Eigenfeldt, Arne, Oliver Bown, and Benjamin Casey. "Collaborative Composition with Creative Systems: Reflections on the First Musebot Ensemble." In *ICCC*, pp. 134-141. 2015.

[7] Haussler, David. "Quantifying inductive bias: AI learning algorithms and Valiant's learning framework." *Artificial intelligence* 36, no. 2 (1988): 177-221.

[8] Li, Yuezun, Ming-Ching Chang, and Siwei Lyu. "In ictu oculi: Exposing ai generated fake face videos by detecting eye blinking." *arXiv preprint arXiv:1806.02877* (2018).

[9] Rubin, Donald B. "The bayesian bootstrap." *The annals of statistics* (1981): 130-134.

[10] Roberts, Adam, Curtis Hawthorne, and Ian Simon. "Magenta. js: A javascript api for augmenting creativity with deep learning." (2018).

[11] Talbot, David. "Vulnerability seen in amazon's cloud-computing." *MIT Tech Review* (2009).

[12] Johanson, Brad, and Riccardo Poli. GP-music: An interactive genetic programming system for music generation with automated fitness raters. University of Birmingham, Cognitive Science Research Centre, 1998.

[13] Khan, Badrul Huda, ed. *Web-based instruction*. Educational Technology, 1997.

[14] Offenhartz, John Ken, and Dana Dawes. "Dynamic generated web UI for configuration." U.S. Patent 9,753,747 issued September 5, 2017.

[15] Abadi, Martín, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin et al. "Tensorflow: A system for large-scale machine learning." In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265-283. 2016.

[16] Mikolov, Tomáš, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. "Recurrent neural network-based language model." In *Eleventh annual conference of the international speech communication association*. 2010.

[17] Zou, Daniel Dore Joey. "DJamBot: Music Generation with Music Theory and Dynamics."

[18] Burges, Christopher, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Gregory N. Hullender. "Learning to rank using gradient descent." In *Proceedings of the 22nd International Conference on Machine learning (ICML-05)*, pp. 89-96. 2005.

[19] Boenn, Georg, et al. "Anton: Answer set programming in the service of music." Proc. 12th Int. Workshop Non-Monotonic Reason. 2008.

[20] Leach, Jeremy, and John Fitch. "Nature, music, and algorithmic composition." Computer Music Journal 19.2 (1995): 23-33.

[21] Worth, Peter, and Susan Stepney. "Growing music: musical interpretations of L-systems." Workshops on Applications of Evolutionary Computation. Springer, Berlin, Heidelberg, 2005.

[22] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors." Cognitive modeling 5.3 (1988): 1.

[23] Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." Advances in neural information processing systems. 2014.

[24] Graves, Alex, Abdel-rahman Mohamed, and Geoffrey Hinton. "Speech recognition with deep recurrent neural networks." 2013 IEEE international conference on acoustics, speech and signal processing. IEEE, 2013.

[25] Mayer, Hermann, et al. "A system for robotic heart surgery that learns to tie knots using recurrent neural networks." Advanced Robotics 22.13-14 (2008): 1521-1537.

[26] Yao, Yuanshun, et al. "Automated crowdturfing attacks and defenses in online review systems." Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2017.

[27] Yi, Xiaoyuan, Ruoyu Li, and Maosong Sun. "Generating chinese classical poems with rnn encoder-decoder." Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data. Springer, Cham, 2017. 211-223.