Music Generation

Abd Al-Muttalib Ibreighith
dept. Software Engineering
Bethlehem University
Bethlehem, Paslestine
officialabdib@gmail.com

***Abstract*—This research paper explores the use of deep learning techniques, specifically the WaveNet architecture, for automatic music generation. The objective is to create enjoyable music without the constraint of adhering to traditional musical theory rules. The learning process focuses on modeling audio waveforms directly, utilizing Long Short-Term Memory (LSTM) networks in conjunction with WaveNet. The WaveNet architecture, built upon Causal Dilated 1D Convolution layers, allows for capturing long-range dependencies and producing high-quality music outputs. The research utilizes a dataset of classical music MIDI files, and the training phase involves predicting subsequent amplitude values given a sequence of input values. The generative phase involves iteratively selecting the most probable values from the model's learned patterns to construct a new musical composition. Overall, this research contributes to the advancement of deep learning in the field of automatic music generation.**

Keywords—Long short term-memory LSTM

I. Introduction

Music composition is an art form, and even playing written music requires significant effort on the part of people. It is difficult and worthless to create an algorithm that can complete both tasks simultaneously at this degree of complexity and abstraction. Since written music is used as training data to identify useful musical patterns, it is simpler to model this as a learning issue.

The objective is to produce music that is enjoyable to listen to but may not necessarily sound like music played by humans. We anticipate that the learning algorithm will locate areas where music sounds well without imposing any requirements that it follows to rules of musical theory. The learning or creation process is therefore not aided by any elements like notes, notation, or chords; rather, we deal directly with the outcome, which is represented by audio waves.

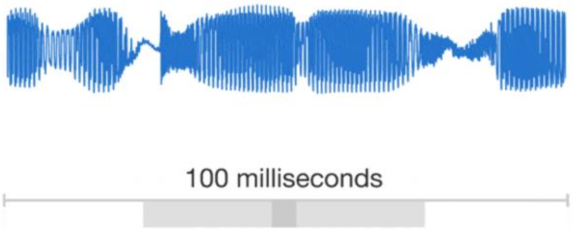

Figure 1: Audio Wave form

Audio waveforms are one-dimensional signals that change with time in such a way that audio fragments from different timesteps transition smoothly from one another. A straightforward representation of the raw audio waveform is shown in Fig. 1. A recurrent neural network would be a logical choice of design to simulate a time-varying function due to its capacity to share parameters over time. To represent the signals, we will specifically use Long Short-Term Memory (LSTM) networks beside WaveNet architecture[1].

One of the interesting ideas of generating music is done by Iannis Xenakis [2] who created what is known as stochastic music in the early 1950s by composing music using the ideas of statistics and probability. According to his definition, music is a collection of components (or sounds) that happens by accident or chance [3]. He, therefore, used stochastic theory to formulate it. His choice of random components wholly relied on mathematical ideas.

Recently, Deep Learning architectures have advanced to the state-of-the-art in Automatic Music Generation. Thus, a lot of deep learning architectures have the capability of creating audio for a sequence of music. WaveNet architecture is one of the methods used for automatic music composition which is used in this research.

II. Related Work

There has been a lot of research on applying musical elements like notes, chords, and notations to create music using LSTMs. These studies offer encouraging findings and show that LSTMs can collect the necessary long-range data for music production. In these methods, a common architecture involves structured input of a music notation from MIDI files that are fed into an LSTM at each timestep. The next timestep's encoding is predicted by the LSTM, and so on. The error is computed as a negative log loss of the predicted value and the true value.

The greatest findings are found in the paper [4], which employs an extremely deep dilated convolutional network to generate samples one at a time sampled at 16KHz. There has also been work that directly models the raw audio recordings. They were able to capture long-range dependencies and produce a decent representation of the audio by increasing the degree of dilatation at each depth. It is simple to picture the level of complexity they had to achieve to collect sufficient time information to effectively encode the music. Because the task was approached as a classification problem, where the generated audio sample was classified into one of 255 values, training the network was rather simple even though the huge depth. As a result, they were able to employ the negative log loss rather than the mean squared loss. Due to the lowered risk of overfitting to outliers, convergence time was shortened. Although this method works, the depth makes it incredibly computationally demanding; on Google's GPU clusters, it takes nearly a minute to produce one second of audio, pushing us to think about a faster option.

Nayebi et al.'s recent work [5] has also worked with audio samples, however they focus on the frequency domain of the audio rather than trying to learn and generate from the raw audio samples. Because the network can train and predict using a collection of samples that make up the frequency domain rather than just one sample, this method is substantially faster. There are no limitations on the type of music that can be produced by the frequency domain because it can still represent all audible audio signals. The samples in the Fourier domain are given as input at each timestep into a single LSTM architecture that they employ. The output of the LSTM is a Fourier representation of the signal for the following timestep. During network training, the mean squared difference between the anticipated output and the true frequency representation is employed as the cost function.

These advancements in LSTM-based music generation, along with alternative approaches focused on the frequency domain, present exciting opportunities for generating high-quality music while addressing computational efficiency concerns. By combining the strengths of deep learning architectures with insights from audio signal processing, researchers are making significant strides towards creating AI-generated music that is both artistically compelling and computationally viable.

III. What are the Constituent Elements of Music?

Music is essentially composed of Notes and Chords. Let me explain these terms from the perspective of the piano instrument:

- Note: The sound is produced by a single key.
- Chords: The sound produced by 2 or more keys simultaneously is called a chord. Generally, most chords contain at least 3 key sounds.
- Octave: A repeated pattern is called an octave. Each octave contains 7 white and 5 black keys.
- Pitch: refers to the frequency of the sound, or how high or low it is.

IV. Methodology

A. *WaveNet Architecture*

WaveNet is a Deep Learning-based generative model for raw audio developed by Google DeepMind. WaveNet's primary goal is to create fresh samples from the data's initial distribution. Consequently, it has the name "Generative Model." Similar to an NLP language model is Wavenet. A language model attempts to predict the following word given a list of words. In WaveNet, given a series of samples, it attempts to predict the following sample, much like a language model.

The WaveNet architecture, used for various tasks like speech synthesis and music generation, relies on Causal Dilated 1D Convolution layers as its fundamental building blocks. To better grasp the significance of these components, let's explore related concepts.

Convolution, a mathematical operation, is employed to extract features from an input. In image processing, convolution involves a linear combination of specific image portions with a kernel. In the context of 1D convolution, a kernel or filter moves along a single direction to process the input sequence as shown in Fig. 2. The objective of 1D convolution is akin to that of the Long Short-Term Memory (LSTM) model, addressing similar tasks. In 1D convolution, the kernel moves along the input sequence in one direction, impacting the output based on factors such as kernel size, input shape, padding, and stride.

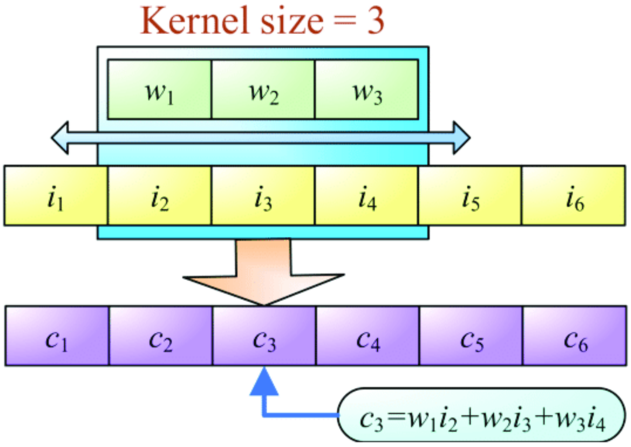

Figure 2: Kernal or filter

Now, let's delve into the significance of using Dilated Causal 1D Convolution layers by exploring different types of padding. When employing "valid" padding (see Fig. 3), the input and output sequences have different lengths, with the output being shorter. Conversely, with "same" padding (see Fig. 4), zeroes are added on both sides of the input sequence to ensure equal input and output lengths. So, Causal 1D Convolution is distinguished by its padding, specifically the addition of zeroes to the left (as shown in Fig. 5) of the input sequence. This preserves the autoregressive principle, allowing convolutions to consider only past or earlier timesteps. The receptive field of a network, indicating the number of inputs influencing an output, is significantly limited in causal convolutions.

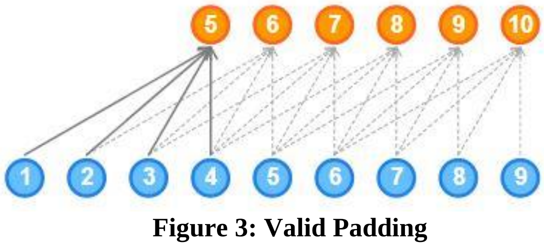

Figure 3: Valid Padding

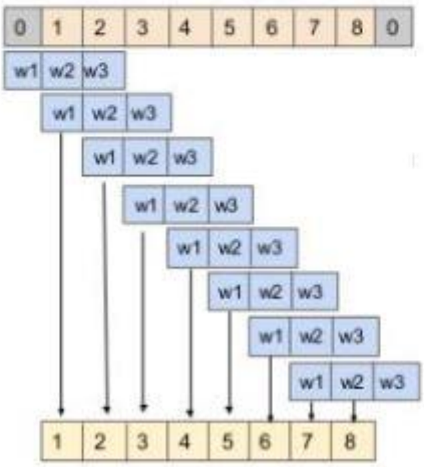

Figure 4: Same Padding

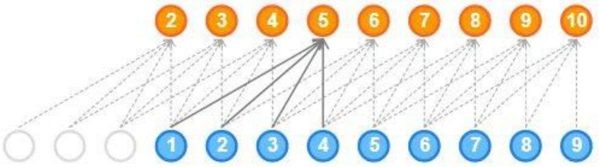

Figure 5: Adding zero to the left padding

As shown in Fig. 6, the output is influenced by only 5 inputs. Hence, the Receptive field of the network is 5, which is very low. The receptive field of a network can also be increased by adding kernels of large sizes but keep in mind that the computational complexity increases.

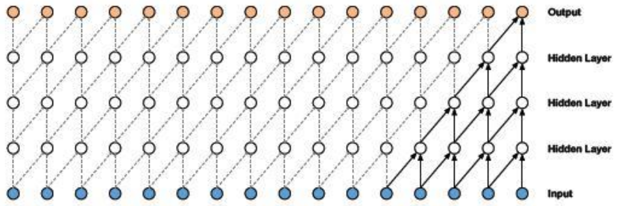

Figure 6: Very low networks' receptive field

To address this limitation, Dilated 1D Causal Convolution introduces the concept of dilations or spaces between the values of a kernel. The dilation rate determines the number of spaces, thereby influencing the receptive field. By increasing the dilation rate at each hidden layer, the dilated 1D convolution network exponentially expands the receptive field, capturing more context from the input sequence. As shown in Fig. 7, the output is influenced by all the inputs. Hence, the receptive field of the network is 16. The WaveNet architecture utilizes Residual Blocks that incorporate Residual and Skip connections to expedite model convergence. These connections aid in information flow and facilitate the training process.

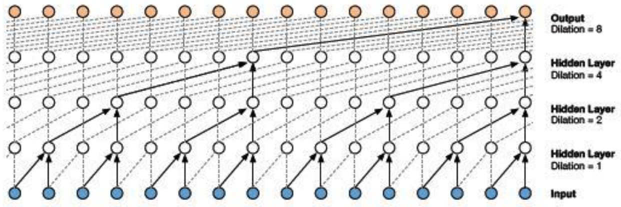

Figure 7: Receptive field influenced by all inputs

As we can see in Fig. 8, the workflow of WaveNet involves:

- Feeding the input into a causal 1D convolution,
- Followed by two dilated 1D convolution layers with sigmoid and tanh activations.
- The element-wise multiplication of the activations produces a skip connection.
- Additionally, the residual connection involves the element-wise addition of the skip connection and the output from the causal 1D convolution, enabling better modeling and representation.

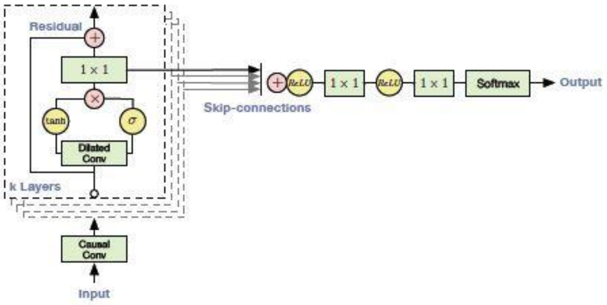

Figure 8: The workflow of WaveNet

By leveraging Causal Dilated 1D Convolution layers and incorporating Residual and Skip connections, WaveNet exhibits powerful capabilities for tasks like music generation and speech synthesis, with the ability to capture long-range dependencies and produce high-quality outputs.

B. Dataset

The dataset utilized in this research comprises a collection of classical music files obtained from various sources [6]. The original dataset consisted of 1662 MIDI files, encompassing diverse lengths ranging from 1 to 40 minutes. To accommodate hardware resource limitations and enhance trainability, the dataset was partitioned into a smaller subset containing 500 MIDI files with durations ranging from 1 to 5 minutes. This subset was further divided into an 80% training set and a 20% testing set to facilitate model training and evaluation.

C. Training and Testing Phase

The training phase is a Many to One problem where the input is a sequence of amplitude values and the output is a subsequent value. WaveNet accepts a piece of a raw audio wave as input. The term "raw audio wave" describes how a wave is represented in the time series domain. As previously seen in Fig. 1, an audio wave is represented in the time-series domain as amplitude values that are captured at various time intervals.

WaveNet aims to predict the next amplitude value given the sequence of the amplitude values. Consider, for example, a 5-second audio wave with a sampling rate of 16,000 (i.e., 16,000 samples per second). For the past five seconds, 80,000 samples have been captured at various intervals. Let's divide the audio into equal-sized segments, say 1024 (this number is a hyperparameter).

As shown in Fig. 9, each chunk's output is dependent exclusively on the past data (i.e., past timesteps), not on future timesteps. So, both the task and the model are referred regarded as autoregressive tasks.

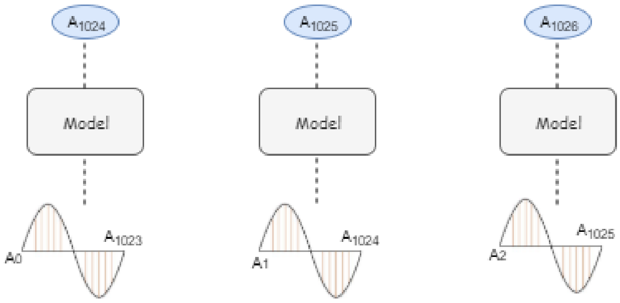

Figure 9: Input output autoregressive task

D. Generating Phase

In the generative phase of automatic music generation, a series of steps are employed to create new musical compositions using deep learning techniques. The process begins by:

1- Selecting a random array of sample values as the initial starting point for the model.

2- The model then generates a probability distribution over all the possible samples based on the learned patterns and rules from the training data.

3- From this distribution, the value with the highest probability is selected and added to an array of samples, representing a musical element in the composition.

4- The first element of the array is then removed, serving as input for the next iteration to generate the subsequent musical element.

5- This iterative process continues as the model repeatedly generates probability distributions and selects the most probable values, iteratively constructing the composition by appending them to the sample array.

This cycle of generating probabilities and updating the input continues for a predetermined number of iterations, resulting in a final musical composition that is a product of the model's learned patterns and the random seed values initially provided.

NOTE: All information and code retrieved from “Want to Generate your own Music using Deep Learning? Here’s a Guide to do just that!” Article [7].

V. Results

As seen in Fig. 10, and during the training process, the model underwent three epochs, and the loss values for each epoch were 5.1239, 4.6501, and 4.5855, respectively. The initial epoch started with a loss value of 5.1239, indicating a relatively high level of error or mismatch between the predicted and actual values. As the training progressed to the second epoch, there was a notable improvement, with the loss decreasing to 4.6501. This reduction in loss signifies that the model learned from the training data and adjusted its parameters to better approximate the desired outputs. In the final epoch, the loss further decreased to 4.5855, indicating continued refinement and convergence of the model. These decreasing loss values across the epochs suggest that the model is making progress in minimizing the discrepancy between its predictions and the actual values, leading to improved performance and accuracy in music generation tasks.

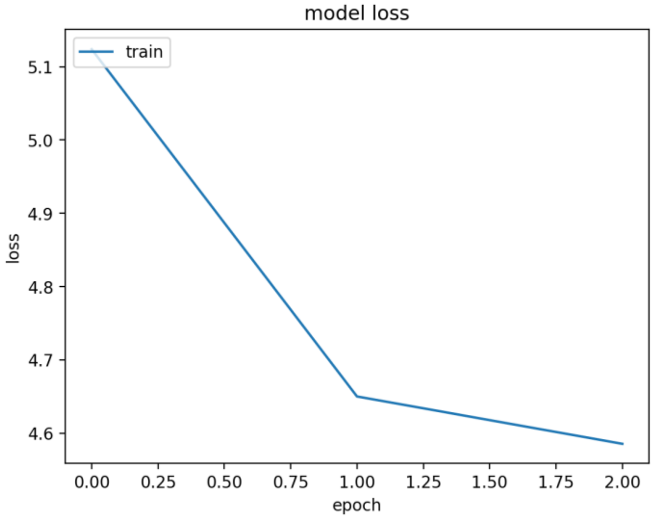

Figure 10: Training results

VI. Conclusion

In summary, the research findings highlight the effectiveness of the WaveNet architecture in automatic music generation. Over the course of three training epochs, the model consistently improved, as indicated by decreasing loss values of 5.1239, 4.6501, and 4.5855, respectively. By incorporating Causal Dilated 1D Convolution layers and Residual and Skip connections, WaveNet successfully captured long-range dependencies and produced high-quality music outputs. While the generated music may not strictly adhere to traditional musical theory, it aimed to create enjoyable and unique compositions. These results underscore the potential of deep learning techniques, particularly WaveNet, in the realm of music composition, offering innovative possibilities for creating captivating musical pieces.

VII. Discussion

The limited size of the training dataset highlights the significance of fine-tuning pre-trained models to enhance the system's robustness. Leveraging existing knowledge and models can be a valuable strategy to improve the performance of music generation systems in the face of data scarcity. However, it is crucial to emphasize the importance of collecting a large and diverse training dataset. Deep learning models excel when trained on extensive and varied datasets, enabling them to capture comprehensive patterns and generate music that is diverse and creative. To unlock the full potential of deep learning for automatic music generation, the future vision entails a concerted effort to amass vast and diverse training data, enabling the models to push the boundaries of musical composition and produce increasingly innovative and captivating musical pieces.

References

[1] WaveNet: A generative model for raw audio. RSS. (n.d.). https://www.deepmind.com/blog/wavenet-a-generative-model-for-raw-audio

[2] Wikimedia Foundation. (2023, May 8). Iannis xenakis. Wikipedia. https://en.wikipedia.org/wiki/Iannis_Xenakis

[3] Sweetwater. (2004, December 14). Stochastic music. inSync. https://www.sweetwater.com/insync/stochastic-music/

[4] Oord, A. van den, Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., &amp; Kavukcuoglu, K. (2016, September 19). WaveNet: A generative model for raw audio. arXiv.org. https://arxiv.org/abs/1609.03499

[5] Nayebi, A., and Vitelli, M. 2015. Gruv: Algorithmic Music Generation using recurrent neural networks. (n.d.). https://cs224d.stanford.edu/reports/NayebiAran.pdf

[6] Google. (n.d.). Dataset. Google Drive. https://drive.google.com/drive/folders/1bmUGIbHLD2VNoK52jqKOAMdWkuaLmjBb?usp=sharing

[7] Pai, A. (2021, January 4). *Want to generate your own music using Deep learning? here's a guide to do just that!* Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/01/how-to-perform-automatic-music-generation/