Title: Code Quaver: Automatic Music Generation from CODE

Sadia Nasrin Tisha, Mushfika Rahman, Minakshi Debnath, Maisha Binte Rashid

8th December 2021

Abstract

This project is about producing music from source code. It is a deep learning-based model that can be utilized to understand the beauty of code through the art of music. The input of the model will be snippets of high-level language code, the output will be music. The project will correspond to how artistic the code is for generating music. To generate the music deep learning techniques like Long short term architecture and wave net architecture are used here. For calculating the fractal dimension of code Block Artefact Method is used here to find out how beautiful the code is through music.

1 Introduction

The process of turning data to music has been around for a long time. Sonifiation is the process of hearing the data which is the use of non-speech audio to represent information, taking data of some kind and create sound with it [1]. It is observed that, sound adds another layer to understanding like in the cinemas, without looking at screen anyone can understand the difference in output. [2]. Music is the ultimate language. It has its own art. Many great composers have created compositions that were both innovative and purposeful throughout history. Composers like Bach were recognized for their meticulous attention to detail when creating compositions with a lot of underlying melodic structure.
On the other hand, every programming language has its own pattern which beautifies them from one another. There is an art in every line of codes[3]. Creating music from the programming language helps to identify how beautiful a code can be. It can be heard nicer if the code is generated in a great pattern.
Music is the collection of tones of different frequencies. So, the Automatic Music Generation is a process of composing a short piece of music with minimum human intervention. It all started with choosing sounds at random and merging them to create a piece of music. Wolfgang Amadeus Mozart composed "Musikalisches Würfelspiel" in 1787, a musical technique for producing a 16-measure waltz by rolling dice. Each measure is chosen from a collection of 11 pre-composed musical bits in this method [8]. Another intriguing concept was to compose music using musical language. Musical Grammar entails the understanding required for the right organization and combining of musical sounds, as well as the proper performance of musical works. Iannis Xenakis employed the notions of statistics and probability to produce music in the early 1950s, which became known as Stochastic Music. He described music as an accidental succession of components (or sounds). As a result, he used stochastic theory to formulate it. His random element selection was entirely based on mathematical notions[9].
Recently Deep Learning architectures have become the state of the art for Automatic Music Generation. There are several deep learning based networks that will produce music from one sequence to another sequence. Deep Learning is a branch of artificial intelligence that is inspired by the structure of the human brain. These networks are capable of learning any non-linear function and extracting features from a dataset automatically. Neural Networks are known as Universal Functional Approximators because of this. Hence, Deep Learning models are cutting-edge in domains such as Natural Language Processing (NLP), Computer Vision, Speech Synthesis, and others. This project is a demonstration of the uses of automatic music

generation using deep learning sequence to sequence modeling architecture from a line of code.
Moreover, The fractal feature of many types of music was investigated in the time domain. The Variation technique is used to determine the fractal dimension of a large number of distinct musics. The use of an analysis of variance reveals that fractal dimension aids in the classification of different types of music[10]. This project measure fractal dimension from source code and also from music to find out how beautiful a code can be. Block Artefact Method(BAM) is used here to calculate the fractal dimension. So the main motivation of this project is understand the beauty of code through music where the music can be generated any kind of data by using deep learning method.

2 Literature Review

Coleman et [3] showed that, beauty and readability are related as we hypothesized and beauty appears to measure a unique property in code called aesthetic appeal. Their data further suggests that indentation is reliably correlated with readable code, more than mnemonics or comments and of the four styles, GNU style is the most correlated with readability. They studied with relativistic and fractal measure of programming style which they have mentioned as beauty factor or beauty as short. And they came up with two hypothesis. The first one is programming with increasing beauty makes the code more readable. And the second thing is Aesthetic appeal of code which is different from readability. They have define this aesthetic appeal of code as a unique property of code which can be measured by the beauty.

Forsythe et [4] mentioned in their work that, ratings of beauty are thought to depend primarily on judgments of visual complexity and artistic images contain arrangements of visual elements intended to appeal to the senses or emotions. In the study of aesthetic processes, the curvilinear relationship between beauty and visual complexity has received the most attention. In such studies, fractal dimension has established as a very useful factor in both the understanding and authenticity of major work of art. They have mentioned in their study that, Beauty and/or interest are thought to depend, primarily on judgments of visual complexity. Also fractal dimension is related to visual complexity. They hypothesized in their study that, measuring fractal dimension and visual complexity can be taken account for more of the variance in judgment of beauty than judgments of visual complexity alone. Mainly, , they evaluated an automated measure of visual complexity as an unbiased measure of complexity in art works.

Huang et [5] worked with music generation using deep learning. They have worked to represent notes in music as a vector and build a generative neural network architectures that effectively express the notions of harmony and melody. They used 2 layered LSTN and RNN architecture. And they evaluate their model's quality if the model incorporating the notion of musical aesthetic. And by musical aesthetic it means how good the music is. They have shown that their multi-layer LSTM, character-level language model applied to two separate data representations is capable of generating music that is at least comparable to sophisticated time series probability density techniques prevalent in the literature.

Kalingeri, V. et al. also worked with deep learning while generating music[6] . They deals with th generation of music using raw audio files in the frequency domain relying on various LSTM architectures. To capture rich characteristics in the frequency domain and improve the quality of music generated, fully connected and convolutional layers are combined with LSTMs.

3 Dataset

The dataset is collected from GitHub Code Snippets [7] which is an SQLite file called snippets-dev.db. This file contains a single relation called snippets where it contains 21 high level programming languages includes Bash, C, C++, CSV, HTML, JSON, Java, JavaScript, Python, Ruby, and so on. From here Python has been filtered. Figure 1 shows the attributes of the dataset. Here the id is an integer primary key, snippet contains each block of code, file name, GitHub report, license, hash commit, number of lines, and chunk size. The size of the full dataset is over 60GB but after filtering into Python it becomes 68 MB where there are 196466 rows and 21 columns.

	id	snippet	language	repo_file_name	github_repo_url	license	commit_hash	starting_line_number	chunk_size
0	186314	version = '7'\nhtml_title = "Guzzle Documentat...	Python	guzzle/guzzle/docs/conf.py	https://github.com/guzzle/guzzle	MIT	55d46a8ba3239be2439fb660b5cc9fba69155113\n	15	5
1	186322	# Path to a touch icon\n # "touch_icon"...	Python	guzzle/guzzle/docs/conf.py	https://github.com/guzzle/guzzle	MIT	55d46a8ba3239be2439fb660b5cc9fba69155113\n	55	5
2	186323	"base_url": "http://guzzlephp.org"\n\n ...	Python	guzzle/guzzle/docs/conf.py	https://github.com/guzzle/guzzle	MIT	55d46a8ba3239be2439fb660b5cc9fba69155113\n	60	5
3	250563	from tensorflow.python.estimator.model_fn impo...	Python	tensorflow/tensorflow/tensorflow/core/platform...	https://github.com/tensorflow/tensorflow	Apache-2.0	686406b968467e7e226fb86601af0f8616ff9736\n	25	5
4	250577	self.assertFalse(gfile.Exists('ram://exist...	Python	tensorflow/tensorflow/tensorflow/core/platform...	https://github.com/tensorflow/tensorflow	Apache-2.0	686406b968467e7e226fb86601af0f8616ff9736\n	95	5

Figure 1: A portion of Dataset

4 Methodology

This project is divided into two part. At first, fractal dimension is calculated from a line of code using BAM and then music generation from block of codes. Figure 2 demonstrate the workflow of this project.

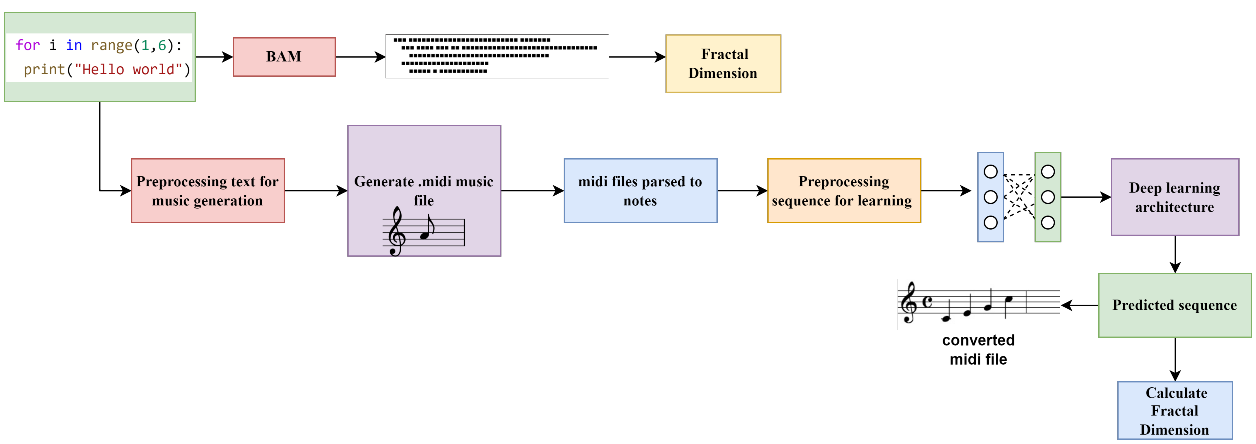

Figure 2: Work Strategy of project

4.1 Block Artefact Method(BAM)

Block Artefact Method is used here to encodes the bitmap with block characters in place of string characters. In the dataset, each line of code was separated in different rows. For pre processing the data, files are concatenated in the same row where for each row the chunk size is 5. Here for the first row, the staring line number is 15 and the file is 40. After using GroupBy method the data are grouped into 52204 rows for snippet and file name column. Then binarized 2D array implemented but convert the text into 1 and blank into 0 shown as Figure 3 . From this binary array Unicode is generated for encode this. Here value 1 is converted into black box and 0 is Unicoded as a blank box and saved the Unicode value as an image format for each array. Figure 4 shows the encoded values.

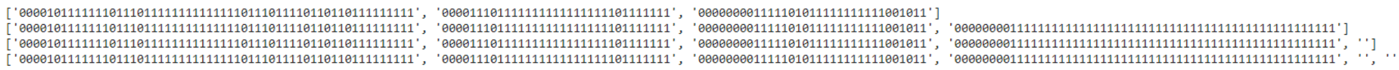

Figure 3: 2D binary array of source code

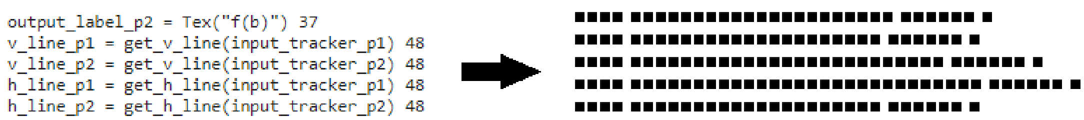

Figure 4: Encoded version of source code

Fractal dimension is one method to appreciate the beauty in something visual. It will help to find out the difference between visual things by fractal dimension coefficient. Therefore for each encoded image Fractal dimension is applied. The FD value for first ten encoded image is given below:-
FD: 1.486017717938463
FD: 1.692707993003542
FD: 1.6854831635503211
FD: 1.5993498640793822
FD: 1.6174012686640085
FD: 1.5644849233800364
FD: 1.4447103626531086
FD: 1.5738532482890069
FD: 1.5069017075889537
FD: 1.4530465659197969

4.2 Music Generation

The next part of this project is to generate music from raw code. For generate music first we need to preprocess the data for music generation. Later we will use deep learning model for training the music.

4.2.1 Data Preprocessing

- **Preprocessing for Creating Music**

  Piano has 8 notes (A,a,b,C,c,D,E,F,f,g). Each block of source code is converted to get the frequency of piano notes and rest note. All the character is mapped to piano notes and rest note. From the converted piano notes calculated the frequency for each note using the equation :

$$note_{freq} = base_{freq} * 2^{\frac{n}{12}} \tag{1}$$

  The base frequency for our experiment was 261.63. By adding the continuous frequency produced sine wave which by applying some over tunes and with help of python music21 library generated music of .midi format. We contrasted the created music with human subjective sense of quality due to a lack of quantitative performance measures.

- **Preprocessing midi files for training**

  The generated midi files serves as a dataset for training the neural network. As we want to work as a language model we parsed the piano notes and chords from the midi files and added to a vector. Prepared an input sequence and corresponding output sequence for the network. As neural network works well with the numerical data, mapped both the input output sequence to numeric representation.

4.2.2 Deep learning Model

Before training the model, we split the sequence in 80-20 ratio to create test and validation set. The validation loss served as the checkpoint for saving model's weights. After that we have used two deep learning architecture LSTM architecture and WaveNet architecture.

- **Wave Net Architecture**

  If we want to explain the process of generating music in short, we can say it is a randomly selecting sounds and combining them to form a piece of music. Notes and Chords make up the majority of music. Deep Learning is a branch of machine learning that is inspired by the structure of the human brain. These networks are capable of learning any non-linear function and extracting features from a dataset automatically. Google DeepMind developed WaveNet, a Deep Learning-based generative model for raw audio. WaveNet's primary goal is to produce new samples from the data's original distribution. As a result, it's called a Generative Model. Wavenet is similar to an NLP language model. A language model attempts to predict the next word given a sequence of words. WaveNet, like such a language model, seeks to predict every next sample given a sequence of samples. As an input, WaveNet takes a piece of a raw audio wave. WaveNet is a deep generative audio waveform model. For our experiment we used wavenet like language model. Given some music notes it will predict sequence for next music notes. WaveNet uses convolutional neural networks to create predictions that are influenced by all past observations, which is an unusual method. It tries to anticipate the next amplitude value given the series of amplitude values. Causal Dilated 1D Convolution layers are the foundation of WaveNet. Dilated convolutions are used by WaveNet to save past information. Dilated convolutions can capture global context while while preserving long-term network dependencies.

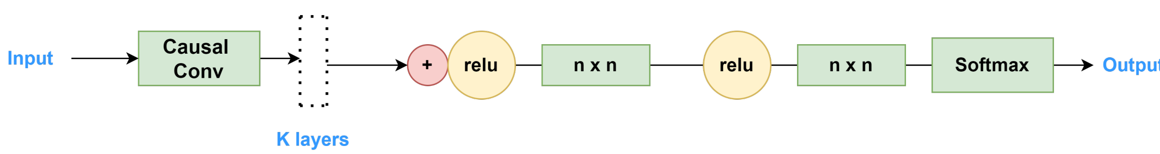

Figure 5: Wavenet Architecture

  causal 1D convolution is fed as input. The output is then fed into two dilated one-dimensional convolution layers. An embedding is a low-dimensional space into which high-dimensional vectors can be translated. Machine learning on big inputs, such as sparse vectors representing words, is made easier via embedding.

  An embedding captures some of the input's semantics by clustering semantically related inputs together in the embedding space. A model's embedding can be learned and reused. In our mode the dense representation of embedding layer is 100. For training our model we kept our batch size 128 and epoch number 50. Due computional barrier it made the trainning process slower and model stops doesn't finish the trainning process We have tried with higher epoch and batch size. And for activation function we used 'Relu'. The learning rate was low at first but then we kept the learning rate .001. For optimizer we have used 'adam'. We have used sparse categorical cross entropy as our loss function. One advantage of sparse categorical cross entropy is that it saves time in both memory and computation because each class is represented by a single integer rather than an entire vector.

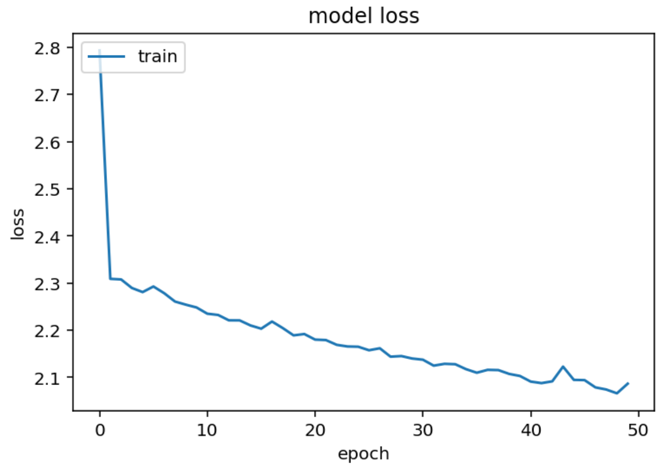

Figure 6: WaveNet Model Loss

- **Long Short Term Memory (LSTM) Architecture**

  Long Short Term Memory Model, popularly known as LSTM, is a variant of Recurrent Neural Networks (RNNs) that is capable of capturing the long term dependencies in the input sequence. LSTM has a wide range of applications in Sequence-to-Sequence modeling tasks.

  Because of this features, LSTM was a choice of model for training purpose in our project. This project used the LSTM sequence to sequence model for generating music. Converting the Midi file to ABC notation sequence as a input to generate music in LSTM was a challenging part of this project. LSTM performs really good with Abc notation, we faced challenge converting midi file to abc notation but prediction was not producing any music. So we generate the midi file as sequence and train the model. Due computational barrier it made the training process slower and model stops and doesn't finish the training process. Later experimented by parsing the notes and chords there again produced sequence from which converting to music was providing invalid sequence. Figure 7 shows the model loss of LSTM model for 2000 iterations.

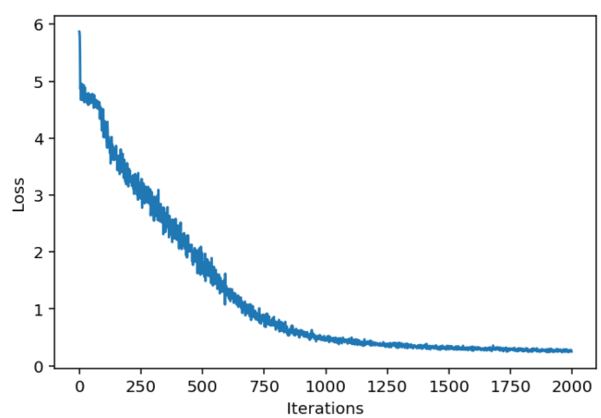

Figure 7: LSTM Model Loss

5 Result Analysis

We have generate music in midi file by using LSTM and wavenet model. In wavenet architecture, from a random note we have generated 100 length sequence and generate roughly one miniute music. The generated music is more melodious than the music produced by LSTM model. As LSTM provide us invalid sequence and that sequence of music is not as melodious as the music generated by wavenet. After experimenting LSTM and WavNet , we reached to the conclusion that waveNet is the model better suited for our project. LSTM is computationally more expensive than Wavenet. WaveNet also converges faster than LSTM.

6 Future Works

Music has some special characteristic that makes the music more beautiful. To produce a melodious song there needs to be some quantitative measurement that will help us to generate better music. In this project, we have generated music directly from the source code using string characters. In the future, we will try to implement some quantitive measurements of music for generating melodious music.

Secondly, we have generated some sample midi file music for training where the size of sample files is limited. Deep learning sequence to sequence method train data based on the input data. The more data it trains, we got the better results from this. In future, we will try to increase the size of sample files for generating melodious, lengthy, and better music by training.

In addition, in case of generating a better quality of music, we need to over-tune the music using music architecture. For that, we will get more melodious music in both preprocessing and post-processing phases. Also, we will use different deep learning architecture for music generation to get better performances.

Moreover, to find out how beautiful a code is through music, we will calculate the fractal dimension of generated music and compare that with the previous fractal dimension value where we calculate FD from a block of Code. This FD will help us to determine how beautiful the code is. For calculating the FD we will generate the music sheet in the black and white image from the generated music file and calculate the FD from the image of the music sheet.

Also, there are some other approaches to determine how beautiful a code can generate music instead of FD. So we will try alternative approaches to determine the beauty of code through music.

7 Conclusion

Our project is a demonstration of how we can use deep learning architecture to produce music automatically from lines of source code. Also we have illustrated the usefulness of fractal dimension which has been very helpful specially when people deals with works of art. It works as the measurement of aestheticness. We have shown two different approach to generate music and analysis the final output. Then we have came to a point that wavenet worked better for our process. We implemented wavenet as a generative model where it works as sequence to sequence modeling manner. We implemented wavenet model as a language model, we want to experiment with the raw input files also. For more improvement we will try different approach . And also we will implement the fractal dimension calculation process to finally state that how much beautiful the code is.

References

[1] https://aaresearch.psu.edu/artist/sonifications-of-the-universe-and-more/what-is-sonification.

[2] https://twotone.io/about/, Datavized Technology and Google news

[3] Coleman, Ron, and B. Boldt. "Aesthetics Versus Readability of Source Code." International Journal of Advanced Computer Science and Applications 9.9 (2018): 12-18.

[4] Forsythe, Alex, et al. "Predicting beauty: fractal dimension and visual complexity in art." British journal of psychology.102.1 (2001): 49-70.

[5] Huang, A. et al. " Deep Learning for Music"arXiv:1606.04930v1,2016.

[6] Kalingeri, V. et al."Music Generation Using Deep Learning" arXiv:1612.04928,2016.

[7] https://www.kaggle.com/simiotic/github-code-snippets-development-sample

[8] https://jythonmusic.me/ch-6-randomness-and-choices/

[9] https://ccrma.stanford.edu/ blackrse/algorithm.html

[10] Bigerelle, M., Iost, A. (2000). Fractal dimension and classification of music. Chaos, Solitons Fractals, 11(14), 2179-2192.