Available online at http://www.advancedscientificjournal.com
http://www.krishmapublication.com
IJMASRI, Vol. 1, issue 10, pp. 369 - 373, Dec. -2021
https://doi.org/10.53633/ijmasri.2021.1.10.019

INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY ADVANCED SCIENTIFIC RESEARCH AND INNOVATION (IJMASRI)

ISSN: 2582-9130 **IBI IMPACT FACTOR 1.5** **DOI: 10.53633/IJMASRI**

RESEARCH ARTICLE

MUSIC GENERATION USING DEEP LEARNING.

[1]Vijeet Gahlawat and [1]Aakash

[1]Department of Information Technolopgy Maharaja Agrasen Institute of Technology,Rohini, Delhi, India.

Abstract

We examine how lengthy short-term memory neural networks (NNs) may be utilised to create music compositions and offer a method for doing so in this study. Bach's musical style was chosen to train the NN in order for it to make similar music works. The recommended method converts midi files to song files before encoding them as NN inputs. Before feeding the files into the NNs, they are augmented, which converts them into distinct keys, and then they are fed into the NN for training. The final phase is the creation of music. The primary purpose is to assign an arbitrary note to the NN, which it will gradually modify until it produces a good piece of music. Several tests have been conducted in order to identify the ideal parameter values for producing good music.

Keywords: lstm, music generation, deep learning, machine learning, neural networks

Introduction

Nowadays, music generation is quite crucial. It has a wide range of applications. Musicians and artists build on the machine's output to create their own creative work. Software-generated music or art is sometimes marketed by the companies or individuals that created it. Corporations and retailers can licence them for advertising, shop music, and other uses. Aiva, an artificial intelligence music generator, is an example of this. It has already issued its own copyrighted albums of generated soundtracks in the electronic music genre (Travers, 2018). A picture called Portrait of Edmond Belamy, created by a GAN network, was auctioned for $432,500, demonstrating the usefulness of utilising neural

networks (NNs) for generating art or music (Christies.com, 2018). Jukedeck is another example, which may generate music based on the genre and beats per minute set by the user.

For many years, automated music creation has been a frequently investigated way of creating new music. Prior to machine learning, generative grammars were the most widely used method for creating music. These prior methods resulted in simplified music with simple melodies and several flaws, such as predictable repeated themes. NNs is the most recent (and arguably greatest) technology for producing music at the moment (Agarwala, Inoue, & Sly, 2017). NNs may be trained to make their own original music pieces using Bach's music; the network learns the pattern of the supplied music and then uses it to generate its own. DeepBach (Hadjeres, Pachet, & Nielsen, 2017) and the BachBot (Liang, 2016) are two examples of recurrent NNs in use. After being taught with hundreds of Bach's chorales, these music generators were able to make music in the style of Bach (Hadjeres et al., 2017).

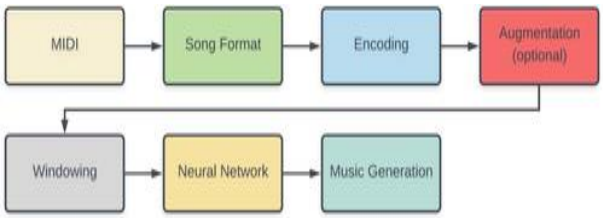
Related work

Over the last decade, music composition automation has been used in a variety of ways. Rule-based systems (grammars) and algorithms that exploit randomization, such as Langston's riffology algorithm, were among the first methods used (1988). L-Systems were a common type of grammar for music. These string rewriting grammars could generate reasonably simple note sequences (McCormack, 1996). In recent years, the focus for music creation and generation has switched to machine learning and artificial intelligence techniques. This is due to NNs and deep learning's capacity to produce tunes that are less predictable and more melodically complex than their random algorithm and grammar system equivalents. LSTM networks have been used to construct the bulk of music generators in recent years. However, they differ in the encoding of MIDI inputs and other features of the NN structure.

Eck and Schmidhuber (2002) were the first to use LSTMs instead of RNNs to create blues music. Blues is a jazz-influenced music genre that features a lot of riffs, or guitar solo note sequences. The network was successful in producing music with a structure that resembled traditional blues music. They didn't use MIDI files to represent note sequences, instead using their own format. Their LSTM network was able to learn chord sequences and recognise the structure of blues chords. Skuli (2017) recently developed a music generator using a dataset of Final Fantasy (the video game) MIDI tracks. Skuli utilised MIDIs to train an LSTM NN and then generated his own MIDIs, which is comparable to our work.

To summarise, we avoided using Generative Adversarial Network (GAN) ideas in our research (Skymind, 2017). A generating network plus a discriminator network make up a GAN. The generator takes as input random vectors and produces a new output. The discriminator uses a sample dataset to identify whether the generated output is a sample or a fake result, and it forecasts the generated output's category. The purpose of this network structure is to produce outputs that can be included in the original dataset (Skymind, 2017). The primary issue with GAN is that the training between the two networks is insecure. This was proved by Agarwala et al. (2017), who found that the GAN network did not produce any significant output due to the model's instability. Only if there is some form of training balance will it function (Tchircoff, 2017).

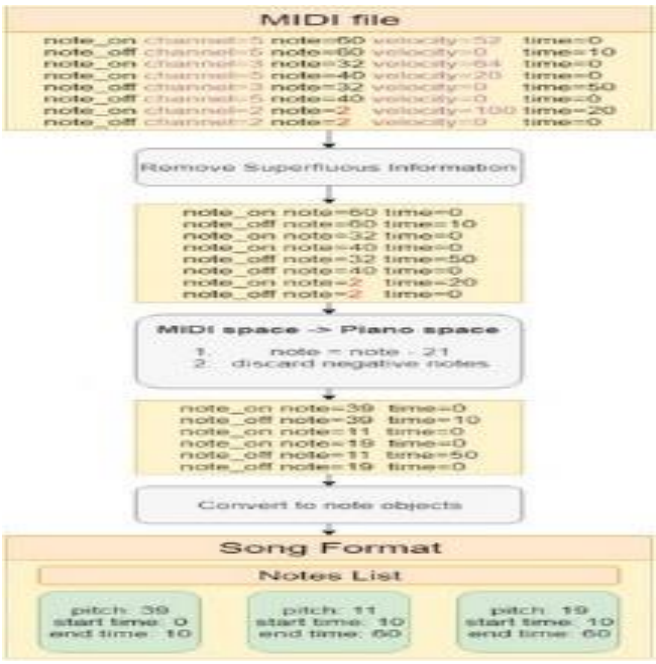
Methodology and Architecture



- ***MIDI files conversion to song format***

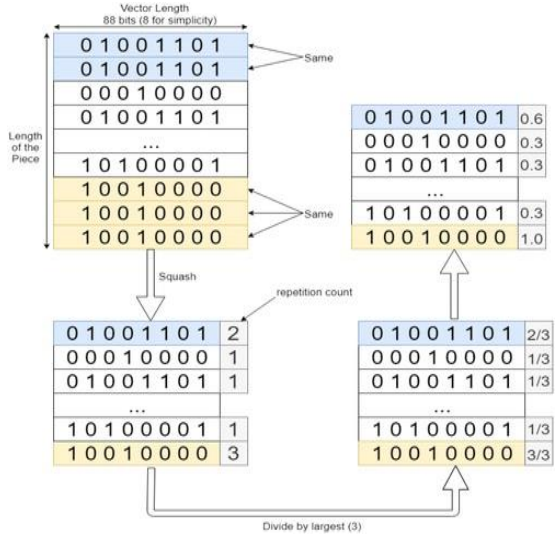
All midi files have been transformed into a custom-made intermediary format (Song Format) that allows us to work with notes in a more natural way. Notes performed in all 16 MIDI channels are captured into the Song Format while repeating over

the events of each file. It's worth noting that MIDI files will occasionally send a 'note on' message with a 'velocity' of 0 to signal a 'note off.' A number corresponding to the time (in ticks) since the last message is stored in the 'time' attribute of each MIDI event. The Song Format is a pair of classes that are responsible for storing note information and converting MIDI files. As a result, the Song Format class handles the functionality of exporting and importing MIDI into the encoded format.

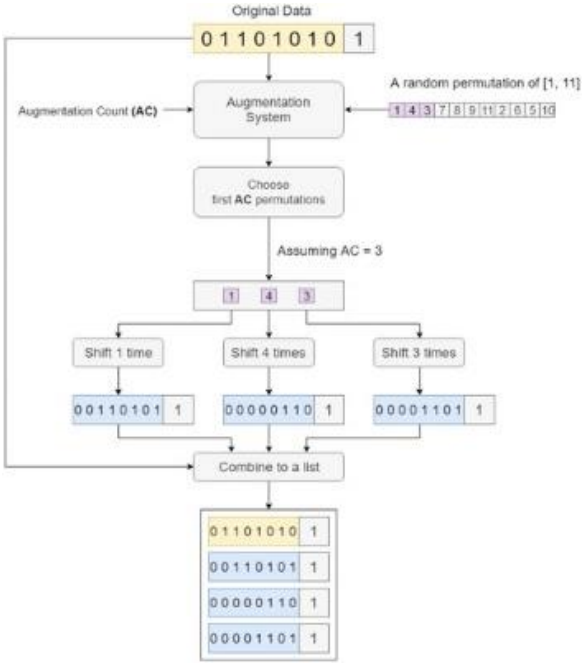


- ***Encoding***

Because there is no method to directly feed midi values into the network and make music, it is necessary to encode the music data into a network-readable format. For each MIDI file, the encoding method produces a matrix of binary vectors with a size of N 89, where N is the number of vectors in the piece. The first 88 bits of the vector represent the 88 notes on the piano keyboard. A zero denotes that no note is currently being played, while a one denotes the contrary. The repetition count is represented by the 89th bit in the vector. The repetition count indicates how many times this vector is repeated, or how long the present state of the keyboard is maintained until it changes. The repeat count in our initial studies was a natural number. However, having a natural number that is not normalized will mess up the loss calculation and convergence. So, at the end we used a real number in the range (0,1]. Where a number close to 0 meaning less repetitions, and 1 meaning it has the most inside it's MIDI file. The intervals are half-open because no vector should have a repetition value of zero, since it is semantically wrong. The least repetition value for a file will be 1/Maximum Repetition.



- ***Data augmentation***

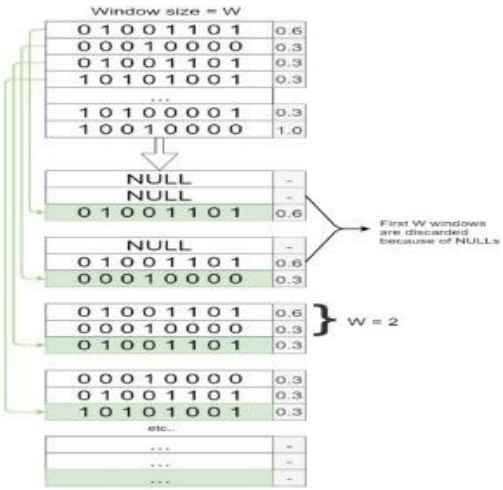


A melody can be played in any key and with extremely diverse notes while still having the same semantic meaning and sounding the same. Figure 9 shows how the same tune can be played in two

different keys using musical notation. The melody is played in the 'C major' musical key in the first line, and in the 'F major' musical key in the second. Even though they have completely distinct notes, these two melodies are equal when listened separately. Figure 9 shows how the same tune can be played in two different keys using musical notation. The melody is played in the 'C major' musical key in the first line, and in the 'F major' musical key in the second. Even though they have completely distinct notes, these two melodies are equal when listened separately. Augmentation is a solution to this problem.

- ***Timesteps***

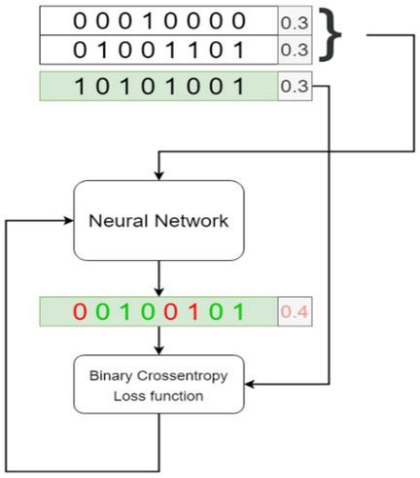
Since LSTMs expect the input shape to be in the format (timesteps, feature count), we must transform our data to be in this format. To do this we use a 'Sliding Window' method so that at each instance in the training, the sample contains itself and the previous *N* samples as shown in figure.



- ***Neural network***

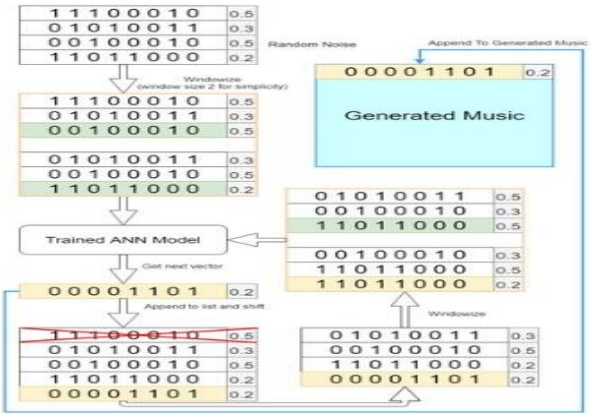
Having the windowed encoded dataset, it is fed into the NN. We separate the examples and labels such that we give an example at sample n with its timesteps in the range [n – w, n – 1] and let the NN to predict the sample at n + 1.

Figure uses a window from the data generated from the windowing in figure and shows how this window will be fed into the network.



- ***Generating music***

During network training, the network makes music at predetermined intervals and saves it in



MIDI format so that it can be listened to and

analysed. Figure 13 depicts how music is generated following training. The network is fed a matrix of either noise or a random matrix from the dataset and asked to predict the next vector to make music. We append this vector to the matrix, as well as the list of created vectors, and pop the last vector from the matrix after predicting the next vector. The same procedure is followed to anticipate the following vector. This is done multiple times until you get a piece of the appropriate length.

Conclusion

We attempted to investigate the behaviour of LSTM to generate music in this study by constructing a six-step approach that included MIDI format conversion to song format, encoding, augmentation, windowing, LSTM learning, and music generation. Diverse tests with various characteristics have been undertaken to study the improvement of learning. Our network's findings were not very outstanding in terms of music, but they weren't horrible either. Our model demonstrates a basic grasp of rhythm and harmony. The works produced have a simple framework with a beginning, an ending, and motif repetition. A motif network, similar to Walder's (2018) model, could have addressed some of this. Some of the unsolved issues include the model's output of a few nice notes followed by all zeros in some circumstances. It can also produce brief melodies that keep repeating indefinitely. These two issues should be thoroughly explored, and a post-processing procedure implemented to prevent such unusual situations. The third issue was that the loss function failed to account for the fact that the 89th bit is a float value. The model performance is musically acceptable and produces nice harmony with less sound if the first two flaws are ignored because they do not usually occur. For the third challenge, the LSTM structure should prioritise the 89th bit by linking the input directly to the output, allowing it to have a direct impact on the NN and loss function.

References

1. Chen, C. (2001). Creating melodies with evolving recurrent neural networks. [ebook]. Washington, DC: IEEE. Retrieved from http://nn.cs.utexas.edu/downloads/papers/chen.ijcnn01.pdf. [Google Scholar]
2. Christies.com. (2018). Is artificial intelligence set to become art's next medium? | Christie's. [online]. Retrieved from https://www.christies.com/features/A collaboration-between-two-artists-one-human-one-a-machine-9332-1.aspx. [Google Scholar]
3. Classicalarchives.com. (2018). Johann Sebastian Bach – Classical archives. [online]. Retrieved from https://www.classicalarchives.com/midi/composer/2113.html. [Google Scholar]
4. Koch, H and Dommer, A. (1865). H. Ch. Koch's Musikalisches lexicon. Heidelberg: J.C.B. Mohr. [Google Scholar]
5. Langston, P. (1988). Six Techniques for Algorithmic Music Composition. [ebook]. Morristown, New Jersey: Bellcore. Retrieved from http://peterlangston.com/Papers/amc.pdf. [Google Scholar]
6. Learningmusic.ableton.com. (2018). Keys and scales | Learning Music (Beta). [online]. Retrieved from https://learningmusic.ableton.com/notes-and-scales/keys-and-scales.html. [Google Scholar]
7. Rouse, M and Good, R. (2005). What is wavetable? – Definition from WhatIs.com. [online] WhatIs.com. Retrieved from https://whatis.techtarget.com/definition/wavetable. [Google Scholar]
8. Skuli, S. (2017). How to generate music using a LSTM neural network in Keras. [online]. Towards Data Science. Retrieved from https://towardsdatascience.com/how-to-generate-music-using-a-lstm-neural-network-in-keras-68786834d4c5. [Google Scholar]
9. Skymind. (2017). A beginner's guide to Generative Adversarial Networks (GANs). [online]. Retrieved form https://skymind.ai/wiki/generative-adversarial-network-gan. [Google Scholar]
10. Agarwala, N., Inoue, Y and Sly, A. (2017). Music composition using recurrent neural networks. [online]. Web.stanford.edu. Retrieved from https://web.stanford.edu/class/cs224n/reports/2762076.pdf. [Google Scholar]

******